Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
УЧЕБНЫЙ ЦЕНТР ОБЩЕЙ ФИЗИКИ ФТФ

УНИВЕРСИТЕТ ИТМО

Группа M32081 К работе допущен

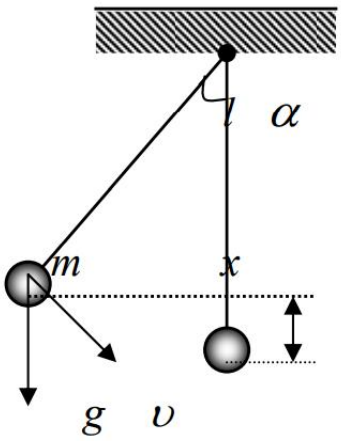
Студент Эседулаева З.А. Работа выполнена

Преподаватель Рахманова Г.Р. Отчет принят

Рабочий протокол и отчет по моделированию №1

Моделирование физического маятника

Задача:



Пусть на неподвижном шарнире подвешен маятник – груз массы m, находящийся на конце стержня длины l.
Шарнир считается идеально гладким в том смысле, что в нем не происходят потери энергии на трение. Стержень считается невесомым и абсолютно жестким, т.е. его кинетическая и потенциальная энергии равны нулю, а груз не может совершать движений вдоль оси стержня.
Груз имеет небольшие размеры по сравнению с длиной стержня (материальная точка); ускорение свободного падения g постоянно.

Рабочие формулы:

Колебания маятника под действием сил тяжести:

$$\frac{d^2\alpha}{dt^2} = -\frac{g}{l}\sin\alpha$$

Частота собственных колебаний:

$$\omega_0^2 = \frac{g}{l}$$

Учитывая трение на шарнирах:

$$\frac{d^2\alpha}{dt^2} = -\omega_0^2 \sin\alpha - \gamma\frac{d\alpha}{dt},$$

где коэффициент затухания γ представляет меру тормозящей силы.

Исходные данные:

$$g = 9.8\ \frac{\text{м}}{\text{с}^2}$$
$$\gamma = 0.6$$
$$l = 1.0\ \text{м}$$

Решение поставленных задач:

Введём угловую скорость $\omega(t) = \frac{d\alpha}{dt}$.

Тогда дифференциальное уравнение можно представить как:

$$\omega'(t) = -\omega_0 \sin\alpha - \gamma\omega(t)$$

Такое ДУ решается в рабочем коде с помощью функции *odeint* в подключенной библиотеке *scipy.integrate*:

```
def differential(self, angles, t):
    self.graph_data[0].append(self.time_elapsed)
    self.graph_data[1].append(angles[0])
    self.graph_data[2].append(angles[1])

    return [self.get_angular_velocity(),
            - g / self.length * sin(self.get_alpha()) - self.stopping_force * self.get_angular_velocity()]

def calculate_position(self):
    x = np.cumsum([self.center[0], self.length * sin(self.get_alpha())])
    y = np.cumsum([self.center[1], - self.length * cos(self.get_alpha())])
    return (x, y)

def update(self, dt):
    self.angles = diff.odeint(self.differential,
                              self.angles,
                              [0, dt])[1]
    self.time_elapsed += dt
```

Графики строятся по результатам работы программы. Демонстрацию работы физического маятника можно посмотреть в видео pendulum_demonstration.mp4.

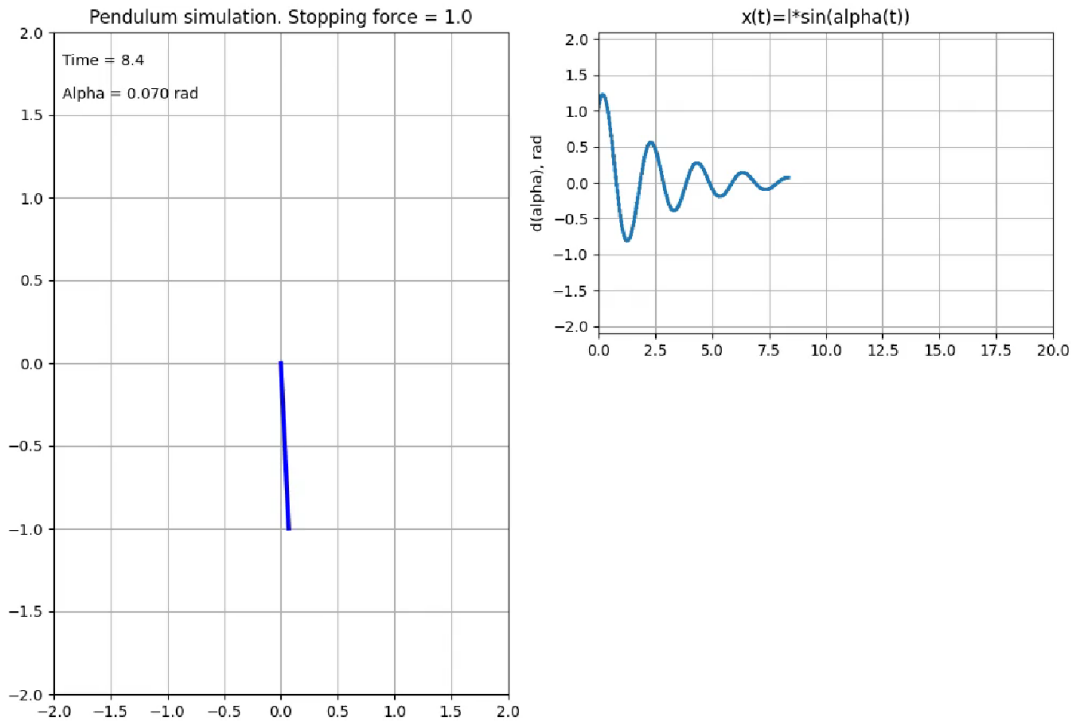
Сравнение периодов (п.6):

Без тормозящей силы за 12 секунд было совершено 6,5 колебаний.
С тормозящей силой за то же время было совершено 5 колебаний.

$$T_1 = \frac{12}{6{,}5} = 1{,}85$$
$$T_2 = \frac{12}{5} = 2{,}4$$
$$T_1 < T_2$$

Результаты работы:

см. ***pendulum_demonstration.mp4*** для анимации полученных результатов
пример конечного кадра:



Выводы:

Была смоделирована работа физического маятника с тормозящей силой, полученный результат совпадает с предложенными выходными данными. Были вычислены периоды при уравнениях с тормозящей силой и без нее, а также было произведено их сравнение.