ME 639: Assignment 01 Report

ROS 2 + MuJoCo Simulations

Monisha Kavuri (23110162)

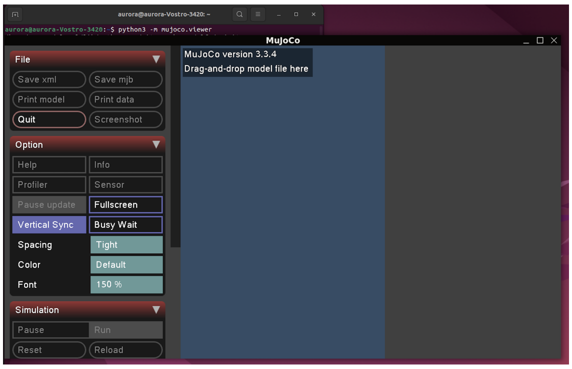
Objective

Set up ROS2 and MuJoCo and run basic robot simulations. Observe the behavior of different robots in a physics simulator under different conditions.

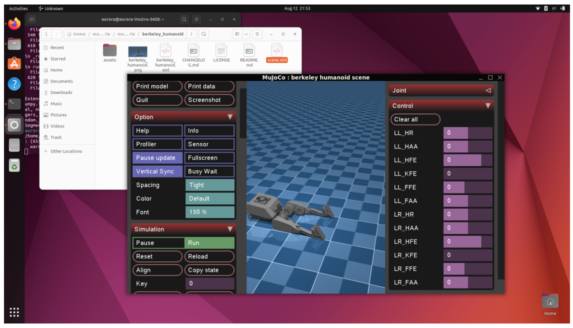
1 Environment Validation

What I ran: Installed ROS2 and MuJoCo. Launched MuJoCo using the command `python3 -m mujoco.viewer`.

- Once the MuJoCo viewer opened, I was able to load different robot files from the `mujoco_menagerie` repository.
- I was able to explore different settings such as gravity, contact, controls, etc.



(a) Launch MuJoCo

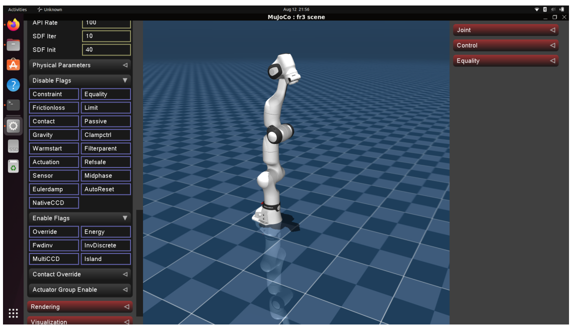


(b) Loading the berkeley humanoid from mujoco_menagerie

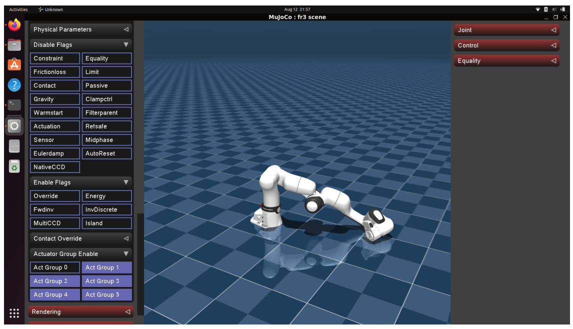
2 Serial Chain Manipulator (Franka Research 3)

Tests: Observed behaviour with gravity on vs off. Kept contact with the ground on. Used the control option to move the different joints.

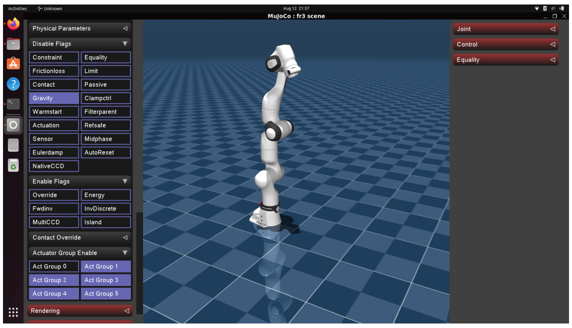
- With gravity *off*, links hold initial pose; with the actuator group *on* and *off*.
- With gravity *on*, the arm falls till contact or limit when the actuator is *off*, and remains upright when the actuator is *on*.



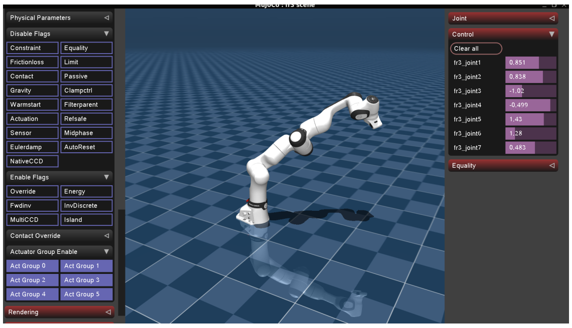
(a) Gravity ON; Actuator ON



(b) Gravity ON; Actuator OFF (arm falls)



(c) Gravity OFF



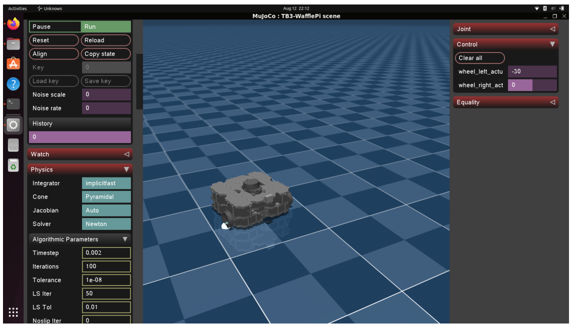
(d) Controlling movement of different joints

- When the Gravity was off, the arm did not fall regardless of whether the actuators were enabled or not.

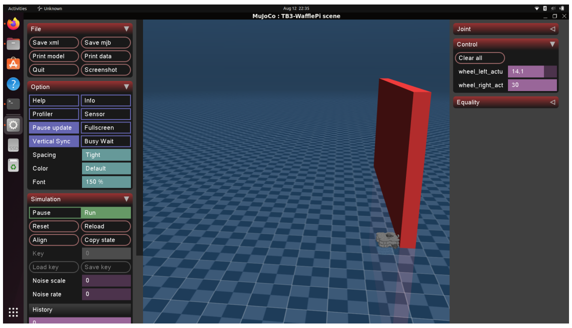
3 Wheeled Mobile Robot (TurtleBot Waffle Pi)

Tests: Apply forces; observe slip/drift while controlling the motor speeds. Add a wall to observe effects on the robot on collision.

- When one wheel (motor) is rotated at full speed (30 rad/s) and the other is kept stationary, the robot rotates. At this time, a small amount of drift is observed in the wheel which is not spinning.
- Upon removing the friction, slight slip in the wheel is observed when motors are given some speed.
- When the robot hits the wall at an angle (one wheel first), it wobbles on impact and drifts until the wall fully stops any further motion.



(a) Push test

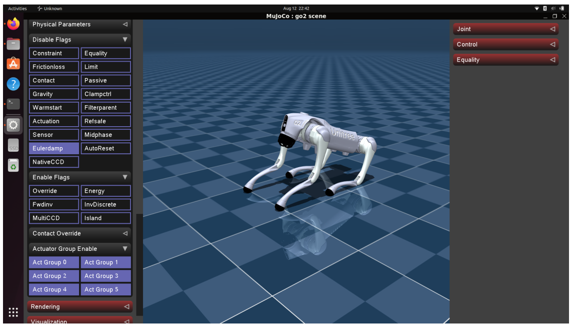


(b) Collision with a wall

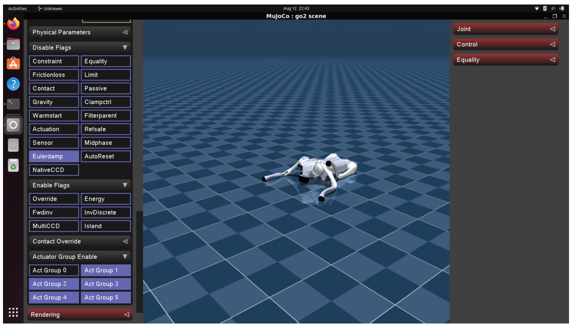
4 Quadruped (Unitree Go2 EDU)

Tests: Stability under gravity, joints and joint limits, and interactions with the ground.

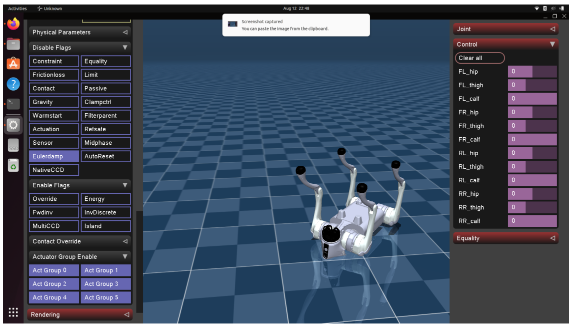
- When the robot is initially spawned, with gravity *on* and actuators *on*, it immediately collapsed to a 'sitting position', as seen in Figure (a). When the actuators are disabled, all 4 hip joints collapse till joint limit/contact with the ground, visible in Figure (b).
- Upon re-enabling the actuators, the robot rolls into the position shown in Figure (c). This is likely caused by the sudden torque applied at all joints, creating an imbalance; combined with its interaction with the ground, this causes it to roll over into place.
- There were a total of 12 joints, 3 at each leg. They were the hip, thigh, and calf. The hip joints moved the front and back legs inward and outward. The thigh joints moved them up and down along the torso, and the calf joints rotated the 'elbow and knee.'
- The limits of the joints were explored in the control panel (depicted in Figure (d)), where each joint was driven to a different angle limit.
- I observed that while it was in the kneeling position, motion to the front joints did not cause it to tip forward, the actuators in the rear legs provided balance. This is displayed in Figures (e) and (f).



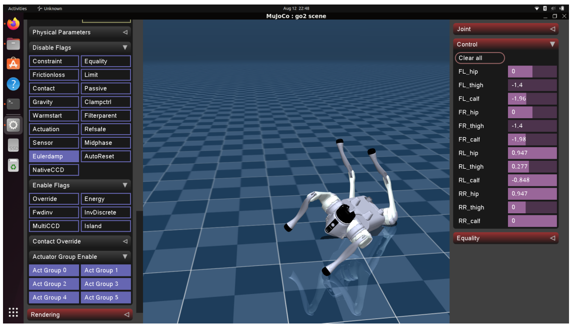
(a) Gravity ON, Actuator ON



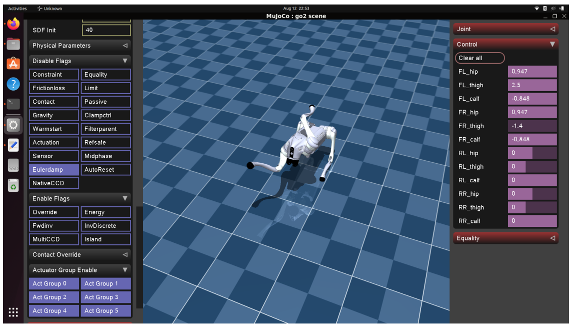
(b) Gravity ON, Actuator OFF



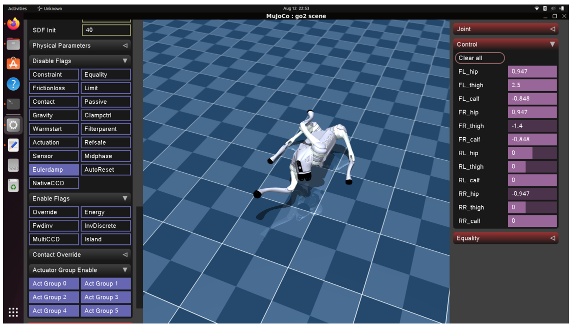
(c) Actuator re-enabled, roll over observed



(d) Joint limit test



(e) Balance on rear legs

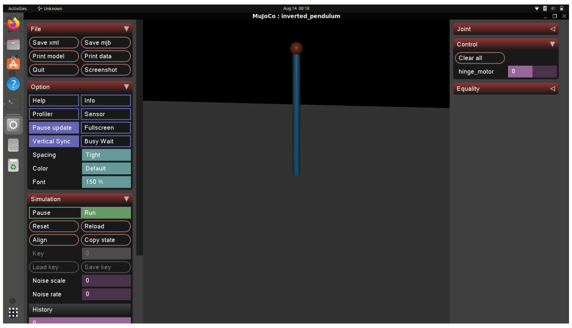


(f) Balance on rear legs when they are at their limit

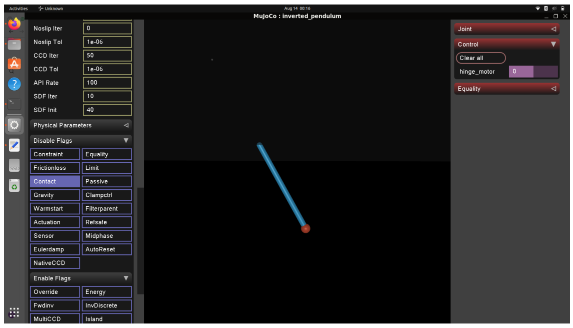
5 Inverted Pendulum

Tests: Varying initial positions, contact with ground, motor settings.

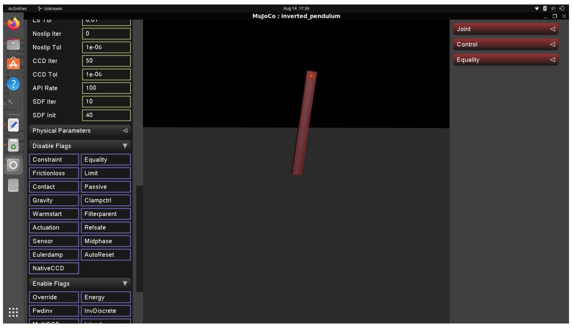
- Once the pendulum was spawned, as it was perfectly vertical, it was balanced (Figure (a)). Upon applying a small force to displace the bob, it fell to the ground (Figure (c)).
- I disabled the contact with the ground, to observe oscillations. I noticed that while the gravity was ON and the motor was OFF, it would simply oscillate until rest (Figure(b)), never going full circle.
- When gravity was OFF, or a slight torque was given through the motor initially before setting it back to 0, the pendulum would go in full circles, slowing down until it started to oscillate, eventually coming to a stop.
- I added a motor at the hinge with gear=5 (so 1 $N \cdot m$ of command produces 5 $N \cdot m$ at the joint). With contacts enabled, once the bob fell to one side, the available motor torque wasn't enough to lift it back up against gravity. After lowering the bob's density (reducing mass) to 700 (from 1200) and increasing the gear ratio, the same command was able to drive the pendulum back up. This is demonstrated in Figure (d).



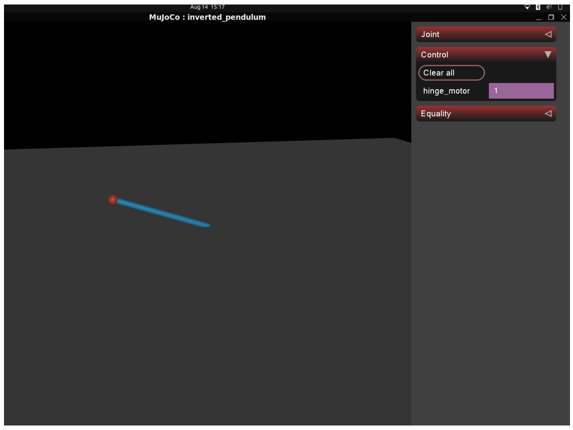
(a) Initial spawn; gravity ON.



(b) Gravity ON, Contact with ground OFF, Motor OFF. Slightly displaced from vertical position and released. Observe oscillations.



(c) The figure shows manual displacement from the vertical. Gravity ON, Contact with ground ON, Motor OFF. Observe the pendulum fall to the ground.



(d) Using motor torque to lift pendulum back up after it fell.

Challenges & Fixes

- **mujoco_ros documentation:** There's no official documentation for mujoco_ros, so I used the MuJoCo viewer directly to load the models and observe the physics/contacts for the purpose of the assignment. I did install an open-source repo, 'mujoco_ros2_control' and loaded the demo files. I was not able to control them in the interface, and will have to further explore it.
- **Inverted Pendulum:** While creating the inverted pendulum .xml file, initially my pendulum was fixed at the joint, so it would not fall with gravity even when the initial position was not perfectly vertical. This was because the pole of the pendulum was massless, so I modified it by giving it a density. I further added a motor to the hinge joint, so that I could control the initial conditions.
- **Robot Control:** In the GUI alone, it is not possible to control multiple joints at the same time. So, establishing coordination between joint movements was difficult, especially in the case of the Unitree robot. Even for the Waffle-Pi, moving it in a straight line required giving both wheels equal speed. Controlling them through python scripts and ros would allow for smoother control.