МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет систем управления и робототехники

Практическая работа №4
по дисциплине
«Имитационное моделирование робототехнических систем»

по теме:
«Управление суставами в MuJoCo»

Студент:
Группа № R4136c Носов А.С.

Предподаватель:
Ассистент СУиР Ракшин Е.А.

Санкт-Петербург 2025

СОДЕРЖАНИЕ

ЦЕЛИ ВЫПОЛНЕНИЯ РАБОТЫ

Построение модели сустава в среде MuJoCo.
В работе решаются следующие задачи:

– Создание модели в среде MuJoCo с двигателями и датчиками;
– Создание ПД контроллера для слежения за траекторией;
– Моделирование, построение графиков.

1 ПОСТРОЕНИЕ МОДЕЛИ В СРЕДЕ MUJOCO

1.1 Условие задания

Система заданная вариантом

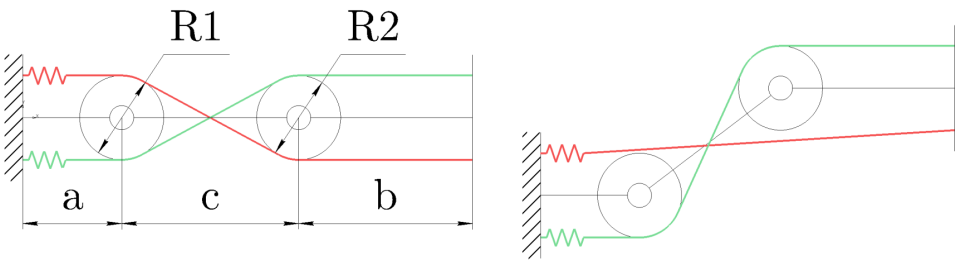

Рисунок 1 — Схема объекта

Параметры варианта:

- $R_1 = 0.05$м
- $R_2 = 0.033$м
- $a = 0.031$м
- $b = 0.091$м
- $c = 0.074$м

Параметры синусоидального сигнала, за которым производится слежение:

- $AMP_1 = 11.27°$
- $FREQ_1 = 3.91$Гц
- $BIAS_1 = 27.8°$
- $AMP_2 = 41.61°$
- $FREQ_2 = 1.41$Гц
- $BIAS_2 = 9.3°$

1.2 XML модель

Листинг 1.1 — Модель MuJoCo в XML

```
<mujoco>
  <default>
```

```
    <geom type="capsule" size="0.002"/>
    <site rgba="0 0 0 1"/>
    <joint type="hinge" axis="0 1 0" range="-180 180"/>
</default>

<asset>
    <texture type="skybox" builtin="gradient" rgb1="1 1 1" rgb2="0.5 0.5 0.5" width
 ="265" height="256"/>
    <texture name="grid" type="2d" builtin="checker" rgb1="0.1 0.1 0.1" rgb2="0.6
 0.6 0.6" width="300" height="300"/>
    <material name="grid" texture="grid" texrepeat="10 10" reflectance="0.2"/>
</asset>

<worldbody>
    <camera name="side view" pos="0.0 -1.5 1.0" euler="90 0 0" fovy="60"/>
    <light pos="0 0 1"/>
    <body name="wall_left" pos="0 0 0">
        <geom type="box" size="0.002 0.05 0.05" rgba="0.7 0.7 0.7 1"/>
    </body>

    <!-- A -->
    <body name="link_a" pos="0.002 0 0">
        <geom fromto="0 0 0 0.032 0 0"/>
        <site name="s1" pos="0 0 0.025"/>
        <site name="s2" pos="0 0 -0.025"/>

        <!-- R1 -->
        <body name="r1" pos="0.032 0 0">
            <!-- <joint name="j_p1"/> -->
            <geom name="r1" type="cylinder"
                fromto="0 0.025 0 0 -0.025 0"
                size="0.025" rgba="0 0 1 0.5"/>
            <site name="center_r1" pos="0 0 0" rgba="1 0 0 1"/>
            <site name="s3" pos="0 0 0.025"/>
            <site name="s4" pos="0 0 -0.025"/>

            <!-- C -->
            <body name="link_c" pos="0 0 0">

                <joint name="j_p1"/>
                <geom fromto="0 0 0 0.074 0 0"/>
                <site name="s5" pos="0.04 0 0"/>
```

```
            <!-- R2 -->
            <body name="r2" pos="0.074 0 0">
                <!-- <joint name="j_p2"/> -->
                <geom name="r2" type="cylinder"
                fromto="0 0.0165 0 0 -0.0165 0"
                size="0.0165" rgba="0 0 1 .5"/>
                <site name="center_r2" pos="0 0 0"  rgba="1 0 0 1"/>
                <site name="s6" pos="0 0 0.0165"/>
                <site name="s7" pos="0 0 -0.0165"/>
                <site name="center B" pos="0 0 0"/>

                <!-- B -->
                <body name="link_b" pos="0 0 0">
                    <joint name="j_p2"/>
                    <geom fromto="0 0 0 0.091 0 0"/>

                    <site name="s8" pos="0.091 0 0.0165"/>
                    <site name="s9" pos="0.091 0 -0.0165"/>

                    <body name="wall_right" pos="0.091 0 0">
                        <geom type="box" size="0.002 0.02 0.02" rgba="0.7 0.7 0.7 1"/
>
                        <site name="end_site" pos="0 0 0" size="0.005" rgba="1 1 0 1
"/>
                    </body>
                </body>
            </body>
        </body>
    </body>




</worldbody>

<tendon>
    <spatial name="tendon_1" stiffness="700" width="0.001" rgba="1 0 0 1"
 springlength="0.2" damping="15">
        <site site="s1"/>
        <!-- <site site="s3"/> -->
```

```
            <geom geom="r1" sidesite="s3"/>
            <site site="s5"/>
            <!-- <site site="s7"/> -->
            <geom geom="r2" sidesite="s7"/>
            <site site="s9"/>
        </spatial>
        <spatial name="tendon_2" stiffness="700" width="0.001" rgba="0 1 0 1"
     springlength="0.2" damping="15">
            <site site="s2"/>
            <!-- <site site="s4"/> -->
            <geom geom="r1" sidesite="s4"/>
            <site site="s5"/>
            <!-- <site site="s6"/> -->
            <geom geom="r2" sidesite="s6"/>
            <site site="s8"/>
        </spatial>
    </tendon>

    <sensor>
        <!-- <framepos objtype="site" objname="center_r1"/>
        <framepos objtype="site" objname="end_site"/> -->
        <!-- <jointpos joint="j_p1"/>
        <jointpos joint="j_p2"/> -->
        <jointpos name="j1_pos" joint="j_p1"/>
        <jointpos name="j2_pos" joint="j_p2"/>

        <jointvel name="j1_vel" joint="j_p1"/>
        <jointvel name="j2_vel" joint="j_p2"/>
    </sensor>

    <actuator>
        <motor name="motor_r1" joint="j_p1" gear="1"/>
        <motor name="motor_r2" joint="j_p2" gear="1"/>
    </actuator>

</mujoco>
```

2 МОДЕЛИРОВАНИЕ СИСТЕМЫ

Был написан python скрипт с ПД-регулятором для слежения за траекторией.

2.1 Python код

Листинг 2.1 — Скрипт симуляции

```
import mujoco.viewer
import matplotlib.pyplot as plt
import time
from math import sin
import numpy as np

model = mujoco.MjModel.from_xml_path('task_4_model.xml')
data = mujoco.MjData(model)

timestamp = 0.001
simul_time = 15.0

num_steps = int(simul_time / timestamp)


# variant data

AMP_1 = 11.27
FREQ_1 = 3.91
BIAS_1 = 27.8

AMP_2 = 41.61
FREQ_2 = 1.41
BIAS_2 = 9.3



def change_stiffness(stiffness):
    tendon_id_1 = mujoco.mj_name2id(model, mujoco.mjtObj.mjOBJ_TENDON, "
     tendon_1")
```

```
    tendon_id_2 = mujoco.mj_name2id(model, mujoco.mjtObj.mjOBJ_TENDON, "
     tendon_2")
    model.tendon_stiffness[tendon_id_1] = stiffness
    model.tendon_stiffness[tendon_id_2] = stiffness

def change_damping(damping):
    tendon_id_1 = mujoco.mj_name2id(model, mujoco.mjtObj.mjOBJ_TENDON, "
     tendon_1")
    tendon_id_2 = mujoco.mj_name2id(model, mujoco.mjtObj.mjOBJ_TENDON, "
     tendon_2")
    model.tendon_damping[tendon_id_1] = damping
    model.tendon_damping[tendon_id_2] = damping


# colors = ['red', 'green', 'blue', 'orange', 'purple']

def controller(pos, goal_pos, vel, kp, kd):
    return kp * (goal_pos - pos) - kd * vel


plt.figure(figsize=(12, 8))

time_steps = []
sensor_1_data = []
sensor_2_data = []
goal_pos_1_data = []
goal_pos_2_data = []
error_1_data = []
error_2_data = []

damp = 5
stiffness = 0.01
change_damping(damp)
change_stiffness(stiffness)

with mujoco.viewer.launch_passive(model, data) as viewer:
    for step in range(num_steps):
        mujoco.mj_step(model, data)

        goal_pos_1 = np.deg2rad(AMP_1*sin(FREQ_1*step*timestamp) + BIAS_1)
        goal_pos_2 = np.deg2rad(AMP_2*sin(FREQ_2*step*timestamp) + BIAS_2)
```

```
        # cur_pos_1 = data.sensordata[0]
        # cur_pos_2 = data.sensordata[1]
        cur_pos_1 = data.sensor("j1_pos").data[0]
        cur_pos_2 = data.sensor("j2_pos").data[0]
        cur_vel_1 = data.sensor("j1_vel").data[0]
        cur_vel_2 = data.sensor("j2_vel").data[0]
        # print(data.sensordata)

        data.ctrl[0] = controller(cur_pos_1, goal_pos_1, cur_vel_1, 5.0, 0.01)
        data.ctrl[1] = controller(cur_pos_2, goal_pos_2, cur_vel_2, 1.0, 0.001)

        goal_pos_1_data.append(goal_pos_1)
        goal_pos_2_data.append(goal_pos_2)
        sensor_1_data.append(cur_pos_1)
        sensor_2_data.append(cur_pos_2)
        error_1_data.append(goal_pos_1 - cur_pos_1)
        error_2_data.append(goal_pos_2 - cur_pos_2)

        #print(f"cur_pos_1 = {cur_pos_1} cur_pos_2 = {cur_pos_2}")
        time.sleep(timestamp)

        #sensor_data.append(data.sensordata[2])

        time_steps.append(timestamp * step)
        # print(sensor_data[2])

        viewer.sync()


plt.plot(time_steps, sensor_1_data, label=f"sensor 1", linewidth=2)
plt.plot(time_steps, sensor_2_data, label=f"sensor 2", linewidth=2)
plt.plot(time_steps, goal_pos_1_data, label=f"goal pos 1", linewidth=2, linestyle='--')
plt.plot(time_steps, goal_pos_2_data, label=f"goal pos 2", linewidth=2, linestyle='--')
# plt.plot(time_steps, error_1_data, label=f"error 1", linewidth=2)
# plt.plot(time_steps, error_2_data, label=f"error 2", linewidth=2)

plt.xlabel('t, s', fontsize=12)
plt.ylabel('angle, rad', fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend(fontsize=10, loc='best')
plt.tight_layout()
```

```
plt.savefig(f'sensor_data_comparison.png', dpi=300, bbox_inches='tight')
plt.show()

plt.close()

plt.plot(time_steps, error_1_data, label=f"error 1", linewidth=2)
plt.plot(time_steps, error_2_data, label=f"error 2", linewidth=2)

plt.xlabel('t, s', fontsize=12)
plt.ylabel('angle error, rad', fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)
plt.legend(fontsize=10, loc='best')
plt.tight_layout()

plt.savefig(f'error_data_comparison.png', dpi=300, bbox_inches='tight')
plt.show()
```

2.2 Моделирование в MuJoCo

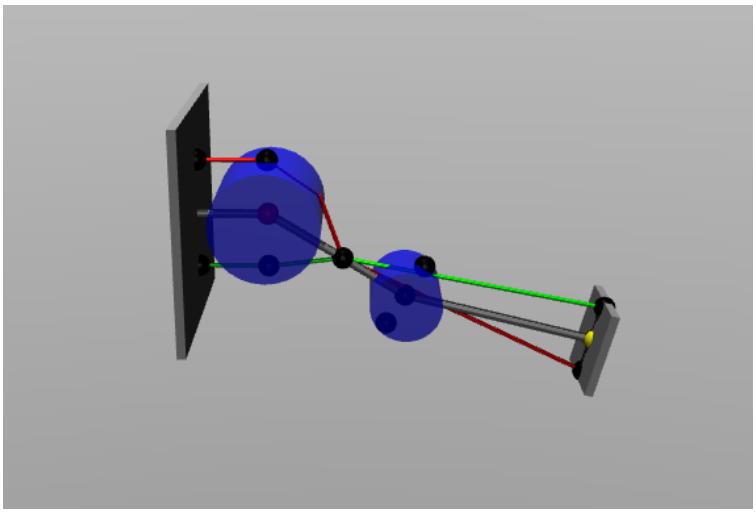

Рисунок 2 — Модель в действии

2.3 Графики моделирования

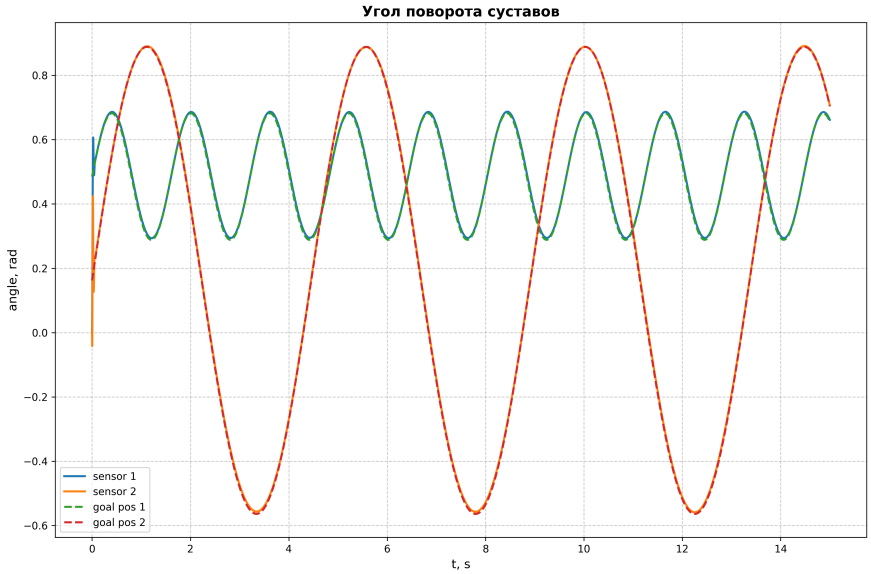

Рисунок 3 — График углов суставов

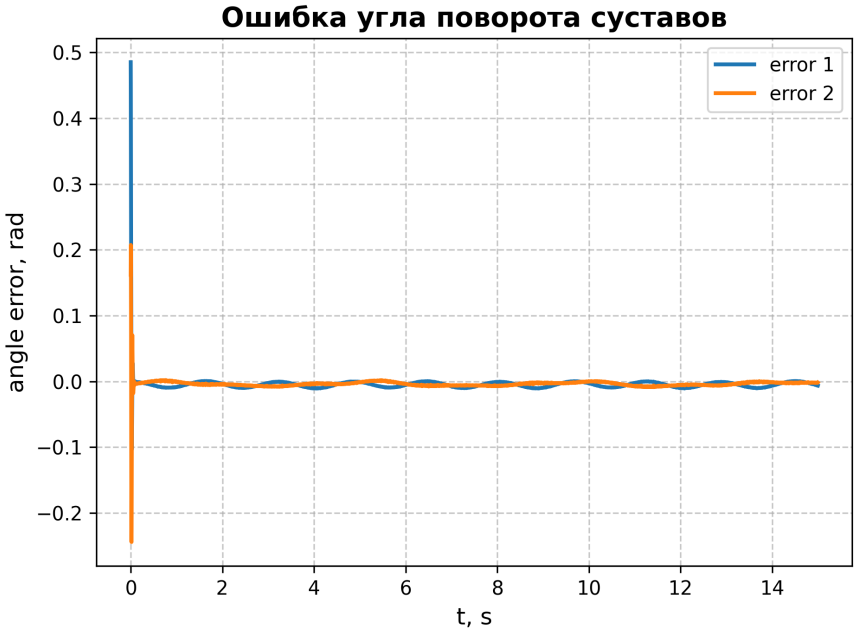

Рисунок 4 — График ошибок слежения

2.4 Анализ графиков

Как видно на графиках два ПД регулятора(по одному регулятору на каждый сустав) справляется с поставленной задачей.

ОБЩИЕ ВЫВОДЫ

В ходе практической работы были выполнены следующие задачи:

- К модели в среде MuJoCo были добавлены двигатели и датчики;
- Написан python код с регулятором для слежения за линией;
- Произведено моделирование и построены графики