Solar system simulation

Joakim Ginste

Physics:

The first part of solving this problem was getting the physics down and implemented into the python code. The main equation used for this project is Newton's law of universal gravitation, as can be seen below in equation 1.

$$F = G * \frac{m_1 * m_2}{r^2} \qquad \text{(Equation 1)}^1$$

The other equation used is the equation for force as can be seen in equation 2, where the subscript 1 is for the planets/object that the force is acting on.

$$F = m_1 * a_1 \text{ (Equation 2)}^2$$

Combining these two equations by the common variable F, and rearranging equation 3.a is obtained.

$$a_1 = G * \frac{m_2}{r^2} \qquad \text{(Equation 3.a)}$$

As the project are working with a solar system that is mapped onto 2D, where the sun is positioned at the centre [0,0] of an xy-grid, the equations must be reframed by using vectors. Firstly, we need to find the unit vector for the force that one planet acts upon another. This is done by getting the distance between the two planets "r".

$$\vec{r} = r_1 - r_2 \qquad \text{(Equation 4)}$$

$$\hat{r} = \frac{\vec{r}}{|r|} \qquad \text{(Equation 5)}^3$$

With equation 5, equation 3.a with vectors can now be written as:

$$a_1 = G * \frac{m_2}{|\vec{r}^2|} * \hat{r} \qquad \text{(Equation 3.b)}$$

With the help of equation 3.b, an acceleration can be calculated. This acceleration is then used to update the position and velocity of the same object. To update the position and velocity of the object Verlet-leapfrog predictor corrector integrator was used as recommended by Nbabel.org and can be seen below in equation 6 and 7.[4]

$$r_{i+1} = r_i + v_i * dt + \frac{1}{2} * a_i * dt^2 \qquad \text{(Equation 6)}$$

$$v_{i+1} = \frac{1}{2} * (a_i + a_{i+1}) * dt \qquad \text{(Equation 7)}$$

---

[1] Wikipedia, *Newton's law of universal gravitation,* https://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation

[2] Wikipedia, *Force*, https://en.wikipedia.org/wiki/Force

[3] Wikipedia, *Unit vector*, https://en.wikipedia.org/wiki/Unit_vector

[4] Nbabel, *Equations,* http://nbabel.org/equations

Python programming:

To make the programming easier, object-oriented programming was used, where each planet/sun/star was made into an object that contained data on its name, position, color, mass, velocity, radius, and acceleration. The radius was set to specific numbers for the plot to make more sense (otherwise the smaller planets would not be visible due to being small compared to the bigger planets). Here the suns radius is very small as can be see in the figures below, but this was made to be able to see the smaller planets such as Mercury etc.

To solve the orbits of each planet, firstly data was collected from NASA of each planets mass, position (Average distance from sun) and orbital velocity.[5] These values were converted into SI units. Using the physics described above a function was made that takes two different objects as input and outputs the acceleration gained to the first object by the gravitational force of the second object. Thereafter with the new acceleration, both the new position and velocity is gained as described above equation 6 and 7.

To get all the planets/star/sun effect on each other a planet a for loop in a for loop was made to go through for each planet the acceleration each other planet has on the specific planet and added together to get the final acceleration for that time step on that planet. Thereafter a loop goes through for 100 years and stores all the positions for each planet at each time step. This data was then used for plotting of the solar system, and how the planets have moved over 100 years. For plotting matplotlib was used.

The python program is designed so that all planets start on a line on the x-axis, and with their initial velocity (orbital velocity) pointing upwards in the y direction. In figure 1 below we can see the starting position of the solar system. This was obtained using the first code named "Solar_normal". In the same code we also get figure 2 shown below that shows how the solar system behaves during 100 years without an extra star, where the dotted lines are the previous positions for each planet.

[5] Nasa, *Planetary Fact Sheet – Metric*, https://nssdc.gsfc.nasa.gov/planetary/factsheet/

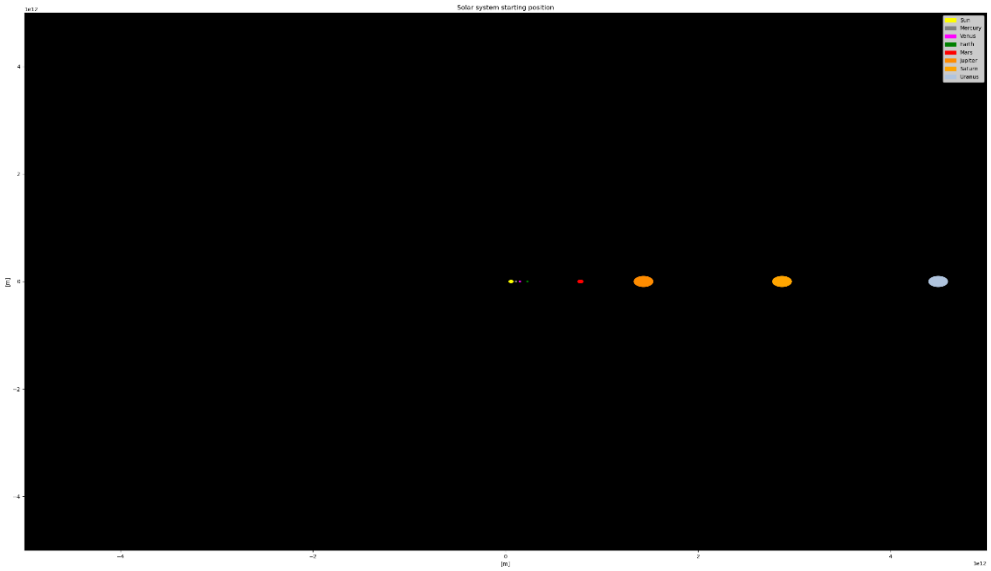

Figure 1. Starting position for solar system without an extra star

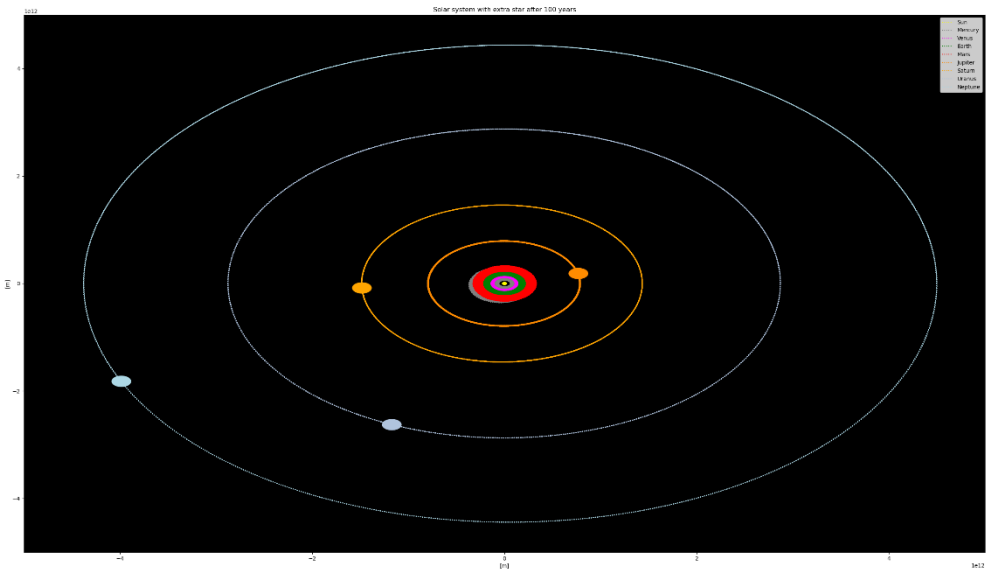

Figure 2. Solar system without an extra star after 100 years.

For the simulation of the regular solar system that is shown in figure 2, a step size of 2 minutes was used. It can be seen for the planets closer to the sun that this step size might be too low of s their orbits are very wide. Preferable a lower step size would be used, but due to lack of time this could not be done. It took around 7 hours to simulate figure 2 with the step size of 2 minutes.

For the simulation of the solar system with the extra star the python code “Solar_with_extra_star” was used. The same starting position is used for the simulation with

the extra star, but the star is given a randomize position on Neptune's orbit around the sun as that is assumed to be the utmost part of the solar system. Thereafter the velocity vector for the extra star is calculated in such a way that if it would go in a straight line for 100 years it would get as close to the sun as the Earth is. This is shown in figure 3, where the predicted path of the star is shown. To find the vector for the velocity four different equations were used depending on what quadrant in the ellipse around the sun that the star is starting in. To calculate the magnitude of velocity for the star equation 8 below was used. Where system length was 2 times Neptune's distance from sun at the start.

$$v_{star} = \frac{system\ length}{100*8760*3600}\ [\frac{m}{s}] \qquad \text{(Equation 8)}$$

In figure 4 below we see the effects the Star will have on the solar system whilst it is flying through under 100 years. As can be seen it quickly turns into chaos and all the planets/stars go outside the solar systems boundaries. This simulation was made with a step size of 6 minutes and took around 2 hours to complete. As there is an extra object in this simulation using a step size of 2 minutes as done in the first simulation, would take a considerable amount of time to finish, and therefore the step size was increase. This might affect the results and to get the best result possible a low step size as possible would be ideal for the simulation.

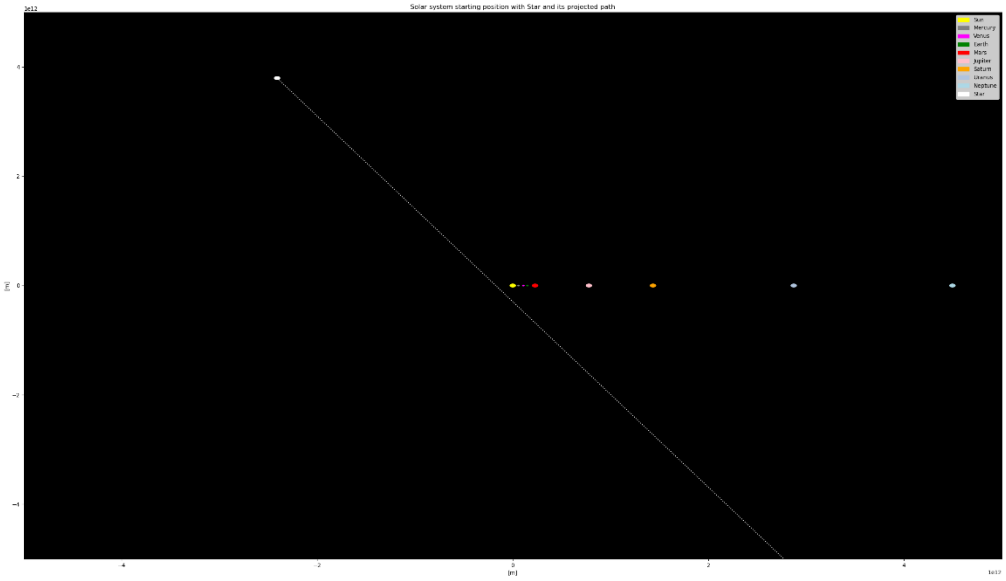

Figure 3. Starting position of solar system with extra star and its projected path

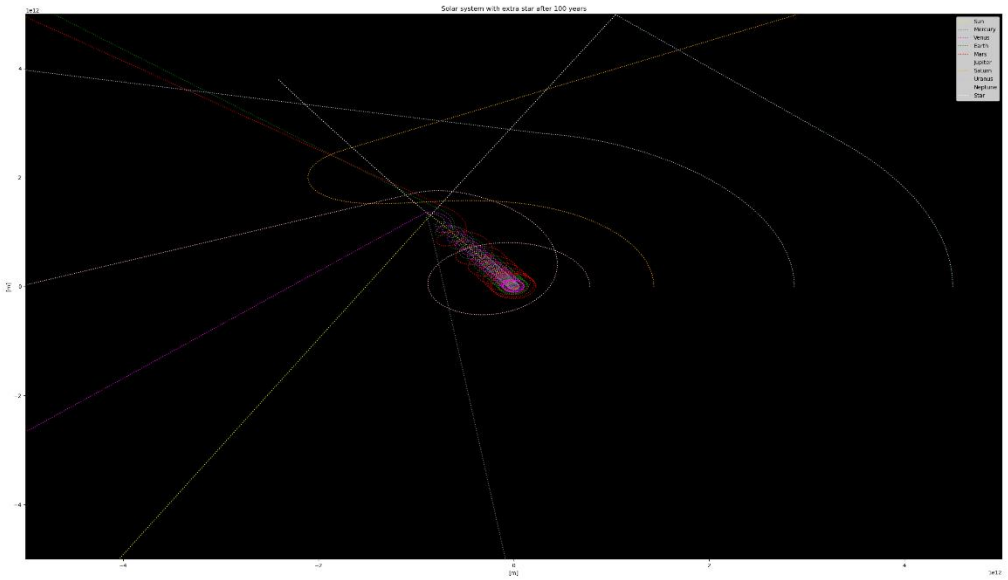

Figure 4. Solar system with extra star after 100 years.

References:

Nasa, *Planetary Fact Sheet – Metric*, https://nssdc.gsfc.nasa.gov/planetary/factsheet/ [obtained 2021-07-18]

Nbabel, *Equations*, http://nbabel.org/equations, [obtained 2021-07-16]

Wikipedia, *Force*, https://en.wikipedia.org/wiki/Force, [obtained 2021-07-25]

Wikipedia, *Newton's law of universal gravitation*, https://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation, [obtained 2021-07-24]

Wikipedia, *Unit vector*, https://en.wikipedia.org/wiki/Unit_vector, [obtained 2021-07-26]