Scientific Computing IIa - Planetary Motion Simulator

Tuomas Tynkkynen

October 30, 2011

1 Description

This program simulates the motion of arbitrary number of celestial objects using the Velocity Verlet[1] numerical integration method. The results of the simulation can then be visualized as an animated or static graph. Also, a separate program is included for an interactive 3d view of the simulation, created mostly for my own learning.

1.1 Compiling

Run `make` to compile and link the program. The main executable program `planets` should then be created.

To use the included animation script, both ImageMagick and Octave has to be installed as well.

SDL, OpenGL and GLU libraries and their development packages (i.e. header files) are required for building the 3d interactive viewer. Run make glplanets to compile it.

2 Usage

2.1 Running

Usage: `planets` *`infile`* `[`*`outfile`*`]`

The planets program reads the initial parameters for the simulation from the file *infile* and writes the results to *outfile*. If *outfile* is omitted, the results are written to standard output instead.

2.2 File formats

The first line of the input file should be of the following format:

$$n\ L\ \Delta t\ N_\omega$$

where n is the number of bodies; L the duration of the simulation in seconds; Δt the timestep of the simulation; N_ω the number of iteration steps written to the output file.

The following n lines should contain definitions of the planets in the following format:

$$m_i\ s_{x_i}\ s_{y_i}\ s_{z_i}\ v_{x_i}\ v_{y_i}\ v_{z_i}$$

where m_i is the mass of the body i in kilograms; s_{x_i}, s_{y_i}, s_{z_i} the x,y,z coordinates of the position of the body in meters; v_{x_i} v_{y_i} v_{z_i} the x,y,z coordinates of the velocity of the body in meters per second.

The corresponding output file will contain N_ω lines of the form

$$n\ t\ m_1\ s_{x_1}\ s_{y_1}\ s_{z_1}\ m_2\ s_{x_2}\ s_{y_2}\ s_{z_2}\ \cdots\ m_n\ s_{x_n}\ s_{y_n}\ s_{z_n}$$

where n is the number of bodies; t the time from the start of simulation in seconds; m_i the mass of body i in kilograms; s_{x_i} s_{y_i} s_{z_i} the x,y,z coordinates of the position of body i in meters.

2.3 Creating an picture or animation of the results

An utility script, `animate.sh` can be used to create an animation of the simulation results. To create an animation, run `./animate.sh` ***input-file*** `[-`***n***`]`. In addition to the output file, this will generate a GIF animation and a static PNG image of the simulation in the `output/` folder. Use the `-n` parameter to skip creating the animation, since it usually takes a while.

For example, running `./animate.sh testfiles/earthsun.in` creates the files `output/earthsun.out`, `output/earthsun.gif`, `output/earthsun.png`.

2.4 Interactive 3d visualization

The `glplanets` program can be used to interactively move around in space and see the planets moving. Run the `glplanets` program with the input file as argument.

2.4.1 Moving around

W,A,S,D Move the camera forward, left, backward or right. Hold Shift to move slower. Hold Alt to move faster.

Arrow keys, mouse Rotate the camera (look around).

1-9 Lock on to the n:th planet described in the input file. Your movement is now relative to the movement of the planet selected..

Tab Teleport on top of the planet you are locked on to.

0 Makes your movement free again.

2.4.2 Controlling the simulation

Esc Exit.

P Pause/unpause the simulation.

T Hide/show planet trails.

R Reset trails.

2.4.3 Changing simulation parameters

Hold Alt to change a setting 10 units at time. Hold Shift to change a setting 0.1 units at time.

F1, F2 Increase/decrease planet scaling by 1. Default: 1 (normal size).

F3, F4 Increase/decrease simulation speed by 1 Earth day/second. Default: 1.

2.4.4 An example run with glplanets

Run `./glplanets testfiles/earthfall.in`, which contains the Sun and the Earth, but with one tenth of the normal orbital speed. You'll start at the top of a red sphere, which is the Sun. The small green spot is Earth, far, far away. Move around a bit with the W,A,S,D keys and mouse and you'll see the light red, green and blue lines representing the positive x, y and z-axes.

Then press 2 and then Tab and you will be located near the green Earth. Position the camera such that you can see both the Earth and the Sun. You can use F1 and F2 to make them larger. Then press P. Earth fill start moving towards the Sun, and the camera will be moving along, but you can still move the camera relative to the Earth.

You can use Alt-F3 and F3 to make everything go faster. After some time you'll see Earth "bouncing" off the Sun in a hyperbelic motion. You can then press 0 to detach the camera from Earth and move around freely. Pressing Tab will teleport you back where you started. You can see the green dotted line which shows the path of Earth.

3 Analysis of results

3.1 Jupiter

Timesteps of 10^n s for $n \in [0; 7]$ were tried. $n = 0$ gave just nonsense, and $n = 1$ still seemed a bit too small, and $n = 7$ is too large. Between 10^2 seconds and 10^6 seconds the time step didn't see to matter too much, so 10^4 seconds was chosen.

With the time step of 10^4 seconds, at $t = 374080000$ s Jupiter was closest to it's starting point (calculated by an awk script). So the orbital period estimated by the simulation is about 11 years and 312 days, which is off only by around three days. Using the same time step, after the actual orbital period, the simulated Jupiter has moved about 2.93×10^6 km ahead. The Sun's orbital period seems to be about one half of Jupiter's period. It seems to "orbit" a "half-ellipse"-shaped path, with the semi-minor axis of 28×10^6 km and semi-major axis of 46×10^6 km.

3.2 Human timespans

Using a simulation with the Earth and the Sun, the Earth's orbital period (a.k.a a year) is about 365 days and 1.7 hours with a time step of 600 seconds. Using a time step of 60 seconds, the orbital period of Moon around Earth (i.e. a month) was approximately 26 days 18.3 hours.

3.3 Multiple bodies

A simple model of Sun, Earth and Moon stayed stable for at least few dozen Earth's orbital periods, but lost its stability after periods several magnitudes longer, though it is quite possible that too large time step was being used at that time so that the computation wouldn't take too long.

A stable triple star system was not able to be created.

References

[1] Wikipedia: Verlet integration – http://en.wikipedia.org/wiki/Verlet_integration#Velocity_Verlet