C Programming - Final Project Report - 1308234

Project 1 - Restricted Three-Body Problem
Two bodies (m_a and m_b) are fixed at (-1,0) and (1,0) respectively. A free particle moves coplanar to them in 2D space using Cartesian coordinates.

I. Numerical Methods Used

Part 1 - Simulation Parameters: user defines mass of bodies *m_a* (-1,0) & *m_b* (1,0). As part of the extension to the project a boundary in *x* & *y* can be set and an integer number of particles can be run.

Part 2 - Particle Parameters: several values are initialised - time step dt=0.005, velocity components x_v & y_v, positions x & v for each step are stored in arrays of size [200000], acceleration a_x & a_y are also stored in arrays of size [200000]. Initial particle position x & y, initial particle velocity *x_v* & *y_v* are defined by the user.

Part 3 - Computation: first positions x[1] and y[1] are computed with the formula:

$$x_1 = x_0 + (x_v * dt)$$

I use a 'for' loop to calculate the particle's position for every iteration labeled n, stopping at n=100000. First, acceleration were computed using the formula:

$$a(r) = -m_a \frac{(r + r_p)}{|r + r_p|^3} - m\_b \frac{(r - r_p)}{|r - r_p|^3}$$

Where $r_p = (1,0)$. Using this result we compute the position using the equation:

$$r_{n+1} = 2r_n - r_{n-1} + a(r_n)dt^2$$

Where 'r_n' represents either x or y at an iteration n.

Part 4 - Collision Detection: We define a 'sensitivity' s=0.05, if a moving particle is detected within this radius from a fixed body then a collision is signaled and the program is stopped. We achieve this using a series of 'if' loops and break functions. If a particle strays outside the user-defined boundaries, a collision is signaled and the program stopped.

II. Results & Figures (All figures computed with MATLAB)

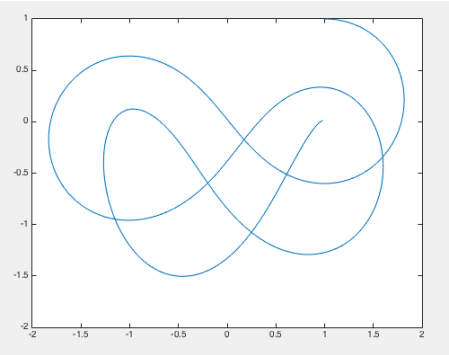

Fig. 1 - General Example of Particle Trajectory
$m_a = 1$; $m_b = 1$

$x = 1$; $y = 1$
$v_x = 1$; $v_y = 0$

Collision occurs at m_b.

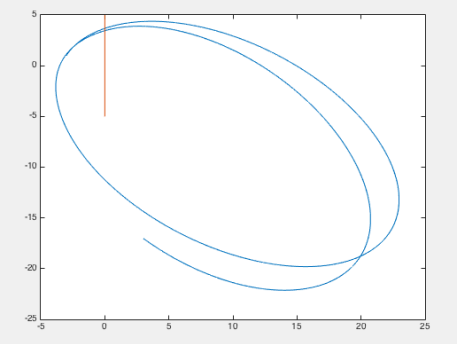

Fig. 2 - Two Particle Trajectories Simulated at Once
$m_a=1$ $m_b=1$

$x_1 = -3$; $y_1 = 1$
$v_{x1} = 0.5$; $v_{y1}=1$ This particle follows a stable orbit around m_b which is deviated by the attraction of the body m_a.

$x_2=0$; $y_2= -5$
$v_{x2}=0$; $v_{y2}=0$ This particle follows SHM

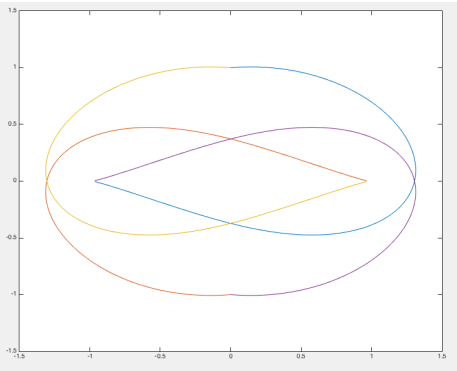

Fig. 3 - Symmetrical Trajectory of Four Particles
$m_a=1$ $m_b=1$

$x_1 = 0$; $y_1 = 1$ $x_3 = 0$; $y_3 = 1$
$v_{x1} = 1$; $v_{y1}= 0.1$ $v_{x3} = -1$; $v_{y3}= 0.1$

$x_2 = 0$; $y_2 = -1$ $x_4 = 0$; $y_4= -1$
$v_{x2} = 1$; $v_{y2} = -0.1$ $v_{x4} = -1$; $v_{y4}= -0.1$

Collisions occur at m_a for particles 1 & 2 and at m_b for particles 3 & 4.

III. Extension

As an extension I added collision detection with regard to both fixed bodies and user-defined boundaries, using 'if' loops with conditions related to position. In addition to this, I added a 'for' loop which enables the program to re-run its computational portion several times - it is therefore possible to run a user-defined number of particles at once (although these do not interact with each other - comparable to comets orbiting a binary star system), as visible in **Fig. 2** and **Fig. 3**.