FYS3150
Project 3 - The Verlet Underground

Hugounet, Antoine & Villeneuve, Ethel

October 2017
University of Oslo
`https://github.com/kryzar/Perseids.git`

Abstract

The aim of this project is to create a simulation of the Solar System. We will consider three cases.

1. The first one will be the simplest, an Earth-Sun system. With it, we can test our program and compare the Euler's and Verlet's method as we know that the orbit of the Earth is supposed to be circular around the Sun, which will be set as the center-of-mass.
2. In the second case, we will add Jupiter. With this model, we can compare the influence of taking into account the real center-of-mass instead of the Sun as center-of-mass.
3. Finally, the third case will be the complete Solar System (without moons, only the eight planets). After checking that our program is stable and viable, we can simulate the Solar System around the years and centuries.

Moreover, we will use the program to find the escape value of the Earth from the Sun and to study the Mercury's perihelion precession with and without a relativistic correction to Newton's law. These are two concrete ways to use this simulation program, but it basically works for many phenomena.

As we expected, Verlet's method is way more precise than Euler's method. That is why we will use Euler's one only for the two-bodies model. In contrast, changing the center-of-mass from the Sun to the real one does not implies big changes as it is really close to the Sun compared to the other distances.

Contents

Introduction

This project is a good initiation to scientific object-oriented programing. The concept of class allowed us to create planets and make them interact very easily with a clear syntax instead of creating a ton of variables and a ton of functions. Considering this and the simplicity of Verlet's algorithm as a simple simplification - yet reliable - of elliptical orbits, we could enjoy with very different scenario like add three Suns, forty-five Earth, and see how those models interact together. We will first study the theory of our models, then see how we implemented our program, and compare our results with the theory.

Chapter 1

Theory

In a first place, we will begin with an Earth-Sun system with the Earth orbiting around the Sun. We will test our simulation on this simple case and add the other planets afterwards.

1.1 Earth-Sun system

1.1.1 Physical conditions of the system

The only force applied to this system is the gravity. According to the Newton's law, we have

$$F_G = \frac{GM_\odot M_\oplus}{r^2} \tag{1}$$

with F_G the norm of the gravitational force, G the gravitational constant ($G = 6.674 \times 10^{-11}$N.m^2.kg^{-2}), $M_\odot$ the mass of the Sun, $M_\oplus$ the mass of the Earth and r the distance between the Earth and the Sun.
We will neglect the motion of the Sun here as the mass of the Sun is much larger than the mass of the Earth ($M_\odot = 2 \times 10^{30}$kg against $M_\oplus = 6 \times 10^{24}$kg). We want to establish the motion of the Earth around the Sun. Moreover, we will assume that the orbit of the Earth around the Sun is coplanar in the xy-plane.

The Newton's second law of motion states that the total force applied to any system in a Galilean referential is $\overrightarrow{F} = M\overrightarrow{a}$ with M the mass of the body concerned and $\overrightarrow{a}$ the acceleration. Applied to our case, we have $\overrightarrow{F_G} = M_\oplus \overrightarrow{a}$. Since a is the second derivative of the position :

$$F_{G,x} = M_\oplus \frac{d^2x}{dt^2}$$
$$F_{G,y} = M_\oplus \frac{d^2y}{dt^2}$$

or

$$\frac{d^2x}{dt^2} = \frac{F_{G,x}}{M_\oplus} \tag{2}$$
$$\frac{d^2y}{dt^2} = \frac{F_{G,y}}{M_\oplus} \tag{3}$$

with $F_{G,x}$ and $F_{G,y}$ the components of the gravitational force.

Remark 1.1. *We will not use the SI units but the Astronomical units (AU) for the distances (with $1AU$=average Earth-Sun distance $= 1.5 \times 10^{11} m$), kg for masses and years for time units. Thus the initial position of the Earth will be $x_\oplus = 1,\ y_\oplus = 0$ with the Sun the origin ($x_\odot = 0,\ y_\odot = 0$).*

To simplify a little bit, we will assume that the Earth's orbit is circular around the Sun. So we can write :

$$F_G = \frac{M_\oplus v^2}{r} = \frac{GM_\odot M_\oplus}{r^2} \tag{4}$$

with v the velocity of Earth. From here we have

$$v^2 r = GM_\odot = 4\pi^2 \mathrm{AU}^3/\mathrm{yr}^2 \tag{5}$$

1.1.2 Escape velocity

In the case where only the gravitational force is applied, the mechanical energy $E_m = E_c + E_p$ with E_c the kinetic energy and E_p the potential energy is conserved (the gravitational force is conservative). For a planet at a distance of 1 AU from the Sun (let's take the Earth as an example), the kinetic energy is given by $E_c = \frac{1}{2} M_\oplus \times v^2$ and the potential energy by $E_p = -\frac{GM_\odot M_\oplus}{r}$ with $r = 1$ AU.
We want this planet to escape from the gravitational attraction of the Sun on it. The escape velocity will be the minimal speed v_e at which the planet escapes from the gravitational influence of the Sun. We consider two positions of the planet : p_i the initial position when $r_i = 1$AU and p_f the final position when the planet has escaped and is at an infinite distance from the Sun $r_f = \infty$.

Let's consider the mechanical energy at the final position with $E_{m_f} = E_{c_f} + E_{p_f}$. $E_{c_f} = 0$ for $v_f = 0$ since at a point at infinity, the Sun still has an attraction on the Earth. Consequently its velocity slightly deceases with r increasing until it is zero at an infinite distance from the Sun. $E_{p_f} = 0$ since $r_f = \infty$ and $E_{m_f} = 0$.
So, with the planet at 1 AU from the Sun at its initial position, of which velocity is the escape velocity v_e, in the case where there is only the gravitational force applied to the system planet-Sun, we have :

$$E_{c_i} + E_{p_i} = \frac{1}{2} M_\oplus \times v_e^2 - \frac{GM_\odot M_\oplus}{r_i} = E_{c_f} + E_{p_f} = 0$$

according to the law of conservation of energy. From this we can write

$$\frac{1}{2} M_\oplus \times v_e^2 - \frac{GM_\odot M_\oplus}{r_i} = 0$$
$$\Rightarrow v_e = \sqrt{\frac{2GM_\odot}{r_i}}$$
$$\Rightarrow v_e = \sqrt{2GM_\odot}$$

which only depends on the mass of the Sun.

$$\begin{aligned} v_e &= \sqrt{2 \times 4\pi^2}, \quad \text{from (5)} \\ v_e &= 2\sqrt{2}\pi \ \text{AU/yr} \end{aligned}$$

Then, for a velocity $v \geq 2\sqrt{2}\pi$, the Earth would escape from the Sun's gravitational attraction.

1.2 Three-body problem

1.2.1 The Sun as the center-of-mass

We add Jupiter to the previous system and place it on the same $x - axis$ that the Earth is on with an initial velocity on the $y - axis$. In the previous case, the Earth's motion was a stable circular orbit in time. Adding Jupiter, it will inevitably be disturbed. We can write the gravitational between the Earth and Jupiter as

$$F_{\oplus - ♃} = \frac{G M_{♃} M_{\oplus}}{r^2_{\oplus - ♃}}$$

with $M_{♃}$ the mass of Jupiter and $r_{\oplus - ♃}$ the distance between the Earth and Jupiter.
The algorithm used for this case will be exactly the same as the previous case except that the total force applied on the Earth is the sum of the Force exerted by the Sun and Jupiter. We will see in section 1.4 how to discretize and implement the algorithm.

1.2.2 Real center-of-mass

To get closer to the reality, we now do not consider the Sun as a static body anymore ; each of three bodies are in motion and considered as planets. The center-of-mass will not be the Sun but the real one. Let's compute the new coordinates of the celestial bodies.

Basic equation

$$\begin{cases} x_c = \dfrac{1}{M_{tot}} \displaystyle\sum_i M_i x_i \\ y_c = \dfrac{1}{M_{tot}} \displaystyle\sum_i M_i y_i \end{cases} \tag{6}$$

Considering the system Earth-Jupiter-Sun :

$$\begin{aligned} M_{tot} &= M_{\oplus} + M_{♃} + M_{\odot} \\ &= 6 \times 10^{24} + 1.9 \times 10^{27} + 2 \times 10^{30} \\ &= 2.00191 \times 10^{30} \text{kg} \end{aligned}$$

$$\begin{cases} x_c = \dfrac{1}{M_{tot}}[M_\oplus x_{\oplus_i} + M_{♃} x_{♃_i} + M_\odot x_{\odot_i}] \\ y_c = \dfrac{1}{M_{tot}}[M_\oplus y_{\oplus_i} + M_{♃} y_{♃_i} + M_\odot y_{\odot_i}] \end{cases}$$

We fix the initial positions with respect to the new center of mass.

Calculations

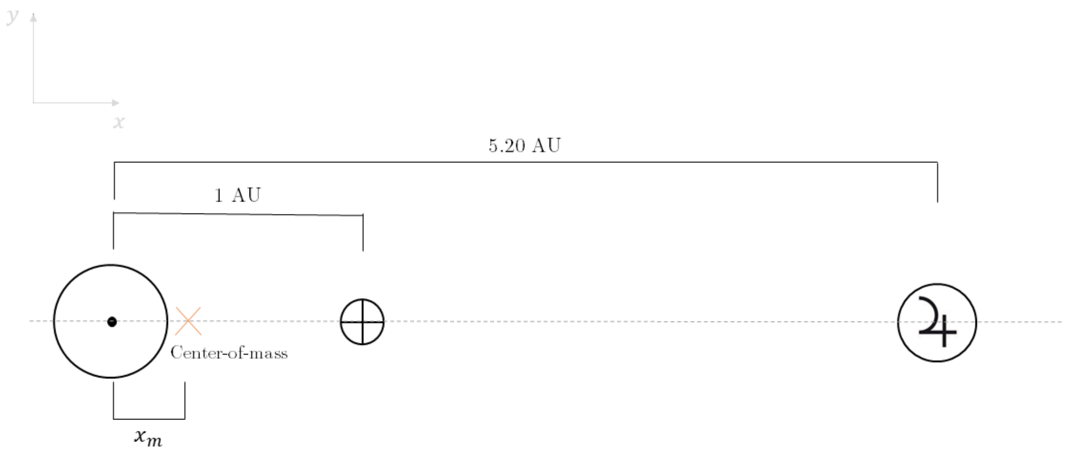


Figure 1.1: Representation of the initial position of our three-bodies system.

So we look for x_m the x-component of the position of the Sun such that $x_c = y_c = 0$.

$$\begin{cases} x_c = \dfrac{1}{M_{tot}}[(2\times 10^{24})\times(1+x_m) + (1.9\times 10^{27})\times(5.20+x_m) + (2\times 10^{30})\times x_m] = 0 \\ y_c = 0 \end{cases}$$

$$\implies \begin{cases} x_m = -4.936\times 10^{-3} \\ y_m = 0 \end{cases} \tag{7}$$

These are the coordinates of the Sun when the center-of-mass of the system is set as the origin. Let's now compute the Sun's initial velocity such that the total momentum of the three-body system equals 0.

$$\overrightarrow{p_{tot}} = \sum_k M_k \overrightarrow{k} = \overrightarrow{0} \tag{8}$$

$$\implies \begin{pmatrix} 0 \\ 0 \end{pmatrix} = M_\oplus \begin{pmatrix} v_{\oplus_x} \\ v_{\oplus_y} \end{pmatrix} + M_{♃} \begin{pmatrix} v_{♃_x} \\ v_{♃_y} \end{pmatrix} + M_\odot \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

$$\implies \begin{cases} M_\oplus v_{\oplus_x} + M_{♃} v_{♃_x} + M_\odot v_x = 0 \\ M_\oplus v_{\oplus_y} + M_{♃} v_{♃_y} + M_\odot v_y = 0 \end{cases}$$

We choose the initial velocities of the Earth and Jupiter so that their orbit will be circular. If the orbit of the Earth is circular around the center-of-mass, $v_{\oplus} = 2\pi \times (1 + x_m)$ AU/yr, because the distance travelled in one Earth-year is $2\pi \times r$ with $r = 1 + x_m$ AU.

Similarly $v_{\oplus} = 6.25217$ AU/yr. The velocity is constant so the initial velocity will be $v_{\oplus_i} = 6.25217$ AU/yr. In the initial configuration represented above, the initial velocity is oriented in the direction of y. So, $v_{\oplus_i} = (0, 6.25217)$.
Jupiter has an orbital period of 11.862 yr. The distance travelled by Jupiter in this period is $2\pi \times r = 2\pi \times (5.20 + x_m) = 32.6415$ AU. So $v_{♃_i} = \frac{2\pi \times r}{T} = \frac{2\pi \times (5.20 + x_m)}{11.862} = 2.75177$ AU/yr. Similarly, the initial velocity can be expressed as $v_{♃_i} = (0, 2.75177)$.

$$\begin{cases} (6 \times 10^{24}) \times 0 + (1.9 \times 10^{27}) \times 0 + (2 \times 10^{30}) \times v_x = 0 \\ (6 \times 10^{24}) \times 6.25217 + (1.9 \times 10^{27}) \times 2.75177 + (2 \times 10^{30}) \times v_y = 0 \end{cases}$$

.

$$\implies \begin{cases} v_x = 0 \\ v_y = -2.63294 \times 10^{-3} \end{cases} \tag{9}$$

Therefore, our initial vectors will be :

Body	Position	Velocity
Earth	$\begin{pmatrix} 0.995064 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 6.25217 \end{pmatrix}$
Jupiter	$\begin{pmatrix} 5.19506 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2.75177 \end{pmatrix}$
Sun	$\begin{pmatrix} -4.936 \times 10^{-3} \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ -2.63294 \times 10^{-3} \end{pmatrix}$

1.3 The complete Solar system

As the program had been made so that we can add as many planets as we want, modelizing the complete Solar System is not more complicated than having only the system Earth-Jupiter-Sun. We will take, in that case, the center-of-mass of the complete Solar System as origin. Therefore the Sun has a motion.

1.3.1 Discretization

The discretization is a crucial step to make the equations "understandable" by our program. From Newton's law of motion, the acceleration of a given body can be written as $a = \frac{F_G}{M}$, with M the mass of the body. a_i is the acceleration at the time t_i, and a_{i+1} is the very next discretized value of the acceleration at $t_{i+1} = t_i + h$.

$$a_i = \frac{\sum_k \frac{GMM_{k_N}}{r_{i_k}^2}}{M}$$

with k the number of bodies involved in the system.

$$a_i = \sum_k \frac{GM_{k_N}}{r_{i_k}^2}$$

From (4), we have $GM_\odot = 4\pi^2 \mathrm{AU}^3/\mathrm{yr}^2$ and as $M_\odot = 1$, $G = 4\pi^2 \mathrm{AU}^3/\mathrm{yr}^2$. So,

$$\begin{aligned} a_i &= \sum_k \frac{4\pi^2 M_{k_N}}{r_{i_k}^2} \\ a_i &= \sum_k \frac{4\pi^2 M_{k_N}(x_i - x_{i_k})}{r_{i_k}^3} \end{aligned} \tag{10}$$

Remark 1.2. *To implement the accelerations, we normalize the masses with respect to the mass of the Sun. Thus, $M_\odot = 1$, $M_{k_N} = \frac{M_k}{M_\odot}$ is the normalized mass for each body k. We use Astronomical Units (AU) as the distance unit and the year (yr) as the time unit.*

1.3.2 Euler's method

With Euler's method, we simply compute the derivation formula for a supposed small constant time-step h : $a_i = \dfrac{dv_i}{dt} = \dfrac{v_{i+1} - v_i}{h}$. With $h = \dfrac{b-a}{N}$ and N the number of time-steps, the bigger N, the smaller h, and the more accurate Euler's approximation is.

Remark 1.3. *It is easier to see this formula as the formula of the derivative without its limit, but Euler's demonstrated it in terms of Taylor's expansion.*

$$\begin{aligned} \frac{v_{i+1} - v_i}{h} &= \sum_k \frac{4\pi^2 M_k (x_i - x_{i_k})}{r_{i_k}^3} \\ \Longrightarrow v_{i+1} &= h \sum_k \frac{4\pi^2 M_k (x_i - x_{i_k})}{r_{i_k}^3} + v_i \end{aligned} \tag{11}$$

The velocity can also be discretized as the derivative of the position

$$v_i = \frac{x_{i+1} - x_i}{h}$$

$$\Longrightarrow x_{i+1} = h v_i + x_i \tag{12}$$

The initial position and velocity of the body are obtained with the data from the NASA website.

1.3.3 Verlet's method

Verlet's method is almost the same as the Euler's one in the sense that it is based on Taylor's expansion, but goes one step further. We consider two instants $t_i + h$ as $t_i - h$:

$$\begin{cases} x_{i+1} = x(t_i + h) = x(t_i) + h \dfrac{dx}{dt}(t_i) + \dfrac{h^2}{2!} \dfrac{d^2x}{dt^2}(t_i) + \dfrac{h^3}{3!} \dfrac{d^3x}{dt^3}(t_i) + \mathcal{O}(h^4) \\ x_{i-1} = x(t_i - h) = x(t_i) - h \dfrac{dx}{dt}(t_i) + \dfrac{h^2}{2!} \dfrac{d^2x}{dt^2}(t_i) - \dfrac{h^3}{3!} \dfrac{d^3x}{dt^3}(t_i) + \mathcal{O}(h^4) \end{cases}$$

We sum up the two lines to have

$$x_{i+1} + x_{i-1} = 2x_i + h^2 \frac{d^2 x_i}{dt^2} + \mathcal{O}(h^4)$$
$$\Rightarrow x_{i+1} = 2x_i - x_{i-1} + h^2 \frac{d^2 x_i}{dt^2} + \mathcal{O}(h^4)$$

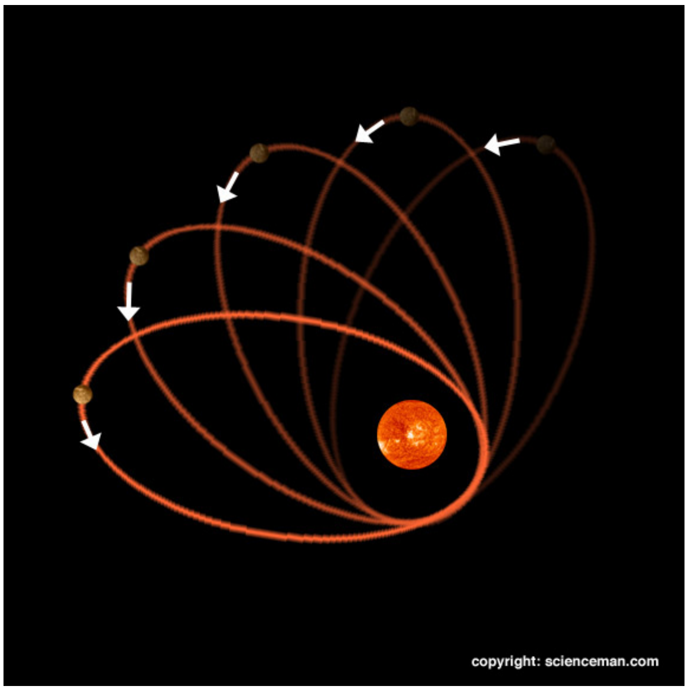
Another way to use the Verlet's method is with :

$$\begin{cases} x_{i+1} = x(t_i + h) = x(t_i) + h\dfrac{dx}{dt}(t_i) + \dfrac{1}{2}h^2 \dfrac{d^2x}{dt^2}(t_i) = x(t_i) + hv(t_i) + \dfrac{1}{2}h^2 a(t_i) \\ v_{i+1} = v(t_i + h) = v(t_i) + \dfrac{1}{2}h\left(\dfrac{dv}{dt}(t_i) + \dfrac{dv}{dt}(t_i + h)\right) = v(t_i) + \dfrac{1}{2}h\left(a(t_i) + a(t_{i+1})\right) \end{cases}$$

where the velocity is explicitly expressed.

1.4 The special case of Mercury

Mercury being the closest planet to the Sun, the influence of the Sun on it is special ; furthermore it is a very light planet, roughly 3.3×10^{23} kg which makes its orbit elliptical and not circular. The point where Mercury is the closest to the Sun is called perihelion. At the initial perihelion, $x_{☿_p} = 0.3075$ [AU], $y_{☿_p} = 0$ and $v_{☿_p} = 12.44$ [AU/yr]. With time, x will decrease whereas y will increase, showing that Mercury's orbit is moving around a circle centered around the Sun. The perihelion precession is a phenomenum that describe the perihelion shift, orbit after orbit.



Mercury's orbit rotating around the Sun.

We observe that by substracting the classical effects (basically, the gravitational attraction between Mercury and the other planets), we still have a perihelion shift with an angle of 43" per century. We need to add the relativistic correction to the Newtonian gravitational force. Thus $F_G = \frac{GM_\odot M_☿}{r^2}$, with $M_☿$ the mass of Mercury and r the distance between Mercury and the Sun ($r = 0.39$AU), becomes

$$F_G = \frac{GM_\odot M_☿}{r^2}\left(1 + \frac{3l^2}{r^2c^2}\right) \tag{13}$$

with $l = |\vec{r} \times \vec{v}|$ the norm of Mercury's orbital angular momentum per mass unit and c the speed of light in vacuum.

This relativistic phenomenum is of course applied to every planets of the Solar System but it is more important in the case of Mercury, and even here we need roughly one century to detect a small perturbation. The difference between the expected value in Newtonian mechanics and the observed value has been explained with the general theory of relativity. The spacetime curvature near the Sun is more important where Mercury is and it is more affected by this phenomenum. This test for giving the general relativity an experimental proof is quite similar in its philosophy to the tests evolving gravitational lenses.

Chapter 2

Implementation

You will find how to run the program in the README.md file from the repository.

The program has been made such as its behavior is exactly the same no matter the number of planets and no matter the center of mass. We created one `planet` class of which role is to initialize planets and one `solver` class of which role is to calculate the interaction between the planets. When one adds a planet to a system simulation, `solver` automatically updates the total mass of the system, calculates the initial accelerations (depending on the relativistic correction or not) and is therefore in capability to compute the center of mass, the kinetic, potential and total energies with associated methods. The two main algorithms to compute positions and velocities around time are Verlet's method and Euler's method.

2.1 Euler's and Verlet's methods

2.1.1 Calculation of the acceleration

The private method `solver::_acceleration` computes the acceleration of a given planet (the planet of index p given as an argument) of the system using (10). It goes through every planet, copies the position and computes the formula. Contrary to the exemples seen in the lectures, the velocity and positions of a planet are stored in vectors of the standard libraries instead of using four distinct doubles. This was made for readability and better connexion with the mathematic reality. However it is a shame that the `std::vector` class does not support basic arithmetic operations, this would have made the code way cleaner[1].

You will notice that this function requires a boolean argument named `relativity`. It is there to tell the function whether it has to compute the relativistic correction or not. The use of the boolean argument allows us to build only one method for both classical and relativistic formulae, instead of two distinct methods. The less code, the less errors, especially since having two similar functions often implies copypasta.

[1]We could have built our own vector class here. It is not very hard to do but it was not the point of this project.

2.1.2 Euler's and Verlet's algorithms

The process of Euler's and Verlet's algorithm are to go through every time-steps of the time period passed as an argument, and inside each time-step, to go through very planet of the system. We will so use two loops, the loop for the planet being inside the loop for the time-steps. For the non-relativistic case, there are 365 time-steps per year, and $360 \times 3600 \times 7 = 9072000$ for the relativistic case (to give a resolution of one arcsecond for the special case of Mercury). For Euler, the program computes (11) and (12), ie the basic derivative formula approximations. For Verlet, the program computes

$$\begin{cases} x_{i+1} = x(t_i) + hv(t_i) + \frac{1}{2}h^2 a(t_i) \\ v_{i+1} = v(t_i) + \frac{1}{2}h\left(a(t_i) + a(t_{i+1})\right) \end{cases}$$

Now suppose we are computing all the planets inside a given run of the time-steps loop. Each planet requires the positions of all the planets at the previous time-step and its own previous position and velocity. For Verlet, it evens needs the next acceleration. The key is to create private vectors of vectors (a vector containing the position vectors of all the planets at a given time t_i) which contains the previous position, velocity and acceleration of each planet. Since the program needs those vales at t_i and that it modifies those values at the same time by replacing them with the new one, it is crucial to save the values of t_i to be sure that the values calculated for $t_i + 1$ are the correct ones.

Euler is straight-forward, but Verlet requires that we compute the next acceleration. Therefore we first compute the next velocity like for Euler, then compute the next acceleration and store it in a vector of vectors, then computes the position and updates the planets. The program can use those vectors to compute the formula, and meanwhile modify the positions and velocities of each planet without impacting the others. The vector of vectors is then reused as the "next-previous" acceleration.

2.2 Validity

2.2.1 Unit-tests

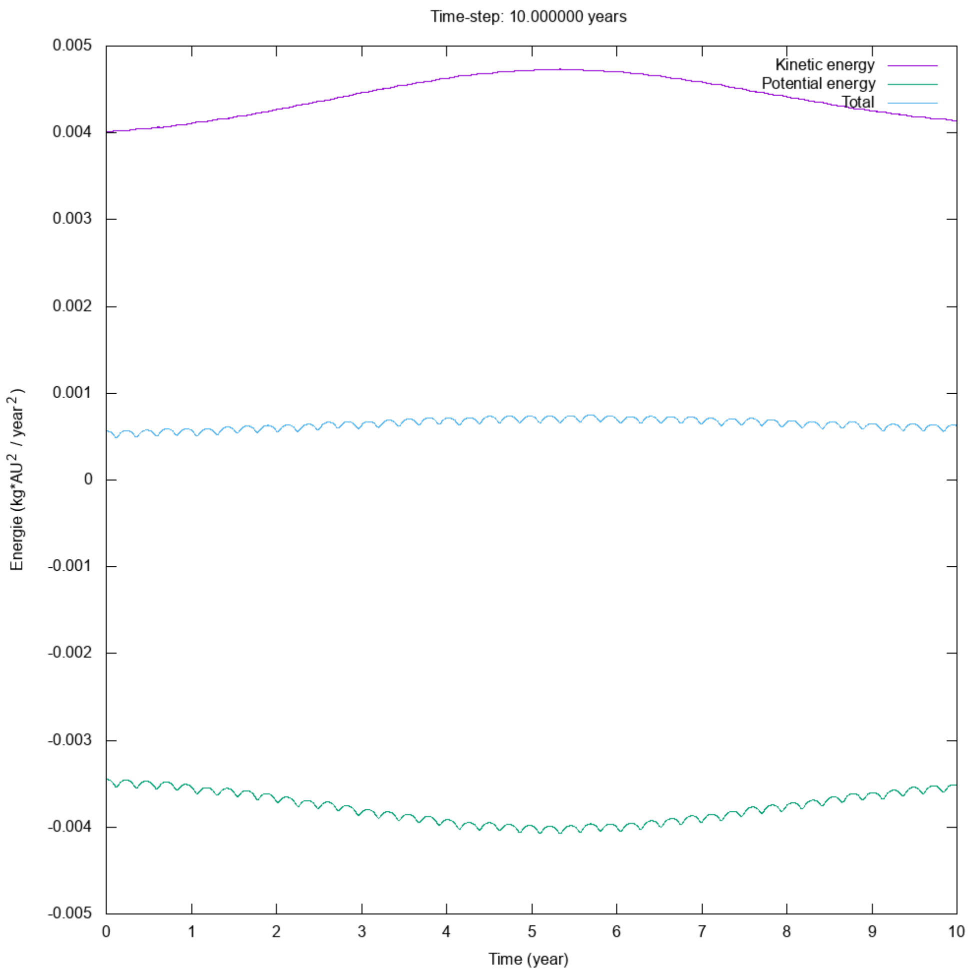
At each run of the program, a battery of unit-tests is computed to validate the results. They focus on auxiliaries functions, such as the methods calculating the center of mass of the system, the distance between two planets, etc etc. If the unit-tests are not validated, the program doesn't compute any result.

2.2.2 Preservation of the energy

The mechanical energy $E_m = E_c + E_p$ should be conserved because the only force taken into account is the gravitational force, which is conservative. The potential energy is constant in time ($E_p = -\frac{GM_\oplus M_\odot}{r} \forall t$), so is conserved. The kinetic energy depends on the velocity of the Earth ($E_c = \frac{1}{2}M_\oplus v^2$, which is constant in time as we have a uniform circular motion of the Earth around the Sun. So as both of the kinetic and potential energies are constant in time, for any t_i and t_f we have

$$\begin{cases} E_{c_i} = E_{p_f} \\ E_{p_i} = E_{p_f} \end{cases} \Rightarrow E_{c_i} + E_{p_i} = E_{c_f} + E_{p_f} \Rightarrow E_{m_i} = E_{m_f}$$

Now if we compute the energies of the full solar-system, we observe that our simulation indeed preserves the energy.



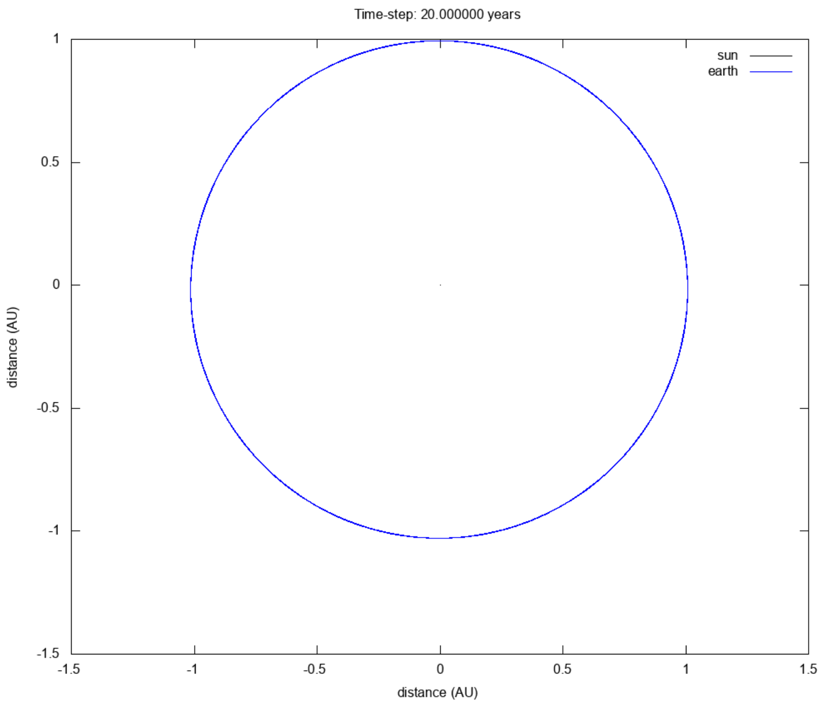
Simulation of the Solar System over 10 years

The "constant" oscillation of the total energy between a min and a max value is due to the imperfection of the circular orbit motions of some bodies. Even with a circular orbit axiom, some bodies like Mercury describe elliptical orbits.

2.2.3 Periodicity of the orbits

Orbits of the planets are supposed circular and must be time-periodic. This is probably the most important validity test of the algorithm and here is for example the orbit of the

Earth around the Sun computed by our Verlet's method. This result is quite reassuring. You will later see the whole solar-system with circular orbits.



Earth's time-periodic orbit around the Sun.

2.3 Performances

The program seems to be quite fast actually. Here is a small time-table to compare Euler and Verlet for an Earth-Sun system :

	Execution time		
	Time-step (years)	Euler (s)	Verlet (s)
Earth-Sun system	1	0.116699	0.19863
	5	0.563373	0.829593
	50	5.80658	9.27182
	200	25.3661	39.8954

According to those values, the execution time is linear to the time-step and the number of planets. To compute the whole Solar system for the next 100 years, the program takes 4.04346 s. However this time is without the writing to output files. Those output operations take a very long time and increase the operation time to... 46.9792 s. However this is a very good performance if we keep in mind that the algorithm computes one couple of position and velocity per day, with the energies of the total system, and the whole for each planet.

Chapter 3

Results

3.1 Earth-Sun system

3.1.1 Comparison between the two algorithms

Let's compare the results between using the Euler's method and the Verlet's method.

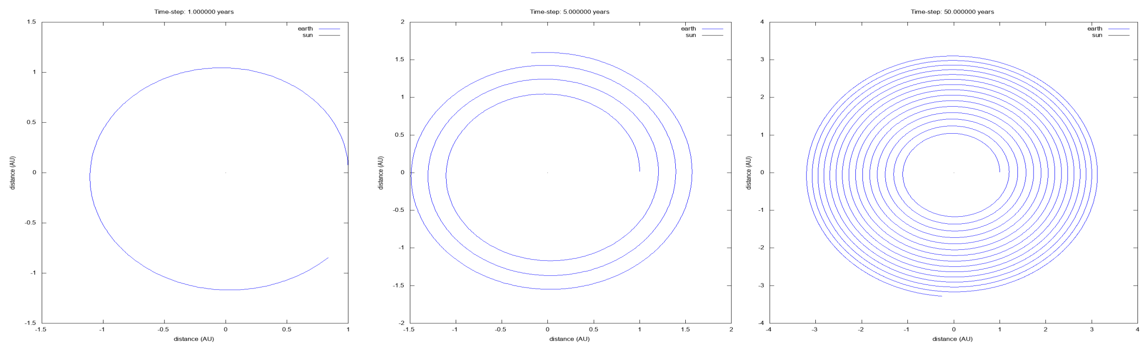


Figure 3.1: Euler's method over 1, 2 and 50 years (from left to right)

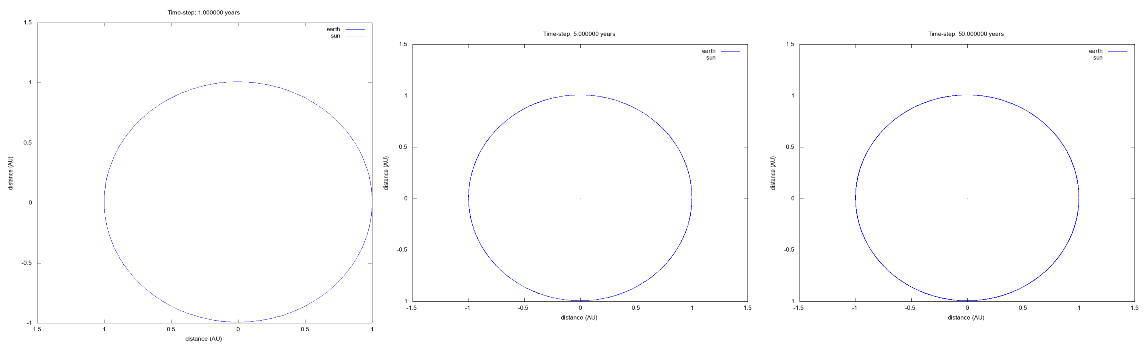


Figure 3.2: Verlet's method over 1, 2 and 50 years (from the left to the right)

We clearly see here the difference of precision between the two algorithms. Verlet goes one order further in the Taylor expansion of the acceleration. With the Euler's method, the Earth is not able to come back at its initial position and its orbit is therefore not periodic.

3.1.2 Escape velocity

We have computed the theorical escape velocity in 1.1.2.

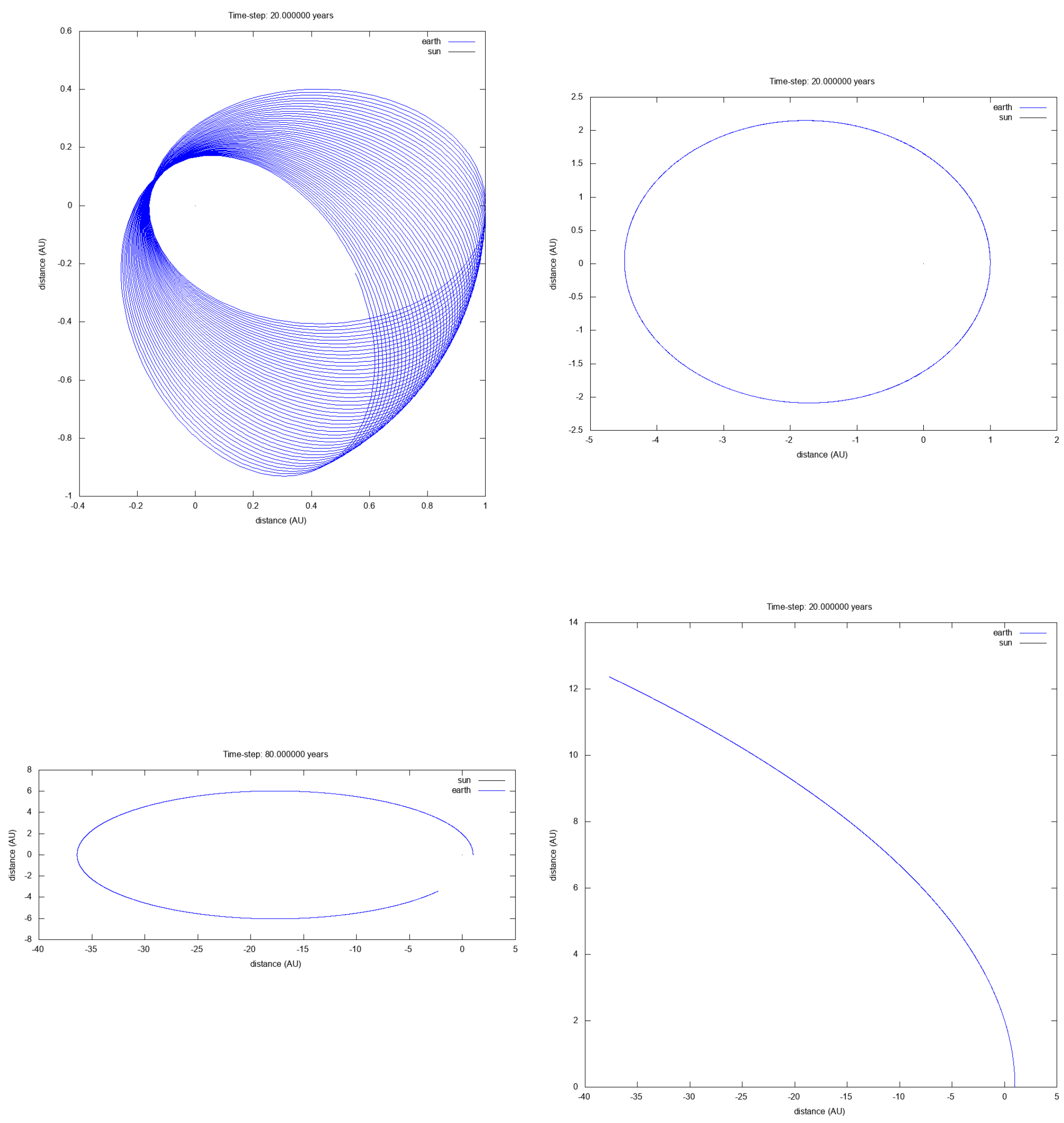


Figure 3.3: The Earth orbit around the Sun for $v_i = 3.287$ AU/yr, $v_i = 8.0355$ AU/yr, $v_i = 8.77$ AU/yr and $v_i = 8.8756$ AU/yr (from the left to the right, top to bottom)

Experimentally, the escape value is $v_{e_e} = 8.8756$ AU/yr which is really close to the theorical value $v_{e_t} = 2\sqrt{2}\pi \approx 8.8858$ AU/yr. We can see that when we approach velocities which are very near to the escape velocities, the Earth takes a way longer time to came back at its initial position and it has an elliptical orbit. When we reach a velocity intently closed to the escape velocity, the return point is a point at infinity. Mathematically there is only one velocity for which the orbit is stable (ie. periodic) in this model and we clearly see that on our results. After the Earth escaped, the motion is linear.

3.2 Three-body problem

3.2.1 The Sun as the center-of-mass

Here, we have kept the Sun as the center-of-mass. We have tested two things : the influence of Jupiter on the Earth, and its influence on the Earth when we change the mass of Jupiter.

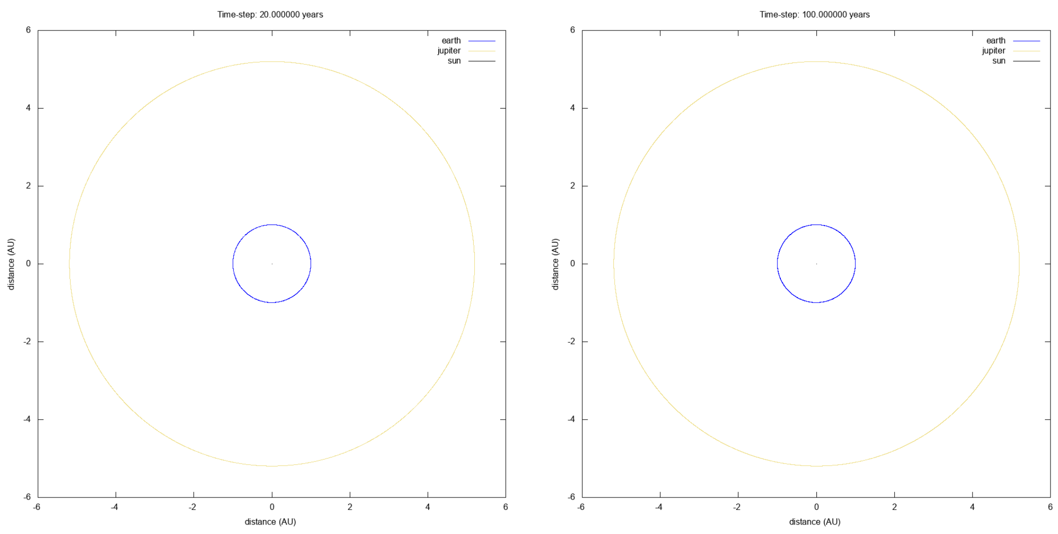


Figure 3.4: The evolution of the position of the Earth, Jupiter and the Sun over 20 and 100 years (from left to right). The system is very stable in time.

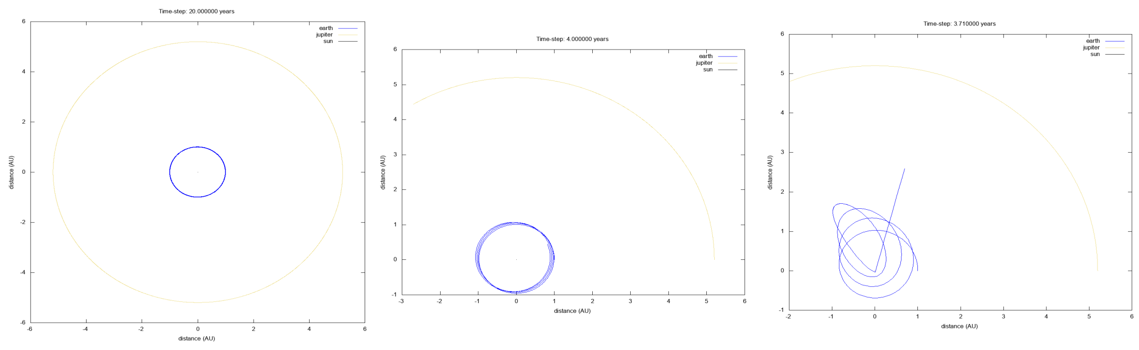


Figure 3.5: The influence of Jupiter on the Earth for $M_{\jupiter} = 10 \times M_{\jupiter}, M_{\jupiter} = 100 \times M_{\jupiter}$ and $M_{\jupiter} = M_{\odot}$ (from the left to the right)

Those results show that the influence of Jupiter on the Earth is not so important, even when Jupiter's mass is multiplied by a factor from 1 to roughly 100. However, when we change the order of magnitude to give Jupiter's the same mass as the Sun's mass, there is a confrontation and Earth goes to Jupiter.

Remark 3.1. *This result is to take with precaution. Indeed, the Sun was here considered as* *the fixed center of mass and had no motion. The situation would be different with a non-zero motion.*

3.2.2 The real center-of-mass

Here, we change the center-of-mass from the Sun to the real one. The coordinates of the three bodies with respect to the new origin have been computed in 1.2.2.

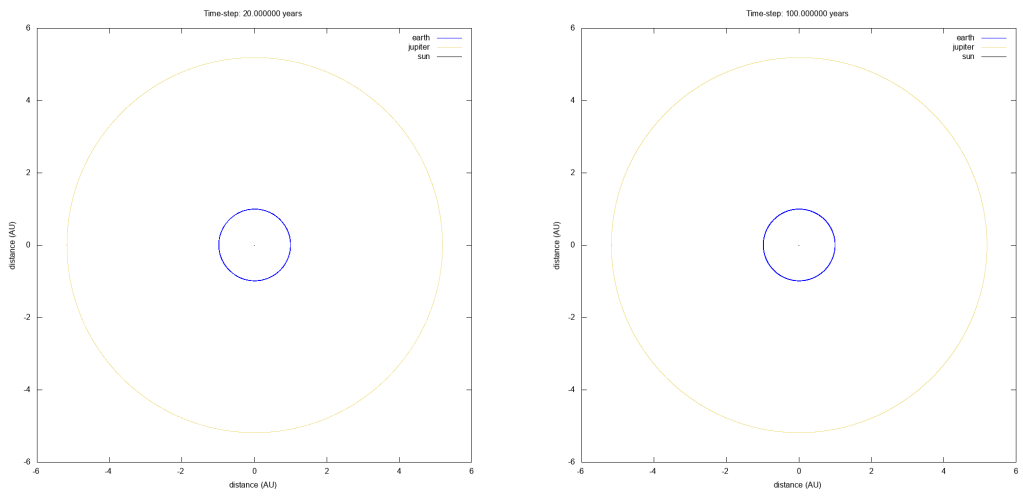


Figure 3.6: The position of the three bodies over 20 and 100 years (from the left to the right)

The orbits of the Earth and Jupiter are quite the same as the case where the center-of-mass was set to the Sun except that the Sun no has a motion.

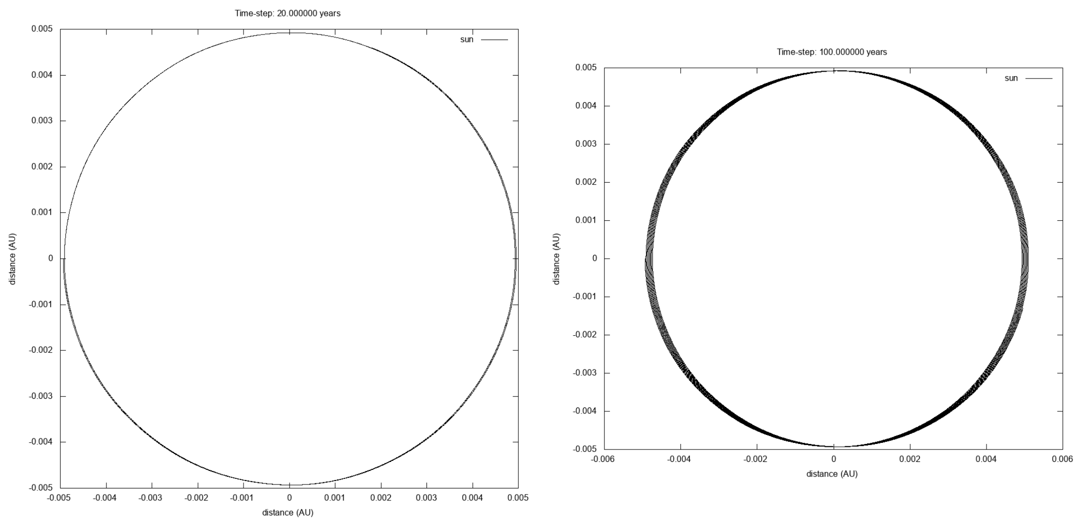
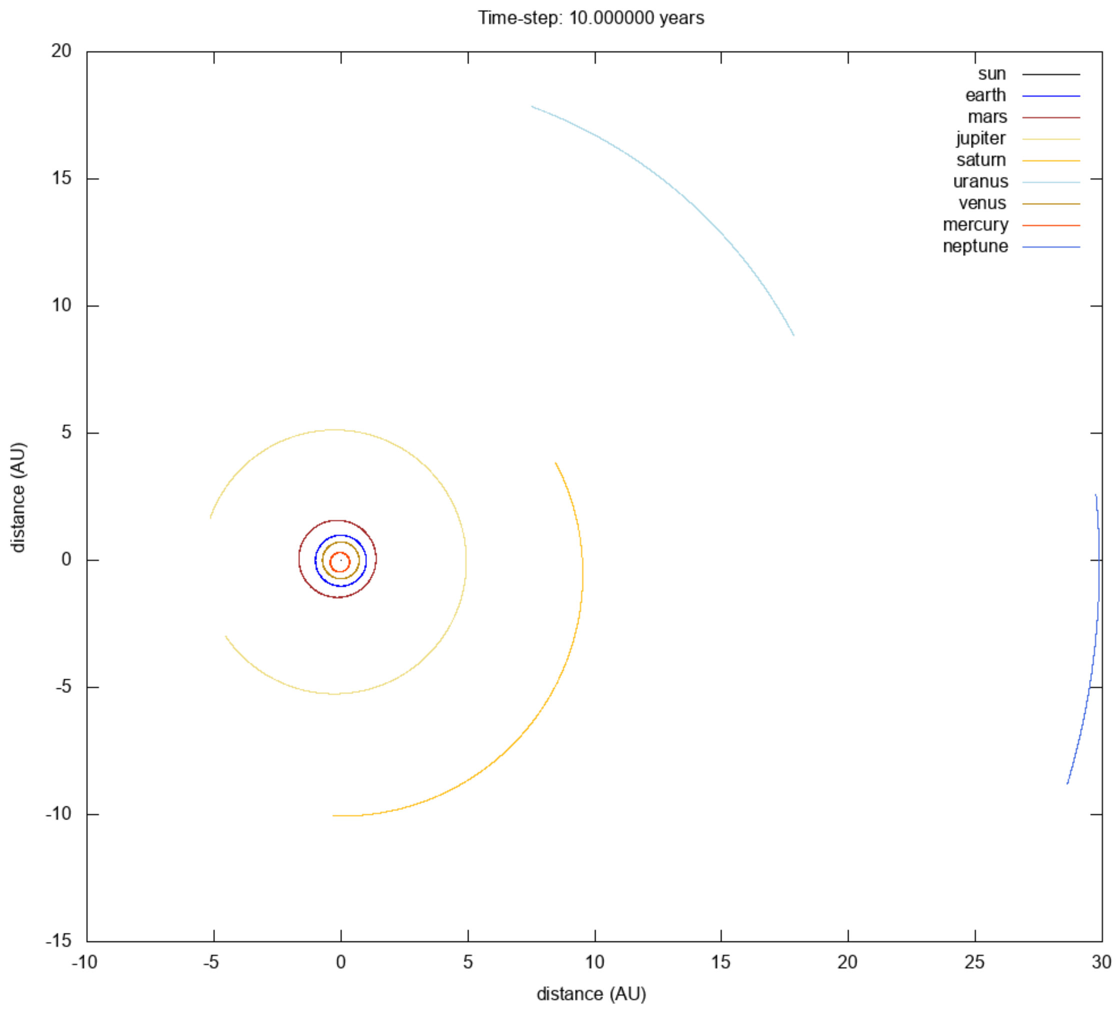


Figure 3.7: The orbit of the Sun around the center-of-mass over 20 and 100 years (from the left to the right)

The good thing is that the center of the Sun never reaches a more than 0.005 UA distance to the center of the system. Compared to the Sun's radius, this distance represents roughly 107% of its radius, which means that even if the Sun has a circular motion, there is always a point of the sun which is located in the center of mass of the system, assuming a 2D motion. This proves that the approximation of considering that the planets orbit around a fix-sun can be a very good approximation for our simulations.

3.3 Complete Solar System



Simulation of the Solar System over 10 years

Cooooooool. We observe that Saturn covers more or less a third of its complete orbit in ten years. It is consistent with the fact that a year on Saturn lasts about 30 years. Similarly, Jupiter covers almost its complete orbit, which is consistent as it covers its whole orbit in around 12 years. Everything seems ok here. We notice that Mars's motion is a bit off-centered of the Sun, but Jupiter and Saturn are definitely very stable and peaceful body. Uranus however seems to have an elliptical orbit, like Neptune.

Remark 3.2. *The initial conditions of the planets come from the NASA for the* 27th *of October at 00:00:00.*

3.4 The special case of Mercury

We had a problem here with our simulation : we computed the difference of Mercury's perihelion precession around a century, and compared the data obtained with a relativistic correction or without. For a Mercury's perihelion after after 99.9037 years, we get a perihelion precession of of roughly 130" ($\theta_{with} - \theta_{without}$) which is way too big compared to the 43" we should have obtained. All the data can be seen here : `https://github.com/kryzar/Perseids/tree/master/Program/Results/Perihelion%20precession`. You will notice that the time-steps of each perihelion is about 0.240732 year, which corresponds to one Mercury year. On top of that you will also notice that the position at any perihelion is very near to the initial position ($x_{p,i} = 0.3075$ and $y_{p,i} = 0$). For example, at $t = t_0 + 99.9037$:

$$x_{p,f} = 0.307499999746$$
$$y_{p,f} = 1.2308857225 \times 10^{-05}$$

The error in the value of θ probably results in a programmatic error.

Conclusion

As we have seen, except for the perihelion precession of Mercury, we have reliable results in accordance with the theory. This is a program which can be used by student who need a fast simulation since it can be run on must laptops. The next step would be to rewrite the program without the approximation such as the circular orbits.

Bibliography

- NASA website `http://ssd.jpl.nasa.gov/horizons.cgi#top`