Assignment 2: Jacobians

Robot Kinematics and Dynamics

Prof. Jeff Ichnowski
Shahram Najam Syed
Yuemin Mao

Contents

1 Overview

This assignment reinforces the following topics:

- Jacobians

2 Background

2.1 The Jacobian

The Jacobian transforms differential joint motions to differential changes in the workspace position and orientation of the robot. This means that it can be used to determine the instantaneous linear and angular velocity of a point on the robot given the current joint angles and joint velocities.

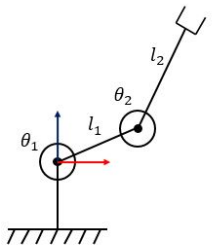
More formally, the Jacobian is a matrix which provides a linear mapping between joint velocities and motion of a coordinate frame attached to the robot (often the end-effector frame). Denoted J, we find it by taking the derivative of the forward kinematics to that frame. To be more concrete, the Jacobian fulfills the following expression

$$\dot{X} = J\dot{\Theta},$$

where $\dot{X}$ is the velocity of the frame in consideration ($\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$ or $\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$ depending on the workspace you are considering) and $\dot{\Theta} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix}$.

Incidentally, this also means the dimension of the Jacobian for a robot with n joints whose workspace is $SE(2)$ is $3 \times n$. Furthermore, we generally refer to $\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$ as the *linear velocity*, whereas $\dot{\theta}$ is generally referred to as the *angular velocity*[1].

First, we need the forward kinematics of a generic RR arm. Let f be the general expression for the forward kinematics of a given arm. We often denote f_x as the forward kinematics with respect to the base frame's x axis. Consider the following robot.



The forward kinematics (from the depicted base to end effector frame) of this arm are

$$\begin{aligned} f_x &= l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \\ f_y &= l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \\ f_\theta &= \theta_1 + \theta_2 \end{aligned}$$

A note on finding the angular component of the forward kinematics in 2D: this can generally be done by inspection; if joint i is a rotational joint, θ_i is added, and if it is prismatic, it does not contribute to the end-effector orientation.

[1] angular velocity is a measure of the speed of rotation, and has units radians/second

Alternatively, if one computes the full homogeneous transform $H^{\texttt{base}}_{\texttt{end effector}}$, the translation components will give f_x and f_y, and the rotation matrix component can be used to extract f_θ given the definition of a rotation matrix $\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$.

We compute the Jacobian by taking the partial derivatives of the forward kinematics with regard to each component. For example, the partial derivative of f with respect to a variable x is computed by treating all the other components as constant, and taking the first-order derivative of f with respect to x. This is denoted $\frac{\partial f}{\partial x}$.

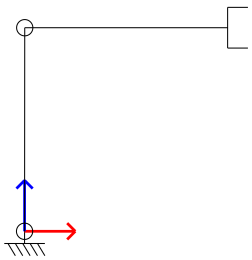
To find J (for the robot in the previous figure), we take the partial derivative of the forward kinematics with regard to each joint angle[2].

$$J = \begin{bmatrix} \frac{\partial f_x}{\partial \theta_1} & \frac{\partial f_x}{\partial \theta_2} \\ \frac{\partial f_y}{\partial \theta_1} & \frac{\partial f_y}{\partial \theta_2} \\ \frac{\partial f_\theta}{\partial \theta_1} & \frac{\partial f_\theta}{\partial \theta_2} \end{bmatrix} = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1+\theta_2) & -l_2 \sin(\theta_1+\theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1+\theta_2) & l_2 \cos(\theta_1+\theta_2) \\ 1 & 1 \end{bmatrix}$$

2.2 Singularities

Informally, a *singularity* is a joint configuration in which a serial mechanism loses a degree of freedom. For example, the end effector of an RR arm in the following configuration can move instantaneously in the x direction, but not in the y direction.

However, in the following configuration, which is away from a singularity, the end effector can move instantaneously in any planar direction via some differential joint motion.



More formally, the singularity concept can be tied to the **rank** of the Jacobian. The rank of a matrix is the dimension of the space spanned by its rows (or columns). Therefore, for the

[2]Typically, for 2-DOF systems, we only consider the x and y terms of the Jacobian; however, this will be described in each question.

Jacobian, the rank is the dimension of the space of possible velocities that can be achieved at the given configuration. Computing the Jacobian (considering x and y rows only) for these two configurations gives the following:

$$\begin{bmatrix} (-l_1 - l_2) & -l_2 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -l_1 & 0 \\ l_2 & l_2 \end{bmatrix}.$$

For the first (singular) configuration, the vectors defined by the columns are parallel—they both point in the $-x$ direction—and therefore they span a 1-dimensional space. For the second, the vectors defined by the columns are at different angles ($\pi/4$ rad or 45 degrees), and therefore span a 2-dimensional space.

Using this terminology, a singular configuration can be thought of as one where the Jacobian loses rank compared to other configurations of the robot. Or as we intuitively noted above, where the kinematics are instantaneously more limited.

2.3 Forces, Torques, and Wrenches

The derivation of the Jacobian is based on differential motions, and therefore naturally relates velocities. However, it also provides a relationship between forces the robot can apply and joint torques. Before defining this relationship, we should formalize the definition of these terms.

A **torque** or **moment** is a force that causes rotation. Whereas a force is applied *along* a line, a moment can be thought of as being applied *about* a line—i.e., causing an object to rotate about that line.

The first concept we will introduce is that of a **wrench**. A wrench is a generalized notion of force. A wrench is a force applied to a body along a given line, and a moment about that line (a torque). In this class, we will represent a wrench by a column vector with the x and y force components and a moment (τ):

$$\begin{bmatrix} f_x \\ f_y \\ \tau \end{bmatrix}.$$

During the course, we will often speak of the wrench that is applied to the robot joints as a *joint torque*, and the wrench applied by the end effector to the world or by gravity to the links as the *end-effector force* or *force due to gravity*. Technically, these are all wrenches, although due to the prevalence of revolute joints, you will often hear phrases such as "the joint torques which cause this end-effector force."

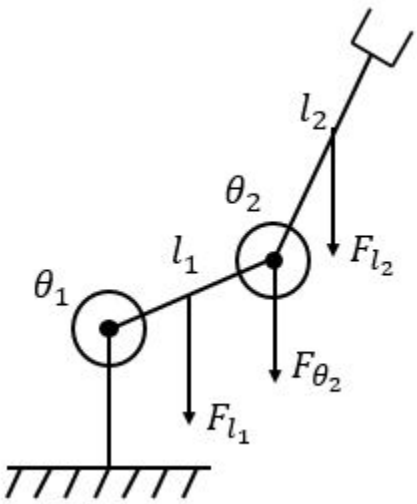
2.4 Jacobian Transpose

A second relationship that the Jacobian provides is between the wrench applied by or on the robot and the joint torques. More formally, the transpose of the Jacobian J^T provides a linear mapping between the wrench at a given location on the robot and the joint torques which generate this wrench:

$$\begin{bmatrix} \tau_{\theta_1} \\ \cdots \\ \tau_{\theta_n} \end{bmatrix} = J^T \begin{bmatrix} f_x \\ f_y \\ \tau \end{bmatrix}.$$

Here, the wrench $\begin{bmatrix} f_x & f_y & \tau \end{bmatrix}^T$ is applied at the point that J was defined from. For example, the Jacobian derived from the kinematic map to the end effector can be used to compute the joint torques used to apply a given force at the end effector (e.g., lift an object of a given weight). Also, the Jacobian derived from the kinematic map to the center of mass of a link can be used to compute the joint torques necessary to counteract the effect of gravity on that link.

Joint torques can be added together to generate multiple forces due to the principle of superposition. This idea can be used to make a robot counteract the effect of gravity, making the robot seem weightless. To do this, one must simply compute the required torques to counteract the effect of each center of mass and then add these torques together. For instance, in the figure below, the Jacobian is calculated to the center of mass of l_1, l_2, and θ_2. Using these Jacobians, the joint torques needed to counteract each force are calculated. Lastly, for each joint, the calculated torques are added together. $\tau = \tau_{l_1} + \tau_{\theta_2} + \tau_{l_2}$.



2.5 Course Logistics

In this class you would be using the following websites regularly:

- Piazza: This is the best way to ask course staff questions, and see course announcements.
- Gradescope: Create an account using your CMU email. You should have already been added to the course. If you haven't, please contact the course staff via email or Piazza post.

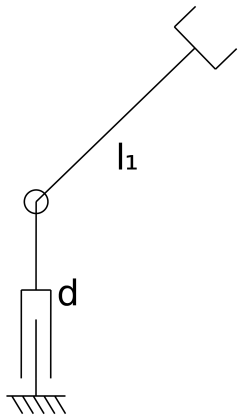
3 Instructions

- The deadline for this assignment is 19th September, 2024 09:00 P.M.
- Zip your code into a single file named `<AndrewId>.zip`. See the complete submission checklist at the end, to ensure you have everything. Submit your PDF file to Gradescope.
- Each question (for points) is marked with a **points** heading.
- **Start early!** This homework may take a long time to complete.
- **During submission indicate the answer/page correspondence carefully when submitting on Gradescope.** If you skip a written question, just submit a blank page for it. This makes our work much easier to grade.
- If you have any questions or need clarifications, please post in Piazza or visit the TAs during the office hours.
- Unless otherwise specified, **all units are in radians, meters, and seconds**, where appropriate.

4 Theory Questions

1) Jacobian

Given the PR Arm illustrated below from Assignment 1.



a. [5 points] Determine the Jacobian of the arm.

b. [5 points] Let $d = 0$ and $\theta = 0$ and $\dot{d} = \dot{\theta} = v$, what is the contribution of each joint to the end-effector's linear velocity? To the angular velocity?

c. [5 points] For this problem, assume that forces due to gravity are pointing in the negative y direction. Assume each **link** has a mass of 0.25 kg, and that each link's center of mass can be assumed to be in the center of that link.

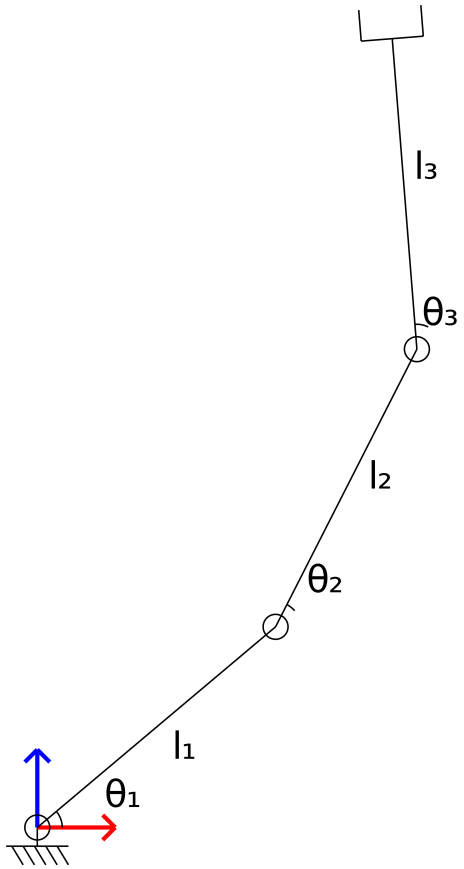
   (i) What are the torques (as a function of the configuration $\Theta = \begin{bmatrix} d \\ \theta \end{bmatrix}$) necessary to counteract the effect of gravity on the center of mass of link 1?

   (ii) Of link 2?

   (iii) What torques are necessary to counteract the effect of gravity on all both links at the same time?

2) Robot Analysis: RRR

Consider the following robot (assume no joint limits, and that the links do not intersect with the ground):



a. [2.5 point] What is the dimension of the end-effector Jacobian? Assume the workspace is $SE(2)$.

b. [5 points] What does each row of J intuitively mean (in 1–2 sentences)?

c. [5 points] What does each column of J intuitively mean (in 1–2 sentences)?

d. [5 points] Find f(the forward-kinematic map to the end effector) for this arm for x, y, and θ as a function of θ_1, θ_2, and θ_3.

e. [2.5 point] Find the partial derivatives of the previous answers with regard to θ_1, θ_2, and θ_3. Using these partial derivatives, what is J?

f. [7.5 points] Let v be some nonzero positive scalar. When the arm is at position $\theta_1 = \theta_2 = \theta_3 = 0$, and $\dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3 = v$, what is the contribution of each joint to the end-effector's linear velocity? To the angular velocity?

g. [7.5 points] At configuration $\theta_1 = \theta_2 = \theta_3 = 0$,

(i) what $\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix}$ must be applied to instantaneously move the end-effector in the planar direction $\begin{bmatrix}-1\\ 0\end{bmatrix}$?

(ii) $\begin{bmatrix}0\\ 1\end{bmatrix}$?

(iii) Now considering rotation, what joint velocities must be applied to move instantaneously in $\begin{bmatrix}0\\ 1\\ 1\end{bmatrix}$ (where the third term is rotation)?

h. [7.5 points] What joints torques are necessary to exert a 5 N force on the end effector in the $\begin{bmatrix}1\\ 1\end{bmatrix}$ direction. Assume the configuration is $\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix}$.

i. [12.5 points] For this problem, assume that forces due to gravity are pointing in the negative y direction. Assume each **link** has a mass of 0.25 kg, and that each link's center of mass can be assumed to be in the center of that link.

(i) What are the joint torques (as a function of the configuration $\Theta = \begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix}$) necessary to counteract the effect of gravity on the center of mass of link 1?

(ii) Of link 2?

(iii) Of link 3?

(iv) What torques are necessary to counteract the effect of gravity on all three links at the same time?

j. [5 points] Now assume the **joints** also each have mass; the joint mass is 0.35 kg, centered on the axis of rotation. What are the new set of torques needed to counteract the

effect of gravity on the entire robot (links and joints), as a function of the configuration $\Theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$?

(i) Joint 1

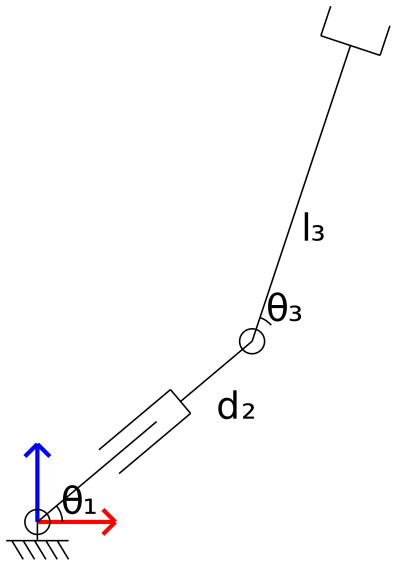
(ii) Joint 2

(iii) Joint 3

(iv) Everything all together (links and joints)

3) Robot Analysis: RPR

Consider the following robot (assume no joint limits, and that the links do not intersect with the ground):



Find the following:

a. [5 points] The forward kinematic map for the end effector.

b. [7.5 points] The Jacobian for the end effector.

c. [12.5 points] The joint torques necessary to counteract the effects of gravity (where gravitational forces are in the negative y direction). For this, assume the entire length between the rotational joints is d_2, and this consists of a mass of 1.0 kg centered at $d_2/2$ along this link. The rotational joints have masses of 0.2 kg centered at their axis of rotation, and the distal link (of length l_3) has a mass of 0.5 kg centered halfway along the link. The end effector is massless. Show your work by writing down the Jacobian and force vector.

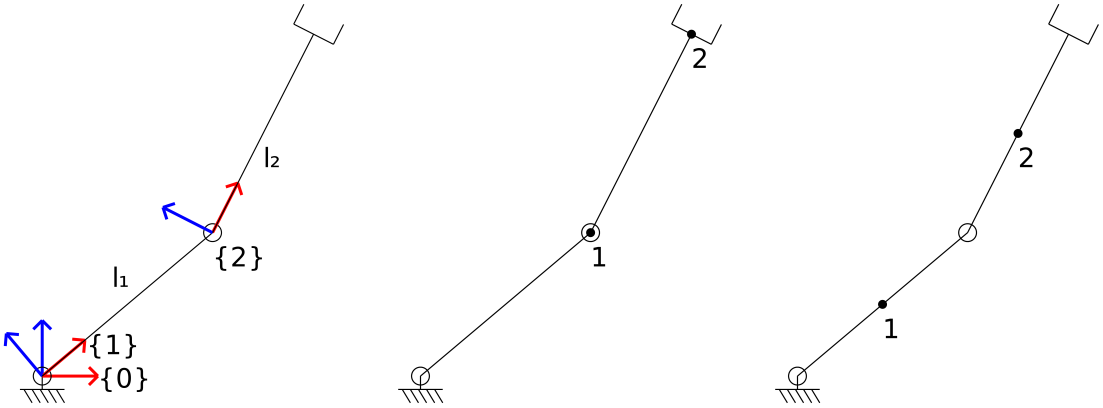
   i. Rotational joint (θ_1)
   ii. Prismatic Joint (d_2)
   iii. Rotational joint (θ_2)
   iv. Link 2 (l_3)
   v. Everything all together (links and joints)

5 Code Questions

In these problems, we'll analytically test some of the work from the previous section.

1) Jacobians and Velocities *25 points*

Begin by opening the `ex_01` directory in your Python environment. In this exercise, you must fill out several Python files. When completed correctly, the first three functions will define the forward kinematics and Jacobian to several points on an RR arm; the final function will provide a self-check for the first three functions.



- `forward_kinematics_RR.py`: As in Exercise 1 from Assignment 1, the forward kinematics to several frames should be filled in. Note: the frames for this problem are a subset of those from the previous assignment. In the code, `H_1_0` and `H_2_0` refer to H_1^0 and H_2^0 respectively (where $\{\mathbf{0}\}$, $\{\mathbf{1}\}$, and $\{\mathbf{2}\}$ are shown in the above left figure).
- `jacobian_link_ends_RR.py`: Jacobian matrices to the end of the robot links should be filled in in this problem. In the code, `J_1` and `J_2` refer to the Jacobians to the end of link 1 and 2 respectively (i.e., points 1 and 2 in the above middle figure).
- `jacobian_coms_RR.py`: Jacobian matrices to the centers of the robot links should be filled in in this problem. In the code, `J_1` and `J_2` refer to the Jacobians to the center of link 1 and 2 respectively (i.e., points 1 and 2 in the above middle figure).
- `sample_path.py`: This function loads data taken from the robot, and draws the path taken by the center and end of each link. You should use the functions you have written above to fill in the position and velocity of these points.

After completing these tasks, run the `sample_path.py` function. You should see four separate lines, representing the path of the center and end of each link. These lines should be formed of arrows, which represent the velocity of each of these points. **Important:** if the arrows are not tangent to the path, there is an error in your code! Verify your code is correct before continuing. As before, when debugging, remember to use simple test cases such as $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ or $\begin{bmatrix} 0 \\ \frac{\pi}{2} \end{bmatrix}$ to check your matrices.

The deliverables for this problem are the four Python files mentioned above, as well as the resulting figure from running `sample_path.py`. This figure should be saved as `path.png`.

Additionally, `sample_path.py` will generate a `results.pkl` file containing the calculated data for each part of the robot. This file will be used for grading, so ensure it is generated correctly.

2) Forces and Gravity Compensation *25 points*

Begin by opening the `ex_02` directory in your Python environment. In this exercise, you must use the Jacobians found in the first problem to compute joint torques which result in forces applied by the robot. This will involve completing two functions:

- `get_joint_torques.py`: In this function, you must determine the joint torques necessary for applying a given force with the end effector (e.g., centered at the end effector origin in a given direction with a given magnitude).
- `get_grav_comp_torques.py`: In this function, you will compute the joint torques which compensate for the force of gravity on each part of the robot, given the current joint angles. Note the direction and magnitude of acceleration due to gravity is passed into this function (assume units of m/s^2).

After completing these functions, run the `sample_torques.py` script. This script will generate plots of your computed torques for a test run of the robot. Two figures will be produced: one containing torques for applying a force with the end effector, and the other containing torques for compensating for gravity.

The deliverables for this problem are the two completed Python files above, as well as the resulting figures from running `sample_torques.py`. These figures will be automatically saved as `torques_ee.png` and `torques_grav_comp.png`, respectively. Please also include copies of these figures in your writeup!

Additionally, the `sample_torques.py` script will save your computed torques as pickle files (`tau_desired_force.pkl` and `tau_grav_comp.pkl`). These will be used for grading purposes, so ensure they are generated correctly.

Note: The `sample_torques.py` script uses functions from the `ex_01` directory. Make sure your Python environment is set up correctly to access these functions.

3) Submission

To submit, run `create_submission.py`. It will first check that your submitted files run without error, and perform a small sanity check. Note, this is not going to grade your submission! The function will create a file called `<andrew_id>.zip`.

6 Submission Checklist

- ☐ Create a PDF of your answers to the written questions with the name `<andrew_id>.pdf` and submit it to Gradescope.
- ☐ Run `create_submission.py` in the python terminal.
- ☐ Upload `<andrew_id>.zip` to Gradescope.
- ☐ Coding questions are autograded, with no limit on number of submissions within the mandated deadline.