Цель работы: изучить и смоделировать вращательное движение твёрдого тела, заданного динамическими и кинематическими уравнениями Эйлера.

1 Теоретические сведения

Кинематика вращательного движения

Кватернионы: Вращение описывается с помощью кватернионов $\mathbf{q} = q_0 + q_1 i + q_2 j + q_3 k$, где i, j, k — кватернионные единицы. Код использует кватернионы для представления ориентации тела (quaternion_to_rotation_matrix, quaternion_multiply).

Уравнения Пуассона: Эволюция кватерниона ориентации описывается уравнениями:

$$\frac{d\mathbf{q}}{dt} = \frac{1}{2}\mathbf{q} \circ \boldsymbol{\omega},$$

где $\boldsymbol{\omega}$ — кватернион угловой скорости ($\omega_0 = 0$, ω_1, ω_2, ω_3 — компоненты угловой скорости). В коде это реализовано численно в функции solve_numeric.

Углы Эйлера (ψ, θ, ϕ):

- ψ — угол прецессии (вращение вокруг оси i_3),
- θ — угол нутации (вращение вокруг линии узлов i'),
- ϕ — угол собственного вращения (вращение вокруг оси e_3).

Угловая скорость $\boldsymbol{\omega}$:

$$\boldsymbol{\omega} = \dot{\psi}\, i_3 + \dot{\theta}\, i' + \dot{\phi}\, e_3$$

Проекции угловой скорости (p, q, r) на оси связанной системы координат (e_1, e_2, e_3):

$$p = \boldsymbol{\omega} \cdot e_1 = \dot{\psi} \sin\theta \sin\phi + \dot{\theta} \cos\phi$$

$$q = \boldsymbol{\omega} \cdot e_2 = \dot{\psi} \sin\theta \cos\phi - \dot{\theta} \sin\phi$$

$$r = \boldsymbol{\omega} \cdot e_3 = \dot{\psi} \cos\theta + \dot{\phi}$$

Кинематические уравнения Эйлера: Производные углов Эйлера выражаются через проекции угловой скорости:

$$\dot{\psi} = \frac{p \sin\phi + q\cos\phi}{\sin\theta}, \quad \dot{\theta} = p\cos\phi - q\sin\phi, \quad \dot{\phi} = r - (p\sin\phi + q\cos\phi)\cot\theta$$

Эти уравнения имеют особенность при $\theta = 0$ (линия узлов не определена).

Теорема Эйлера о конечном повороте: Любое вращение можно представить как поворот вокруг оси $\mathbf{e}$ на угол ϕ:

$$\mathbf{r}' = A\mathbf{r}A^{-1},$$

где $\mathbf{r}$ и $\mathbf{r}'$ — начальное и конечное положение вектора, A — кватернион поворота: $A = \cos\left(\frac{\phi}{2}\right) + \mathbf{e}\sin\left(\frac{\phi}{2}\right)$.

Кинематический винт: Комбинация поступательного движения со скоростью $\mathbf{v}$ и вращательного движения с угловой скоростью $\boldsymbol{\omega}$. Ось винта параллельна $\boldsymbol{\omega}$, а скорость точек на оси винта равна $\mathbf{v}$.

Динамика вращательного движения

- **Момент инерции:** Тензор $\mathbf{I}$ описывает распределение массы тела и его сопротивление вращению. В коде моменты инерции задаются параметром I_values.
- **Момент импульса:**
$$\mathbf{L} = \mathbf{I}\boldsymbol{\omega}.$$
В коде вычисляется функцией compute_angular_momentum.
- **Кинетическая энергия:**
$$T = \frac{1}{2}\boldsymbol{\omega} \cdot \mathbf{I}\boldsymbol{\omega}.$$
В коде вычисляется функцией compute_energy.

  В случае главных осей инерции:
$$T = \frac{1}{2}\left(Ap^2 + Bq^2 + Cr^2\right)$$
где A, B, C — главные моменты инерции, p, q, r — проекции $\boldsymbol{\omega}$ на главные оси.
- **Динамические уравнения Эйлера**

  Общие уравнения: Основаны на теореме об изменении момента импульса:
$$J\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (J\boldsymbol{\omega}) = \boldsymbol{M}$$
где J — тензор инерции, $\boldsymbol{\omega}$ — угловая скорость, $\boldsymbol{M}$ — момент внешних сил. В коде эти уравнения решаются численно в функции solve_numeric.

  Уравнения Эйлера в проекциях на главные оси инерции:
$$A\dot{p} + (C - B)qr = M_1$$
$$B\dot{q} + (A - C)rp = M_2$$
$$C\dot{r} + (B - A)pq = M_3$$
где M_1, M_2, M_3 — проекции момента внешних сил на главные оси.
- **Случай Эйлера (свободное вращение):** $\mathbf{M} = 0$. В коде моделируется при $M_{\text{ext}} = [0, 0, 0]$. Два интеграла движения: $|\mathbf{L}| = \text{const}$, $T = \text{const}$.
- **Случай Лагранжа (тяжелый симметричный волчок):**
$$\mathbf{M} = m\mathbf{g}l \times \mathbf{e},$$
где m — масса, $\mathbf{g}$ — ускорение свободного падения, l — расстояние от точки опоры до центра масс, $\mathbf{e}$ — ось симметрии.

2 Описание работы программы

Программа моделирует вращение твердого тела в трёхмерном пространстве, используя кватернионы для представления ориентации и численный метод Рунге-Кутты 4-го порядка для решения уравнений движения. На выходе программа отображает анимацию вращения тела.

Описание функций

1. `quaternion_to_rotation_matrix(q)`
Описание: Преобразует кватернион в матрицу вращения.
Аргументы:

- `q (numpy.ndarray)`: Кватернион в виде массива NumPy $[q_0, q_1, q_2, q_3]$.

Возвращаемое значение:

- `numpy.ndarray`: Матрица вращения 3×3.

2. `quaternion_multiply(q, p)`
Описание: Выполняет умножение двух кватернионов.
Аргументы:

- `q (numpy.ndarray)`: Первый кватернион $[q_0, q_1, q_2, q_3]$.
- `p (numpy.ndarray)`: Второй кватернион $[p_0, p_1, p_2, p_3]$.

Возвращаемое значение:

- `numpy.ndarray`: Результирующий кватернион.

3. `rk4_step(q, w, dt, torque, I, I_inv)`
Описание: Выполняет один шаг интегрирования методом Рунге-Кутты 4-го порядка для вычисления нового кватерниона ориентации и новой угловой скорости.
Аргументы:

- `q (numpy.ndarray)`: Кватернион ориентации.
- `w (numpy.ndarray)`: Угловая скорость.
- `dt (float)`: Шаг по времени.
- `torque (numpy.ndarray)`: Момент сил.
- `I (numpy.ndarray)`: Тензор инерции.
- `I_inv (numpy.ndarray)`: Обратный тензор инерции.

Возвращаемое значение:

- `tuple`: Новый кватернион ориентации и новая угловая скорость.

4. `solve_numeric(q0, w0, I_values, r, F, M_ext, t_max, dt)`
Описание: Численно решает уравнения движения вращающегося тела с помощью метода Рунге-Кутты 4-го порядка. Вычисляет кватернионы ориентации, угловые скорости и энергии (потенциальную, кинетическую и полную) в каждый момент времени.
Аргументы:

- `q0 (numpy.ndarray)`: Начальный кватернион ориентации.
- `w0 (numpy.ndarray)`: Начальная угловая скорость.
- `I_values (list)`: Список значений моментов инерции $[I_x, I_y, I_z]$.

- `r (numpy.ndarray)`: Радиус-вектор точки приложения силы относительно центра масс.
- `F (numpy.ndarray)`: Вектор силы.
- `M_ext (numpy.ndarray)`: Вектор внешнего момента сил.
- `t_max (float)`: Максимальное время моделирования.
- `dt (float)`: Шаг по времени.

Возвращаемое значение:

- `tuple`: Кортеж, содержащий массивы времени, кватернионов, угловых скоростей, потенциальной, кинетической и полной энергии.

5. `compute_angular_momentum(w, I)`
Описание: Вычисляет момент импульса.
Аргументы:

- `w (numpy.ndarray)`: Угловая скорость.
- `I (numpy.ndarray)`: Тензор инерции.

Возвращаемое значение:

- `numpy.ndarray`: Момент импульса.

6. `compute_kinetic_energy(w, I)`
Описание: Вычисляет кинетическую энергию вращения.
Аргументы:

- `w (numpy.ndarray)`: Угловая скорость.
- `I (numpy.ndarray)`: Тензор инерции.

Возвращаемое значение:

- `float`: Кинетическая энергия.

7. `update_plot(frame, q, w, ax, I, potential_energy, kinetic_energy, total_energy)`
Описание: Обновляет 3D-график на каждом кадре анимации, отображая вращение тела и выводя значения момента импульса и энергий.
Аргументы:

- `frame (int)`: Номер кадра.
- `q (numpy.ndarray)`: Массив кватернионов.
- `w (numpy.ndarray)`: Массив угловых скоростей.
- `ax (matplotlib.axes.Axes)`: Оси для построения графика.
- `I (numpy.ndarray)`: Тензор инерции.
- `potential_energy (numpy.ndarray)`: Массив потенциальной энергии.
- `kinetic_energy (numpy.ndarray)`: Массив кинетической энергии.
- `total_energy (numpy.ndarray)`: Массив полной энергии.

Связь кода и описания

Код предоставляет численное решение уравнений движения. Анимация визуализирует вращение, а графики позволяют анализировать динамику. Описание предоставляет теоретическую основу для понимания результатов.

Ограничения кода

Код моделирует только вращение с неподвижной точкой и не учитывает упругие деформации, использует численное интегрирование.

2.1 Вывод результатов в консоль

Результаты работы программы представлены ниже.

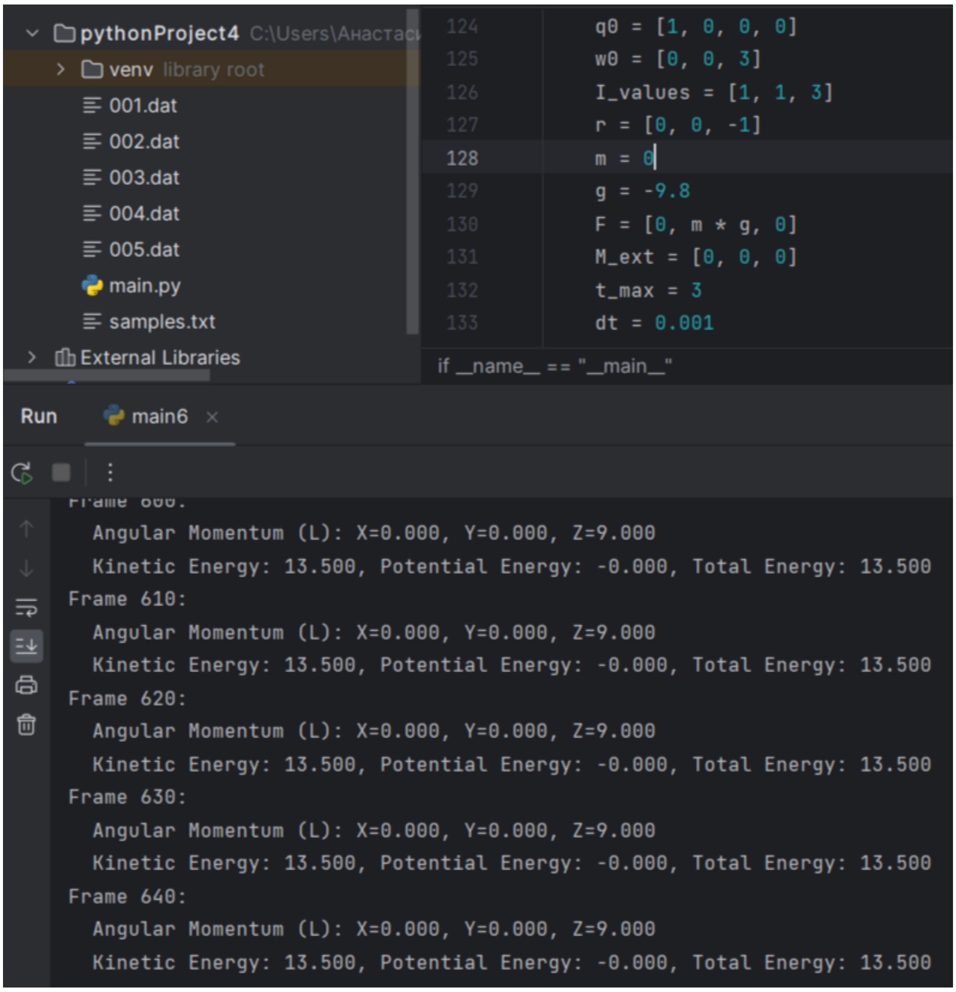

Рис. 1: Случай Эйлера.

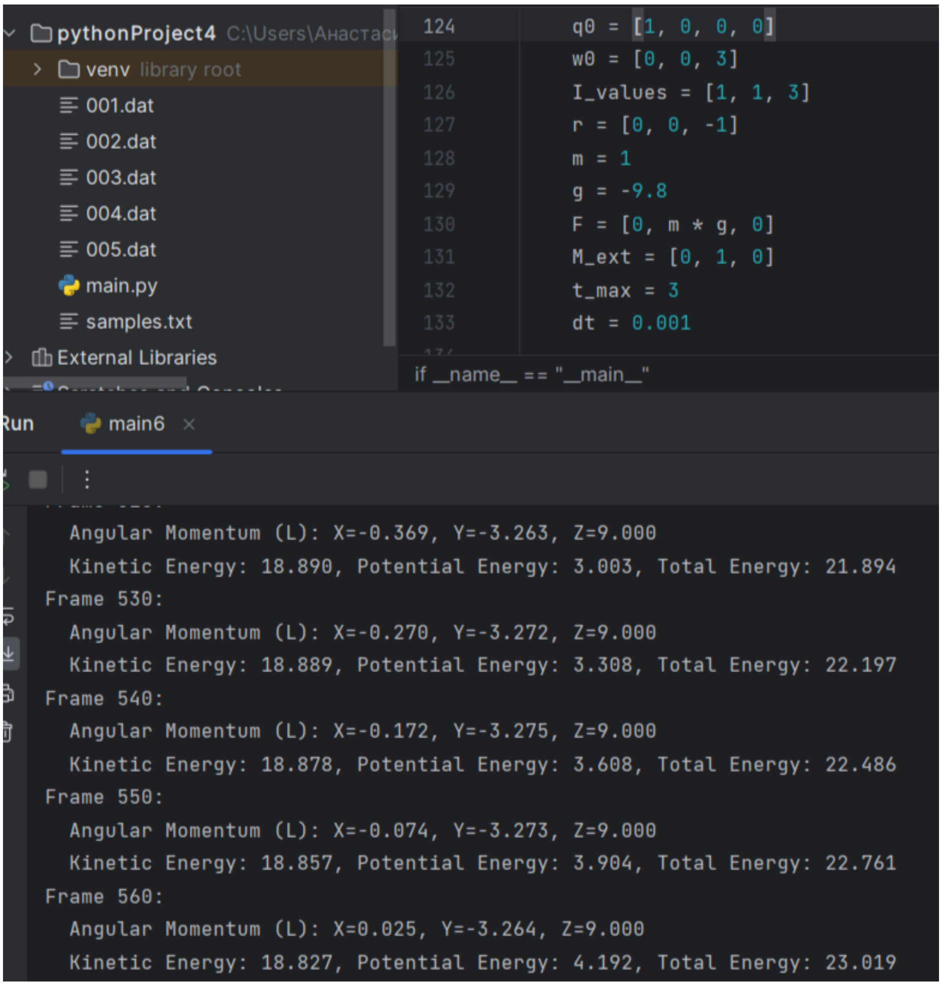

Рис. 2: Общий случай

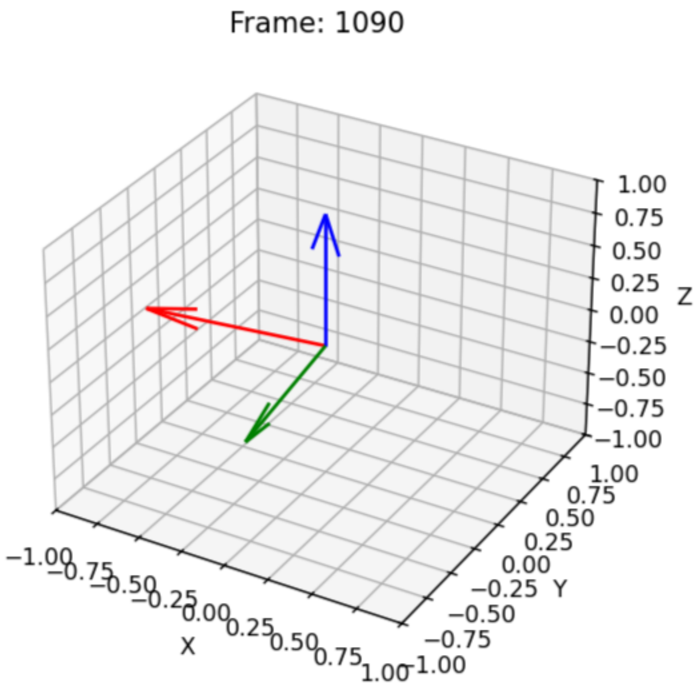

Рис. 3: 3D-модель

3 Итоги

В данной работе проведено моделирование процесса вращения твердого тела, основанное на кинематических и динамических уравнениях Эйлера. Была разработана программа на языке Python, реализующая численное решение этих уравнений с использованием метода Рунге-Кутта. Полученное решение позволило визуализировать вращение тела в виде анимации, демонстрирующей изменение его ориентации во времени.

Анимация наглядно проиллюстрировала влияние начальных условий на характер вращения. Было показано, что при отсутствии внешних моментов (случай Эйлера) величина момента импульса и кинетическая энергия остаются постоянными, а ось вращения прецессирует. В случае наличия внешнего момента (случай Лагранжа) наблюдалось нутационное движение.

В целом, работа подтвердила применимость уравнений Эйлера для моделирования вращательного движения.