Университет ИТМО
Физико-технический мегафакультет
Физический факультет

Группа M3213 К работе допущен

Студент Губанов Константин Романович Работа выполнена

Преподаватель Хуснутдинова Наира Рустемовна Отчет принят

Рабочий протокол и отчет по моделированию 1 “Лунолет”

Вариант 17

1. Цель работы.

   1.1 Построить графики зависимости вертикальной скорости Vy, ускорения ay, и высоты H от времени, чтобы визуализировать динамику движения аппарата.

   1.2 Определить значение вертикальной скорости при достижении высоты 0 м (в момент посадки).

2. Задачи, решаемые при выполнении работы.

   2.1 Рассчитать скорость, высоту и ускорение на каждом шаге времени, используя уравнение Мещерского, которое учитывает истечение продуктов сгорания и изменение массы аппарата.

   2.2 Построить графики зависимости вертикальной скорости V_y, ускорения a_y, и высоты H от времени для анализа параметров движения.

   2.3 Определить вертикальную скорость аппарата в момент приземления на поверхность (когда высота становится равной нулю).

3. Объект исследования.

   Движение лунного аппарата с учетом действия сил гравитации Луны и управляемого реактивного двигателя, работающего на керосине и жидком кислороде.

4. Метод экспериментального исследования.

   Проведение численного моделирования движения аппарата по вертикальной оси с использованием уравнений движения, учитывающих изменение массы и сил, действующих на аппарат, с дальнейшей визуализацией результатов в виде графиков зависимости параметров движения от времени.

   Моделирование производится при помощи кода, написанного на языке Python и использовании библиотек NumPy и Matplotlib.

5. Рабочие формулы и исходные данные.
   Исходные данные:

   $g_л$ =1.62 м/с^2 - ускорение силы тяжести на Луне
   M = 2150 кг - масса корабля 2000 кг, плюс пилот в скафандре 150 кг
   V_p = 3660 м/с - скорость истечения продуктов сгорания
   m = 150 кг - топливо и окислитель
   $\dot{m}$ = 15 кг/с - расход топлива двигателем
   H_0 = 2000м – начальная высота
   V_{0y} = 29 м/с – начальная вертикальная скорость
   V_{max} = 3 м/с – максимальная допустимая скорость при посадке

Уравнения кинематики (свободное падение):

Уравнение скорости:

$$V = V0 + g \cdot t$$

Уравнение высоты:

$$H = H0 - V0 \cdot t - \frac{1}{2} g \cdot \mathrm{t}^2$$

Уравнения движения ракеты с переменной массой (уравнение Мещерского):

Уравнение тягового ускорения:

$$athrust = -\frac{Vp \cdot \dot{m}}{m}$$

Уравнение полного ускорения:

$$atotal = athrust - g$$

Изменение параметров корабля при управляемой посадке:

Уравнение скорости:

$$Vnew = Vcur + atotal \cdot \Delta t$$

Уравнение высоты:

$$Hnew = Hcur - Vcur \cdot \Delta t - \frac{1}{2} atotal \cdot (\Delta t)^2$$

Уравнение массы:

$$mnew = mcur - \dot{m} \cdot \Delta t$$

6. Код программы

Используемые библиотеки и исходные параметры

```
import numpy as np
import matplotlib.pyplot as plt

# Исходные параметры
g_L = 1.62  # ускорение на Луне, м/с^2
M = 2150  # масса аппарата без топлива, кг
m = 150  # начальная масса топлива, кг
Vp = 3660  # скорость истечения продуктов сгорания, м/с
m_ = 15  # расход топлива, кг/с
H0 = 2000  # начальная высота, м
V0y = 29  # начальная вертикальная скорость, м/с (вниз положительная)
Vmax = 3  # допустимая скорость при посадке, м/с
dt = 0.1  # Шаг времени, с
```

Функция, моделирующая свободное падение аппарата с выключенным двигателем

```
def simulate_free_fall(H0, V0, gravity, dt):
    time, height, velocity = [0], [H0], [V0]
    while height[-1] > 0:
        V = velocity[-1] + gravity * dt
        H = height[-1] - velocity[-1] * dt - 0.5 * gravity * dt ** 2
        time.append(time[-1] + dt)
        velocity.append(V)
        height.append(max(H, 0))  # Ограничиваем высоту до нуля
        if H <= 0:
            break
    return np.array(time), np.array(height), np.array(velocity)
```

Функция, симулирующая движение аппарата при включенном двигателе (используется уравнение Мещерского)

```
def simulate_powered_descent(H0, V0, initial_mass, gravity, exhaust_velocity, fuel_rate, max_speed, dt):
    time, height, velocity, mass, acceleration = [0], [H0], [V0], [initial_mass], []
    while height[-1] > 0 and mass[-1] > M:
        m_current = mass[-1]
        thrust_accel = -exhaust_velocity * fuel_rate / m_current
        total_accel = thrust_accel - gravity
        V = velocity[-1] + total_accel * dt
        H = height[-1] - velocity[-1] * dt - 0.5 * total_accel * dt ** 2
        m_new = max(m_current - fuel_rate * dt, M)
        time.append(time[-1] + dt)
        velocity.append(V)
        height.append(max(H, 0))
        mass.append(m_new)
        acceleration.append(total_accel)
        if H <= 0 or V <= max_speed:
            height[-1] = 0
            break
    return np.array(time), np.array(height), np.array(velocity), np.array(acceleration), np.array(mass)
```

Функция, которая вычисляет высоту, на которой нужно включить двигатель для безопасной посадки

```
def find_engine_start_point(t_free_fall, H_free_fall, V_free_fall, base_mass, fuel_mass, dt):
    for idx in reversed(range(len(t_free_fall))):
        if H_free_fall[idx] <= 0:
            continue
        H_engine_on, V_engine_on = H_free_fall[idx], V_free_fall[idx]
        total_mass = base_mass + fuel_mass
        t_powered, H_powered, V_powered, a_powered, m_powered = simulate_powered_descent(
            H_engine_on, V_engine_on, total_mass, g_L, Vp, m_, Vmax, dt
        )
        if V_powered[-1] <= Vmax:
            return idx, t_powered, H_powered, V_powered, a_powered
    return None, None, None, None, None
```

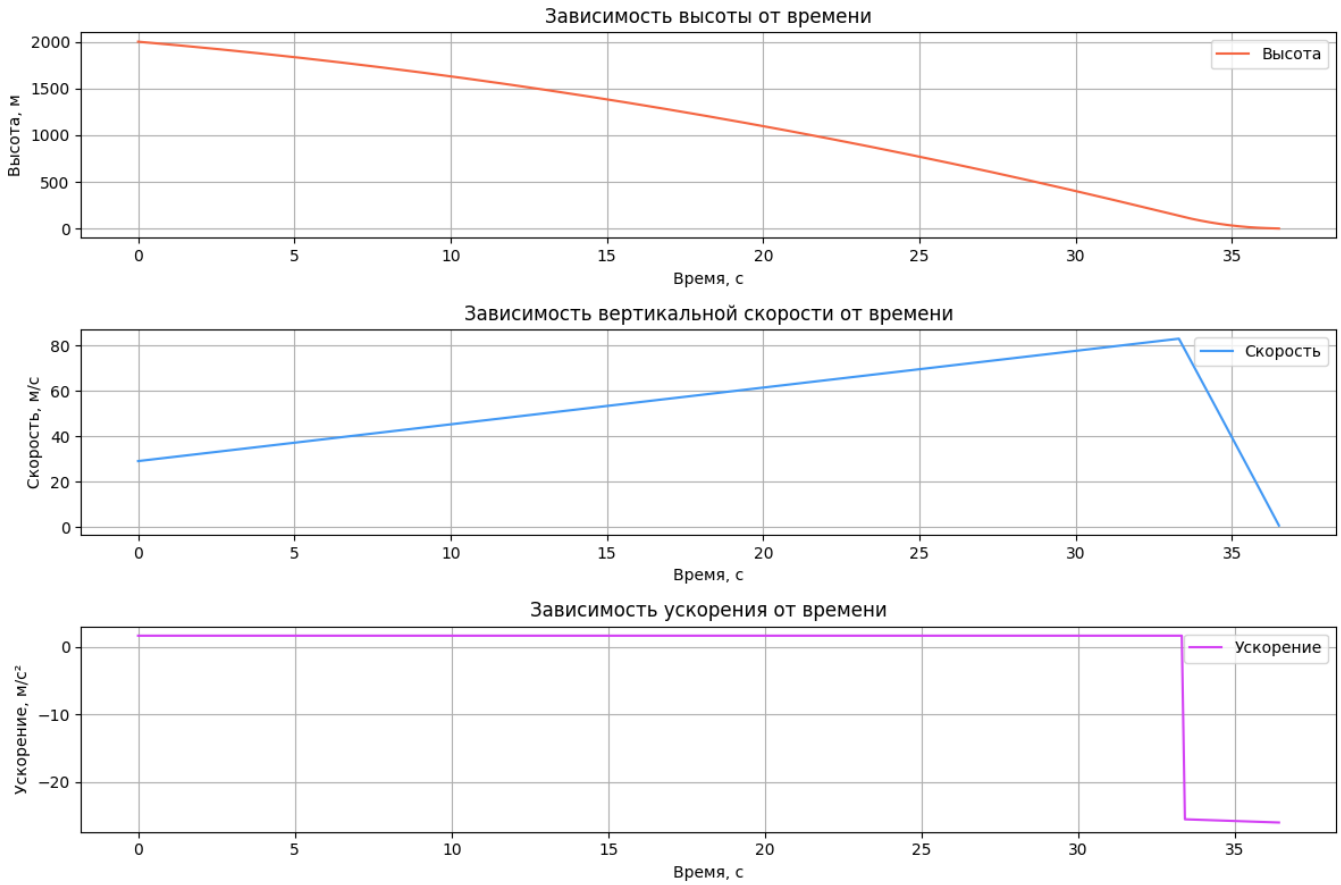
Функция, отвечающая за построение графиков

```
def plot_results(time, height, velocity, acceleration):
    plt.figure(figsize=(12, 8))

    plt.subplot(3, 1, 1)
    plt.plot(time, height, color='#f56642', label='Высота')
    plt.title('Зависимость высоты от времени')
    plt.xlabel('Время, с')
    plt.ylabel('Высота, м')
    plt.grid(True)
    plt.legend()

    plt.subplot(3, 1, 2)
    plt.plot(time, velocity, color='#4299f5', label='Скорость')
    plt.title('Зависимость вертикальной скорости от времени')
    plt.xlabel('Время, с')
    plt.ylabel('Скорость, м/с')
    plt.grid(True)
    plt.legend()

    plt.subplot(3, 1, 3)
    plt.plot(time[:-1], acceleration, color='#d442f5', label='Ускорение')
    plt.title('Зависимость ускорения от времени')
    plt.xlabel('Время, с')
    plt.ylabel('Ускорение, м/с²')
    plt.grid(True)
    plt.legend()

    plt.tight_layout()
    plt.show()
```

Точка входа программы и исполнение основного кода

```
def main():
    t_free_fall, H_free_fall, V_free_fall = simulate_free_fall(H0, V0y, g_L, dt)

    engine_idx, t_powered, H_powered, V_powered, a_powered = find_engine_start_point(
        t_free_fall, H_free_fall, V_free_fall, M, m, dt
    )

    if engine_idx is not None:
        t_total = np.concatenate((t_free_fall[:engine_idx + 1], t_free_fall[engine_idx] +
t_powered[1:]))
        H_total = np.concatenate((H_free_fall[:engine_idx + 1], H_powered[1:]))
        V_total = np.concatenate((V_free_fall[:engine_idx + 1], V_powered[1:]))
        a_total = np.concatenate((np.full(engine_idx + 1, g_L), a_powered[:-1]))

        plot_results(t_total, H_total, V_total, a_total)

        print(f"Высота на которой нужно включить двигатель: {H_free_fall[engine_idx]:.2f}
м")
        print(f"Вертикальная скорость при посадке: {V_total[-1]:.2f} м/с")
    else:
        print("Безопасная посадка невозможна")


if __name__ == "__main__":
    main()
```

7. Графики



8. Окончательные результаты.

8.1 На основе численного моделирования движения аппарата по вертикальной оси были построены графики зависимости высоты H, вертикальной скорости V_y и ускорения a_y от времени.

8.2 Итоговая вертикальная скорость аппарата при посадке составила примерно 0.6 м/с, двигатель нужно будет включить на высоте примерно 136.1 м.

8.3 Достигнутая вертикальная скорость значительно меньше безопасной посадочной скорость в 3 м/с, что говорит о возможности безопасного приземления.

9. Выводы и анализ результатов работы.

9.1 Построены графики, демонстрирующие изменение высоты, скорости и ускорения аппарата при его спуске, включая моменты свободного падения и работы двигателя.

9.2 Высота для безопасного включения двигателя составляет примерно **136.1 м**. Это позволяет начать торможение на оптимальной высоте для снижения посадочной скорости.

9.3 Вертикальная скорость аппарата при посадке составила около **0.6 м/с**, что значительно ниже допустимых 3 м/с, обеспечивая безопасную посадку.

Модель показала, что при заданных параметрах аппарат может безопасно приземлиться, если включить двигатель на рассчитанной высоте.