

Crash course on Linux, ORCA software etc.

Yasunori Toshimitsu
PhD Candidate, SRL
05.10.2026



Linux basic commands

Basic commands need to navigate through terminal, aws instance etc.

- **pwd** → Show current directory/path
- **mkdir my_dir** → Make directory directory (fancy name for “folder”)
- **cd my_dir** → Change directory to dir/Enter that directory
- **cd ..** → Go up a directory
- **ls** → List files of current directory

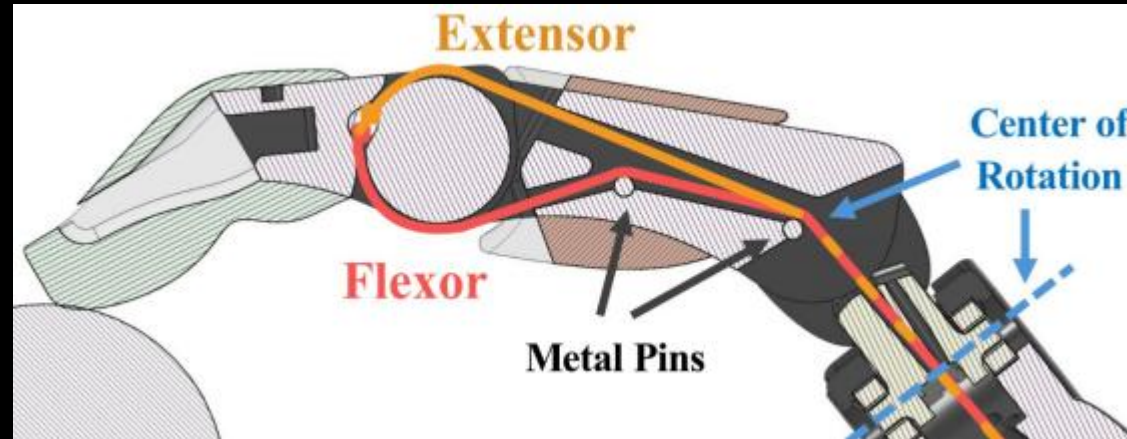
Linux basic commands

Basic commands need to navigate through terminal, aws instance etc.

- `chmod 775 my_file` → Change mode of file to 775 (read write execute permissions)
- `ls /dev/ttyUSB* /dev/ttyACM*` → Find available USB ports
- `sudo` → "superuser do". Run programs with root (administrator) privileges. Use with care!

ORCA Calibration

All joints have hard stops by design → Provide us a known Range of Motion (ROM)



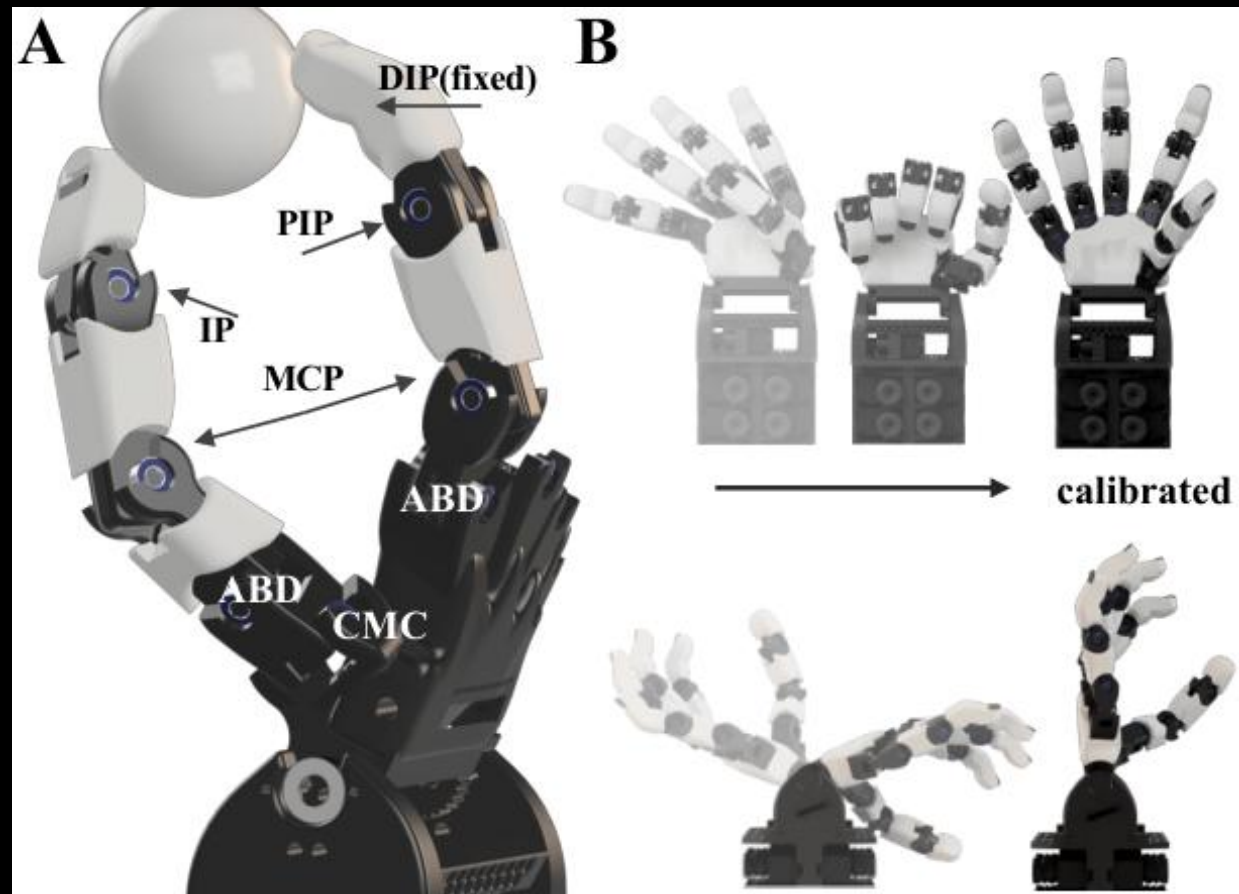
- Tendon wraps around target joint **around a pulley** → linear correlation of servo angle and joint angle
 - Routing is very **close to center of rotation** → each tendon is only affected by target joint
- Simple to control: linear, one-to-one mapping between joint and tendon (and in extension, servo motor)

If you want to know about modeling & controlling complex tendon driven mechanisms, read [this post](#)

ORCA Calibration

Calibration: Move each joint to its physical limits/hard-stops and record ROM of servo. Then we can interpolate: servo angle $\rightleftharpoons$ joint angle

For more details, read the [original ORCA paper \(2025\)](#)



ORCA Calibration know-how

Calibration should be run first time each day, but also after some hours of use or by feel
depends on how aggressive teleop/policy is

Check with TAs (and teach each other) to get a *feel* of how much the tendons should be tensioned
too loose → Joint control accuracy is reduced
too tight → Increases friction and creates “springiness” on fingers.



Nolan et al., 2026

Code Framework



orca_core

- OSS Python library to control Orca hand
- Abstracts complexities to provide simple interface

Python

faive_system

- SRL closed repo to control our robot hands & arms

hand_control_node.py, hand_controller_v2.py

- Acts as wrapper around orca_core
- Interface Orca with ROS 2
- Standardized interface allowing usage with other faive_system programs



Faive Hand (2023)

ROS 2

ROS 2

Control programs

- IL
- RL
- Teleoperation
- Etc.

ORCA / ROS 2 basic commands

Install ROS 2 with robostack

- If on Windows, either dual boot Ubuntu (long term better) or use WSL 2 (short term easier)
- Might work on macOS, haven't tested
- Installing on native Ubuntu is better for the later RL & IL portions of the course

See README of faive_system for installation instructions of the framework

See README of faive_system/src/hand_control for ORCA v2 specific instructions

Basic steps are: Tension → Calibrate → Run controller

rqt_graph to see the ROS 2 communication network

ros2 topic list to list out ROS 2 topic names, **ros2 topic echo <topic_name>** to intercept the messages

Resources

[Robostack - Getting Started](#): official tutorial to set up ROS 2 on Robostack

[Nailing your Robotics project at SRL - and beyond!](#): Ben (mimic founding engineer, SRL alumni)'s kickoff tutorial about good coding / developing practices. Also see his [slides from his RWR talk last year](#)

[Installing ROS2 in a Virtual Environment Using RoboStack](#): my post about setting up a ROS 2 environment

[Yasu's Programming Environment Setup](#): recommendations for creating a comfortable programming environment (mostly for Linux)