Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

Report for Mini Project in Computer Graphics - COMP 342

on

‘**escape**’

Submitted By:

Ranjan Lamsal (26)

Reewaj Khanal (61)

Submitted to:

Mr. Dhiraj Shrestha

Assistant Professor

Department of Computer Science and Engineering

School of Engineering

Kathmandu University

Dhulikhel, Kavre

Submission Date: Saturday 13 April 2024

Introduction:

The Escape project is a mini project in Computer Graphics, focusing on implementing a rocket launch simulation program using OpenGL and Pygame. This project delves into the technical aspects of 3D graphics rendering, simulation, and user interaction. By leveraging OpenGL's capabilities for rendering complex graphics and Pygame for managing user interface elements, the project provides a platform to explore the intricate mechanics of rocket propulsion and trajectory analysis. Through this endeavor, students gain practical experience in applying computer graphics concepts to simulate real-world phenomena, enhancing their understanding of graphics programming techniques and their applications.

Objectives

- To simulate the physics and mechanics of a rocket launch process.
- To provide users with a visually engaging and interactive experience.
- To educate users about rocket science and the physics of space exploration.
- To allow users to adjust parameters and observe the impact on the launch process.

Background Information

Since the dawn of the space age, rockets have served as our primary means of reaching space. From the pioneering days of the Apollo missions to the modern era of commercial spaceflight, rockets have evolved to become more powerful, efficient, and versatile. However, the fundamental principles that govern their operation remain the same, rooted in the laws of physics and the ingenuity of human ingenuity.

Rocket Physics

At its core, the physics of rocketry revolves around Newton's laws of motion and the principles of conservation of momentum and energy. The key concepts involved in rocket physics include:

- **Mass:** The mass of a rocket encompasses both its structural components and the fuel it carries. As fuel is burned during the launch, the rocket's mass decreases, affecting its acceleration and velocity.
- **Acceleration:** Acceleration is the rate of change of velocity over time and is crucial for propelling the rocket upward. It is determined by the net force acting on the rocket, which includes thrust generated by the engines, gravitational pull, and atmospheric drag.
- **Thrust:** Thrust is the force produced by the rocket's engines, propelling it in the opposite direction to the exhaust gases expelled. It is directly proportional to the rate of fuel consumption and the exhaust velocity of the propellant.
- **Trajectory:** The trajectory of a rocket describes its path through space, from liftoff to reaching its intended destination. Factors such as initial velocity, launch angle, gravitational pull, and atmospheric conditions influence the trajectory of a rocket.

Understanding these fundamental principles is essential for comprehending the challenges and complexities involved in launching a rocket into space. The Rocket Launch Simulation Program aims to provide users with a hands-on experience of these concepts, allowing them to explore the physics of rocketry in a visually engaging and interactive manner. Through realistic simulations and dynamic visualizations, users will gain a deeper appreciation for the science and technology that make space exploration possible.

Implementation

Programming Languages:

- Python: Utilized for overall project development.
- OpenGL: A graphics rendering library for creating 3D visualizations of Earth's surface, rocket structure, and trajectory.
- Pygame: Used for managing the user interface and interactions, including graphical elements, user inputs, and audio.

Libraries:

- OpenGL: Handles complex graphics rendering.
- Pygame: Manages the user interface and interactions.
- Math: Provides mathematical functions for calculations involved in the simulation.

Functions and Features

The program consists of several functions responsible for rendering various elements and simulating the rocket launch process:

Rendering Functions

- **stars():** Renders stars in the background, creating a visually appealing night sky effect.

```
# Function to Display stars
def stars():
   glColor3f(1.0,1.0,1.0)
   glPointSize(2)
   glBegin(GL_POINTS)
   for s in range(50):
       x=random.randint(0,800)
       y=random.randint(0,800)
       glVertex2i(x,y)
   glEnd()
```

- **moon(radius):** Renders a moon with the specified radius, simulating its movement across the sky.

```
# Function to display moon
def moon(radius):
   global tx,ty,DEG2RAD
   glBegin(GL_POLYGON)
   for i in range(359):
       degInRad = i*DEG2RAD

glVertex2f(300+ty+math.cos(degInRad)*radius,500-tx+(math.sin(degInRad))*ra
dius)
   glEnd()
   tx=tx+0.1
   ty=ty+0.1
```

- **grass():** Renders grass on the ground, adding detail to the Earth's surface.

```
# Function to display grass
def grass():
   glColor3f(0.0,0.9,0.0)
   glPointSize(3)
   glBegin(GL_POINTS)
   for s in range(1000):
       x=random.randint(0,800)
       y=random.randint(0,250)
       glVertex2i(x,y)
   glEnd()
```

Control and Simulation Functions:

- **control():** Controls the rendering of different parts of the rocket launch process, including liftoff, ascent, and trajectory.

```
# Function to control the rendering parts of rocket launch
def control():
   global launch,sky_color,count,count1,tx,ty,fumes,DEG2RAD
   count1+=1
```

```
if(count1==775659):
    launch=1
elif (launch == 1 and (count1 == 805407 )):
    rocket_position()
elif (launch == 1 and count1 >= 1000000):
    Moving_Rocket()
```

- **Rocket_on_Ground():** Renders the initial scene with the rocket on the ground, showcasing the launch pad, rocket body, and boosters.
- **rocket_position():** Simulates the rocket's movement within Earth's atmosphere, displaying changes in its position and appearance.
- **Moving_Rocket():** Simulates the rocket's trajectory in lower Earth orbit, showcasing its continued ascent and orbit insertion.

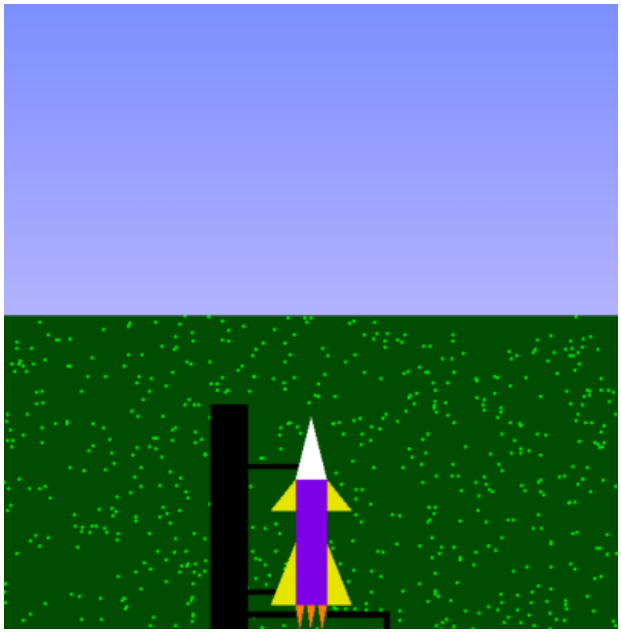
User Interaction Functions:

- **iterate():** Initializes the OpenGL environment and sets up the rendering parameters.
- **Showscreen():** Handles the display of the simulation screen, including the rendering of different scenes and updating the display based on the rocket's state.

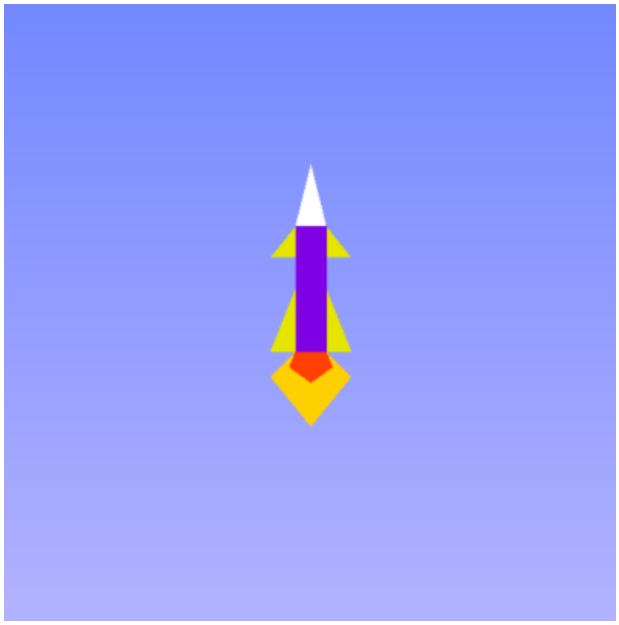
The implementation follows a modular approach, with separate functions responsible for different aspects of the simulation. The use of OpenGL ensures efficient rendering of 2D graphics, while Pygame facilitates user interaction and interface management. Overall, the implementation aims to provide a realistic and interactive simulation of a rocket launch process.

Output

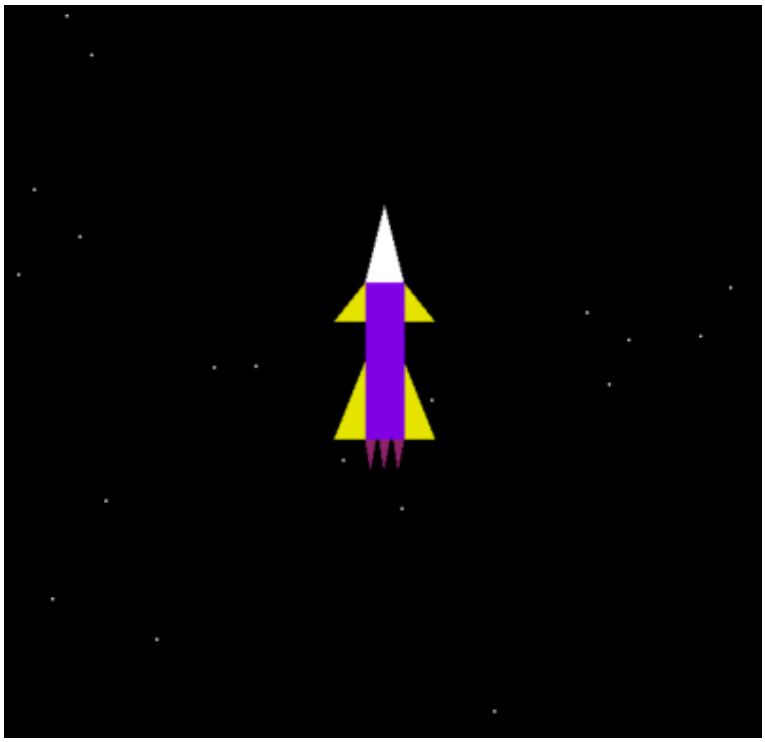
Rocket on Ground



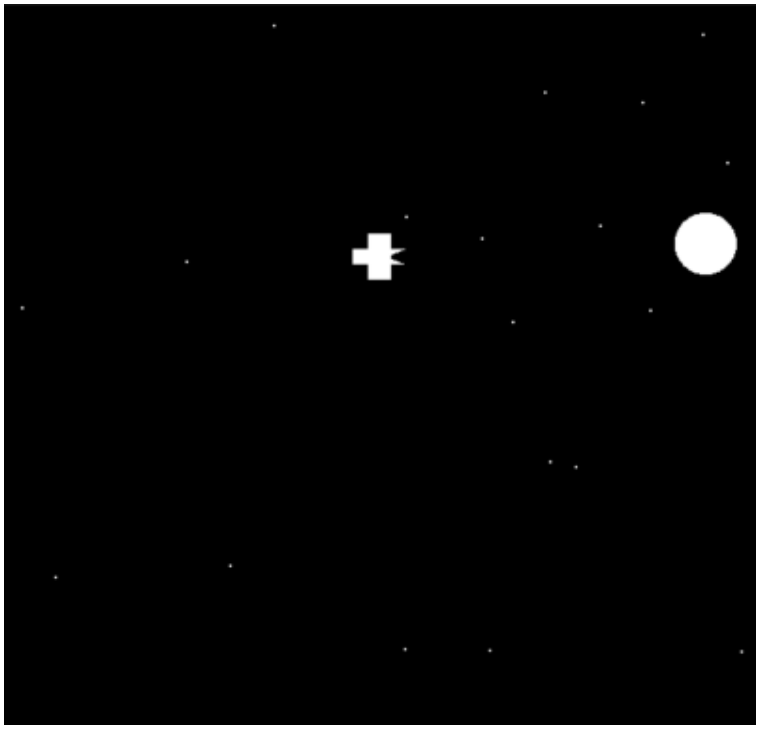
Rocket Space Trajectory



Rocket in Space



Rocket (Trunk Detatach)



Conclusion

The Rocket Launch Simulation Program aims to provide users with an immersive and educational experience of the rocket launch process. By combining realistic physics simulation with interactive visualization, the program offers insights into the complexities of space exploration and inspires curiosity about the wonders of the universe.

References

- Eerland, W. J., Box, S., Fangohr, H., & Sóbester, A. (2017). An open-source, stochastic, six-degrees-of-freedom rocket flight simulator, with a probabilistic trajectory analysis approach.
- Mehlhase, A. (2014). A Python framework to create and simulate models with variable structure in common simulation environments.
- Sells, R. (2020, March). Julia programming language benchmark using a flight simulation.