Lab 4: Jacobians and Velocity Kinematics

MEAM 520, University of Pennsylvania

October 23, 2020

This lab consists of two portions, with a pre-lab due on **Friday, October 30, by midnight (11:59 p.m.)** and a lab report due on **Friday, November 6, by midnight (11:59 p.m.)**. Late submissions will be accepted until midnight on Monday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Piazza to request an extension if you need one due to a special situation.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. When you get stuck, post a question on Piazza or go to office hours!

Individual vs. Pair Programming

Work closely with your partner throughout the lab, following these guidelines, which were adapted from "All I really needed to know about pair programming I learned in Kindergarten," by Williams and Kessler, *Communications of the ACM*, May 2000. This article is available on Canvas under Files / Resources.

- Start with a good attitude, setting aside any skepticism, and expect to jell with your partner.
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot) while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every 30 minutes, *even if one partner is much more experienced than the other.* You may want to set a timer to help you remember to switch.
- If you notice an error in the equation or code that your partner is writing, wait until they finish the line to correct them.
- Stay focused and on-task the whole time you are working together.
- Take a break periodically to refresh your perspective.
- Share responsibility for your project; avoid blaming either partner for challenges you run into.
- Recognize that working in pairs usually takes more time than working alone, but it produces better work, deeper learning, and a more positive experience for the participants.

1 Pre-lab Tasks (due October 30, 5 pts)

Directions: For the pre-lab component of this lab, you turn in your own **individual** work. Show your work to receive full credit. These calculations should be typed or written legibly. Submit a **pdf** on Gradescope containing your work.

1. Derive the forward linear velocity kinematics for the Lynx robot for when you are tracking the end effector. (Note, you do not need to do all the computations by hand, but you should explain your steps and your final result. Your submission should include a copy of any code you write for this step.)
2. Do a sanity check: Let's say the Lynx starts in the zero position and only one of the joints moves. What do you expect the corresponding velocity of the end effector to be? Does your FK reflect this velocity?

2 Lab (due November 6, 45 pts)

The remainder of the lab should be done with a partner. You may work with anyone you choose, but you must work with them for all parts of this assignment. You will both turn in the same report and code (see Submission Instructions), for which you are jointly responsible and you will both receive the same grade.

2.1 Methods

1. Let's say that you are tracking the position and orientation of joint i on the robot (where $i = 0$ is the base, $i = 1, \dots, 5$ is a joint, and $i = 6$ is the center of the gripper). The robot is currently in configuration q and the joints are moving with velocity $\dot{q}$. What are the linear and angular velocity of joint i in the world frame?
2. Let's say that you are tracking the position and orientation of joint i on the robot and would like the joint to move with a linear velocity ${}^0\mathbf{v}$ and angular velocity ${}^0\omega$, both expressed in the world frame. The robot is currently in configuration q. What joint velocities should the robot execute to move joint i in the desired direction?

2.2 Coding

2.2.1 Simulation Environment Updates

We've made a few tweaks to the simulation environment, and you must take the following steps to have an up to date simulator. **These steps are needed for all students!**

1. **Update the Gazebo simulator:** On the Virtual Machine, open a terminal and run the command

   ```
   cd ~/meam520_ws/src/meam520_sim && git pull
   ```

   This will update the code on your machine to match the current version.
2. **Update the Core:** From Canvas, redownload the `Core.zip` file for your respective language and extract it in the same location as for Lab0.
3. **(Gazebo Pro Tip:)** Right-click on the robot and select 'Move To' to automatically zoom your view into the robot. Hold shift, click the robot's base, and drag to reorient the view.

To launch the simulation simply use:

```
$roslaunch al5d_gazebo lab4.launch
```

2.3 Your Tasks

Download the file `lab4.zip` attached to this assignment. The zip file contains two empty functions, which you should fill in

1. `FK_velocity`:
    - Inputs:
        - `q` - a 1×6 vector corresponding to the robot's current configuration
        - `dq` - a 1×6 vector of joint velocities
        - `joint` - an integer $\in [0, 6]$ corresponding to which joint you are tracking (with 0 being the base and 6 being the end-effector)
    - Outputs:
        - `v` - the resulting linear velocity of the joint in the world frame
        - `omega` - the resulting angular velocity of the joint in the world frame
2. `IK_velocity`:
    - Inputs:
        - `q` - a 1×6 vector corresponding to the robot's current configuration
        - `v` - the desired linear velocity of the joint in the world frame
        - `omega` - the desired angular velocity of the joint in the world frame
        - `joint` - an integer $\in [0, 6]$ corresponding to which joint you are tracking (with 0 being the base and 6 being the end-effector)
    - Outputs:
        - `dq` - a 1×6 vector of joint velocities according to:
            * If it is possible to achieve the exact input velocities `v` and `omega`, then `dq` should be the joint velocities that do so.
            * If it is not possible to achieve the exact input velocities `v` and `omega`, then `dq` should be the joint velocities that minimize least squared error between the resulting and desired velocities.
            * If any of the inputs in `v` or `omega` contain `NaN`, then that velocity is unconstrained and can be anything. For example, `v=[1,NaN,NaN];omega=[1,0,0]` means that we want the end effector to move with angular velocity `[1,0,0]` rad/s while translating in the x direction, but the translation in the y and z direction does not matter.
3. Not required, but probably helpful: Write a function `calcJacobian` that takes inputs `q` and `joint` and calculates the corresponding linear and angular velocity Jacobians.

The zip also contains `calculateFK_sol` code for the Lynx.

Additionally, the zip contains a script `TestVelocity_Sim`. This script is an example script for how to send a velocity command to the simulated robot.

2.4 Evaluation

Evaluate your expressions to check that they are correct. Some tests to consider are:

- Start in the zero position and assume only one of the joints moves. What do you expect the corresponding velocity of the end effector to be? Does your FK reflect this velocity?

- Using geometric intuition, where are the singularities? Under what conditions do you expect there to be no solutions to the velocity IK? Explain. Are these configurations reflected in your mathematical expressions?
- What trajectory does the end effector trace out when all joints are moving at a constant velocity?
- What joint angle trajectories should the robot execute to have the end effector follow a straight line? A circle? What if you also want to control the orientation?

2.4.1 Analysis

Discuss the Lynx forward and inverse velocity kinematics in the context of your data and observations. Do your results make sense? What movements is the robot good at? What movements is it bad at? Under what circumstances might you want to use velocity control over position control (which you have done in previous labs)?

3 Submission Instructions

Submit the assignment. One person from each pair should submit code and a pdf copy of the report to the Gradescope assignment for Lab 4. After selecting the files and uploading them, the website will take you to the next page, where in the top right corner you should add your group members. If you do not add your group members they will not get credit for the assignment.

3.1 Report

The format of the report is up to you, but you should make sure that it is clear, organized, and readable. The report should include:

1. Your answers to the conceptual questions in the Methods section, typed or legibly hand-written.
2. A short 1-pg description of how the concepts are incorporated into your code. Include pointers to important line numbers of subfunctions in your code. This will help the graders understand and provide feedback on your work. (This description can be bulleted. No need to use full sentences.)
3. Your experimental results, including a description of your experimental setup (i.e., what were your inputs) and collected data.
4. Your analysis comparing expectations against reality and extrapolations to general conclusions you would make from this lab.

3.2 Code Submission

Your code should be cleaned up so that it is easy to follow. Remove any commented-out commands that you are not using, and add comments to explain the tricky steps. Clearly indicate which parts of the code correspond to which parts of the lab. Your code submission should include:

1. Your `FK_velocity`.
2. Your `IK_velocity`.
3. Any additional functions needed to run your code not included in the original code.

Each file should be attached separately to Gradescope. Do **not** zip them into a single file attachment.