

УСТАНОВКА И ЗАПУСК MUJOCO + СИМУЛЯЦИЯ РОБОТА UNITREE (ДЛЯ НОВИЧКОВ)

Эта инструкция поможет Вам установить MuJoCo — физический симулятор — и запустить в нём робота от компании Unitree (например, собаку или гуманоида H1). Всё делается в терминале Linux (например, Ubuntu).

Шаг 1: Скачиваем MuJoCo

- Перейдите на официальную страницу релизов MuJoCo:

<https://github.com/google-deepmind/mujoco/releases>

- Найдите и скачайте архив **mujoco-3.3.6-linux-x86_64.tar.gz** (или последнюю версию для Linux).

Обычно он сохраняется в папку **Загрузки** (или `~/Downloads`).

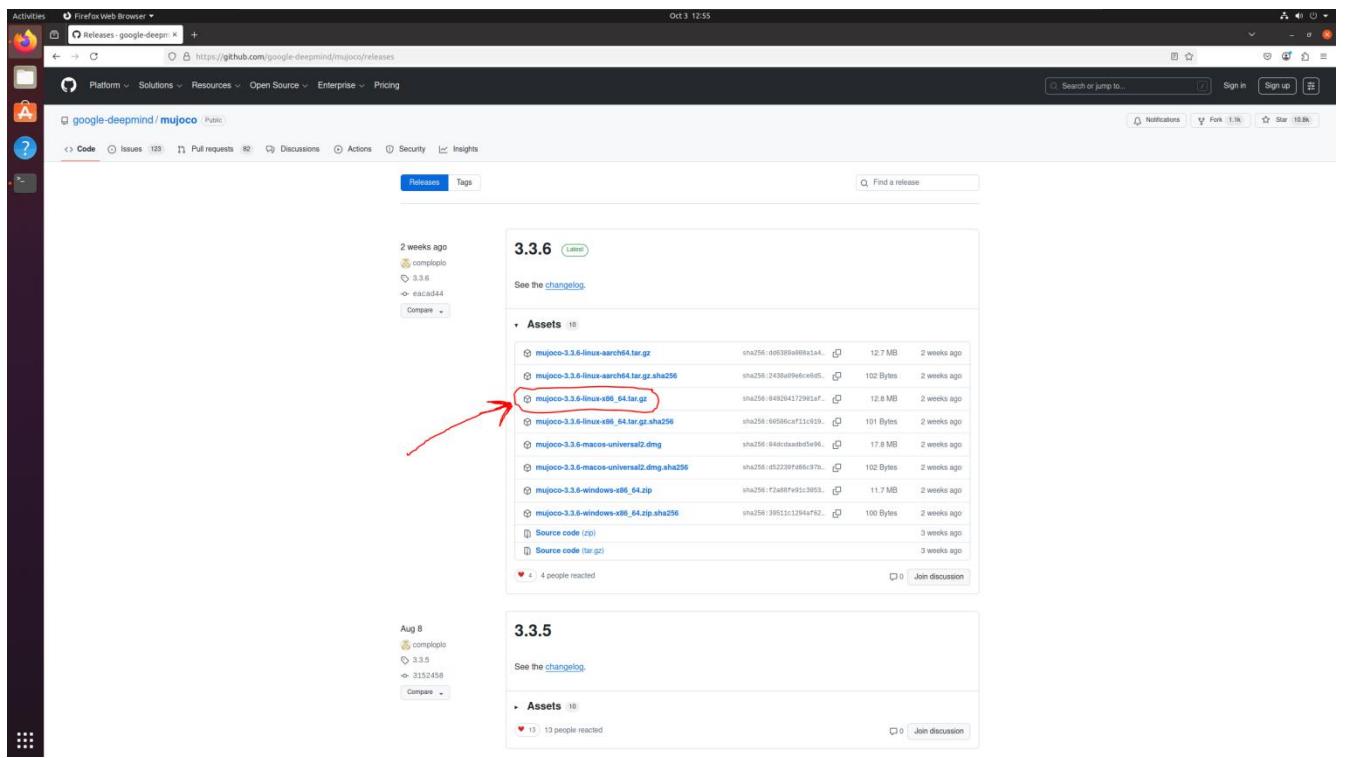


Рисунок 1 - Расположение архива на Github.

Шаг 2: Распаковываем MuJoCo

Откройте **терминал** и выполните команду:

```
sudo tar -xvzf ~/Downloads/mujoco-3.3.6-linux-x86_64.tar.gz -C /opt/
```

Эта команда распакует MuJoCo в системную папку `/opt/`, чтобы он был доступен всем пользователям.

Шаг 3: Делаем MuJoCo доступным из любой папки

Чтобы можно было запускать симулятор просто командой `simulate`, создадим символьическую ссылку:

```
sudo ln -s /opt/mujoco-3.3.6/bin/simulate /usr/local/bin/simulate
```

Шаг 4: Проверяем установку

В терминале введите:

```
simulate
```

Если всё сделано правильно, откроется окно симулятора MuJoCo:

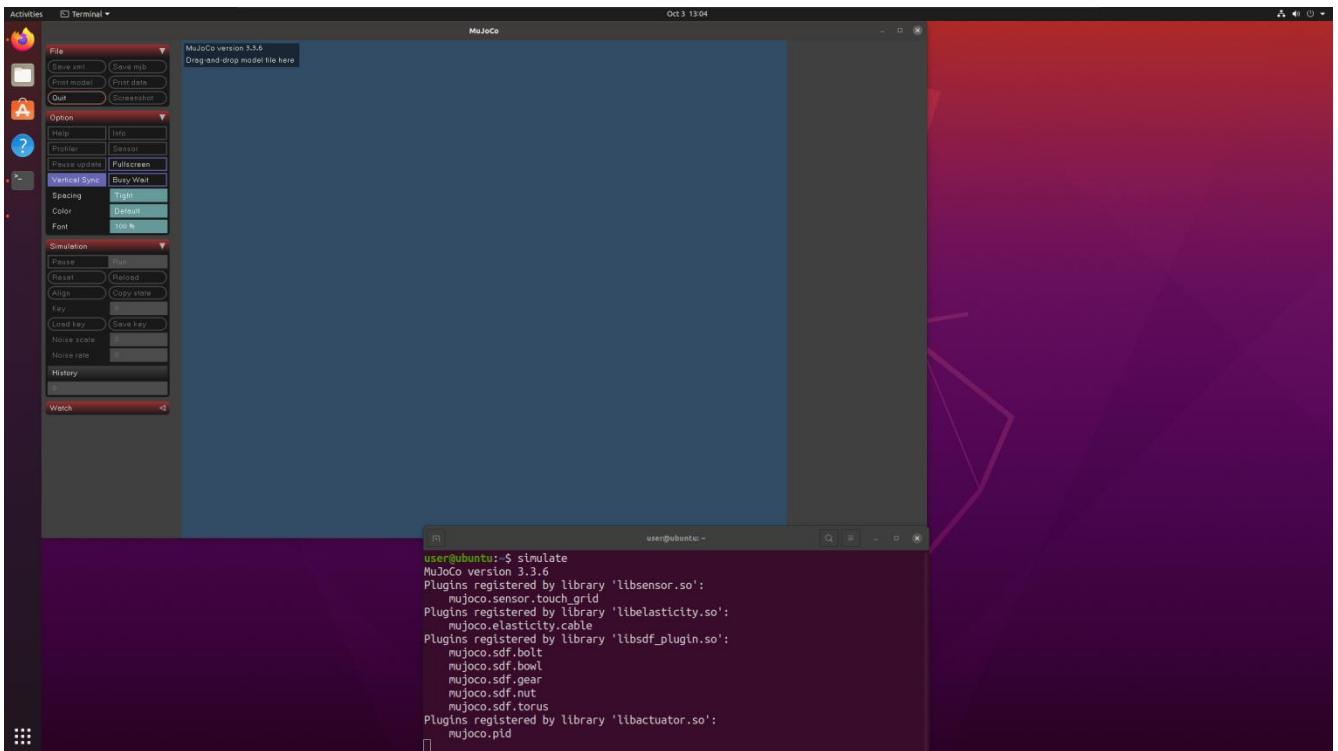


Рисунок 2 - Тестовый запуск MuJoCo.

Теперь MuJoCo установлен! Закройте окно, оно более нам не понадобится.

Шаг 5: Клонируем симуляцию робота Unitree

Выполните в терминале:

```
cd ~  
git clone  
https://github.com/cyberbanana777/unitree\_mujoco\_mirea\_olympiad.git
```

Это скачает код для запуска роботов Unitree (собака, гуманоид и т.д.) в MuJoCo.

Шаг 6: Устанавливаем зависимости Python

MuJoCo и интерфейс управления работают через Python. Установим нужные библиотеки:

```
pip3 install mujoco pygame
```

Убедись, что у Вас установлен pip3. Если нет — установите через

```
sudo apt install python3-pip
```

Ничего не нужно делать, если появилась вот такая ошибка:

```
ERROR: importlib-resources 6.4.5 has requirement zipp>=3.1.0;  
python_version < "3.10", but you'll have zipp 1.0.0 which is  
incompatible.
```

Просто перейдите к шагу 7.

Шаг 7: Запускаем антропоморфного робота Unitree H1

Перейдите в папку симуляции и запустите скрипт:

```
cd ~/unitree_mujoco_mirea_olympiad/simulate_python  
python3 unitree_mujoco.py
```

Должен появиться робот **Unitree H1**, который стоит на месте (может не двигаться):

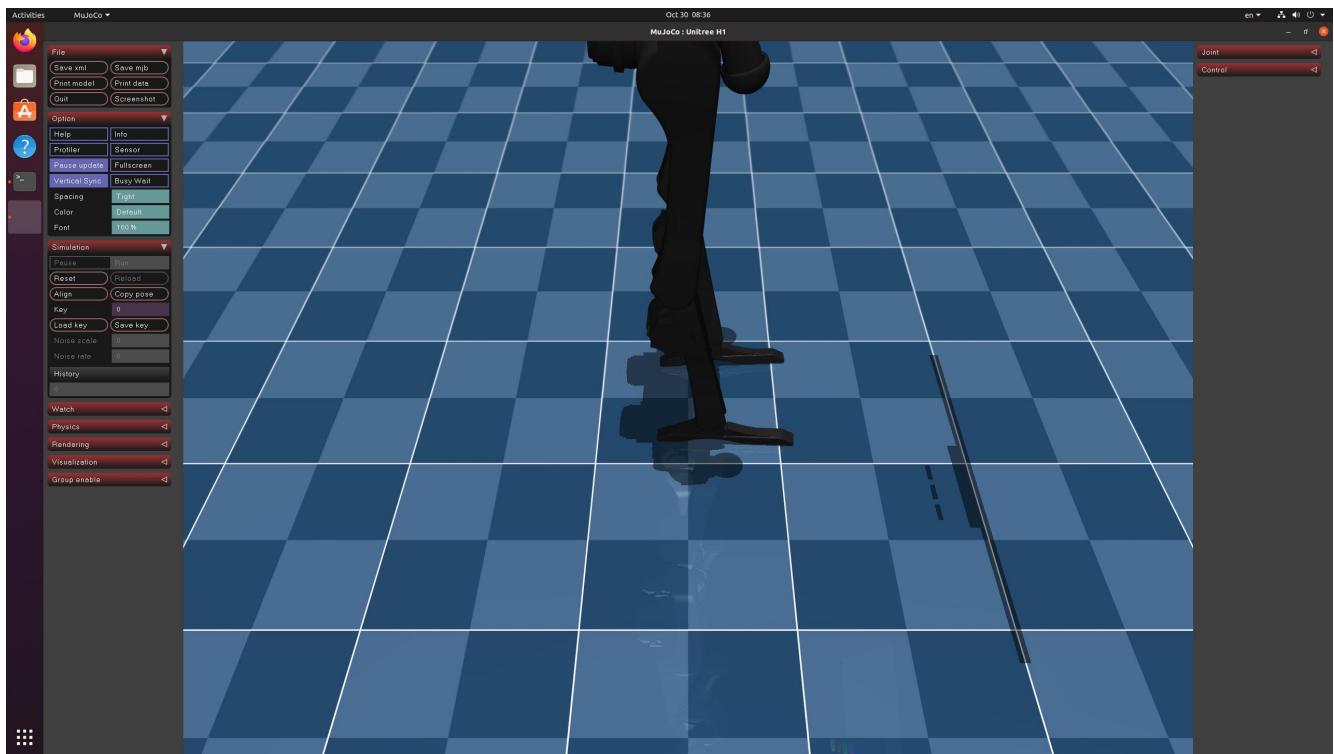


Рисунок 3 - Окно mujoco при запуске с роботом Unitree H1.

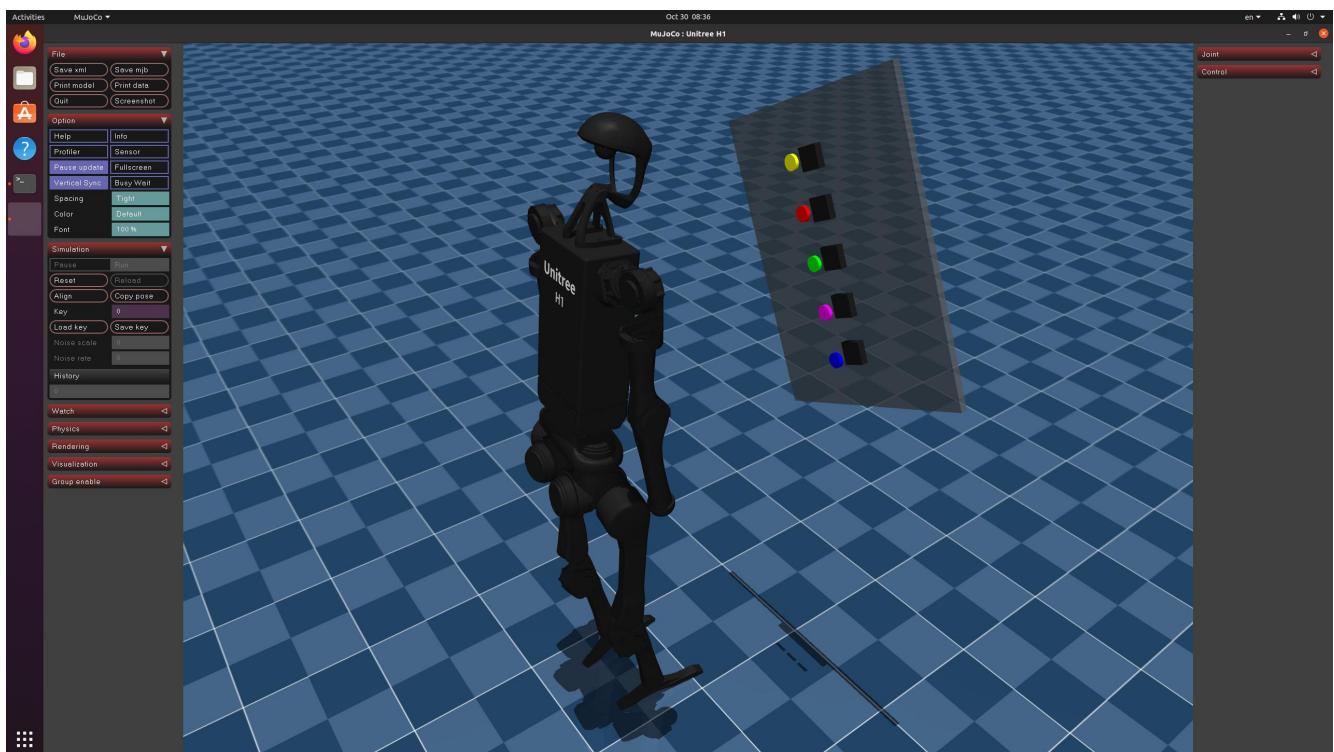


Рисунок 4 - Окно mujoco с роботом Unitree H1 при изменении ракурса.

Если робот не двигается и симуляция как будто остановилась, выполните шаги 8 и 9, а если нет — перейдите к шагу 10.

Шаг 8: Настраиваем подключение

Сейчас симуляция не “общается” с ROS2. Нужно указать правильный сетевой интерфейс и выбрать нужного робота.

1. Откройте файл настроек:

```
cd ~/unitree_mujoco_mirea_olympiad/simulate_python  
nano config.py
```

Он выглядит так:

```
GNU nano 4.8 config.py
ROBOT = "h1" # Robot name, "go2", "b2", "b2w", "h1", "go2w", "g1"
ROBOT_SCENE = "../unitree_robots/" + ROBOT + "/scene_with_lab_stand.xml" # Robot scene
DOMAIN_ID = 0 # Domain id
INTERFACE = "ens33" # Interface

USE_JOYSTICK = 0 # Simulate Unitree WirelessController using a gamepad
JOYSTICK_TYPE = "xbox" # support "xbox" and "switch" gamepad layout
JOYSTICK_DEVICE = 0 # Joystick number

PRINT_SCENE_INFORMATION = True # Print link, joint and sensors information of robot
ENABLE_ELASTIC_BAND = True # Virtual spring band, used for lifting h1

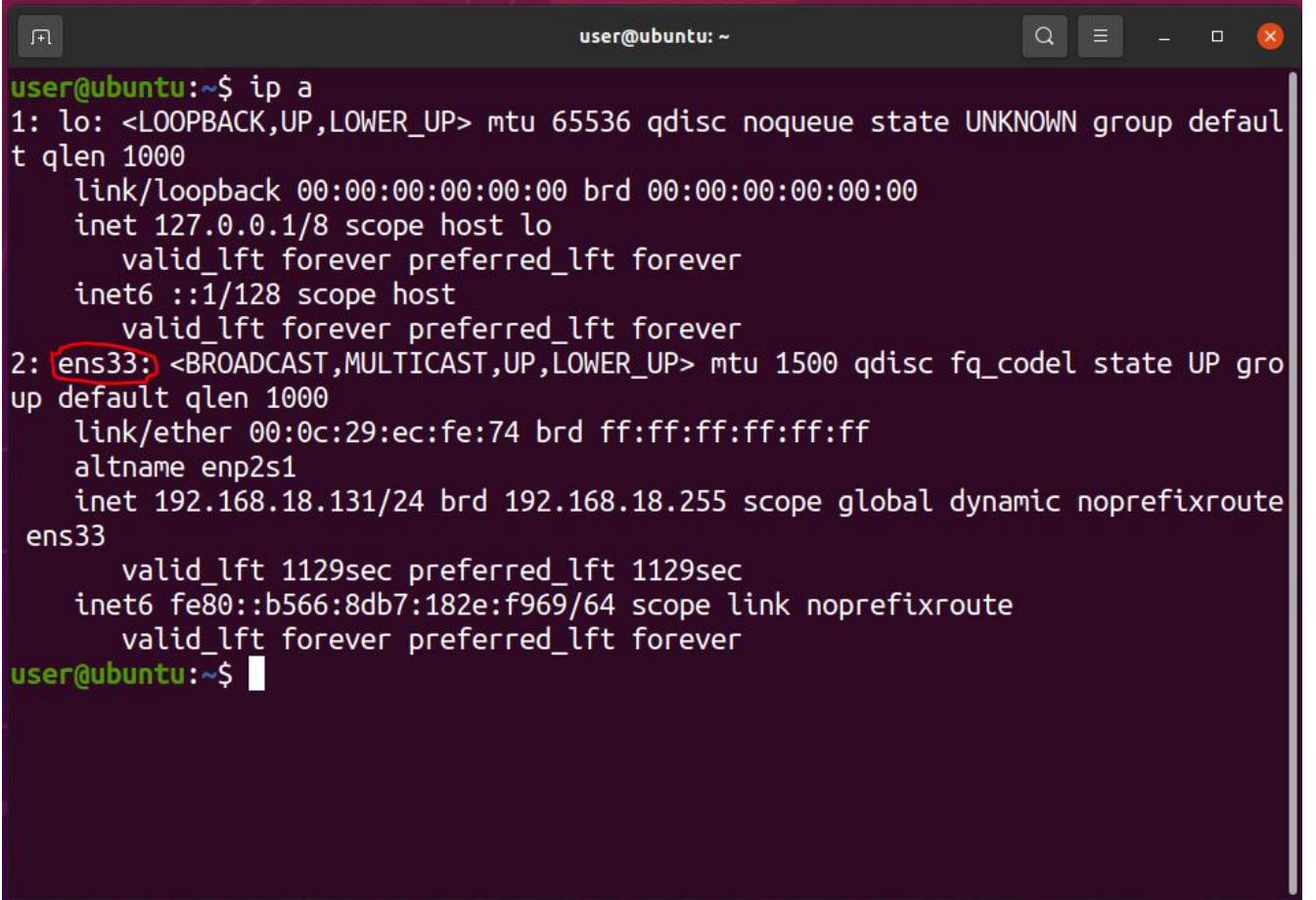
SIMULATE_DT = 0.005 # Need to be larger than the runtime of viewer.sync()
VIEWER_DT = 0.02 # 50 fps for viewer
```

Рисунок 5 - Оригинальная конфигурация запуска MuJoCo.

2. Откройте **новый терминал** и выполните:

```
ip a
```

Найдите активный сетевой интерфейс (обычно wlan0, eth0 или enp...):



The screenshot shows a terminal window on an Ubuntu system. The command `ip a` is run, displaying information about network interfaces. The output shows two interfaces: `lo` (loopback) and `ens33` (Ethernet). The `ens33` interface is highlighted with a red box. The configuration includes MTU, queueing discipline (qdisc), link layer information, and IP addresses (inet and inet6) with their respective subnet masks, broadcast addresses, and lifetimes (valid_lft and preferred_lft).

```
user@ubuntu:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:ec:fe:74 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.18.131/24 brd 192.168.18.255 scope global dynamic noprefixroute ens33
        valid_lft 1129sec preferred_lft 1129sec
    inet6 fe80::b566:8db7:182e:f969/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
user@ubuntu:~$
```

Рисунок 6 - Где посмотреть сетевой интерфейс.

3. Вернитесь в config.py и измените:

- `interface` — укажите имя Вашего сетевого интерфейса (например, `"wlan0"`).

Пример исправленного файла:

e

```
GNU nano 4.8 config.py
ROBOT = "h1" # Robot name, "go2", "b2", "b2w", "h1", "go2w", "g1"
ROBOT_SCENE = "../unitree_robots/" + ROBOT + "/scene_with_lab_stand.xml" # Robot scene
DOMAIN_ID = 0 # Domain id
INTERFACE = "ens33" # Interface

USE_JOYSTICK = 0 # Simulate Unitree WirelessController using a gamepad
JOYSTICK_TYPE = "xbox" # support "xbox" and "switch" gamepad layout
JOYSTICK_DEVICE = 0 # Joystick number

PRINT_SCENE_INFORMATION = True # Print link, joint and sensors information of robot
ENABLE_ELASTIC_BAND = True # Virtual spring band, used for lifting h1

SIMULATE_DT = 0.005 # Need to be larger than the runtime of viewer.sync()
VIEWER_DT = 0.02 # 50 fps for viewer

^G Get Help      ^O Write Out     ^W Where Is      ^K Cut Text      ^J Justify      ^C Cur Pos
^X Exit          ^R Read File     ^\ Replace       ^U Paste Text    ^T To Spell     ^_ Go To Line
```

Рисунок 7 - Необходимая для работы с Unitree H1 конфигурация запуска MuJoCo.

4. Сохраните файл:

В nano нажмите Ctrl+O → Enter → Ctrl+X.

Шаг 9: Запускаем симуляцию снова

```
cd ~/unitree_mujoco/simulate_python
python3 unitree_mujoco.py
```

Теперь робот должен двигаться, и симуляция будет работать корректно, пример в коротком видео по ссылке:

[Unitree H1 в mujoco при корректном запуске](#)

Шаг 10: Настраиваем ROS2 (чтобы появлялись топики)

Чтобы симуляция “разговаривала” с ROS2, нужно убедиться, что транспортный уровень (DDS) работает. Смотреть наличие специальных топиков нужно, очевидно при запущенной симуляции Unitree H1 в mujoco.

1. Добавьте путь к локальным программам в `.bashrc`:

```
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc
```

2. Перезагрузите настройки терминала:

```
source ~/.bashrc
```

3. Проверьте, работает ли CycloneDDS (транспорт ROS2) (mujoco с роботом H1 должна быть запущена, чтобы топики были):

```
cyclonedds ps
```

Должен появиться список процессов:

Host	Application	Pid	Participants	Topics
ubuntu	python3	13875	0110e32d-5edc-9202-2...	rt/wirelesscontrol... rt/lowstate rt/sportmodestate rt/lowcmd rt/parameter_events ros_discovery_info rt/rosout
ubuntu	python3	13904	0110d5d1-5670-9622-c...	

Рисунок 8 - Активные топики (DDS)

4. Посмотрите список ROS2-топиков:

```
ros2 topic list
```

Правильный вывод должен включать топики от робота:

```
/lowcmd  
/lowstate  
/parameter_events  
/rosout  
/sportmodestate  
/wirelesscontroller
```

Если видите только **/parameter_events** и **/rosout** — значит, ROS2-демон не видит симуляцию.

```
/parameter_events  
/rosout
```

Исправление:

```
ros2 daemon stop  
ros2 daemon start  
ros2 topic list
```

Теперь топики должны появиться!