

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИТМО»**

Университет ИТМО

дисциплина

«Имитационное моделирование робототехнических систем»

Отчет по лабораторной работе № 3

Выполнил:

Смирнов И. Д. R4136с

Проверил:

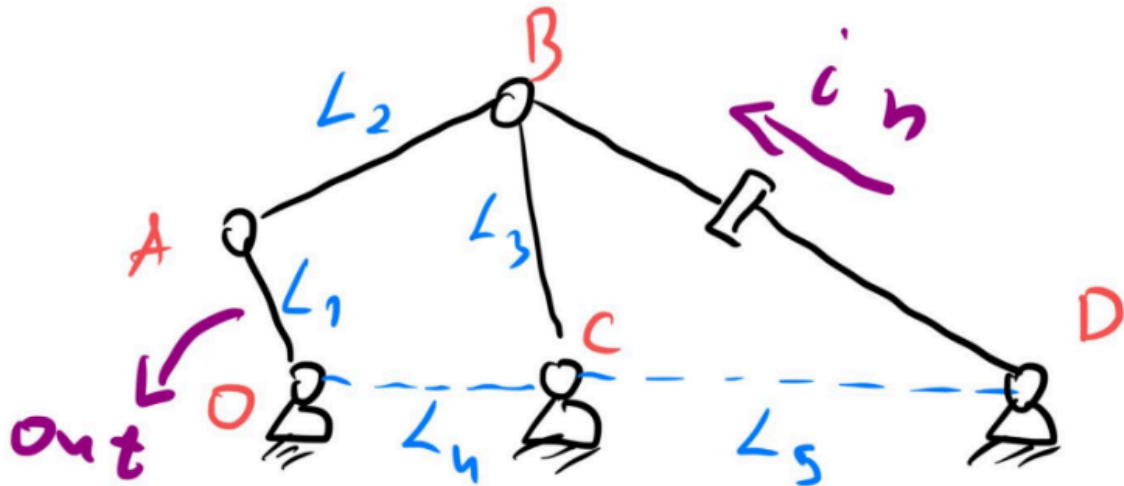
Ракишин Е. А.

Санкт-Петербург

2025

Задание

1) Написать скрипт на python с использованием методов model, data и viewer для коленчатого механизма с замкнутой цепью (optimus):



L1, m	L2, m	L3, m	L4, m	L5, m
0.036	0.0468	0.054	0.036	0.18

Ход работы

Выбран механизм *Optimus knee closed-chain*, представляющий собой планарный четырёхзвенный шарнирный механизм с дополнительным параллельным звеном.

Кинематическая схема включает точки:

- O, C, D - базовые шарниры, закреплённые на земле (плоскость $Z = 0$);
- A, B - промежуточные точки, образующие подвижный контур.

Механизм образует два замкнутых контура:

1. Основной OABC
2. Вспомогательный OABC (реализован в модели для повышения жёсткости)

Составим конфигурацию:

```

<?xml version='1.0' encoding='UTF-8'?>
<mujoco model="Optimus Mechanism">
  <option timestep="1e-3" gravity="0 0 -9.8"/>

  <asset>
    <texture type="skybox" builtin="gradient" rgb1="0.08 0.08 0.08" rgb2="0.15 0.15 0.15"
width="256" height="256"/>
    <texture name="grid" type="2d" builtin="checker" rgb1="0.88 0.88 0.88" rgb2="0.72 0.65
0.55" width="300" height="300"/>
    <material name="grid" texture="grid" texrepeat="10 10" reflectance="0.2"/>

    <material name="link_mat" reflectance="0.3"/>
  </asset>

  <worldbody>

    <light directional="false" pos="0 0 2.5" diffuse="0.9 0.9 0.9" specular="0.3 0.3
0.3"/>

    <geom type="plane" size="0.5 0.5 0.01" material="grid"/>

    <body name="OAB1" pos="0 0 0">
      <joint name="O" type="hinge" axis="0 -1 0" stiffness="0" springref="0"
damping="0"/>

      <geom name="point_O" type="sphere" pos="0 0 0" size="0.006" rgba="0.95 0.15 0.15
1.0" contype="0"/>

      <geom name="link_OA" type="cylinder" pos="0 0 0.018" size="0.005 0.018"
rgba="0.90 0.30 0.25 0.8" material="link_mat" contype="0"/>

      <body name="AB1" pos="0 0 0.036">
        <joint name="A" type="hinge" axis="0 -1 0" stiffness="0" springref="0"
damping="0.1"/>

        <geom name="point_A" type="sphere" pos="0 0 0" size="0.006" rgba="0.95 0.15
0.15 1.0" contype="0"/>

        <geom name="link_AB1" type="cylinder" pos="0 0 0.0234" size="0.005 0.0234"
rgba="0.90 0.65 0.20 0.8" material="link_mat" contype="0"/>

        <site name="sC1" pos="0 0 0.0468" size="0.003"/>
      </body>
    </body>

    <body name="CB2" pos="0.036 0 0">
      <joint name="C" type="hinge" axis="0 -1 0" stiffness="0" springref="0"
damping="0.1"/>

      <geom name="point_C" type="sphere" pos="0 0 0" size="0.006" rgba="0.95 0.15 0.15
1.0" contype="0"/>

```

```

    <geom name="link_CB2" type="cylinder" pos="0 0 0.027" size="0.005 0.027"
        rgba="0.25 0.70 0.25 0.8" material="link_mat" contype="0"/>

    <site name="sC2" pos="0 0 0.054" size="0.003"/>
</body>

<body name="DFB3" pos="0.216 0 0">
    <joint name="D" type="hinge" axis="0 -1 0" stiffness="0" springref="0"
damping="0"/>

    <geom name="point_D" type="sphere" pos="0 0 0" size="0.006" rgba="0.95 0.15 0.15
1.0" contype="0"/>

    <geom name="link_DB3" type="cylinder" pos="0 0 0.09" size="0.005 0.09"
        rgba="0.20 0.40 0.90 0.8" material="link_mat" contype="0"/>

    <body name="FB3" pos="0 0 0.18">
        <joint name="slider" type="slide" axis="0 0 1" limited="true"
            range="-0.2 0.2" stiffness="0" springref="0" damping="0"/>

        <geom name="point_B3" type="sphere" pos="0 0 0" size="0.006" rgba="0.95 0.15
0.15 1.0" contype="0"/>

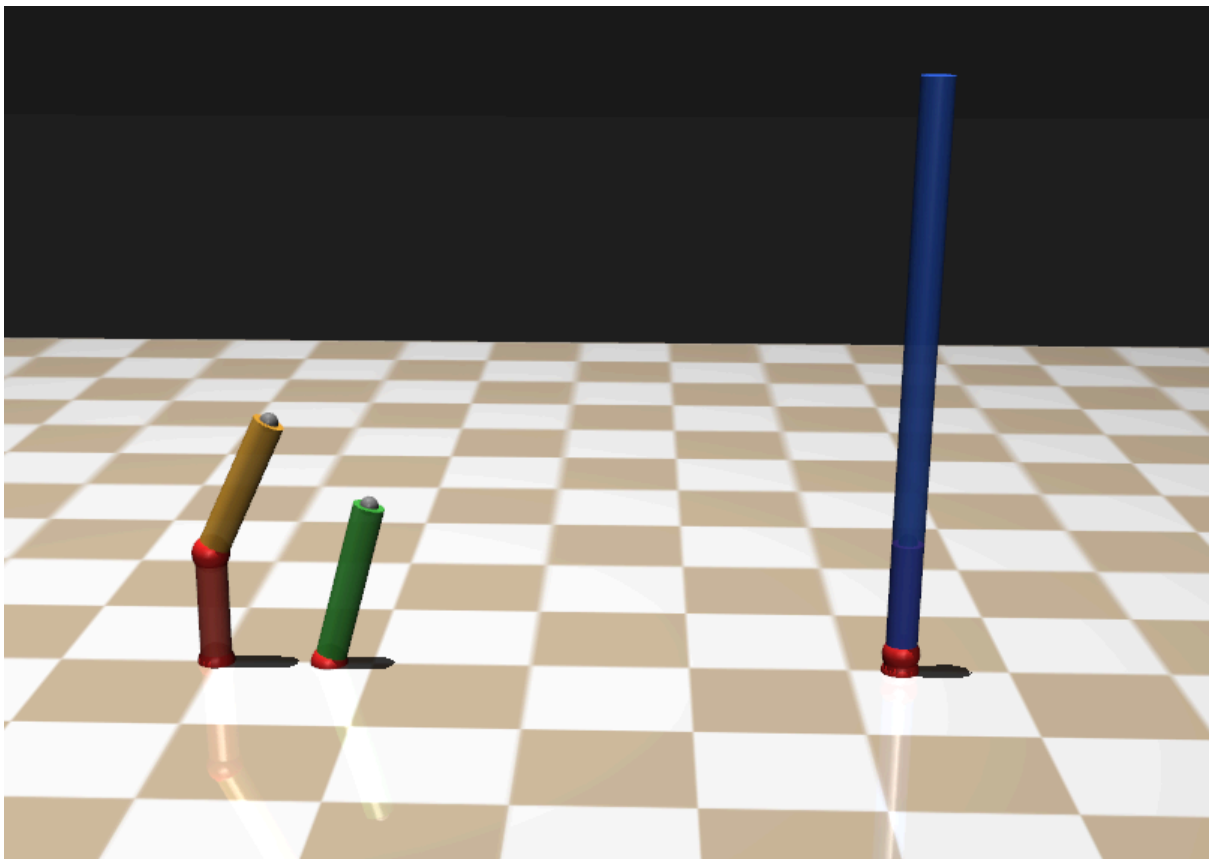
        <geom name="link_FB3" type="cylinder" pos="0 0 0.018" size="0.005 0.018"
            rgba="0.60 0.20 0.80 0.8" material="link_mat" contype="0"/>

        <site name="sC3" pos="0 0 0.036" size="0.003"/>
    </body>
</body>
</worldbody>

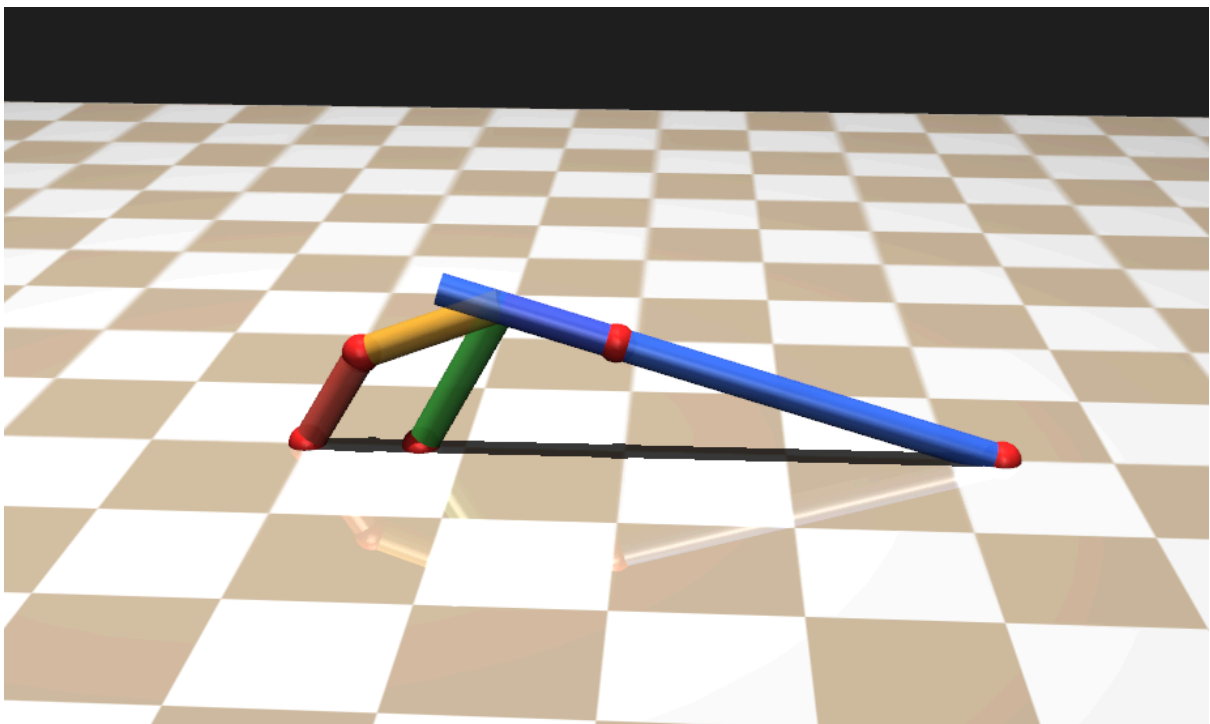
<equality>
    <connect name="eq_C1_C2" site1="sC1" site2="sC2" solimp="0.95 0.99 0.001" solref="0.01
1"/>
    <connect name="eq_C1_C3" site1="sC1" site2="sC3" solimp="0.95 0.99 0.001" solref="0.01
1"/>
</equality>
</mujoco>

```

получим на выходе следующую визуализацию:



добавим связь `<equality>` и получаем необходимый механизм с замкнутой кинематикой



Вывод

Разработанная модель точно отражает заданную геометрию пассивного механизма колена Optimus. Использование связей `<equality>` позволило корректно смоделировать замкнутый кинематический контур — ключевую особенность данного типа механизмов. Симуляция демонстрирует физически правдоподобное поведение, что подтверждает пригодность модели для дальнейшего использования

Листинги программ

```
import time
import mujoco
import mujoco.viewer
import numpy as np

XML_PATH = "optimus.xml"

def print_model_info(model):
    print("\nModel summary:")
    print(f"  nq = {model.nq}")
    print(f"  nv = {model.nv}")
    print(f"  nu = {model.nu}")

    print("\nJoints:")
    for jid in range(model.njnt):
        name = mujoco.mj_id2name(model, mujoco.mjtObj.mjOBJ_JOINT, jid)
        print(f"  {jid}: {name}")

def set_joint_angle(model, data, joint_name, angle_rad):
    jid = mujoco.mj_name2id(model, mujoco.mjtObj.mjOBJ_JOINT, joint_name)
    if jid == -1:
        print(f"[Warning] Joint '{joint_name}' not found!")
        return

    qpos_addr = model.jnt_qposadr[jid]
    data.qpos[qpos_addr] = angle_rad

def main():

    print("Loading model from:", XML_PATH)
    model = mujoco.MjModel.from_xml_path(XML_PATH)
    data = mujoco.MjData(model)

    print_model_info(model)

    # Все координаты = 0
    data.qpos[:] = 0.0
```

```

set_joint_angle(model, data, "O", np.deg2rad(-10))
set_joint_angle(model, data, "A", np.deg2rad(-25))
set_joint_angle(model, data, "C", np.deg2rad(-20))
set_joint_angle(model, data, "D", np.deg2rad(35))
set_joint_angle(model, data, "slider", -0.1)

print("Launching viewer...")

with mujoco.viewer.launch_passive(model, data) as viewer:

    viewer.cam.lookat[:] = np.array([0.09, 0.0, 0.0])
    viewer.cam.distance = 0.5
    viewer.cam.elevation = -30

    step = 0
    while viewer.is_running() and step < 30000:
        step += 1
        mujoco.mj_step(model, data)
        viewer.sync()
        time.sleep(0.001)

    print("Simulation finished.")

if __name__ == "__main__":
    main()

```