

Entropy

I. Introduction

The principle of entropy can be confusing to people. When we discussed it in class, the definitions of it remained unclear in my mind and I could not understand what was meant by an 'ordered' state versus a 'chaotic' state. The examples given did not help me in understanding the principle, either, as in my mind what was called an 'ordered' state was in fact chaotic and vice versa. Here is an example of how I viewed the different states:

If we imagine a 10 x 10 set of jars lying on the floor and then imagine a set of 200 pennies being dropped from high above the jars, we would find that, on average, each jar would contain 2 pennies.

In my mind, this state seems very ordered, as the pennies are spread evenly among the jars, and the alternative, all 200 pennies falling into the same jar, would be highly improbable, and therefore chaotic. The converse, however, is the truth. The pennies being evenly spread among the jars is evidence of high entropy, a chaotic state.

In order to resolve the confusion I felt, I began imagining the same type of situation, only using the movement of particles in a box rather than pennies being dropped. If we were to create an ordered set of particles moving in the box, we could say that perhaps all of the particles are moving with the same speed and in the same direction. We could easily predict where each particle would be at any time. This system would be an ordered, low entropy state.

What would happen when the particles bounce off of the wall and begin to collide? Their directions would no longer be the same, and more collisions would start to take place. Quickly, we would lose the order that was present in the set of particles, and the possibility of being able to predict where a given particle will be at a given time diminished. The particles are now in a chaotic, high entropy state.

I teach a Physics 107 class, in which we once did an experiment using an air table. The students were to take discs of various sizes and analyze their collisions in two dimensions. While I was waiting for the students to finish their reports, I took a few discs of the same size and attempted to replicate the idea of entropy. I pushed all of the discs in one group so that they all moved in the same direction. Once they hit the wall and began colliding, I could see quickly that the discs began moving in random motion. The number of collisions was too great to be able to say where any one of the discs would be at a given time.

II. The Project

For my term project, I have written a java applet that can be viewed with any web browser that demonstrates the principle of entropy as I discussed in the introduction.

A. Preparation

1. **Simplifying Assumptions** - In order to develop this program, I made some assumptions to make the programming simpler:
 - i. All of the particles are spherical
 - ii. All of the particles are of the same size and mass.
 - iii. All of the particles move with the same magnitude of velocity.
 - iv. All of the collisions are perfectly elastic, so that the magnitude of velocity does not change after a collision.

2. **Math Preparation** - To be able to write this program, algorithms needed to be developed to do the following:

- i. **Reflections off of walls**

The first problem to solve deals with telling the computer at what angle each particle moves. The angles tracked by the computer are all measured from the x-axis from 0° to 360° . This means that one particle moving at 45° reflects off of a vertical wall to a direction of $90^\circ + 45^\circ$ or 135° . A simple analysis shows that a reflection off of a vertical wall has no effect on the y component of the direction, and that the x component switches direction. The new direction is $\theta' = 180^\circ - \theta$, where θ represents the original direction of motion. A reflection off of a horizontal wall likewise has no effect on the x component and reverses the y component. The new direction is $\theta' = -1 * \theta$.

- ii. **Reflections off of other particles**

The difficulty here came in finding out how a collision affects the direction of motion of one of the particles. Because we have assumed the particles to be spherical, they will have contact at only one point when they collide. It can be shown that the change in direction is a simple reflection on the line tangent to the two particles.

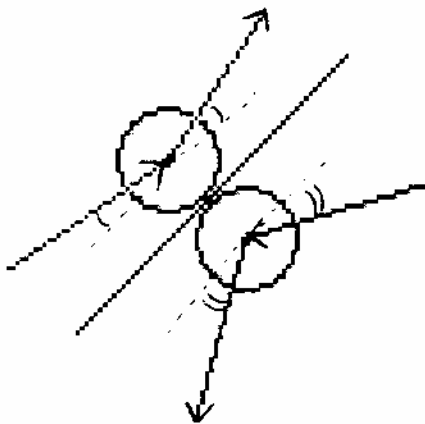


Figure 1: The two particles collide, and the reflection occurs on the line tangent to the two particles, regardless of what direction of motion each particle has originally. The new direction of motion *is* dependent on the angle at which the particles approach the tangent line. The angle of the tangent line will be called α .

The angle of the tangent line, α , must first be found. The line tangent to the two colliding particles can be found by calculating the perpendicular bisector of the line segment joining the positions of the two particles.

This segment will have a slope of: $m = \frac{y_1 - y_2}{x_1 - x_2}$.

The slope of a line perpendicular to that would be: $m_{\text{tan}} = -\frac{x_1 - x_2}{y_1 - y_2} = \frac{x_2 - x_1}{y_1 - y_2}$

We can now calculate α : $\alpha = \tan^{-1}\left(\frac{x_2 - x_1}{y_1 - y_2}\right)$

The next task is to find the new angle θ' at which the particle reflects off of the tangent line:

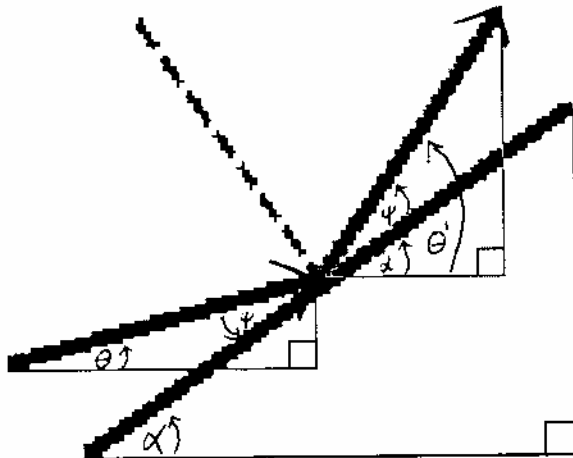


Figure 2 - The original direction is denoted with θ . The angle of the line tangent to the particle off of which it reflects is denoted α . The angle of reflection is ψ . The new angle of direction, as stored by the computer, is $\theta' = \alpha + \psi$.

We find the reflection angle ψ to be $(90^\circ - \theta) - (90^\circ - \alpha) = \alpha - \theta$. Therefore, $\theta' = 2\alpha - \theta$.

A similar procedure using a negative tangent slope reveals that $\theta' = -2\alpha - \theta$. When the computer finds α for a negative tangent slope, however, it finds $-\alpha$. So the general equation for the new direction becomes $\theta' = 2\alpha - \theta$.

iii. Shifting the x and y values on reflections

One of the problems I ran into in writing this program was that the time increments were not infinitesimal, and so a particle could move too far and overlap another particle. Its reflection would not then be sufficient to move away from the other particle, and so the computer would see them colliding again, and they would get stuck into the other particle. A similar thing happened with the walls—one particle would pass a wall completely and bounce back and forth on the other side of the wall. To solve this problem, the position of the particle had to be corrected for having passed an impassible boundary.

The walls were simple to analyze—when the particle passed one of the walls, its position was shifted back twice the amount by which it had passed the wall. These algorithms are found under note 2 in the program (lines 53 to 70).

The particle collisions were more difficult:

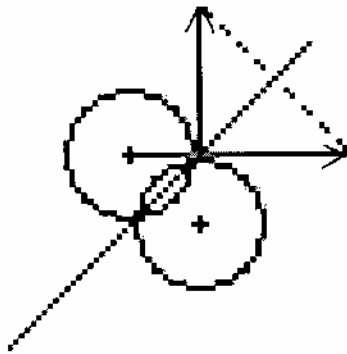


Figure 3 – Here two particles have overshot their paths. Their positions must be corrected to compensate.


A geometrical analysis reveals that the particles will have overshot their path by a total distance of $2r - d$, where r is the radius of a particle and d is the distance between the particles. (Each particle has moved $r - d/2$ too far.) The particles should have collided when $d = 2r$. Using geometrical analysis, we find that the x position of the particle must be

shifted by an amount $\Delta x = (2r - d) \sin(\theta - \alpha) \sin(\alpha)$. The y position must be shifted by $\Delta y = (2r - d) \sin(\theta - \alpha) \cos(\alpha)$. Further analysis shows that in the computer we must add the x shift and subtract the y shift. The calculations of θ and α compensate for the different directions that particles have overshot. These shifts are found in lines 121 to 124 of the code.

B. Writing the Program

The program itself was simple enough, once all of the mathematical difficulties were resolved. Attached to this report is a printout of the code. The included disk contains the code, the classes, and a simple HTML document that will allow the code to be run on a web browser.

A few things should be noted about the program: as is stated in note 4, there are variables declared for the particle class that are not used. I included these to show that this program could be further developed to make use of real data. Velocities could be calculated knowing the temperature and mass of a particle. Particles with different temperatures could be used to show the transfer of heat and what happens when thermal equilibrium is reached. This code could be the basis of many other Java applets that could demonstrate principles of thermodynamics.



III. Conclusion

The program runs well, and demonstrates very well what would happen to the ordered state of particles. One quickly sees the particles begin to scatter and move in all directions, colliding with the other particles. The demonstration clearly shows that the natural tendency of our universe is to move into a more chaotic, high entropy state.

This project was a lot of fun to do, and helped me think more about entropy and the motion of particles in thermodynamics. The program that I wrote could be used to help explain the principle of entropy to others who may have had the same difficulty that I did.

Entropy is not a difficult concept to understand, but can be confusing. Chaotic, high entropy states may seem ordered to us because it is common to see things in high entropy states. After all, if the universe tends to increase in entropy, it would be unlikely to find ordered, low entropy states. They do exist, but as can be seen with this program, left alone they do not remain there for long.