

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»

Студен:

Группа R4134c

I. Ковылин

Преподаватель:

Ассистент

E.A. Ракин

Санкт-Петербург 2025 г.

ЦЕЛЬ РАБОТЫ.

- В модель, созданную в предыдущем задании, необходимо добавить исполнительные механизмы — два исполнительных механизма ($q1$ и $q2$).
- Измените XML-файл, добавив контейнеры `<actuaor>` и `<sensor>`
- Определите управляющее усилие с помощью ПД-регулятора. $qdes = AMP \cdot \sin(FREQ \cdot t) + BIAS$.

ЛИСТИНГ ПРОГРАММ

```
1 import mujoco
2 import mujoco.viewer
3 import numpy as np
4 import time
5 import os
6 import matplotlib.pyplot as plt
7
8
9 # [геометрия]
10 def calculate_crossed_tangent_points(c1, r1, c2, r2):
11     c1 = np.array(c1)
12     c2 = np.array(c2)
13     d_vec = c2 - c1
14     d = np.linalg.norm(d_vec)
15     if d < 1e-8:
16         return None
17
18     unit_d = d_vec / d
19     perp_d = np.array([-unit_d[1], unit_d[0]])
20
21     r2_eff = -r2
22     cos_theta = (r1 - r2_eff) / d
23     if abs(cos_theta) > 1.0:
24         return None
25
26     sin_theta = np.sqrt(1 - cos_theta**2)
27
28     tangents = []
29     for side in [1, -1]:
30         dir_r = cos_theta * unit_d + side * sin_theta * perp_d
31         p1 = c1 + r1 * dir_r
32         p2 = c2 + r2_eff * dir_r
33         tangents.append((p1, p2))
34
35     return tangents
36
37
38
39 # [Обновление положений]
40
41 def update_tendon_sites(model, data):
42     block1_pos = data.body("body_block1").xpos[:2]
43     block2_pos = data.body("body_block2").xpos[:2]
44
45     r1 = model.geom("block1").size[0]
46     r2 = model.geom("block2").size[0]
47
48     tangents = calculate_crossed_tangent_points(block1_pos, r1, block2_pos, r2)
49     if tangents is None:
50         return
51
52     tangents = sorted(tangents, key=lambda t: t[0][1], reverse=True)
53     (t1_b1, t1_b2), (t2_b1, t2_b2) = tangents
54
55     z = 0.0
56     b1_xmat = data.body("body_block1").xmat.reshape(3, 3)
57     b2_xmat = data.body("body_block2").xmat.reshape(3, 3)
58
59     rel_t1_b1 = np.dot(b1_xmat.T, [t1_b1[0], t1_b1[1], z]) - data.body("body_block1").xpos
60     rel_t2_b1 = np.dot(b1_xmat.T, [t2_b1[0], t2_b1[1], z]) - data.body("body_block1").xpos
61     rel_t1_b2 = np.dot(b2_xmat.T, [t1_b2[0], t1_b2[1], z]) - data.body("body_block2").xpos
62     rel_t2_b2 = np.dot(b2_xmat.T, [t2_b2[0], t2_b2[1], z]) - data.body("body_block2").xpos
63
64     data.qpos[model.joint("t1_block1_joint").qposadr[0]]:
65         model.joint("t1_block1_joint").qposadr[0] + 2] = rel_t1_b1[:2]
66
67     data.qpos[model.joint("t2_block1_joint").qposadr[0]]:
68         model.joint("t2_block1_joint").qposadr[0] + 2] = rel_t2_b1[:2]
```

Рисунок 1.1 – листинг скрипта на python

```

70     data.qpos[model.joint("t1_block2_joint")].qposadr[0]:
71     |   |   |   model.joint("t1_block2_joint").qposadr[0] + 2] = rel_t1_b2[:2]
72
73     data.qpos[model.joint("t2_block2_joint")].qposadr[0]:
74     |   |   |   model.joint("t2_block2_joint").qposadr[0] + 2] = rel_t2_b2[:2]
75
76
77     # Хранение траекторий
78     time_log = []
79     q1_log, q1_des_log = [], []
80     q2_log, q2_des_log = [], []
81
82
83
84     # И
85
86
87     def control_callback(model, data):
88
89         k = 1
90
91         KP1, KD1 = 135550.0 / k, 715.0 / k
92         KP2, KD2 = 135550.0 / k, 715.0 / k
93
94         AMP1, FREQ1, BIAS1 = 18.7 / k, 3.8 / k, -36.7 / k
95         AMP2, FREQ2, BIAS2 = 54.03 / k, 2.62 / k, 25.3 / k
96
97         t = data.time
98
99         # Ожидаемые
100        q1_des = AMP1 * np.sin(FREQ1 * t) + BIAS1
101        dq1_des = AMP1 * FREQ1 * np.cos(FREQ1 * t)
102
103        q2_des = AMP2 * np.sin(FREQ2 * t) + BIAS2
104        dq2_des = AMP2 * FREQ2 * np.cos(FREQ2 * t)
105
106        # Текущие
107        q1 = data.joint("block1_joint").qpos[0]
108        dq1 = data.joint("block1_joint").qvel[0]
109
110        q2 = data.joint("block2_joint").qpos[0]
111        dq2 = data.joint("block2_joint").qvel[0]
112
113        # И
114        data.ctrl[0] = KP1 * (q1_des - q1) + KD1 * (dq1_des - dq1)
115        data.ctrl[1] = KP2 * (q2_des - q2) + KD2 * (dq2_des - dq2)
116
117        time_log.append(t)
118        q1_log.append(q1)
119        q1_des_log.append(q1_des)
120        q2_log.append(q2)
121        q2_des_log.append(q2_des)
122
123        update_tendon_sites(model, data)
124
125        current_dir = os.path.dirname(os.path.abspath(__file__))
126        model_path = os.path.join(current_dir, "model.xml")
127
128        model = mujoco.MjModel.from_xml_path(model_path)
129        data = mujoco.MjData(model)
130
131        mujoco.set_mjcb_control(control_callback)
132
133        viewer = mujoco.viewer.launch(model, data)
134

```

Рисунок 1.2 – листинг скрипта на python

```
134
135
136
137 # Отображение траектории
138 plt.figure(figsize=(12, 6))
139
140 plt.subplot(2, 1, 1)
141 plt.plot(time_log, q1_des_log, '--', label="q1 desired")
142 plt.grid()
143 plt.legend()
144 plt.title("Block 1 trajectory")
145
146 plt.subplot(2, 1, 2)
147 plt.plot(time_log, q2_log, label="q2 actual")
148 plt.plot(time_log, q2_des_log, '--', label="q2 desired")
149 plt.grid()
150 plt.legend()
151 plt.title("Block 2 trajectory")
152
153 plt.tight_layout()
154 plt.show()
155
156
157
158 if viewer is None:
159     raise RuntimeError("Не удалось запустить MuJoCo viewer. Возможно, проблема с GL или GLFW.")
160
161 while viewer.is_running():
162     viewer.sync()
163     time.sleep(0.002)
164
165 # Окно закрыто пользователем []
166 # закрываем viewer корректно
167 viewer.close()
168
169
170
```

Рисунок 1.3 – листинг скрипта на python

Рисунок 2.1 – листинг xml файла

```
70
71             <site name="s1_b2" pos="0 -0.035 0" size="0.003"/>
72             <site name="s2_b2" pos="0 0.035 0" size="0.003"/>
73
74     </body>
75         <body name="load" pos="0.046 0 0">
76             <geom type="box" size="0.001 0.03 0.02" rgba=".7 .7 .7 1"/>
77             <site name="s1_load" pos="0 -0.025 0"/>
78             <site name="s2_load" pos="0 0.025 0"/>
79         </body>
80     </body>
81
82 </worldbody>
83
84 <tendon>
85     <spatial width="0.003">
86         <site site="s1_wall1"/>
87         <geom geom="block1" sidesite="s1_b1"/>
88         <site site="t1_block1"/>
89         <site site="t1_block2"/>
90         <geom geom="block2" sidesite="s1_b2"/>
91         <site site="s1_load"/>
92     </spatial>
93
94     <spatial width="0.003">
95         <site site="s2_wall1"/>
96         <geom geom="block1" sidesite="s2_b1"/>
97         <site site="t2_block1"/>
98         <site site="t2_block2"/>
99         <geom geom="block2" sidesite="s2_b2"/>
100        <site site="s2_load"/>
101    </spatial>
102 </tendon>
103
104 <actuator>
105     <position joint="block1_joint" kp="1"/>
106     <position joint="block2_joint" kp="1"/>
107 </actuator>
108
109
110 </mujoco>
111
112
```

Рисунок 2.2 – листинг xml файла



Рисунок 3 – внешний вид модели

Из – за слишком больших значений и маленькой геометрии тяжело уследить за траекторией, поэтому траектории были выведены на plot, при обычных значениях синусоиды, и уменьшенных в 10 раз.

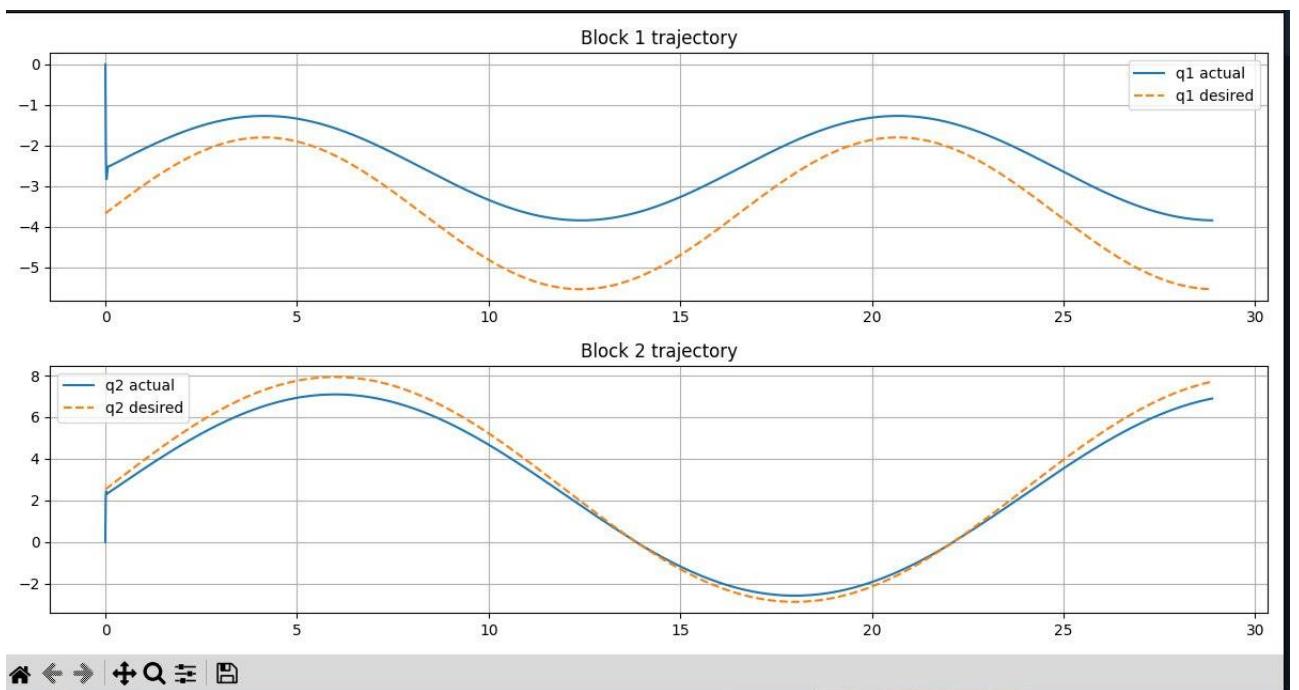


Рисунок 4 – график при уменьшенных параметрах синусоиды в 10 раз

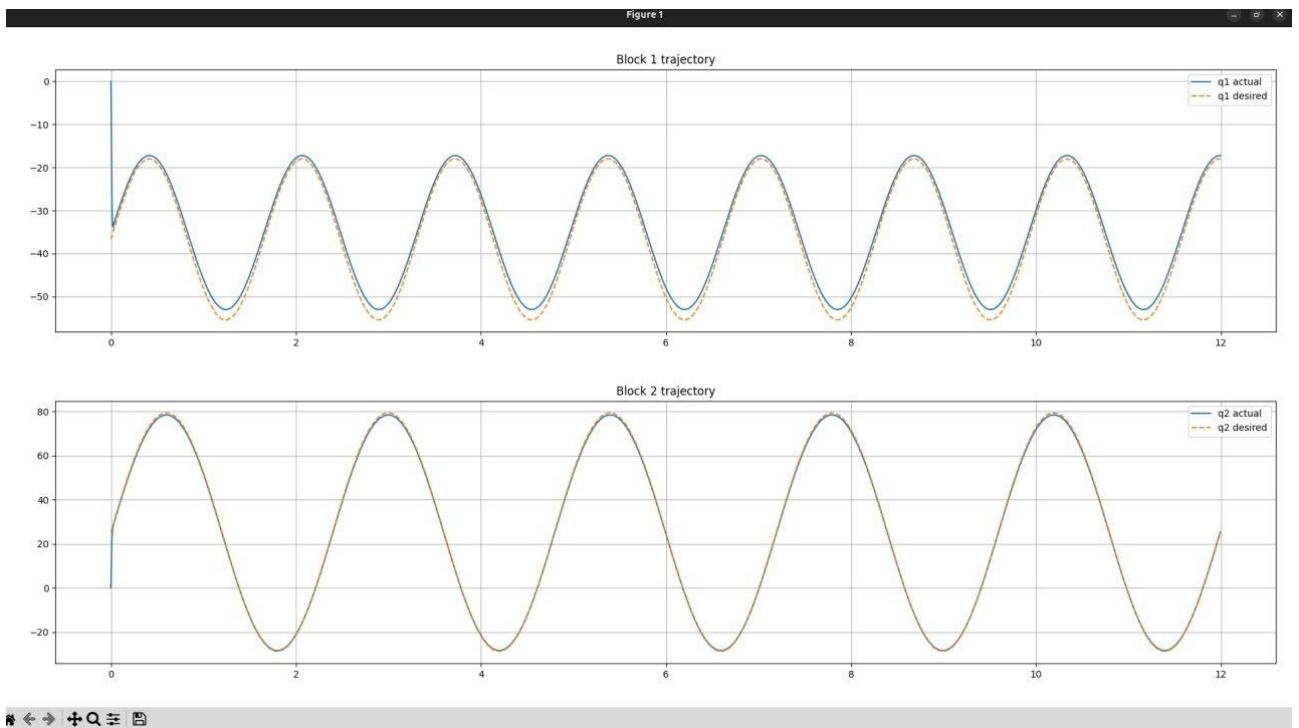


Рисунок 5 – график при обычных параметрах синусоиды в 10 раз

ВЫВОД.

В данной работе была разработана модель сухожильно-связанного 2R плоского механизма с помощью библиотеки Mujoco.