Optimus' knee closed-chain mechanism with PD Controller

Ahmad Ahmad

November 2025

1 Introduction

The Optimus knee mechanism is a closed-chain, multi-link knee articulation used in advanced prosthetics and humanoid robotics. Unlike a simple hinge knee, this mechanism uses several interconnected links to create nonlinear motion, load distribution, and controlled torque generation, similar to the biomechanics of the human knee. Your diagram shows a 5-link structure that reproduces the roll-and-slide motion of a natural knee.

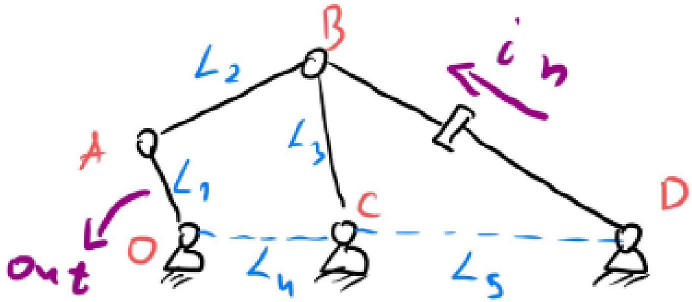

Figure 1: mechanism

2 Mechanism Structure

In this mechanism we have 3 links L1 , L2, L3 that connect to joints O,A,B,C . and the input for forces is the link DB that is like an adjustable link. the output of motion we have to observable is the link oA , where joint A makes a curve of motion with respect to other joints .

3 ADD Actuator and Sensors

3.1 Actuator

An actuator is added to joint D, and its motion will be linear according to a specific equation :

$$q^{des} = AMP.sin(FREQ.t) + BIAS \tag{1}$$

where :
AMP=47.7 deg , FREQ=3.1 HZ , BIAS =-35.2

```
    <actuator>
        <motor name="m1" joint="D" gear="1"/>
    </actuator>
```

3.2 Sensors

First, we edit the XML file to add position and velocity sensors to joints A, C, and D. we detect each joint with two sensors

```
    <sensor>
    <jointpos joint="A"  name="A_pos"/>
    <jointvel joint="A"  name="A_vel"/>
    <jointpos joint="D"  name="D_pos"/>
    <jointvel joint="D"  name="D_vel"/>
    <jointpos joint="C"  name="C_pos"/>
    <jointvel joint="C"  name="C_vel"/>
</sensor>
```

4 PD Controller

4.1 Definition

A PD Controller (Proportional-Derivative Controller) is a type of feedback controller used in mechanical and electrical systems to generate the control effort required to move the system toward a desired reference.
P – Proportional: Depends on the difference between the desired value and the current value (error).
D – Derivative: Depends on the rate of change of the error to reduce oscillations and improve system stability.

4.2 Mathematical Equation

The controller computes the control signal as follows:

$$force = K_p.e(t) + K_D.e^{'}(t) \tag{2}$$

where :
K_P : proportional gain
K_D:derivative gain
$e(t) = q^d es - q$: the error between the desired position q^{des} and current position q
q: the error in velocity
The K_P term drives the system toward the reference, while the K_D term slows it down near the target to prevent overshoot.the goal ensure the links move smoothly along the sinusoidal trajectory without collisions or excessive oscillations.
the term $K_p.e(t)$ pushes the system toward the reference; larger errors produce stronger corrective action. and the term $K_D.e^{'}(t)$ damps oscillations and improves stability by resisting rapid changes in error.

```
        q_des = AMP * np.sin(FREQ * t) + BIAS
        qd_des = AMP * FREQ * np.cos(FREQ * t)

        q = data.qpos[0]
        qdot = data.qvel[1]

        #PD control
        force = KP*(q_des - q) + KD*(qd_des - qdot)
        data.ctrl[0] = force
```

5 Results and Dicsuss :

The results presented in the figure 2 a comprehensive depiction of the dynamic behavior of joints A, D, and C when subjected to a sinusoidal reference input regulated by a PD controller. At the onset of the simulation, all measured variables display a pronounced transient response, primarily attributed to the initial discrepancy between the system's actual state and the desired trajectory, which induces a substantial corrective effort from the controller. Following this transient interval, the system exhibits a consistent convergence toward steady-state behavior, characterized by smooth, periodic oscillations for all joints. The actuated joint D demonstrates accurate and stable tracking of the prescribed sinusoidal reference, whereas joints A and C exhibit oscillatory motion governed by the geometric and kinematic constraints of the closed-chain mechanism. The velocity profiles similarly attain sinusoidal patterns with distinct phase shifts, reflecting the inherent interdependence among the joints. Importantly, all state variables remain bounded over the entire simulation horizon, with no evidence of instability or drift, thereby confirming both the adequacy of the PD tuning parameters and the correctness of the mechanism's modeling and constraint formulation.

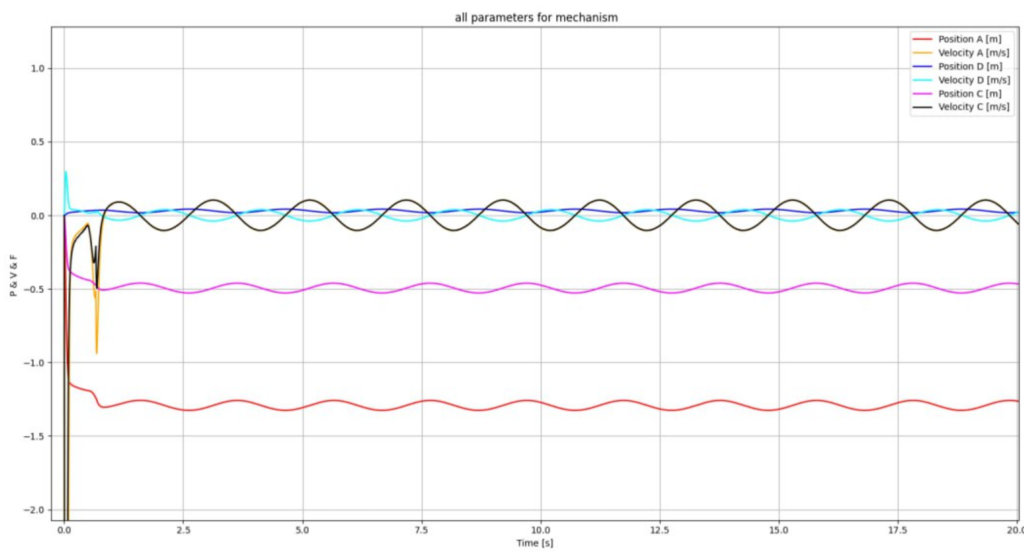

Figure 2: position and velocity

References

[1] A. PID Controllers ,2nd edition ,karl and Tore ,1995.

[2] b. Ivan.B ,Simulation of Robotic System course, ITMO University ,2025.

[3] DeepMind, *MuJoCo Documentation*, Version 3.1, 2024. Available at: https://mujoco.readthedocs.io/. Accessed: 1 Feb. 2025.

A Program Code Appendix

A.1 Python code

```
import mujoco
import mujoco.viewer
```

```python
import numpy as np
import matplotlib.pyplot as plt

model = mujoco.MjModel.from_xml_path("Task4.xml")
data = mujoco.MjData(model)

AMP = np.deg2rad(47.7)
FREQ = 3.1
BIAS = np.deg2rad(-35.2)

KP = 100
KD = 15

t = 0
dt = model.opt.timestep
time_list=[]
pos_list_A=[]
vel_list_A=[]
pos_list_C=[]
vel_list_C=[]
pos_list_D=[]
vel_list_D=[]
force_list=[]

with mujoco.viewer.launch_passive(model, data) as viewer:
    while viewer.is_running():
        q_des = AMP * np.sin(FREQ * t) + BIAS
        qd_des = AMP * FREQ * np.cos(FREQ * t)

        q = data.qpos[0]
        qdot = data.qvel[1]

        # ---- PD control ----
        force = KP*(q_des - q) + KD*(qd_des - qdot)
        data.ctrl[0] = force

        mujoco.mj_step(model, data)
        time_list.append(t)
        force_list.append(force)
        pos_list_A.append(data.sensordata[0])
        vel_list_A.append(data.sensordata[1])
        pos_list_D.append(data.sensordata[2])
        vel_list_D.append(data.sensordata[3])
        pos_list_C.append(data.sensordata[4])
        vel_list_C.append(data.sensordata[5])

        viewer.sync()
        t += dt

plt.figure(figsize=(12,6))
plt.plot(time_list, pos_list_A, label="Position A [m]",color="red")
plt.plot(time_list, vel_list_A, label="Velocity A [m/s]", color="orange")
plt.plot(time_list, pos_list_D, label="Position D [m]",color="blue")
plt.plot(time_list, vel_list_D, label="Velocity D [m/s]", color="cyan")
plt.plot(time_list, pos_list_C, label="Position C [m]",color="magenta")
plt.plot(time_list, vel_list_C, label="Velocity C [m/s]", color="black")
#plt.plot(time_list, force_list, label="force [m/s]", color="green")
plt.xlabel("Time [s]")
plt.ylabel("P & V & F ")
plt.title(" all parameters for mechanism")
plt.grid(True)
plt.legend()
plt.show()
```

A.2 XML code

```
    <mujoco>
    <option timestep="1e-4"/>
    <option gravity="0 0 -0.98"/>

    <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>
        <light pos="0 0 10 "/>
        <geom type="plane" size="0.5 0.5 1" material="grid"/>

        <camera name="side_view" pos="0.1 -1.5 1.0" euler="0 0 0" />
        <camera name="upper_view" pos="0 0 1.5 " euler="0 0 0 "/>

        <body name="OA" pos="0 0 0" euler="90 0 0">
                <joint name="O" type="hinge" axis="0 0 1" stiffness="0" springref="0" damping="0"/>
                <geom name="point O" type="cylinder" pos="0 0 0" size="0.02 0.02" rgba="0.89 0.14 0.16 0.5" euler="0 0 0" contype="0"/>
                <geom name="link OA" type="cylinder" pos="0 0.046 0 " size="0.01 0.046" rgba="0.21 0.32 0.82 0.5" euler="90 0 0" contype="1"/>
                <body name="AB" pos="0 0.092 0" euler="0 0 0">
                    <joint name="A" type="hinge" axis="0 0 1" damping="0" stiffness="0" springref="0"/>
                    <geom name="point B" type="cylinder" pos="0 0 0" size="0.02 0.02" rgba="0.89 0.14 0.16 0.5" euler="0 0 0" contype="0"/>
                     <geom name="link AB" type="cylinder" pos="0 0.0598 0 " size="0.01 0.0598" rgba="0.21 0.32 0.82 0.5" euler="90 0 0" contype="1"/>
                     <site name="AB" size="0.015" pos="0 0.1196 0"/>

                </body>
                <body name="BC" pos="0.057 0 0" euler="0 0 0 " >
                            <joint name="C" type="hinge" axis="0 0 1" damping="0" springref="0" stiffness="0"/>
                            <geom name="point C" type="cylinder" pos="0 0 0" size="0.02 0.02" rgba="0.89 0.14 0.16 0.5" euler="0 0 0" contype="0"/>
                            <geom name="link BC" type="cylinder" pos="0 0.069 0 " size="0.01 0.069" rgba="0.21 0.32 0.82 0.5" euler="90 0 0" contype="1"/>
                            <site name="BC" pos="0 0.138 0" size="0.02"/>
                            <body name="BJ" pos="0 0.138 0" euler="0 0 0">
                                     <joint name="BJ" type="hinge" axis="0 0 1" damping="0" springref="0" stiffness="0"/>
                                     <geom name="point BJ" type="cylinder" pos="0 0 0" size="0.02 0.02" rgba="0.89 0.14 0.16 0.5" euler="0 0 0" contype="1"/>
                                </body>
                </body>
        </body>
        <body name="BD" pos="0.287 0 0" euler=" 90 0 0 ">
                    <joint name="D" type="slide" axis="1 0 0"/>
                    <geom name="point D" type="cylinder" pos="0 0 0 " size=" 0.02 0.02" rgba=" 0.8 0.8 0.8 0.8" euler=" 0 0 0" contype="0" />
                     <geom name="link DB" type="capsule" fromto="0 0 0  -0.23 0.138  0" size="0.01" rgba="0.21 0.32 0.82 0.5" euler="90 0 0" contype="1"/>
                    <site name="DB" size="0.015" pos="-0.23 0.138 0"/>
        </body>
    </worldbody>
<equality>
<connect site1="DB" site2="BC"/>
    <connect site1="BC" site2="AB"/>
</equality>
    <actuator>
```

```
        <motor name="␣m1" joint="D" gear="1"/>
    </actuator>
<sensor>
    <jointpos joint="A"  name="A_pos"/>
    <jointvel joint="A"  name="A_vel"/>
    <jointpos joint="D"  name="D_pos"/>
    <jointvel joint="D"  name="D_vel"/>
    <jointpos joint="C"  name="C_pos"/>
    <jointvel joint="C"  name="C_vel"/>
</sensor>
</mujoco>
```