Pool simulation

Michał Skrzyszowski

October 2024

0.1 Problem

Let a 4x2 rectangle, which will represent a pool table. On this table, in any place inside its edges we place a ball. Next, we shoot the ball in any given direction. We're interested in the path the ball takes.

We assume that the centre point of the table is the beginning ($P(0,0)$) of the cartesian coordinate system, that the shot strength is equal to the displacement of the ball on an infinite table (without edges), e.g. a ball shot with strength of 20 would travel 20 units. We also assume that the direction the ball would go in will be determined by a vector ($v[x,y]$). Upon reaching the edge of the table the ball bounces off it and loses 5% of its energy (e.g. if it was shot with strength of 22 and hit an edge after 2, it would have only 19 and not 20 units of energy left). We also assume that the table can be bigger than the base, given a parameter to the program ($m \geq 1$). Let $m = 3$, the table would be a 12x6 rectangle:

$$P = \{(x,y) \in \mathbf{R}^2 : -6 \leq x \leq 6, -3 \leq y \leq 3\}$$

To summarise, the task is to write a program which will take 6 arguments:

1. $m \geq 1$ - table size multiplier
2. $s > 0$ - shot strength
3. x - x coordinate of ball's initial placement
4. y - y coordinate of ball's initial placement
5. wx - x coordinate of ball's initial vector
6. wy - y coordinate of ball's initial vector

The program must print a drawing of the path the ball will take in console, if it travels from the initial point, with the given strength, initially in the direction described by the given vector.

0.2 Concept

My approach to solving this problem is based on fundamental concepts of analytical geometry in the Cartesian plane. It should be noted that the path taken by the ball after a strike is a line segment on a straight line. Knowing two points (the starting point and the point calculated using the starting point and a given vector), we can calculate the equation of this line using the following formula:

$$y - y_a = a(x - x_a)$$

where slope:

$$a = \frac{\Delta y}{\Delta x}$$

Since the table's edge is perpendicular to the coordinate system axes, the slope of the next line (in which the segment the ball will move after a bounce is contained) is the opposite of the slope of the previous line. Using transformed formulas derived from the line equation above:

$$y = a(x - x_a) + y_a$$

$$x = \frac{y - y_a}{a} + x_a$$

we can calculate from which edge the next bounce will occur, substituting either x or y accordingly, then checking if the calculated y or x lies within the bounds of the table calculated using:

$$P = \{(x, y) \in \mathbf{R}^2 : -2 * m \leq x \leq 2 * m, -1 * m \leq y \leq 1 * m\}$$

where m is the table size multiplier provided as a program parameter. By tracking the sign of the current vector coordinates, we know whether the bounce is from the upper or lower edge (for bounces off edges parallel to the X-axis) or from the right or left edge (for bounces off edges parallel to the Y-axis). After bouncing from a given edge, this sign is reversed. This can be represented by a boolean value in implementation.

Knowing two points, we can calculate the length of the segment between them using the formula:

$$|AB| = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$$

The calculated length should be subtracted from the current strength of the ball's strike after each bounce, considering a five percent loss of strength.

After each bounce calculation, we need to determine if the ball still has enough strength to cover the required distance. If not, we need to calculate where it will stop. Since we know the segment's length and slope, we can transform the above formulas in this way:

$$x_b^2 + (ax_b)^2 - 2x_a x_b - 2a^2 x_a x_b + (ax_a)^2 - |AB|^2 = 0$$

and

$$y_b = ax_b - ax_a + y_a$$

This quadratic equation will give us two valid points. One of them should be discarded based on the current sign of the relevant vector coordinate and the slope.

Each bounce, as well as the starting and ending points, is recorded in a list, which will later be used to display the ball's path in the terminal. To display each segment, Bresenham's algorithm is used.