FYS3150 Prosjekt 3

tobias

Oktober 2017

1 Introduction

2 Method/Approach

2.1 The Earth-Sun system

As mentioned in the introduction, this project is all about developing a code for simulating the solar system, and we will start small with just the Earth-Sun system. Later we will expand our program to include all the planets in the solar system. To begin with we make two assumptions for this case: First we will assume that the Earth's orbit around the Sun is completely circular, and secondly we will assume that the Sun is completely at rest in the origin of our system. The second assumption we can do because the mass of the sun is much larger than that of the earth. It should also be mentioned that throughout this whole project we will measure time in years and lengths in Astronomical units (AU), all masses will be measured in solar masses and that the orbits around the sun is thought of as co-planar in the xy-plane.

The only force at work in our two-body system is the force of gravity, given by Newton's law of gravity. This means that the force on the earth from the sun will be given as:

$$\vec{F_G} = -\frac{Gm_p m_s}{r^2}\vec{e_r}$$

wich can be written in a more practical form as:

$$\vec{F_G} = -\frac{Gm_p m_s}{r^3}\vec{r}$$

where G is the universal gravitational constant, m_p is the mass of the planet (in this case the Earth, later called m_E), m_s is the mass of the sun and $\vec{r}$ is a vector pointing from the sun towards the earth. Of course, the earth will also exert a force on the sun in reality, but we ignore this here because we have assumed that the sun will be at rest in our system.

We choose the initial position of the earth to be on the x-axis, and we know that it is at a distance of 1 AU away. This gives the initial position of the earth: x = 1.
To determine the initial velocity v_E of the earth we take advantage of the circular orbit, wich means that the earth will have a sentripetal acceleration $a_s = \frac{v^2}{r}$.

Using Newton's second law $\sum F = ma_s$, this gives us an equation that we can solve for v_E:

$$G\frac{m_s m_E}{r^2} = m_E\frac{v_E^2}{r} \Leftrightarrow v_E^2 = G\frac{m_s}{r}$$

In this case we have $m_s = r = 1$, so that $v_E = \sqrt{G} = 2\pi$, where we have used that $G = 4\pi^2$ in astronomical units.

Now that we have the earth's initial conditions, it's time to compute the earth's orbit. This motion is governed by a set of coupled differential equations that codify Newton's law of motion due to the gravitational force. We will here use different methods for solving these equations (both the Euler method and the velocity-Verlet method). Using Newtons second law we get the following equations:

$$\frac{d^2x}{dt^2} = \frac{F_{G,x}}{m_E}$$

and

$$\frac{d^2y}{dt^2} = \frac{F_{G,y}}{m_E}$$

After discretizing the above differential equations they will be ready for solving by our numerical algorithm of choice. First we take a look at the Euler method, wich is given by:

$$v_{i+1} = v_i + a * dt$$

$$x_{i+1} = x_i + v_i * dt$$

where i indicates what timestep we are on, and our timestep h is given by $h = \frac{Stop-time}{N}$, where N is the number of timesteps.
Although the Euler method does work, it has some disadvantages. A better choice is the Velocity-Verlet method, given by:

$$x_{i+1} = x_i + hv_i + \frac{h^2}{2}a_i$$

$$v_{i+1} = v_i + \frac{h}{2}(a_{i+1} + a_i)$$

Note that the term a_{i+1} depends depends on the position x_{i+1}. This means that we need to calculate the position at the updated time t+1 before computing the next velocity.

As a test to see if our program is working correctly in the case with circular motion, we will calculate both the total energy and the angular momentum L of the system. The total energy is simply calculated as the sum of kinetic and potential energy, and the angular momentum is calculated as $\vec{L} = \vec{r} \times \vec{p}$. Both of these quantities should be conserved. The total energy should be conserved because the only force working is the gravitational force, wich is a conservative force.
To explain why the angular momentum is conserved, we will make use of the fact that the torque of the earth's orbit around the sun is $\vec{\tau} = \frac{d\vec{L}}{dt}$. The torque can also be expressed as: $\vec{\tau} = \vec{r} \times \vec{F_G}$. As we know, $\vec{r}$ and $\vec{F_G}$ will always be parallell, wich makes the cross product, and therefore the torque, equal to zero. When the torque is equal to zero we have $\frac{d\vec{L}}{dt} = 0$, wich shows that angular momentum is conserved.

2.2 Object orienting

As mentioned in the introduction, we will in this project write an object oriented code. For a project like this, where we will be doing a lot of the same calculations for several different planets, object orienting is very useful. We therefore divide our program into several classes:
We have one class called SolarSystem wich both creates and keep track of our bodies (planets) with the necessary initial conditions. In addition to this, the solarsystem class also calculates the energy and angular momentum of our system, aswell as the center of mass and the motion of the sun (wich isn't important for the earth-sun system, but it becomes important when we start adding more planets to our model).
We also have an Euler class wich implements the Euler method and a velocity verlet class wich implements the velocity-verlet method.
Our class vec3 contains all the basic functionality and operations for vectors. The class celestialbody declares/initializes the necessary vectors and variables to contain important information about our planets, mainly position, velocity and mass.

2.3 Adding the rest of the planets