Quentin_Trombini_Week14

S14_01_a

```
import rebound
import os
import numpy as np

import matplotlib.figure as figure
import matplotlib.pyplot as plt

import matplotlib.backends.backend_agg as agg

# ------------------------------- STAR CREATION ----------------
def make_sphere (x_c, y_c, z_c, radius, colour):
    u = np.linspace (0, 2 * np.pi, 1000)
    v = np.linspace (0, np.pi, 1000)
    x = radius * np.outer (np.cos(u), np.sin(v)) + x_c
    y = radius * np.outer (np.sin(u), np.sin(v)) + y_c
    z = radius * np.outer (np.ones(np.size(u)), np.cos(v)) + z_c
    sphere = ax.plot_surface (x, y, z, color=colour, antialiase
    shade=True, rcount=100, ccount=100)
    return (sphere)

# ----------------------------- CREATE SIMULATION -------------
sim = rebound.Simulation()

sim.add(m=2.063)
sim.add(m=1.018, a = 7.4957, e = 0.59142)

sim.save_to_file('Sirius.bin')

# ---------------------------- ORBITAL INTEGRATION ------------
if not os.path.isfile("./S14/S14_graph000.png"):
    sim.move_to_com()
    part = sim.particles

    interval = 0.5
    n_output = 500
    dt = 0.01

    sim.integrator = 'ias15'
    sim.dt = dt

    for i in range(n_output):

        t = interval * i
        sim.integrate(t)

        star1_x = part[0].x
        star1_y = part[0].y
        star1_z = part[0].z
        star2_x = part[1].x
        star2_y = part[1].y
        star2_z = part[1].z

        file_fig = f"./S14/S14_graph{i:03d}.png"

        fig = figure.Figure(figsize=[15.36, 8.64,])
        fig.subplots_adjust(left=0.0, right=1.0, bottom=0.0, to
        canvas = agg.FigureCanvasAgg(fig)
        ax = fig.add_axes((0, 0, 1, 1), projection='3d')

        fig.set_facecolor ('black')
        ax.set_facecolor ('black')
        ax.grid (False)

        ax.xaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
        ax.yaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
        ax.zaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
        ax.set_xlim(-15, +5)
        ax.set_ylim(-10, +5)
        ax.set_zlim(-5, +5)
        ax.set_aspect('equal')

        siriusA = make_sphere(star1_x, star1_y, star1_z, 0.5, "c
        siriusB = make_sphere(star2_x, star2_y, star2_z, 0.2, "a

        fig.savefig(file_fig, dpi=225)
        plt.close(fig)

# -------------------------- CREATE VIDEO FROM IMAGE ----------
command = "ffmpeg -framerate 30 -pattern_type glob -i './S14/*.p
os.system(command)
```

S14_01_b

https://youtu.be/PyhyXIN1fw4

https://www.youtube.com/watch?v=mQyk7CsifuU

S14_01_c

This is the version representing Sirius movement from this paper:

https://iopscience.iop.org/article/10.3847/1538-4357/aa6af8/pdf

(The first part of the code is the sphere creation but it’s used later so I will explain it when it’s used)

The first line of code is the following if statement that allow me to make different try with ffmpeg without recreating all the images each times

```
if not os.path.isfile("./S14/S14_graph000.png"):
```

Then I will define the simulation and the parameter of it including the stars mass, eccentricity and semi-major axis (found from the paper).

```
t = interval * i
sim.integrate(t)

star1_x = part[0].x
star1_y = part[0].y
star1_z = part[0].z
star2_x = part[1].x
star2_y = part[1].y
star2_z = part[1].z
```

This will compute the time depending on the time step of we loop we are in and then use this time to compute the position of each stars on the three axis

```
fig = figure.Figure(figsize=[15.36, 8.64,])
fig.subplots_adjust(left=0.0, right=1.0, bottom=0.0, top=1.0, ws
canvas = agg.FigureCanvasAgg(fig)
ax = fig.add_axes((0, 0, 1, 1), projection='3d')

fig.set_facecolor ('black')
ax.set_facecolor ('black')
ax.grid (False)
ax.xaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
ax.yaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
ax.zaxis.set_pane_color ((0.0, 0.0, 0.0, 0.0))
```

This will create a black background and a black 3D space to represent space and on which I will be able to place my 3D objects

In the following code I pass the place of my object generated by the simulation to the create_sphere function that will make a 3D sphere at the right place

```
siriusA = make_sphere(star1_x, star1_y, star1_z, 0.5, "orange")
siriusB = make_sphere(star2_x, star2_y, star2_z, 0.2, "aquamari

def make_sphere (x_c, y_c, z_c, radius, colour):
    u = np.linspace (0, 2 * np.pi, 1000)
    v = np.linspace (0, np.pi, 1000)
    x = radius * np.outer (np.cos(u), np.sin(v)) + x_c
    y = radius * np.outer (np.sin(u), np.sin(v)) + y_c
    z = radius * np.outer (np.ones(np.size(u)), np.cos(v)) + z_
    sphere = ax.plot_surface (x, y, z, color=colour, antialiase
    shade=True, rcount=100, ccount=100)
    return (sphere)
```

For each timestamp an image with two sphere of the right size and colour will be generated then I will call a terminal ffmpeg command to make a video out of all the images