

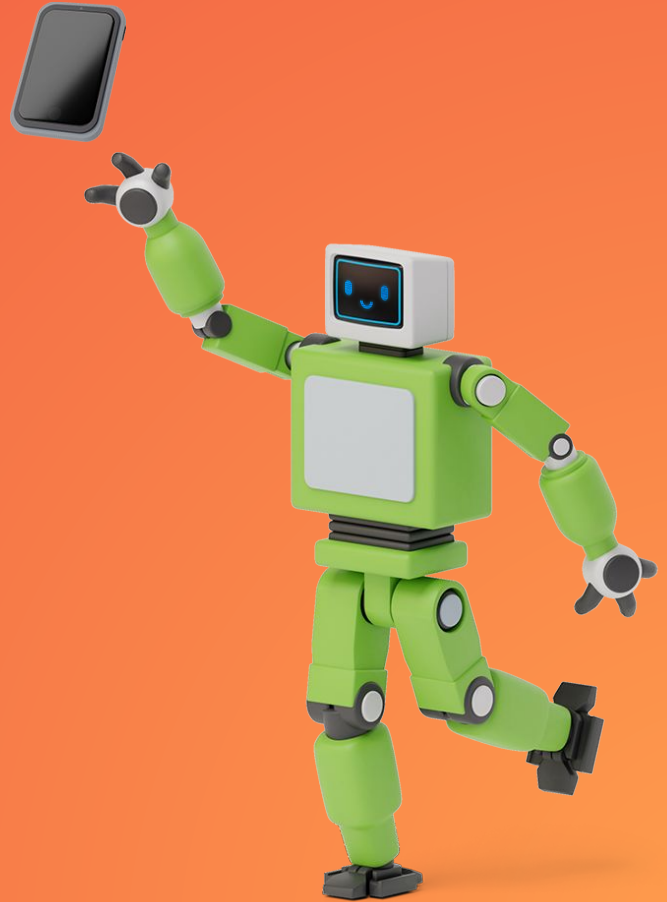
**EECS/BIOE/ME
106A/206A**

LABS 3 & 4



ANNOUNCEMENTS + REMINDERS

- **If you aren't feeling well, don't come to lab**
 - Be respectful to everyone
 - No food/drink in the lab
 - Keep your stations clean
 - Don't work in the lab alone
-
- **Ctrl + C** out of all terminals before leaving
 - Use `kill -u [username]` to log out



ROADMAP

Last Week (9/4-9/8)
Lab 2



This Week (9/11-9/15)
Lab 2 Due
Start Labs 3 and 4



Next Week (9/18-9/22)
Continue Labs 3 and 4



Two Weeks (9/25-9/29)
Buffer Week Labs 3 and 4



You must get checked off for Lab 2 before working on Labs 3 or 4!

You must have completed the Robot Usage Quiz before starting!³

GENERAL ROBOT RESPONSIBILITY

DO:

- **DO** be aware of your surroundings
- **DO** test code in simulation before running it! (if applicable)
- **DO** ask a staff member if you're ever unsure of how to do something!

DON'T:

- **DON'T** make hardware modifications!
- **DON'T** close a terminal without pressing Ctrl+C!
- **DON'T** use robots without a partner!



SAWYER RESPONSIBILITY

DO:

- **DO** be ready to press the E-Stop at a moment's notice so the robot doesn't crash into anything
- **DO** hold the robot arm by the cuff when moving it manually (zero-g) mode
- **DO** be especially careful when handling these robots

DON'T:

- **DON'T** manually move the robot except when holding the cuffs
- **DON'T** get too close to the Sawyer robots!



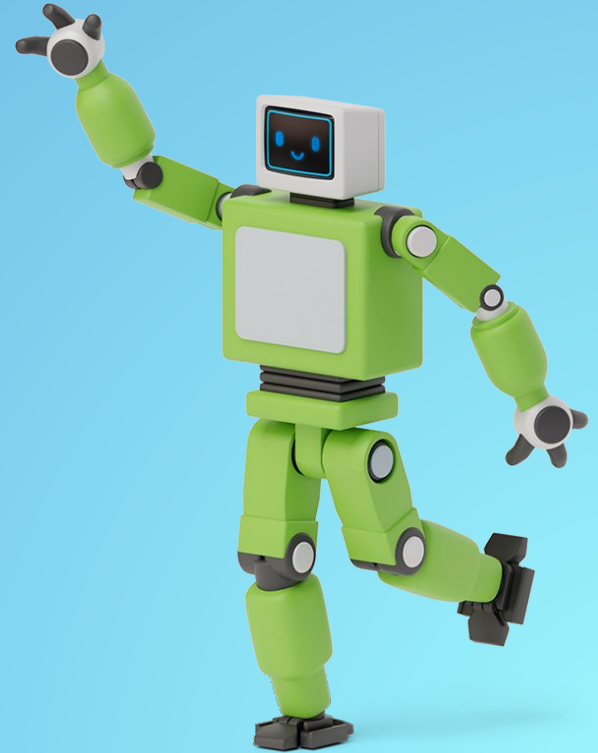
TURTLEBOT RESPONSIBILITY

DO:

- **DO** make sure you're using the correct TurtleBot for your workstation
- **DO** be careful with TurtleBots on the floor!
- **DO** plug in and turn off the TurtleBots when charging

DON'T:

- **DON'T** leave the TurtleBots on the floor after you have finished
- **DON'T** leave the TurtleBots uncharged



LAB 3

Forward Kinematics/Coordinate
Transformations



GOALS

- Practice computing forward kinematics maps
- Show that your FK implementation matches ROS
- Leverage the powerful functionality of tf2
- Control Sawyer to joint-position goals



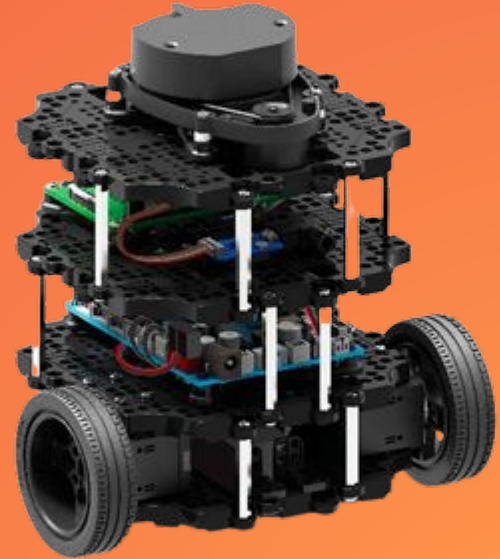
LAB 4

Introduction to TurtleBot



GOALS

- Bringup and control a TurtleBot from keyboard
- Perform SLAM and navigate in the map
- Track the TurtleBot's ARTag with a camera
- Make a TurtleBot track a second ARTag



IMPORTANT INFORMATION

Lab 3:

- Must do pre-lab from HW!
 - Otherwise, this lab will be very painful!
- The `/intera.sh` scripts produce a lot of output - safe to ignore
- Print out your initial joint angles before moving the arm to a different position
- Press the E-Stop immediately if anything looks like it's about to go wrong

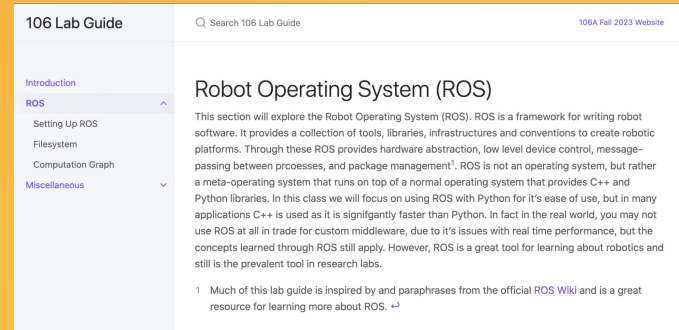
Lab 4:

- Only have one TurtleBot controller (keyboard OR custom) running at a time
 - Otherwise, the two controllers will send conflicting commands!
- Positive X on ARTag is NOT positive X (forwards) on the TurtleBot
 - The lab doc accounts for this already :)
- Pay attention to which commands should be run via SSH vs on workstation
- Turn the TurtleBot off and on again if it's not connecting

LAB USAGE GUIDE IS RELEASED!

<https://ucb-ee106.github.io/lab-guide/>

Very helpful if you have specific questions about lab concepts – goes a lot more in depth than the lab doc!



The screenshot shows the '106 Lab Guide' website. The page title is '106 Lab Guide' and the search bar contains 'Search 106 Lab Guide'. The page is dated '106A Fall 2023 Website'. The left sidebar contains a navigation menu with the following items: 'Introduction', 'ROS', 'Setting Up ROS', 'Filesystem', 'Computation Graph', and 'Miscellaneous'. The main content area is titled 'Robot Operating System (ROS)' and contains the following text: 'This section will explore the Robot Operating System (ROS). ROS is a framework for writing robot software. It provides a collection of tools, libraries, infrastructures and conventions to create robotic platforms. Through these ROS provides hardware abstraction, low level device control, message-passing between processes, and package management¹. ROS is not an operating system, but rather a meta-operating system that runs on top of a normal operating system that provides C++ and Python libraries. In this class we will focus on using ROS with Python for its ease of use, but in many applications C++ is used as it is significantly faster than Python. In fact in the real world, you may not use ROS at all in trade for custom middleware, due to its issues with real time performance, but the concepts learned through ROS still apply. However, ROS is a great tool for learning about robotics and still is the prevalent tool in research labs.' A footnote at the bottom reads: '1 Much of this lab guide is inspired by and paraphrases from the official ROS Wiki and is a great resource for learning more about ROS. ↗'

ANY QUESTIONS?

Help/Checkoff form:

tinyurl.com/fa23-106alab

