# Lab 2: Stretch Plans for Inverses

Zackory Erickson

Carnegie Mellon University
The Robotics Institute

16-762: Mobile Manipulation
Spring 2026

# Lab 2 Goals

**Part #1**

Implement IK in ROS 2, extend it with an additional joint, and deploy your implementation to make contact with objects.

**Part #2**

Implement motion planning (with MoveIt 2) in Python to make contact with objects and plan paths around obstacles.

# What you will submit

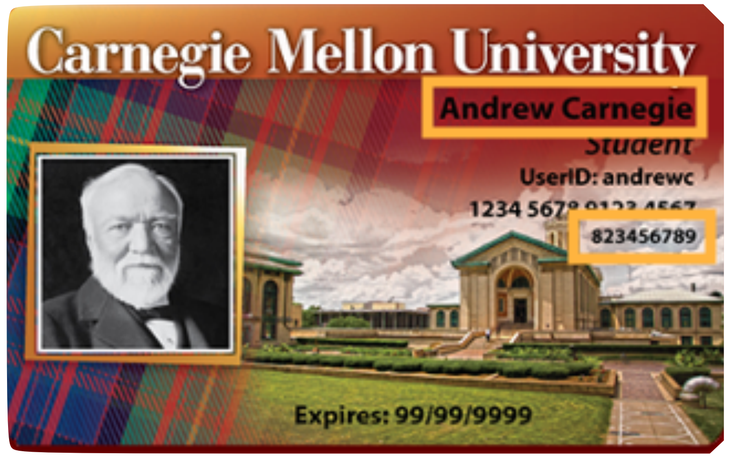
1. Your completed code files (.py) for parts #1 and #2 (see previous slide)
    - Part #1: `ik_ros.py`
    - Part #2: `moveit2.py`
2. Two videos (.mp4) for part #1. One video showing IK to touch an object, and another video showing IK for “something cool”.
3. Video (.mp4) for part #2, showing the robot reach each of the 4 configurations shown in the slides.

# Lab 2 is Teams

- If you don’t already have a team, find one today! In-person or on Canvas Discussions
- Due Date: **Wednesday, February 18th** (2 weeks from now)

# Send your IDs to me

- To get access to the AI Maker Space.
- 9 digit ID on front of your CMU card. (e.g. 822500909)
- Not your Andrew ID.




**https://forms.gle/5LavZWFW6xp9Fuif7**

# Setup a folder on the robot for your team code

```
cd 16762/
mkdir your_name
cd your_name

# Create a Python virtual env (that way your pip installs don't break other people's code)
python3 -m venv env
source env/bin/activate
pip3 install --upgrade pip
pip3 install hello-robot-stretch-body
pip3 install numpy==1.26.4 opencv-contrib-python==4.10.0.84 opencv-python==4.11.0.86

# Create your own GitHub repo.
git clone https://github.com/YourTeamName/YourRepoName.git
```

# Reminder: When you turn on the robot

Put the battery into **SUPPLY** mode.

Run:

```
cd 16762/your_name
source env/bin/activate
stretch_robot_home.py
```

# Reminder: How to connect into the robot

- We have two robots — 3159 and 3160. These numbers are on the back of the robot near the on/off switch.
- Username: hello-robot
- Password: 16762cmu!
- stretch-se3-3159:
    - **SSH**: ssh -X hello-robot@stretch3159.wifi.local.cmu.edu
    - **Anydesk**: 1943010008
- stretch-se3-3160:
    - **SSH**: ssh -X hello-robot@stretch3160.wifi.local.cmu.edu
    - **Anydesk**: 1424568870

# Scheduling time with the robot

- We have a Google Calendar for each robot, which you have all been added to.
- For this **lecture you can reserve max 15 minutes** on the robot.
- In the AI Maker Space, max **3 hour** time block, then let another team use the robot.
- If at the end of your time block no other team has reserved the robot, then you can reserve for another 3 hours. (**Please don't abuse this**)

# Must be in-person to connect to robot

- Do not connect to a robot unless you are right next to the robot.
- **You cannot use a robot remotely. That is not safe.**
- Only connect to Anydesk if you have reserved the robot on Google Calendar and you are sitting next to the robot.

# Part #1: IK in ROS 2

Make a copy of this file and call is `ik_ros.py`

https://github.com/Zackory/mm2026/blob/main/stretch_python/stretch_ik.py

# Part #1: IK in ROS 2

Implement `stretch_ik.py` in ROS 2, call is `ik_ros.py`

In your ROS 2 implementation, add a rotation joint before the translation joint. IK should now be able to plan for robot base rotation, translation, arm lift and extend, and roll, pitch, yaw of the gripper.

Place an object on the ground, roughly 0.5 meters in front of the robot (in front of the laser range finder)

Use your IK implementation to move the end effector to touch the object. The robot base will have to rotate to achieve this.
(In the future we will use vision to perceive and grasp an object)

Now, call IK multiple times to do something fun with the robot. You define what fun is. At the end of your `ik_ros.py` script, you will add at least 3 more new function calls `move_to_grasp_goal(new_target_point, new_target_orientation)`. In essence, you are commanding the robot to move to 3+ new end effector poses, one after another.

# Solution

# Part #2: MoveIt 2 in Python

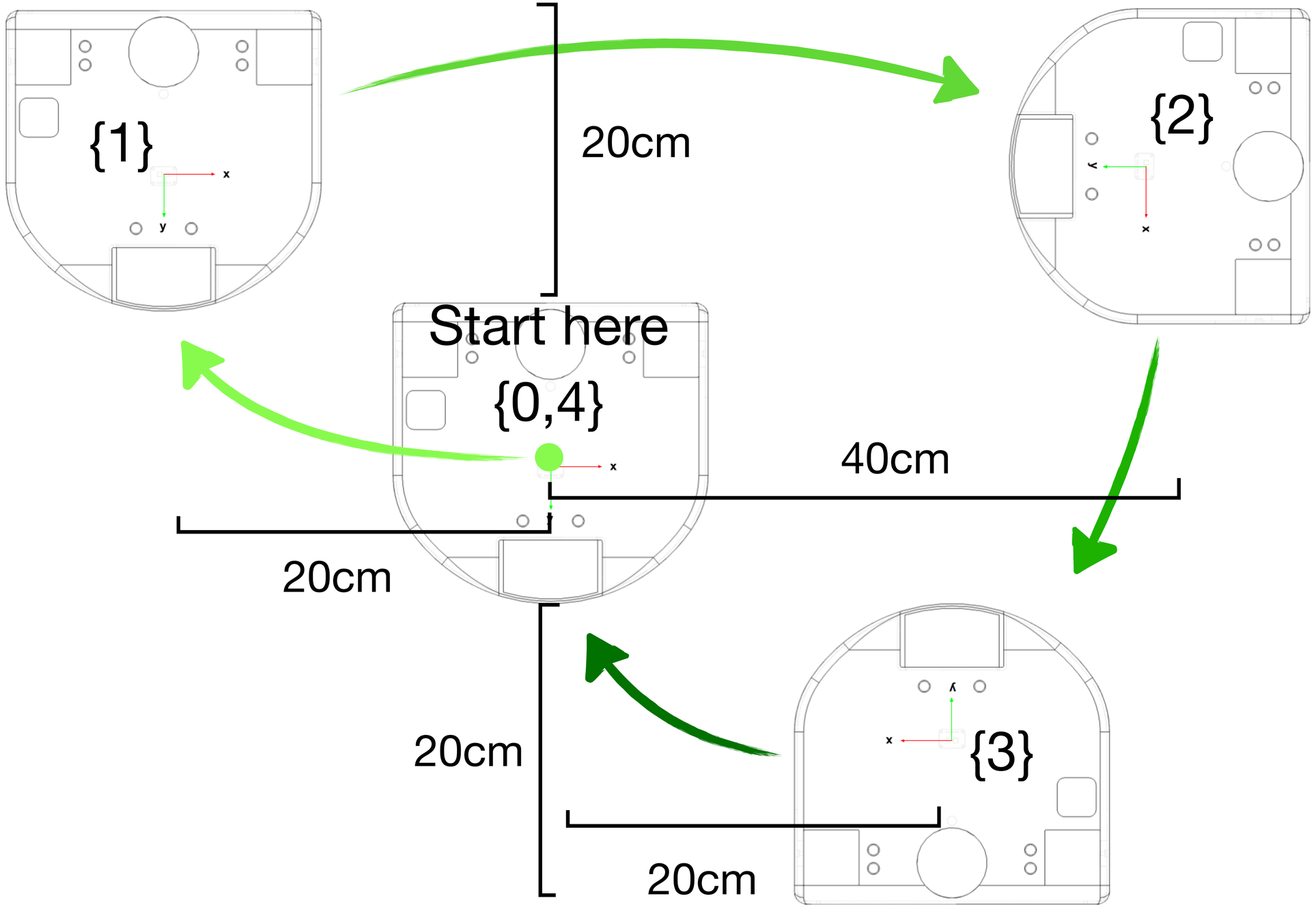
Code: https://github.com/Zackory/mm2026/tree/main/lab2

The objective of Part #2 is to interface with MoveIt 2 through Python and observe the errors that can occur if you use dead reckoning with motion planning. In following lectures and labs we will start leveraging sensors to mitigate compounding errors.

You will code a MoveIt 2 trajectory for Stretch to follow the four configurations shown on the next slide.

You have been provided moveit2.py and moveit2_utils.py files. You should only need to edit the moveit2.py file. The file currently demonstrates how to drive forward 0.3 meters.

- Between poses 0 and 1, the robot will lift its arm to 0.5 m.
- Between poses 1 and 2, the robot will extend its arm to 0.4 m.
- Between poses 2 and 3, the robot will rotate it's wrist 45 degrees along each of the 3 axes.
- Between poses 3 and 4, the robot will bring all of it's arm motors back in to the stow pose.


{1}
x
y
20cm
{2}
y
x
Start here
{0,4}
x
40cm
20cm
20cm
y
x
{3}
20cm

# Solution

**An example path.**

**Your robot may take a different path**