Project Report

Cometo

The Interactive Comet Trajectory Simulation

By

Thatchapon Unprasert

Section 1, ID: 5888220, 1st Year Student

Physical Science and Computation

ITCS161

Abstract

Astronomers have been studying the universe in several ways in order to know how the life forms. One evidence that they can count for is studying properties of comets that they could detect by a telescope. A comet has the existence of water and organic compounds, which might yield important clues of life formation. To collect information from a comet, they not only need to know its physical characteristics, but also need to identify the trajectory of the comet so that they can design the observation when it passes close to the sun. This project is responsible for identifying trajectories of comets, much like what astronomers do, by giving their instantaneous velocities and distances from the sun. To do so, the Python program is designed to perform the simulation using VPython library, based on the fundamental physics formula: gravitational force. As a result, any comets added to the simulation move realistically and have their movement significantly matched with the in-depth approximation by physics laws. To conclude, this project well achieves the trajectory simulation, yet it is not supposed to be an accurate simulator that aims to mimic the real situation.

Before start

Cometo (Comet + “o”) is an interactive simulator that plays major roles on comets’ trajectory. It provides the solar system model as a basic environment with the sun at the center. Six predefined comets are included to demonstrate the user how comets would move under gravitational forces exerted by any massive bodies. Not only the sun, but also planets, and even other comets that affect the trajectory of any particular comet. This is because gravitational force works with any objects with mass as its property. Four predefined comets create the elliptic orbit with a variety of eccentricities, velocities and approaching directions. Others create the hyperbolic orbit with different velocities and directions. The main feature of Cometo is how you control this program. The simulation control window allows you to take control of the simulation. With this, you can adjust the simulation rate to be faster, slower, or just pause it. You can show a marker, trajectory line and mechanical properties for any moving objects. It is also possible to make everything in a true scale. You can even generate a new comet randomly and see how it would move when time passes. If you want to get through the simulator even more, use the console feature. It is the place where you make the most of the simulation. This feature will be completely explained later in this paper.

Key Concepts

1. How objects move?

This project uses a basic physics formula to simulate both comets' and planets' motion, starts with Newton's force of gravity law:

$$F_{gravitational} = -\frac{Gm_1m_2}{r}\hat{r}$$

This means, every object attracts each other by what we called "gravitational force". A force changes an acceleration of the object that the force exerts, follows the 2^{nd} Newton's inertial force law:

$$F = ma$$

Combining these two formulae, we obtain the formula that calculates a change in the velocity of an object (since there's acceleration) with respect to the other object, which attracts it by gravitational force:

$$a_{radial} = \frac{-Gm}{|r|^2}\hat{r}$$

Where: m: mass of other object

r: vector <u>from</u> other object <u>to</u> target object

Any objects added require the initial velocity and distance from the reference (the sun in this case), it otherwise simply falls down into sun with zero speed. That is, it requires an energy to maintain the trajectory with respect to its massive body. After the object is successfully added into the system, this formula controls its motion.

2. Approximate the right way

It is possible to calculate (approximate) most mechanical properties of any comets (any "satellites", for general meanings) that we are able to measure their velocity, mass and distance from the sun. In fact, we need to know their mass. The best guess that can be made is to approximate the mass from the composition of a comet because it produces undetectable gravitational perturbations from which its gravitational mass could be estimated. Having density value and physical sizes (determined by angular size) of the comet, one can derive the approximated mass.

Trajectory type of a comet is rather the first thing that astronomers would like to know. The type can be one of the followings: circular, elliptic, parabolic or hyperbolic. They have different value of eccentricity, and eccentricity of the orbit can be calculated from the total energy of the comet.

The following ordered items show how to estimate the orbit type, together with every other parameter used by this program.

2.1 Energy of comet

Given an object (m) that orbits around a massive object (M), the total energy is the sum of kinetic energy and potential energy of the object, says:

$$E_{total} = E_{kinetic} + E_{potential} = \frac{1}{2}mv^2 - \frac{GMm}{r}$$

Anyway, our system is rather more complex, as the comet's energy is affected by other surrounding objects. Keep in mind that there are other massive bodies (in comet's perspective) apart from the sun. So, the formula can be rewritten for any particular comet c, with respect to the sun and perturbations by any other bodies:

$$E_{c,total} = \frac{1}{2}m_c v_c^2 - \sum_{obj \in S} \frac{G m_{obj} m_c}{r_{obj-c}}$$

S = Set of all surrounding objects, including sun, planets, and other comets

2.2 Semi-major axis

From 2.1, since $-\frac{GMm}{2a} = E_c$ if r = a (no matter what r is, the total energy remains constant) Therefore, we get:

$$a = -\frac{GM_{sun}m_c}{2E_{c,total}}$$

We determine the semi-major axis for several further uses of determination of other physical parameters.

2.3 Reduced mass

A so-called inertial mass for two-body problem of Newtonian mechanics, for two objects can be written this way:

$$\mu = \frac{m_1 m_2}{m_1 + m_2}$$

2.4 Orbital eccentricity

In a two-body problem, every orbit is Kepler orbit, which its eccentricity can be determined by this formula:

$$e = \sqrt{1 + \frac{2E_{total}L^2}{G^2 M^2 \mu^3}}$$

However, the origin can be something else than sun. For instance, Moon orbits around Earth because the gravitational force magnitude is higher than the amount that sun exerts.

Where L is angular momentum of the orbiting object, μ is the reduced mass, and M is the mass of massive body at the origin point.

Substituting these remaining variables by relevant formulae, we obtain:

$$e = \sqrt{1 + \frac{2E_{c,total}\left|r_{c-origin} \times v_c\right|^2 (M + m_c)^3}{G^2 M^5 m}}$$

We finally get eccentricity formula that can tell the orbit type of the comet. (e = 0 for circular, 0 < e < 1 for elliptic, e = 1 for parabolic, and e > 1 for hyperbolic)

2.5 Vis-via equation

It's the formula that models the motion of orbiting bodies. Given the distance from massive body r, and semi-major axis for the orbit, we have:

$$v^2 = GM\left(\frac{2}{r} - \frac{1}{a}\right)$$

We can use this formula to determine the comet speed at its perihelion r_p and aphelion r_a (for elliptic orbits). Since $r_p = a(1 - e)$ and $r_a = a(1 + e)$ according to the geometry:

$$v_p = \sqrt{\frac{GM}{a}\frac{1+e}{1-e}} \text{ and } v_a = \sqrt{\frac{GM}{a}\frac{1-e}{1+e}}$$

2.6 Orbital period

Using the Kepler's law of motion, the orbital period of a comet can be calculated this way:

$$T = 2\pi\sqrt{\frac{a^3}{GM}}$$

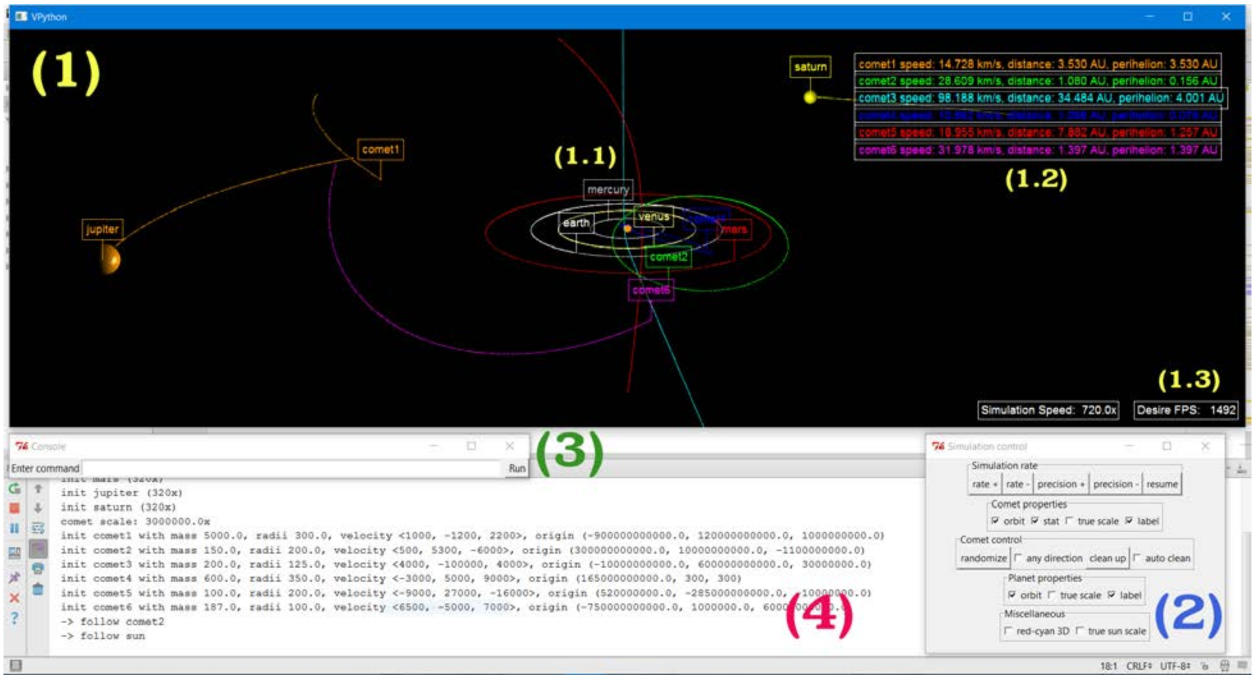
2.7 Terminal speed (for hyperbolic orbits)

The object reaches its terminal speed when it moves forward far enough from the massive body. This speed never changes until it is disturbed by any other massive bodies. At this state, there is no potential energy, so we calculate the speed from vis-viva equation and obtain:

$$v_{terminal} = \sqrt{\frac{GM}{-a}}$$

Program Features

At a glance



Four main components

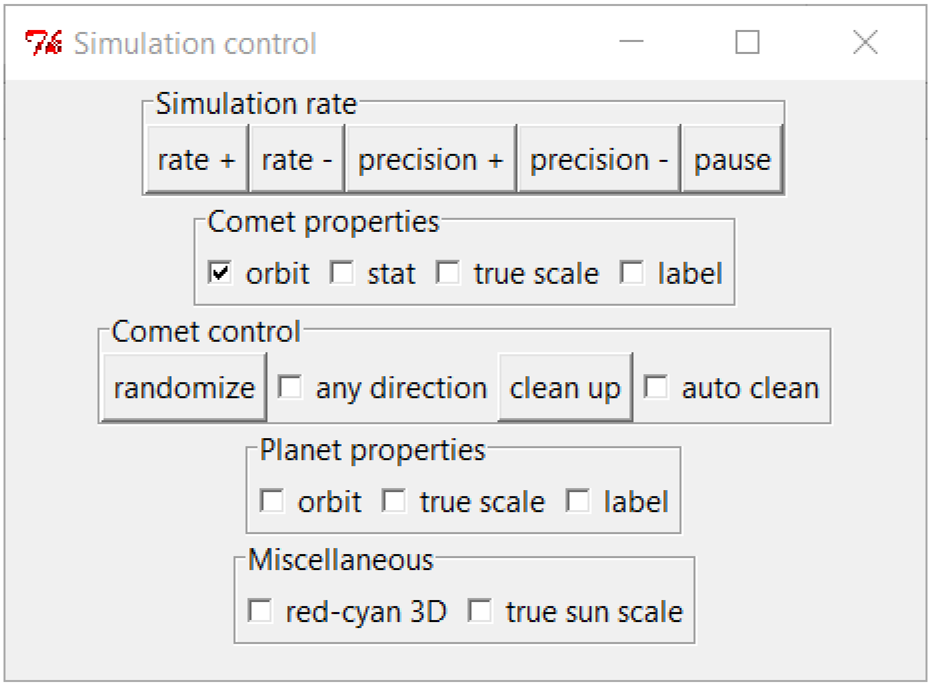
1. Simulation window
    - 1.1 Label and trajectory line
    - 1.2 Comet properties report
    - 1.3 Simulation rate
2. Simulation control panel
3. Console
4. Debugging section

Usually, users are supposed to manage the simulation from simulation control panel. They are anyway allowed to input a command to control many simulation behaviors right from the console window, and get debugging information from debugging section, if applicable.

You can rotate the scene (right-click pan) and zoom the scene (left-and-right click pan).

Simulation Control Panel

Makes the simulation more desirable



1. Simulation rate

This section allows you to adjust the simulation speed at which you prefer. The "rate" buttons adjust the maximum frames per second (desire FPS) of the simulation. More positive adjustment of rate means more speed of simulation, but it also depends on your computer graphics performance. The "precision" buttons adjust the precision of the simulation. More positive adjustment of precision means precise trajectories of everything, but this also slows down your simulation speed. On the other hand, negative adjustment of precision gives you more simulation speed, but you have to accept its poor quality of precision because the delta t (difference in time per calculation) is huge. The last one at the right side is the simulation state toggle. You can pause and resume the simulation at will.

2. Comet properties

This section manages the representation of comets. The "orbit" checkmark toggles the visibility of their orbits. The same for "label" checkmark but is for comets' labels (their names). The "stat" checkmark toggles the visibility of comets properties report (as can be seen from 1.2 in the first program screenshot). The "true scale" toggles the sizes of comets, either scaled or true sizes.

3. Comet control

This section provides randomly generating comets feature. You can generate right from a single click on “randomize” button. You may also want generated comets to only move towards the sun (with predefined range) to see how their trajectories would change, by disabling “any direction” option. According to the fact, some comets might travel close to the sun and start to evaporate because of sun heat. The simulation of those either evaporated or within-sun comets won’t be continued and has their trajectories left behind. Says you want only normal comets to be simulated, you would just click “clean up” button to clear all outdated trajectories. This process can be automatically performed if the “auto clean” checkmark is enabled.

4. Planet properties

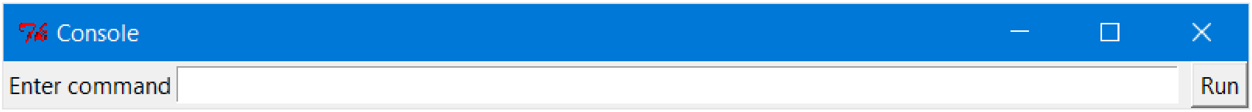
Similar to comets, this section manages the representation of planets, yet the planet properties report toggle (“stat”) isn’t included because we mainly focus on comets.

5. Miscellaneous

This section holds some other options that might be useful for one. The “red-cyan 3D” visualizes the scene in 3D, bring the red-cyan 3D glasses to have even more better experiences. The “true sun scale” option does the job as the name said, enabling it together with comets’ and planets’ enables you to see the solar system and comets in the realistic way.

Console

Empowers the simulation capability



Though the interface is simplistic, you can supply varieties of commands to the simulation. Input the command then either press “Run” button or hit enter to proceed.

Commands List

1. Pause or resume simulation
   `pause`
   `resume`
2. Remove all outdated comets*
   `clean`
3. Print all adjustable variables used by this program*
   `allvars`
4. Center at any particular object
   `(follow|center)` *`object`*
   `follow sun`
   `center comet3`
5. Adjust precision or desire fps rate (multiplying factor for each button click)
   `rate` *`variable-name value`*
   `rate rate 2.0`
   `rate fps 2.0`
   `rate precision 1.5`
   Note: ‘rate’ and ‘fps’ are the same
6. Scale any particular object
   `scale` *`object value`*
   `scale sun 32.5`
   `scale comet5 6e5`
7. Multiply comet’s velocity (either multiply all components or separately)
   `speedmult` *`object value`*
   `speedmult` *`object`* `(`*`mx,my,mz`*`)`
   `speedmult comet1 2.5`
   `speedmult comet8 (2,-1,3.2)`
8. Add a comet to simulation*
   `add mass=`*`mass`* `r=`*`radii`* `v=(`*`vx,vy,vz`*`) o=(`*`ox,oy,oz`*`) (oau)`
   `add mass=100 r=300 v=(2,1,3) o=(1,1.3,-1.1) oau`
   Adding 100-kg, 300-m radii comet, at velocity (2, 1, 3) m/s, at origin (1, 1.3, -1.1) AU
   `add mass=20 r=15 v=(2e3,-1.5e3,1e3) o=(-2e11,5e10,3e11)`
   Adding 20-kg, 15-m radii comet, at velocity (2000, -1500, 1000) m/s, at origin (-2×10^{11}, 5×10^{10}, 3×10^{11}) m

9. Approximate comet's mechanical properties*

```
approx(imate) object
approx comet1
approximate comet3
```

10. Build d-t, v-t, a-t or e-t graph of any object (except sun for v-t, a-t and e-t)

```
graph d(istance)-t(ime) object1 object2
graph (v|speed)-t(ime) object
graph (a|accel)-t(ime) object
graph (e|energy)-t(ime) object
graph distance-time comet6 comet7
graph v-t earth
graph a-t comet4
graph e-t comet5
graph analyze comet3
```

Note: 'graph analyze' works all d-t (compared to sun), v-t, a-t and e-t for given object.

Note: e-t will illustrate both kinetic energy and potential energy for the given object.

Unit for graph:

d-t (AU-Year)

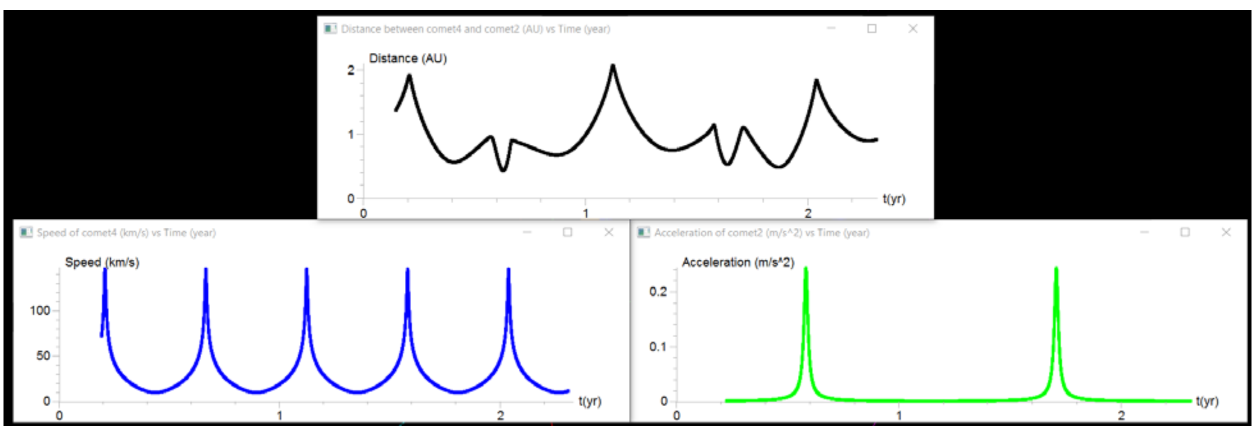
v-t (km/s-Year)

a-t (m/s^2-Year)

e-t (J-Year)

*Options with * marked means they produce output in debugging section as required.*

The graph examples using 'graph' command

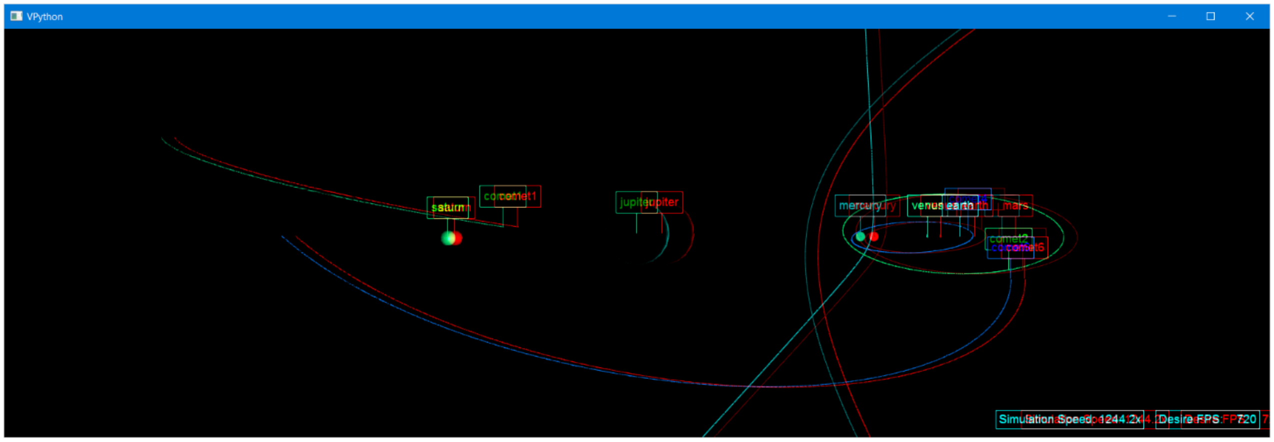


Bottom-left: `graph v-t comet4`

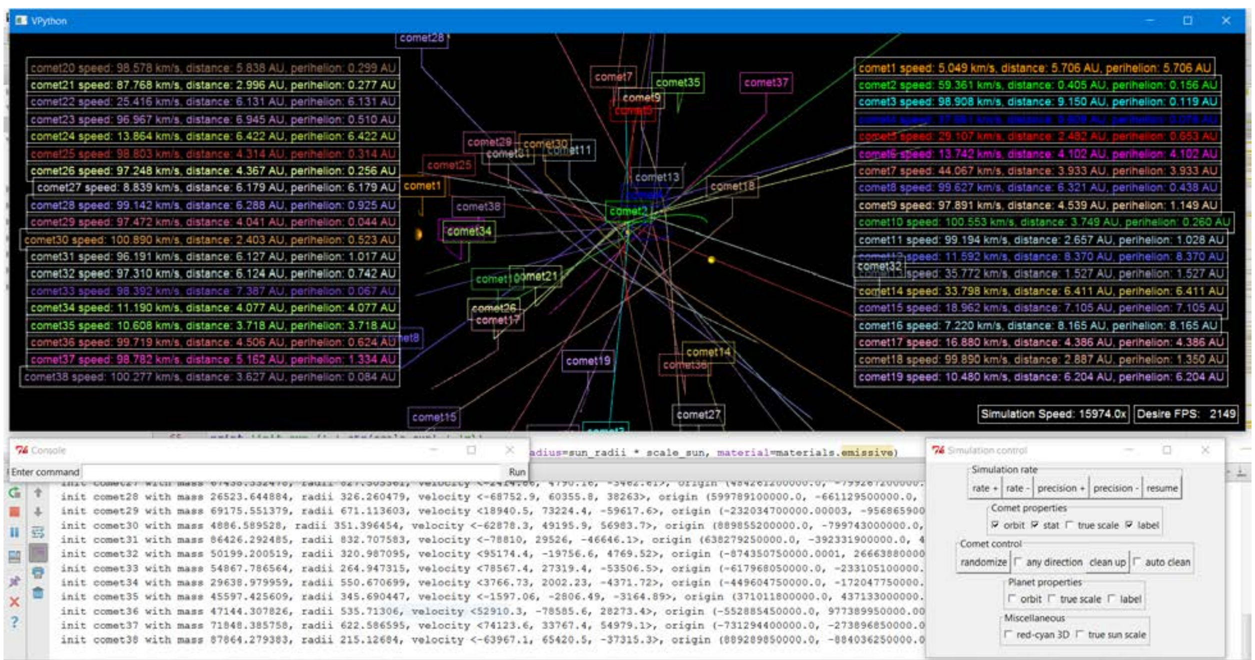
Bottom-right: `graph a-t comet2`

Top-middle: `graph d-t comet4 comet2`

Screenshots

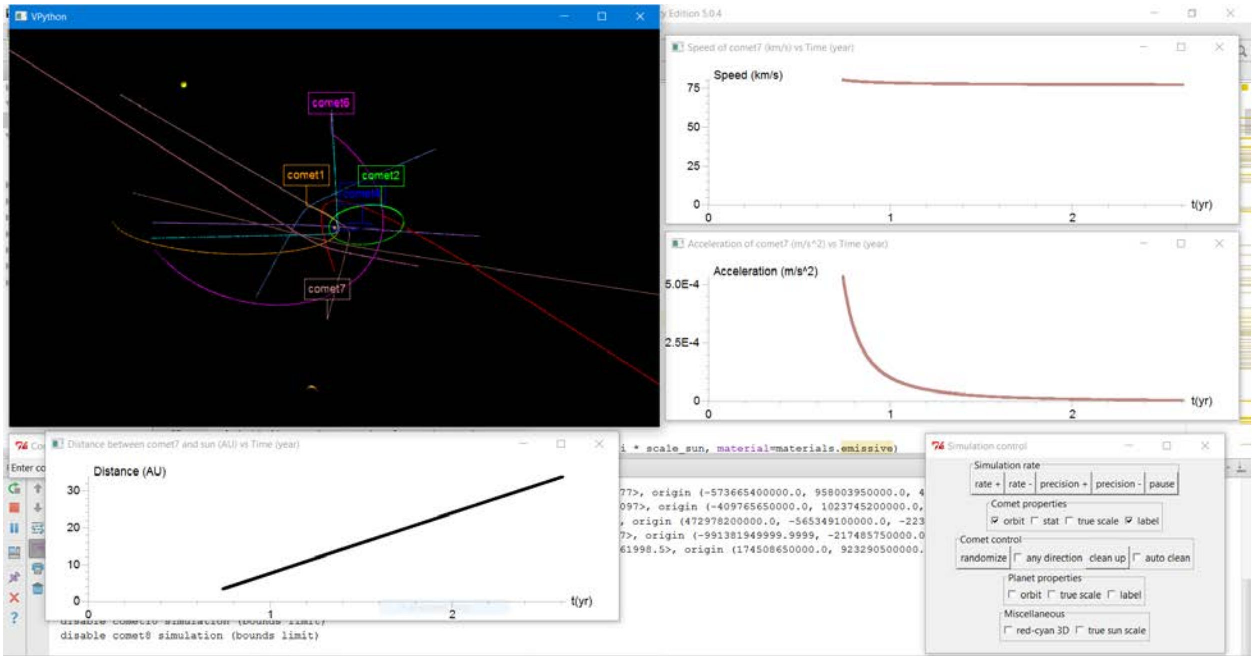


Red-cyan 3D Simulation



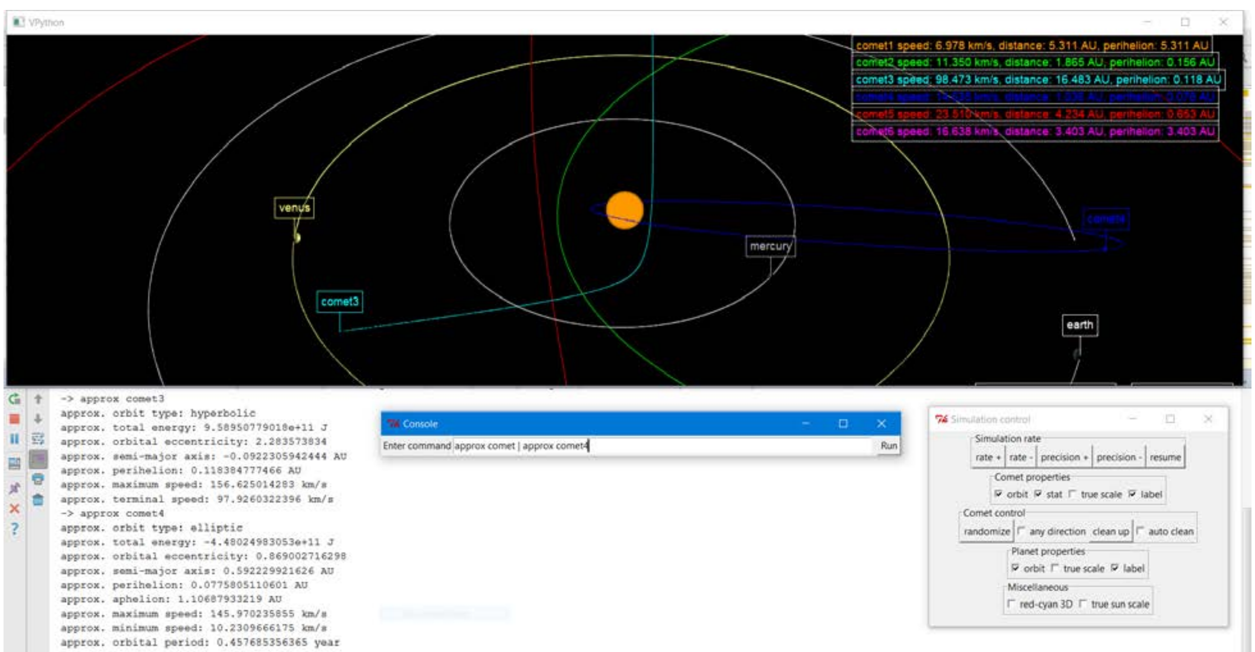
Comets’ Party

Note on comet’s party: though one can add unlimited numbers of comets, one needs to be concerned about the real purposes. This image just shows the dynamic of adding comets.



Comet Analysis

Note on comet analysis: starting point of a graph is current absolute time when you create it.



Comet Approximation

Note on comet approximation: the approximation yields the very close values compared with comets' properties report, or the actual values from the simulation.