Understanding interactions between celestial bodies using Python

Vineet Vijayan*, B.P. Shah**
*Department of Mathematics, **Department of Physics
Faculty of Science, The M.S. University of Baroda, Vadodara 390002

- **Abstract:**

Code simulation is crucial for scientific research, as it enables scientists to create virtual models that mimic real-world systems and phenomena. It also helps scientists test hypotheses and validate theories before conducting costly experiments. Simulations can be used to generate synthetic data for analysis, predict future outcomes, explore "what-if" scenarios, and assess the potential impacts of different factors or interventions. They provide an accessible platform for researchers worldwide. Stimulating space events on computer programs can have several benefits, including enhancing education and training, supporting research and development, lowering risk, improving cost effectiveness, and increasing public participation. Simulations of space events enable astronauts, engineers, and scientists to have immersive training experiences, hone their abilities to manage space missions, and practice the complexity and difficulties of space habitats. These simulations support the design of future space missions, test hypotheses, and understand the behaviour of space systems. Public engagement and outreach can be achieved through virtual simulations, visualizations, and games that create excitement and curiosity about space, encouraging interest in STEM fields. Here in the study and analyses of three programs has been done where stimulation real life space missions and space event has been done and also conducted kinematic analysis. The study aims to advance the knowledge of Python glow-script and underlying physics.

- **Keywords:** Near Earth Asteroids, DART, Artemis-1, Asteroids, Python stimulation, DART stimulation, Artemis-1 stimulation, Near Earth Asteroid stimulation.

- **Introduction:**

Scientists use code simulation to build computational representations of real-world systems and phenomena, which is essential for scientific research. It enables researchers to generate and analyze data, generate hypotheses for testing, improve reproducibility, and optimize designs. Simulations can produce artificial data for analysis, forecast and predict future results, investigate "what-if" scenarios, and determine the potential effects of various factors or interventions. Additionally, they offer researchers from around the world an accessible platform. The use of computer programs to simulate space events can be advantageous in many ways, including improving training and education, advancing research and development, reducing risk, enhancing cost effectiveness, and fostering greater public involvement. This report analyses three programs: Visualizing the Near-Earth Asteroid & Calculating Deflection Angle, Replicating DART: Double Asteroid Redirection Test, and Replicating ARTEMIS 1 (Exploration Mission-1 (EM-1)). The code developed successfully stimulated three real-life previously mentioned events, providing valuable insights into understanding interactions between celestial bodies using Python. Future research directions include replication studies, artificial intelligence and autonomous systems, resilient coding and fault tolerance, and advancing the knowledge base on understanding interactions between celestial bodies using Python.

- **Theory and Algorithm Discussion:**
- **Program 1: Visualizing the Near-Earth Asteroid & Calculating Deflection Angle**

1. **Overview**: This study investigates the movement of Asteroid 2023 BU, which passes by Earth 100 times closer than the moon and 10 times closer than geostationary satellites. The goal is to explore resource potential, identify dangers, develop efficient asteroid deflection and mitigation techniques, and advance human space exploration by studying and tracking these asteroids. The program stimulates a near-Earth asteroid, displaying graphs indicating energy change, deflection angle calculation, and ultimate velocity after deflection. [1]

2. **Programing Algorithm**

   Break the program into small time steps

   1) Find the position vector r (position vector between asteroid and earth).
   2) Calculate the gravitational force F_G using r.
   $$F_G = G * \frac{(m1 * m2)}{r^2}$$
   3) Calculate/Update momentum using F_G.
   $$p_{final} = p_{initial} + F_G \times dt$$
   4) Find the new position using the updated momentum.
   $$x_{final} = x_{initial} + \left(\frac{\Delta p}{(m)}\right) \times \Delta t$$
   5) Calculate kinetic, potential, total energy and update the time.
   6) Repeat the steps from 1 to 5 for a particular time period.
   7) Calculate the final velocity of asteroid using mass and momentum.
   $$V_{final} = \frac{P_{final}}{m}$$
   8) Calculate the deflection angle using dot product.
   $$\theta = \cos^{-1}\left(\frac{(A \cdot B)}{(|A|\ |B|)}\right)$$

- **Program 2: Replicating DART: Double Asteroid Redirection Test**

1. **Overview:** Planetary defence involves identifying and detecting potential impacts from Near-Earth Objects (NEOs) within Earth's orbit. The DART mission aims to control an asteroid's course using a kinetic impactor. The spacecraft collided with the moonlet of Didymos, a binary asteroid system. The impact altered the moonlet's orbit, demonstrating the potential to divert asteroids that may threaten Earth. The program stimulates a twin asteroid system, causing dart spacecraft to collide.
   Two final paths are displayed, representing the asteroid's path without and with collision. Deflection angle and dimorphos orbital time period changes are calculated to minimize the risk of impact. [2]

2. **Programing Algorithm**

   Break the program into small time steps

   1. Find the position vector r (position vector between Didymos and Dimorphos)
   2. Calculate the gravitational force F_G using r.
   $$F_G = G * \frac{(m1 * m2)}{r^2}$$
   3. Calculate/Update momentum of both bodies using F_G.
   $$p_{final} = p_{initial} + F_G \times dt$$
   4. Find the new position for both bodies and the spacecraft using the updated momentum.
   $$x_{final} = x_{initial} + \left(\frac{\Delta p}{(m)}\right) \times \Delta t$$
   5. Apply the condition for change in momentum of the moon Dimorphos only after collision.
   $$p_{final} = p_{initial} + (F_G - F_{Collision}) \times dt$$
   6. Update the time.
   7. Repeat the steps from 1 to 5 for a particular time period.

8. Calculate the deflection angle using dot product.

$$\theta = \cos^{-1}\left(\frac{(A \cdot B)}{(|A|\ |B|)}\right)$$

9. Calculate the initial and final orbital time period of Dimorphos using the given formula

$$T = 2 \times \pi \times \sqrt{\frac{(r^3)}{(G \times (m_1 + m_2))}}$$

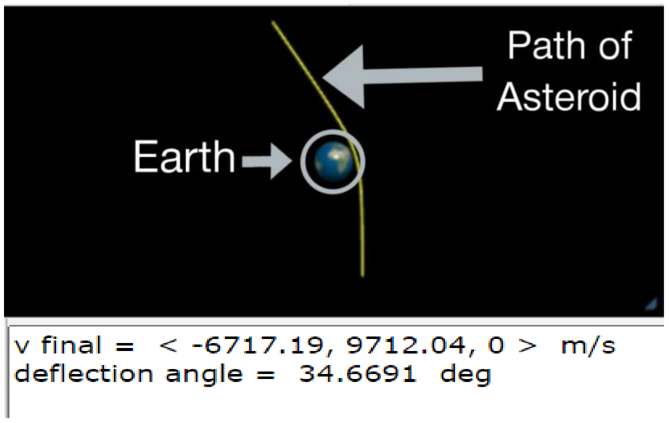
- **Program 3: Replicating ARTEMIS-1(Exploration Mission-1)**

1. **Overview:** Artemis-1 is NASA's first uncrewed mission to return humans to the Moon and establish a sustainable presence. It tests the Orion spacecraft's performance, capabilities, and safety in a deep space environment. The mission will also carry scientific instruments and payloads, gathering data for future crewed missions to the lunar surface. The Artemis-1 mission stimulation program has been modified, allowing Orion spacecraft to accurately direct towards the moon from Earth. The program has been extended to include lunar flyby and re-entry, allowing the Orion spacecraft to replicate its journey in the Artemis-1 mission. [3]

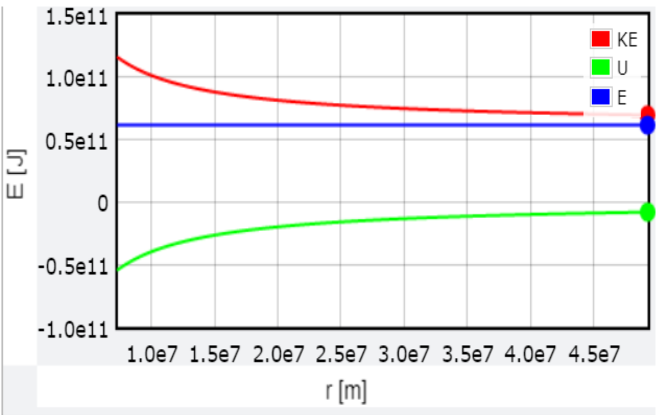
2. **Programing Algorithm**

   1. Give Orion's momentum direction towards moon.
   2. Break the program into small time steps.
   3. Calculate the force values for a particular distance and update the positions of moon, earth and Orion.
   4. Deflect Orion after a particular distance to get into moon's orbit.
   5. Continue calculating the force values for a particular distance while Orion is in the moon's orbit and update the positions of moon, earth and Orion.
   6. Deflect the Orion from moon's orbit after a particular time period and direct it towards earth. Continue calculating the force values while Orion is re-entering earth and update the positions of moon, earth and Orion.

- **Result and discussion:**
- **Output: Program 1: Visualizing the Near-Earth Asteroid & Calculating Deflection Angle**

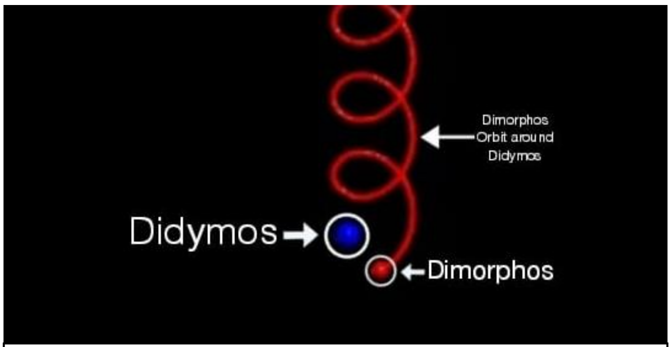


```
v final =  < -6717.19, 9712.04, 0 >  m/s
deflection angle =  34.6691  deg
```

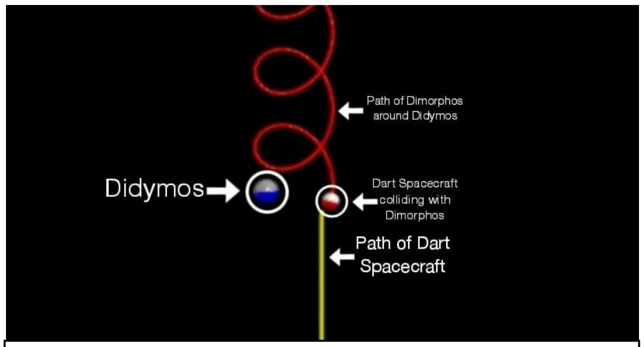


The program displays the trajectory of a near-earth asteroid, with a yellow line representing its path and a blue ball at its center representing Earth. It calculates the final velocity and deflection angle. Additionally, an energy diagram is added, calculating kinetic, potential, and total energy. The graph displays the change in energy of the asteroid as it passes near Earth, with red lines representing kinetic energy, blue lines representing total energy, and green lines representing potential energy.

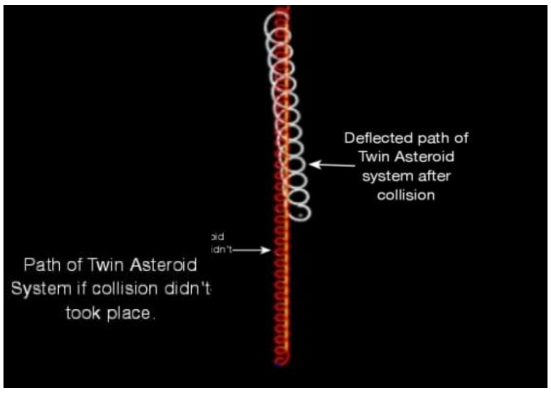
- **Output: Program 2: Replicating DART: Double Asteroid Redirection Test**

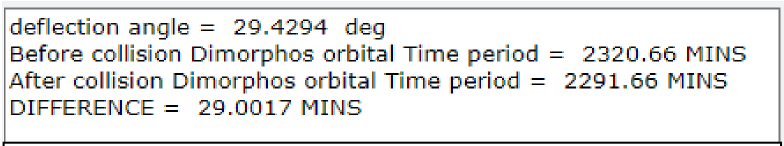


1.Straight path of Didymos (Blue ball) and Dimorphos (Red ball) system in space. Red spiral denotes the trail/path of Dimorphos around Didymos as the system is moving forward


Path of Dimorphos around Didymos
Didymos
Dart Spacecraft colliding with Dimorphos
Path of Dart Spacecraft

2.Time of collision of spacecraft with Dimorphos, yellow line indicates the trail/Path of spacecraft.

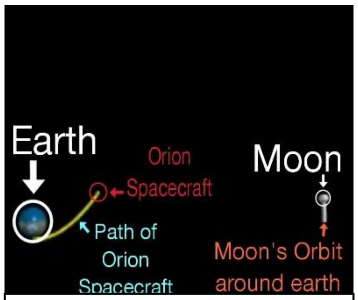

Deflected path of Twin Asteroid system after collision
Path of Twin Asteroid System if collision didn't took place



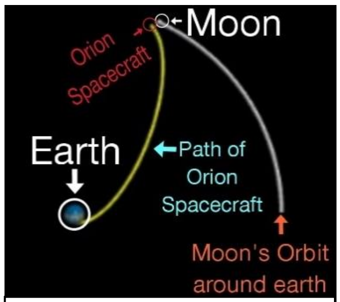
```
deflection angle =  29.4294  deg
Before collision Dimorphos orbital Time period =  2320.66 MINS
After collision Dimorphos orbital Time period =  2291.66 MINS
DIFFERENCE =  29.0017 MINS
```

3.Change in path of the system after the collision with the spacecraft is denoted with the white spiral, while the red spiral denotes the path which the asteroid system should have followed if there was no collision.Orbital time period as well as deflecvion angle has been calculated.

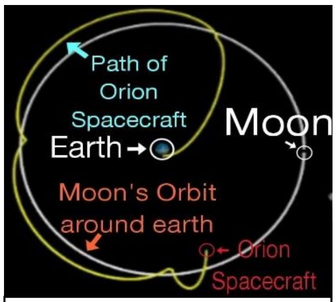
- **Output: Program 3: Replicating ARTEMIS-1(Exploration Mission-1)**



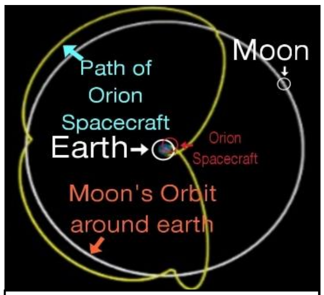
1.Launch directed to



2.Entering moon's orbit



3.Leaving moon's orbit



4.Re-entering Earth

The program displays the trajectory of Orion spacecraft launched to the moon, with yellow lines representing the path followed by the spacecraft, white lines representing the moon's path around Earth, and a blue ball at the centre representing Earth. The program stimulates Orion's path and directs it towards the moon. The program successfully stimulated Orion's path in the Artemis-1 mission, completing the four stages of the mission. The output shows the launch of Orion from Earth, reaching the moon's orbit, leaving the orbit, and re-entering Earth's atmosphere.

- **Conclusion:**

The code developed is able to successfully stimulate three real life scenarios which included Near-Earth Asteroid stimulation where the program is able to calculate energy change, final velocity and deflection angle of asteroid, three stages of DART mission was stimulated and four stages of ARTEMIS-1 mission has been stimulated.

- **Future Aspects:**
  This report has provided valuable insights into **Understanding interactions between celestial bodies using Python** and has stimulated three real life space events, there remain several avenues for future research that can extend and enhance our understanding of this field. The following are some key directions for future investigation:

    Replication Studies: Replicating this research with a larger and more diverse sample would strengthen the generalizability of our findings and increase stimulation accuracy.

    Artificial Intelligence and Autonomous Systems:
    As space missions venture further into the cosmos, the need for autonomous systems becomes paramount. Artificial intelligence (AI) can be integrated into spacecraft and rovers, enabling them to make crucial decisions independently while adapting to unforeseen circumstances. AI-driven coding can lead to improved data analysis, better risk assessment, and enhanced exploration capabilities, ultimately increasing the overall success of space missions.

    Resilient Coding and Fault Tolerance: Hardware malfunctions can be disastrous in space because it is an unforgiving environment. Researchers are working on resilient coding techniques that can withstand flaws and errors in hardware components in order to guarantee the success of long-duration space missions. To maintain mission integrity even in the face of hardware failures, sophisticated error correction codes, redundant systems, and fault-tolerant algorithms will be essential.

  By pursuing these future research directions, we can continue to expand the knowledge base on **Understanding interactions between celestial bodies using Python**, contributing to the advancement of the field and its practical applications.

REFERENCES

[1] M. Dunn, “Asteroid coming exceedingly close to Earth,” phys.org, 2023.

[2] NASA, “Double Asteroid Redirection Test-PRESS KIT”.

[3] NASA, “ARTEMIS 1- Press Kit”.