readme.txt (eng)

Introduction

This is a project for planetary simulation with FORTRAN 90.

The objective of the project is to simulate planetary motions according to the Newton's law of gravity.

The simulation program written in Fortran gets a input file where the data of objects (e.g. mass, initial positions, velocities etc.), time steps, total simulation length, how many data point you want to be written to the output file etc. are. Then the program produces an output file where each objects' positions are written.

I also have codes to produce plots from the output file, written in Python.

The project is my final course project, and it is in Finnish. Maybe later I will improve the program or the whole project, and maybe change the language into English too.

Description

In this project, I used Verlet algorithm (see https://en.wikipedia.org/wiki/Verlet_integration) to simulate the celestial objects. The algorithm is briefly:

$$x_{i+1} = x_i + v_i \Delta t + \frac{1}{2} a_i \Delta t^2 \text{ (similarly for } y_{i+1} = y_i + \ldots \text{ etc.)}$$
$$a_{i+1} = F(x_{i+1})/m$$
$$v_{i+1} = v_i + \frac{1}{2}(a_i + a_{i+1}) \Delta t \,.$$

, where (i+1) indexes indicate the next step and i the current step.

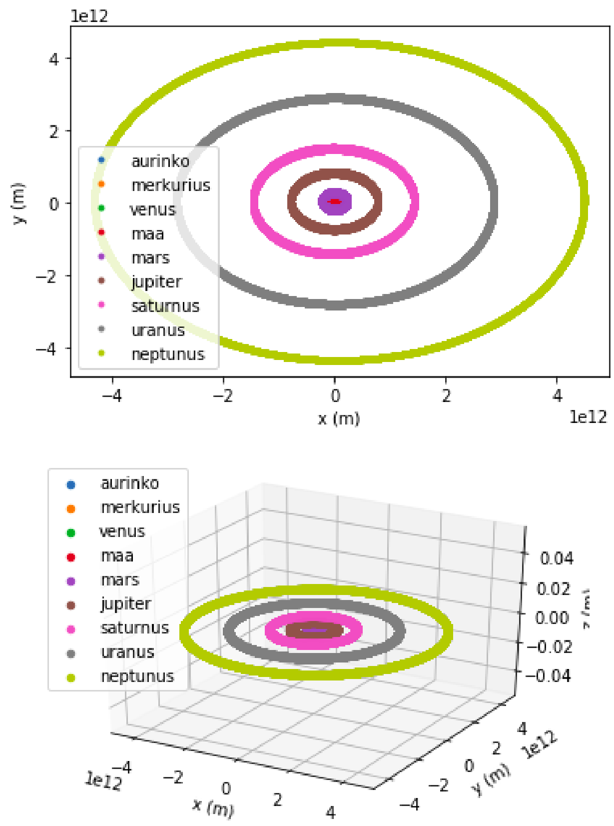
Acceleration is calculated using the universal law of gravity:

$$\vec{a}_{12} = \frac{\vec{F}_{12}}{m} = G\frac{M}{|\vec{r}_2 - \vec{r}_1|^2}\hat{r}_{12} = G\frac{M}{|\vec{r}_2 - \vec{r}_1|^3}(\vec{r}_2 - \vec{r}_1) \tag{1}$$

I made the following assumptions:

- Objects to be simulated are point-like (meaning they do not have volumes)
- The only interactions is gravity.
- Collisions are not considered. Planets will travel past each other.

Visuals



Plots of the Solar System using Matplotlib library in Python.

```
-------------------------------------------------

Number of objects: 9
Time since the beginning: 9.35 s
Steps past: 148669
Datapoints in the output file: 2

The positional data of objects:
1.  (0.1405E+10 , 0.8394E+11 , 0.000)
2.  (0.1741E+11 , 0.1370E+12 , 0.000)
3.  (-0.1454E+11 , -0.2350E+11 , 0.000)
4.  (-0.2325E+11 , -0.6427E+11 , 0.000)
5.  (0.1916E+12 , 0.2089E+12 , 0.000)
6.  (0.1721E+12 , -0.6794E+12 , 0.000)
7.  (-0.1438E+13 , 0.3663E+12 , 0.000)
8.  (0.2848E+13 , -0.2785E+12 , 0.000)
9.  (0.4408E+13 , 0.9568E+12 , 0.000)
```

Expected printout to the terminal when you have runned the Fortran source codes.

Requirements

- GNU Fortran compiler (or possibly other Fortran compilers.)
- Jupyter notebook for running the code for plots.
- an input file (possibly .txt file) where data about objects to be simulated are stored. (Check the format below)

Installation

- Make sure source codes are in the same file.
- Make sure input file are also in the same file.

The following codes can be then copied to the terminal:

1. **Compilation:**

```
gfortran -c parameters.f90  func_type.f90 readingfile.f90 updatealgorithm.f90 writetofile.f90 printti.f90 main.f90
```

2. **Linking:**

```
gfortran func_type.o parameters.o readingfile.o updatealgorithm.o main.o writetofile.o printti.o
```

3. **Running:**

```
./a.out [input_filename] [output_filename]
```

- There is a attached input file of the Solar System. You can use it.

Input file format

Example input file when there are two objects to be simulated: (exclamation mark is explation about the meaning of the numbers)

```
2                     !how many objects are there?
1 2 3 4 5 6 7         !1. object: mass/position: x,y,z / speed: vx, vy, vz
8 9 10 11 12 13 14    !2. object: mass/position: x,y,z / speed: vx, vy, vz
5 500000 2 1           !length of a time step in seconds/ total simulation time in seconds
                       !/between how many steps positional information to be printed to the screen (terminal)
                       !/between how many steps positional information to be read into the specified output file
```

Contributing

Pull requests are welcome. For major changes, please open an issue first to discuss what you would like to change.

License

MIT