RBOT101: Mathematical Foundations of Robotics

Instructor: Dr. Lekan Molu

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

PREAMBLE

Consider this the roadmap for this course. Please read through the syllabus posted on Moodle2 carefully and feel free to share any questions that you may have. Please print a copy of the Syllabus for reference. Some relevant parts of the Syllabus are repeated here but the Moodles reference should serve as your guide throughout the ten weeks of this course.

1.1 Course Description

This course focuses on the algorithmic and mathematical concepts with respect classical and recent methods for solving real-world problems in robotics. While some students may have encountered some of the concepts we will be treating in past courses or avenues of study, we will provide the breadth and depth necessary for equipping students to be world-class roboticists. The topics covered by this course shall include the configuration space, rigid bodies, semi-rigid soft bodies, as well as their motions in $\mathbb{R}^n$, wrenches, homogeneous transformations, optimal algorithms for rigid body rotations, linear systems theory, probability theory, the Kalman filter. The course will begin and end with a self-assessment to allow students to gauge their strengths and weaknesses in these topics. References for further, in-depth study in each topic are provided at the end of this course.

1.2 Course Outcomes

After taking this course, each student will be able to

- Develop mathematical tools for solving fundamental kinematic problems in robot operation;
- Formulate optimal state estimation tools for solving real-time smoothing and filtering operations in robotics;

- Integrate state estimation with rigid and semi-rigid soft bodies to solve real-world automation problems; and 4. Use open-source Python, and C++ tools to solve classical and emerging problems in robotics in our day.

1.3 Prerequisites

An undergraduate-level understanding of linear algebra, analytical mechanics, Python and C++ programming.

1.4 Recommended Texts

- Main Texts
  - Simon, Dan. (2007). Optimal state estimation: Kalman, $H - \infty$, and nonlinear approaches. Choice Reviews Online, Vol. 44, pp. 44-3334-44–3334. https://doi.org/10.5860/choice.44-3334
  - Murray, R. M., Li, Z., and Sastry, S. S. (1994). A Mathematical Introduction to Robotic Manipulation. Book (Vol. 29). Free PDF preprint downloadable from, Murray's website.
  - Theory of Screws: A Study in the Dynamics of a Rigid Body by Robert Stawell Ball, Dublin: Hodges, Foster, and Co., Grafton-Street. a. Textbooks:
- Secondary Text
  - Modern Robotics: Mechanics, Planning, and Control. Free PDF preprint downloadable from Author's Northwestern University Website.
- Auxiliary Text:
  - Theory of Screws: A Study in the Dynamics of a Rigid Body by Robert Stawell Ball, Dublin: Hodges, Foster, and Co., Grafton-Street (Should be downloadable via Interlibrary Loan).

1.5 Recommended Journals

- IEEE Transactions on Robotics.
- The International Journal of Robotics Research.
- The IEEE International Conference on Robotics and Automation (ICRA).
- IEEE/Robotics Society of Japan International Conference on Intelligent Robots and Systems (IROS).
- Robotics and Autonomous Systems, An Elsevier Journal.

1.6 Required Software

- A working knowledge of python and the anaconda environment.
- ROS 1.x Installation Instructions: ros 1.x website.
- ROS 2 installation ros 2.0 website.

1.7 Online Course Content

This course will be conducted completely online using Brandeis' LATTE site. The site contains the course syllabus, assignments, our discussion forums, links/resources to course-related professional organizations and sites, and weekly checklists, objectives, outcomes, topic notes, self-tests, and discussion questions. Access information is emailed to enrolled participants before the start of the course. To begin participating in the course, review the “Welcoming Message" and the “Week 1 Checklist."

1.8 Errata

If in the course of using these notes, you find sentence errors, errata or mistakes in equations, please annotate them and upload it to the discussion forum. Points will awarded, at the discretion of the instructor, for such help.

CHAPTER 2

INTRODUCTION TO MATRIX ANALYSIS.

Our goal here is to introduce the student to the study of matrix theory. Matrices are symbolism of the important transformations in everyday life; these transformations lie at the heart of mathematics and robotics. The contents of this topic are thus positioned toward the aspiration of roboticists, engineers of all stripes and scientists. Specifically, we are concerned with the *theory of symmetric matrices*, which is important for all fields, *matrices and differential equations*, necessary for engineering and robotics, as well as *positive matrices*, necessary for probability theory. Most of the texts im this chapter are drawn from Richard Bellman's Matrix Analysis Book given in the Syllabus.

2.1 Maximization and Minimization

Of importance to us in this section is to ascertain the range of values of *homogeneous quadratic functions* of two variables and how it is connected to the determination of the maximum or minimum of a general function of two variables.

2.1.1 Maximization of Functions of a Variable

Suppose $f(x)$ is a real function of the real variable x for $x \in [a, b]$, and let us suppose that it is a Taylor series of the form

$$f(x) = f(c) + f'(x-c) + f''\frac{(x-c)^2}{2!} + \ldots \tag{2.1.1}$$

around every point in the open interval (a, b). We define a *stationary point* of $f(x)$ to be a point where $f'(x) = 0$ and it is the point that determines if c is a point at which $f(x)$ is a relative maximum, a relative minimum, or a stationary point of a subtle characteristic. If c is a stationary point, we must have

$$f(x) = f(c) + f''\frac{(x-c)^2}{2!} + \ldots \tag{2.1.2}$$

If $f''(c) > 0$, then $f(x)$ has a relative minimum at $x = c$. Otherwise, if $f''(c) < 0$, $f(x)$ has a relative maximum at $x = c$. Whereas, if $f''(c) = 0$, we must needs consider further terms in the expansion.

Quiz 1. Suppose that $f''(c) = 0$, what are the sufficient conditions that c must furnish to be a relative minimum?

2.1.2 Maximization of Functions of Two Variables

Now, suppose that we have two variables x, y as arguments of a function f, defined over the rectangle $a_1 \leq x \leq b_1$, $a_2 \leq y \leq b_2$, and possessing a convergent Taylor series around each point (c_1, c_2) within the region. Then, for sufficiently small $|x - c_1$ and $|y - c_2|$, we have

$$\begin{aligned} f(x,y) = f(c_1,c_2) + (x-c_1)\frac{\partial f}{\partial c_1} + (y-c_2)\frac{\partial f}{\partial c_2} + \frac{(x-c_1)^2}{2}\frac{\partial^2 f}{\partial c_1^2} \\ + (x-c_1)(y-c_2)\frac{\partial^2 f}{\partial c_1 \partial c_2} + \frac{(y-c_2)^2}{2}\frac{\partial^2 f}{\partial c_2^2} + \dots \end{aligned} \tag{2.1.3}$$

where

$$\begin{aligned} \frac{\partial f}{\partial c_1} &= \frac{\partial f}{\partial x} \text{ at } x = c_1, \qquad y = c_2 \\ \frac{\partial f}{\partial c_2} &= \frac{\partial f}{\partial y} \text{ at } x = c_1, \qquad y = c_2 \text{ e.t.c.} \end{aligned} \tag{2.1.4}$$

As before, the stationary point of $f(x, y)$ is defined to be (c_1, c_2) so that $\frac{\partial f}{\partial c_1} = 0$ and $\frac{\partial f}{\partial c_2} = 0$; and the behavior of $f(x, y)$ in the immediate neighborhood of (c_1, c_2) depends on the nature of the quadratic terms in the expansion of (2.1.3),

$$Q_2(x,y) = a(x-c_1)^2 + 2b(x-c_1)(y-c_2) + c(y-c_2)^2 \tag{2.1.5}$$

where $a = \frac{1}{2}\frac{\partial^2 f}{\partial c_1^2}$, $2b = \frac{\partial^2 f}{\partial c_1 \partial c_2}$, and $c = \frac{1}{2}\frac{\partial^2 f}{\partial c_2^2}$.

Suppose we set $x - c_1 = u$ and $y - c_2 = v$, then we can write a quadratic expression in variables u and v $i.e.$

$$Q(u,v) = au^2 + 2buv + cv^2 \tag{2.1.6}$$

whereupon we are interested in the behavior of $Q(u, v)$ in the vicinity of $u = v = 0$ and the fact that $Q(u, v)$ is homogeneous allows us to examine the range of values of $Q(u, v)$ for the set of values on $u^2 + v^2 = 1$.

If $Q(u, v) > 0$ for all u and v distinct from $u = v = 0$, $f(x, y)$ will have a relative minimum at $x = c_1$, $y = c_2$; and if $Q(u, v) < 0$ for all u and v distinct from $u = v = 0$, $f(x, y)$ will have a relative maximum at $x = c_1$, $y = c_2$; The stationary point is a *saddle point* if $Q(u, v)$ can take on both positive and negative values.

2.1.3 Algebraic Approach

How do we determine which of the three situations described in the foregoing occur for any given quadratic form, $au^2 + 2buv + cv^2$, with real coefficients. To determine the sign of $Q(u, v)$, we complete the square in $au^2 + 2buv$ and write $Q(u, v)$ as

$$Q(u, v) = a\left(u + \frac{bv}{a}\right)^2 + \left(c - \frac{b^2}{a}\right)v^2 \tag{2.1.7}$$

provided that $a \neq 0$.

If $a = c = -$, then $Q(u, v) \equiv 2buv$. If $b \neq 0$, then $Q(u, v)$ can be positive or negative. If however, $b = 0$, the quadratic form is eliminated.

If $a \neq 0$, from (2.1.7), we must have a $Q(u, v) > 0$ for all unique u and v different from the pair $(0, 0)$ provided that $a > 0$ and $c - \frac{b^2}{a} > 0$.

In the same vein, $Q(u, v) < 0$ for all nontrivial u and v, provided that we have the inequalities, $a < 0$ and $c - \frac{b^2}{a} < 0$.

Positivity Requirement

A set of *necessary and sufficient* conditions that $Q(u, v)$ be positive for all nontrivial u and v is that

$$a > 0, \qquad \begin{vmatrix} a & b \\ b & c \end{vmatrix} > 0. \tag{2.1.8}$$

2.1.4 Analytic Approach

To find the range of values of $Q(u, v)$, we can examine the set of values that $Q(u, v)$ occupies on the circle $u^2 + v^2 = 1$. If Q is to be positive for all nontrivial values of u and v, we must have

$$\min_{u^2+v^2=1} Q(u, v) > 0 \tag{2.1.9}$$

and to have $Q(u, v)$ negative for all u and v on the unit circle, we must have

$$\max_{u^2+v^2=1} Q(u, v) < 0. \tag{2.1.10}$$

Introducing a Lagrange multiplier, λ, we can rewrite the problem as

$$R(u, v) = aU^2 + 2buv + cv^2 - \lambda(u^2 + v^2). \tag{2.1.11}$$

At the stationary points, we must have $\frac{\partial R}{\partial u} = \frac{\partial R}{\partial v} = 0$ so that

$$\begin{aligned} au + bv - \lambda u &= 0 \\ bu + cv - \lambda v &= 0 \end{aligned} \tag{2.1.12}$$

whereupon, we see that λ satisfies

$$\begin{vmatrix} a - \lambda & b \\ b & c - \lambda \end{vmatrix} = 0 \tag{2.1.13}$$

$$\lambda^2 - (a + c)\lambda + ac - b^2 = 0. \tag{2.1.14}$$

The roots of (2.1.14) are real seeing that the discriminant is non-negative $i.e.$

$$(a + c)^2 - 4\left(ac - b^2\right) = (a - c)^2 + 4b^2, \tag{2.1.15}$$

and as long as $a \neq 0$ and $b \neq 0$, the roots are distinct.

If $b = 0$, the roots of the quadratic in (2.1.14) becomes $\lambda_1 = a$, $\lambda_2 = c$. For $\lambda_1 = a$, the linear set of equations from (2.1.12) becomes

$$(a - \lambda_1)\, u = 0 \qquad (c - \lambda_1)\, v = 0 \tag{2.1.16}$$

which leaves u arbitrary and $v = 0$, if $a \neq c$.

Whereas if $b \neq 0$, we obtain the nontrivial solutions of (2.1.12) by using one equation and discarding the other. Therefore, u and v are connected by the relation

$$(a - \lambda_1)\, u = -bv. \tag{2.1.17}$$

For the exact solution, we can add the normalization requirement that $u^2 + v^2 = 1$ so that the values of u and v are

$$\begin{aligned} u_1 &= -b/\left(b^2 + (a - \lambda_1)^2\right)^{1/2} \\ v_1 &= (a - \lambda_1)/\left(b^2 + (a - \lambda_1)^2\right)^{1/2} \end{aligned} \tag{2.1.18}$$

with another set (u_2, v_2) determined in a similar fashion when λ_2 is used in place of λ_1.

CHAPTER 3

VECTORS AND MATRICES

In the previous chapter, we looked into the problem of the minima and maxima (locally) of a function of a single and two variables. Suppose that we have N variables, and proceed in a similar manner as before, we see that finding basic necessary and sufficient conditions that ensure the positivity of a quadratic form of N variables are of the form

$$Q(x_1, x_2, \ldots, x_N) = \sum_{i,j=1}^{N} a_{ij} x_i x_j \tag{3.0.1}$$

We will thus develop a notation that allows us to solve the problem *analytically* using a minimum of arithmetic or analytical calculation. In this light, we will develop a notation that allows us to study linear transformations such as

$$y_i = \sum_{j=1}^{N} a_{ij} x_j \qquad i = 1, 2, \ldots, N \tag{3.0.2}$$

3.1 Vectors

We shall define a set of N complex-valued numbers as a *vector*, written as

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \tag{3.1.1}$$

The vector $\boldsymbol{x}$ in (3.1.1) shall be called a *column vector*. If the elements of the vector are stacked horizontally, $i.e.$

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \ldots & x_N, \end{bmatrix} \tag{3.1.2}$$

then we shall call it a *row vector*.

Going forward, we shall use the notation of (3.1.1) to represent all forms of vectors we shall be using. When we mean a row vector, we shall use the notation of a transpose of (3.1.1), $i.e.$ $\boldsymbol{x}^T$. Bold font letters such as $\boldsymbol{x}$, or $\boldsymbol{y}$ shall denote vectors and lower-case letters with subscripts i such as x_i, y_i, z_i or p_i, q_i, r_i shall denote the components of a vector. When discussing a particular set of vectors, we shall use the superscripts $\boldsymbol{x}^1, \boldsymbol{x}^2$ e.t.c. N shall denote the dimension of a vector $\boldsymbol{x}$.

One-dimensional vectors are called *scalars* and shall be our quantities of analysis. When we write $\bar{\boldsymbol{x}}$, we shall mean the vector whose components are the complex conjugates of the elements of $\boldsymbol{x}$.

3.1.1 Addition of Vectors

Two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are said to be equal if all of their components, (x_i, y_i) are equal for $i = 1, 2, \dots, N$. Addition is the simplest of the arithmetic operations on vectors. We shall write the sum of two vectors as $\boldsymbol{x} + \boldsymbol{y}$ so that

$$\boldsymbol{x} + \boldsymbol{y} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_N + y_N, \end{bmatrix} \tag{3.1.3}$$

whereupon we note that the " $+$ " sign connecting $\boldsymbol{x}$ and $\boldsymbol{y}$ is different from the one connecting x_i and y_i.

Homework 1. Prove that we have the *commutativity*, $\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$, and the associativity $\boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z}) = (\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z}$

Homework 2. Just as we showed the addition property of two vectors above, show the subtraction property of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$.

3.1.2 Scalar Multiplication

When a vector is multiplied by a scalar, we shall write it out as follows

$$c_1 \boldsymbol{x} = \boldsymbol{x} c_1 = \begin{bmatrix} c_1 x_1 \\ c_1 x_2 \\ \vdots \\ c_1 x_N \end{bmatrix} \tag{3.1.4}$$

3.1.3 The Inner Product of Two Vectors

This is a scalar function of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ defined as

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{N} x_i y_i. \tag{3.1.5}$$

Further to the above, we define the following properties for inner product

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle \tag{3.1.6a}$$

$$\langle \boldsymbol{x} + \boldsymbol{y}, \boldsymbol{u} + \boldsymbol{v} \rangle = \langle \boldsymbol{x}, \boldsymbol{u} \rangle + \langle \boldsymbol{x}, \boldsymbol{v} \rangle + \langle \boldsymbol{y}, \boldsymbol{u} \rangle + \langle \boldsymbol{y}, \boldsymbol{v} \rangle \tag{3.1.6b}$$

$$\langle c_1 \boldsymbol{x}, \boldsymbol{y} \rangle = c_1 \langle \boldsymbol{x}, \boldsymbol{y} \rangle \tag{3.1.6c}$$

The above is an easy way to *multiply* two vectors. The inner product is important because $\langle \boldsymbol{x}, \boldsymbol{x} \rangle$ can be considered as the square of the “length" of the real vector $\boldsymbol{x}$.

Homework 3. Prove that $\langle a\boldsymbol{x} + b\boldsymbol{y}, a\boldsymbol{x} + b\boldsymbol{y} \rangle = a^2 \langle \boldsymbol{x}, \boldsymbol{x} \rangle + 2ab \langle \boldsymbol{x}, \boldsymbol{x} \rangle + b^2 \langle \boldsymbol{y}, \boldsymbol{y} \rangle$ is a non-negative quadratic form in the scalar variables a and b if $\boldsymbol{x}$ and $\boldsymbol{y}$ are real.

Homework 4. Hence, show that for real-valued vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, that the Cauchy-Schwarz Inequality $\langle \boldsymbol{x}, \boldsymbol{y} \rangle^2 \leq \langle \boldsymbol{x}, \boldsymbol{x} \rangle \langle \boldsymbol{y}, \boldsymbol{y} \rangle$ holds.

Homework 5. Using the above result, show that for any two complex vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle|^2 \leq \langle \boldsymbol{x}, \bar{\boldsymbol{x}} \rangle \langle \boldsymbol{y}, \bar{\boldsymbol{y}} \rangle$

Homework 6. Show that the *triangle inequality*

$$\langle \boldsymbol{x}+\boldsymbol{y}, \boldsymbol{x}+\boldsymbol{y}\rangle^{\frac{1}{2}} \leq \langle \boldsymbol{x}, \boldsymbol{x}\rangle^{\frac{1}{2}} + \langle \boldsymbol{y}, \boldsymbol{y}\rangle^{\frac{1}{2}}$$

holds for any two real-valued variables.

3.1.4 Orthogonality

Two vectors are said to be orthogonal if their inner product is 0 $i.e.$

$$\langle \boldsymbol{x}, \boldsymbol{y}\rangle = 0 \tag{3.1.7}$$

When the set of real vectors $\{\boldsymbol{x}^i\}$ possess the property that $\langle \boldsymbol{x}^i, \boldsymbol{y}^i\rangle = 1$, then we say they are *orthonormal.*

Homework 7. show that $\boldsymbol{x}^i$ are mutually orthogonal and normalized $i.e.$ orthonormal for the following N-dimensional Euclidean basis coordinate vectors

$$\boldsymbol{x}^1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \boldsymbol{x}^2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \qquad \boldsymbol{x}^N = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \tag{3.1.8}$$

3.2 Matrices

We can write an array of complex numbers in the form

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{NN} \end{bmatrix} \tag{3.2.1}$$

The matrix of (3.2.1) shall be called a *square matrix*. The quantities x_{ij} are the *elements* of the matrix X; the quantities $x_{i1}, x_{i2}, \ldots, x_{iN}$ are the ith *rows* of the matrix X and the quantities $x_{1j}, x_{2j}, \ldots, x_{Nj}$ are the jth columns of X. We denote matrices with upper case letters or the lower-case subscript notations

$$X = (x_{ij}) \tag{3.2.2}$$

while the *determinant* of the array associated with (3.2.1) shall be denoted $|X|$ or $|x_{ij}$.

Similar to the equality definition between vectors, two matrices are said to be equal if and only if their elements are equal $i.e.$

$$A + B = (a_{ij} + b_{ij}) \tag{3.2.3}$$

Scalar multiplication of a matrix can be expressed as

$$c_1 X = X c_1 = (c_1 x_{ij}) \tag{3.2.4}$$

Lastly, by $\bar{X}$ we shall mean the matrix whose elements are the complex conjugates of X. X is a real matrix if the elements of X are real.

3.2.1 Vector by Matrix Multiplication

Recall the linear transformation

$$y_i = \sum_{j=1}^{N} a_{ij} x_j \qquad i = 1, 2, \ldots, N \tag{3.2.5}$$

where a_{ij} are complex quantities. For two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ related as above, we have

$$\boldsymbol{y} = A\boldsymbol{x} \tag{3.2.6}$$

to describe the multiplication of a vector $\boldsymbol{x}$ by a matrix X.

Homework 8. Consider the identity matrix I, so defined

$$I = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \tag{3.2.7}$$

$i.e.$ $I = (\delta_{ij})$, where δ_{ij} is the Kronecker delta symbol, defined as

$$\delta_{ij} = \begin{cases} 0, & \text{if } i \neq j \\ 1, & \text{if } i = j \end{cases} \tag{3.2.8}$$

Show that

$$\delta_{ij} = \sum_{k=1}^{N} \delta_{ik}\delta_{kj} \tag{3.2.9}$$

Homework 9. Show that

$$\langle Ax, Ax \rangle = \sum_{i=1}^{N} \left(\sum_{j=1}^{N} a_{ij}x_j \right)^2 \tag{3.2.10}$$

3.2.2 Matrix by Matrix Multiplication

Consider (3.2.6). Now, suppose our goal is to generate a second-order linear transformation so defined

$$\boldsymbol{z} = B\boldsymbol{y} \tag{3.2.11}$$

which converts the components of $\boldsymbol{y}$ into components of $\boldsymbol{z}$. To express the components of $\boldsymbol{z}$ in terms of the components of $\boldsymbol{x}$ this, we write

$$z_i = \sum_{k=1}^{N} b_{ik}y_k = \sum_{k=1}^{N} b_{ik} \left(\sum_{j=1}^{N} a_{kj}x_j \right) \tag{3.2.12}$$

$$= \sum_{j=1}^{N} \left(\sum_{k=1}^{N} b_{ik}a_{kj} \right) \boldsymbol{x}_j \tag{3.2.13}$$

Introducing $C = (c_{ij})$ defined as

$$c_{ij} = \sum_{k=1}^{N} b_{ik} a_{kj} \qquad i, j = 1, 2, \ldots, N \tag{3.2.14}$$

we may write

$$\boldsymbol{z} = C\boldsymbol{x} \tag{3.2.15}$$

Since, formally

$$\boldsymbol{z} = B\boldsymbol{y} = B(A\boldsymbol{x}) = B(A\boldsymbol{x}) = (BA)\boldsymbol{x} \tag{3.2.16}$$

so that

$$C = BA \tag{3.2.17}$$

Note the ordering of the matrix product above.

Homework 10. Show that

$$f(\theta_1)f(\theta_2) = f(\theta_2)f(\theta_1) = f(\theta_1 + \theta_2) \tag{3.2.18}$$

where

$$f(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3.2.19}$$

Homework 11. Show that

$$\left(a_1^2 + a_2^2\right)\left(b_1^2 + b_2^2\right) = \left(a_2 b_1 + a_1 b_2\right)^2 + \left(a_1 b_1 + a_2 b_2\right)^2 \tag{3.2.20}$$

Hint: $|AB| = |A||B|$,

3.2.3 Non-Commutativity

Matrix multiplication is not commutative, $i.e.$ $AB \neq BA$. For an example, consider the following 3×3 matrices

$$A = \begin{bmatrix} 5 & 6 & 9 \\ 2 & 1 & 6 \\ 3 & 6 & 9 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 4 & 13 \\ 23 & 6 & 24 \\ 8 & 3 & 9 \end{bmatrix} \tag{3.2.21}$$

where

$$AB = \begin{bmatrix} 215 & 83 & 290 \\ 73 & 32 & 104 \\ 213 & 75 & 264 \end{bmatrix} \qquad \text{and } BA = \begin{bmatrix} 52 & 88 & 150 \\ 199 & 288 & 459 \\ 73 & 105 & 171 \end{bmatrix} \tag{3.2.22}$$

so that $AB \neq BA$. If, however, $AB = BA$, we say A and B *commute*. Note that

$$(AB)^{-1} = B^{-1}A^{-1}. \tag{3.2.23}$$

3.2.4 Associativity

Associativity of matrix multiplication gets preserved unlike the commutativity. So for matrices A, B, and C, we have

$$(AB)\,C = A\,(BC) \tag{3.2.24}$$

that is, the product ABC is unambiguously defined without the parentheses. To prove this, we write the ijth element of AB as

$$a_{ik}b_{kj} \tag{3.2.25}$$

so that the definition of multiplication implies that

$$(AB)\,C = [(a_{ik}b_{kl})\,c_{lj}] \tag{3.2.26}$$

$$A\,(BC) = [a_{ik}\,(b_{kl}c_{lj})] \tag{3.2.27}$$

which establishes the equality $(AB)\,C$ and $A\,(BC)$.

3.2.5 Invariant Vectors

The problem of finding the minimum or maximum of $Q = \sum_{i,j=1}^{N} a_{ij}\boldsymbol{x}_i\boldsymbol{x}_j$ for $\boldsymbol{x}_i$ satisfying the relation $\sum_{i=1}^{N} \boldsymbol{x}_i^2 = 1$ can be reduced to the problem of finding the values of the scalar λ that satisfies the set of linear homogeneous equations

$$\sum_{j=1}^{N} a_{ij}\boldsymbol{x}_j = \lambda\boldsymbol{x}_i, \qquad i = 1, 2, \ldots, N \tag{3.2.28}$$

which possesses nontrivial solutions. Vectorizing, we have

$$A\boldsymbol{x} = \lambda\boldsymbol{x} \tag{3.2.29}$$

Here, $\boldsymbol{x}$ signifies the direction indicated by the N direction numbers $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$, and we are searching for the directions that are invariant.

3.2.6 The Matrix Transpose

We define the transpose of the matrix $A = (a_{ij})$ as $A^T = (a_{ji})$ $i.e.$ the rows of A^T are the columns of A and vice versa. An important consequence of this is that the transformation A on the set of a vector $\boldsymbol{x}$ is same as the transformation of the matrix A^T on the set $\boldsymbol{y}$. This is shown in the following

$$\langle A\boldsymbol{x}, \boldsymbol{y}\rangle = y_1 \sum_{j=1}^{N} a_{1j}\boldsymbol{x}_j + y_2 \sum_{j=1}^{N} a_{2j}\boldsymbol{x}_j + \ldots + y_N \sum_{j=1}^{N} a_{Nj}\boldsymbol{x}_j \tag{3.2.30}$$

which becomes upon rearrangement,

$$\langle A\boldsymbol{x}, \boldsymbol{y}\rangle = x_1 \sum_{i=1}^{N} a_{i1}\boldsymbol{y}_i + x_2 \sum_{i=1}^{N} a_{i2}\boldsymbol{y}_i + \ldots + x_N \sum_{i=1}^{N} a_{iN}\boldsymbol{y}_i \tag{3.2.31}$$

$$= \langle \boldsymbol{x}, A^T\boldsymbol{y}\rangle \tag{3.2.32}$$

We can then regard A^T as the *induced or adjoint transformation* of A. An interesting property of the transpose of amatrix product is that

$$(AB)^T = B^T A^T \tag{3.2.33}$$

3.2.7 Symmetric Matrices

Matrices that satisfy the relation

$$A = A^T \tag{3.2.34}$$

play a crucial role in the study of quadratic forms and such matrices are said to be *symmetric*, with the property that

$$a_{ij} = a_{ji} \tag{3.2.35}$$

Homework 12. Prove that $(A^T)^T = A$

Homework 13. Prove that $\langle A\boldsymbol{x}, B\boldsymbol{y}\rangle = \langle \boldsymbol{x}, A^T B\boldsymbol{y}\rangle$

3.2.8 Hermitian Matrices

The scalar function for complex vectors is the expression $\langle \boldsymbol{x}, \bar{\boldsymbol{y}}\rangle$. Suppose we define $\boldsymbol{z} = \bar{A}^T\boldsymbol{y}$, then

$$\langle A\boldsymbol{x}, \bar{y}\rangle = \langle \boldsymbol{x}, \bar{\boldsymbol{z}}\rangle \tag{3.2.36}$$

$i.e.$ the induced transformation is now $\bar{A}^T$, the complex conjugate of A. Matrices for which

$$A = \bar{A}^T \tag{3.2.37}$$

are called Hermitian. Note that in some literature, the Hermitian matrix is often written as $A^{\star}$.

3.2.9 Orthogonal Matrices

This section has to do with the invariance of distance between matrices, that is, taking the Euclidean measure of distance as the measure of the magnitude of the real-valued vector $\boldsymbol{x}$. The prodding question of interest is to figure out the linear transformation $\boldsymbol{y} = H\boldsymbol{x}$ that leaves the inner product $\langle \boldsymbol{x}, \boldsymbol{z}\rangle$. Mathematically, we express this problem such that

$$\langle \boldsymbol{x}, \boldsymbol{x}\rangle = \langle H\boldsymbol{x}, H\boldsymbol{x}\rangle \tag{3.2.38}$$

is satisfied for *all* $\boldsymbol{x}$. We know that

$$\langle H\boldsymbol{x}, H\boldsymbol{x}\rangle = \langle \boldsymbol{x}, H^T H\boldsymbol{x}\rangle \tag{3.2.39}$$

and that $H^T H$ is symmetric so that (3.2.38), gives

$$H^T H = I. \tag{3.2.40}$$

Orthogonal Matrix

A real matrix H for which $H^T H = I$ is called *orthogonal*.

3.2.10 Unitary Matrices

This is the measure of the distance of a complex vector, akin to the invariance condition of real-valued matrices (3.2.40). We define the unitary property as follows:

$$H^\star H = I. \tag{3.2.41}$$

Matrices defined as in the foregoing play a crucial role in the treatment of Hermitian matrices, such as the role that orthogonal matrices play in symmetric matrices theory.

3.2.11 Matrix Determinant

The determinant of a scalar is same as the scalar while the determinant of a matrix shall be inductively defined for square matrices. Suppose we have an $n \times n$ matrix A, its determinant is defined as

$$|A| = \sum_{j=1}^{N} (-1)^{i+j} a_{(i,j)} |a^{(i,j)}| \tag{3.2.42}$$

for any value of $i \in [1, n]$, where (3.2.42) is called the Laplace expansion of $|A|$ along its ith row. Equation (3.2.42) shows us that the determinant of the square matrix A is found in terms of the determinants of the $(n-1) \times (n-1)$ matrices. Similarly, the determinants of $(n-1) \times (n-1)$ matrices are defined by $(n-2) \times (n-2)$ and so on until we get to the determinant of 1×1 matrices which are scalars. We can also define the determinant of A as

$$|A| = \sum_{i=1}^{N} (-1)^{i+j} a_{(i,j)} |a^{(i,j)}| \tag{3.2.43}$$

for any value of $j \in [1, n]$. This is termed the Laplace expansion of A along its jth column. It follows that

$$|A_{11}| = A_{11} \tag{3.2.44a}$$

$$\det \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = A_{11}A_{22} - A_{12}A_{21} \tag{3.2.44b}$$

and that

$$\det \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = A_{11}(A_{22}A_{33} - A_{23}A_{32}) - \tag{3.2.44c}$$

$$A_{12}(A_{21}A_{33} - A_{23}A_{31}) + \tag{3.2.44d}$$

$$A_{13}(A_{21}A_{32} - A_{22}A_{31}) \tag{3.2.44e}$$

3.2.12 Properties of the Matrix Determinant

1. $|AB| = |A||B|$, where A and B are assumed to be of equal dimensions.

2. $|A| = \prod_{i=1}^{N} \lambda_i$, where λ_i are the eigenvalues of A.

3. The inverse of A is said to exist if $AA^{-1} = I$. Such a matrix is said to be *non-singular*. Note that A must be a square matrix in order for it to have a determinant. A square matrix whose inverse does not exist is said to be *singular*.

Take for example,

$$\begin{bmatrix} 3 & 0 \\ 2, 1 & \end{bmatrix} \begin{bmatrix} 1/3 & 0 \\ -2/3, 1 & \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0, 1 & \end{bmatrix}. \tag{3.2.45}$$

Then we say that the two matrices on the left are inverses of one another. Among other ways of stating the nonsingularity of A are that

- A's rows or columns are linearly independent.
- $|A| \neq 0$.
- $Ax = b$ has a unique solution x for all b.
- The rank of $A = n$.
- 0 is not an eigenvalue of A.
- A^{-1} exists.

3.2.13 The Matrix Trace

The trace of a matrix exists if and only of the matrix is square. It is defined as the sum of its diagonal elements.

$$\text{Tr}(A) = \sum_{i=1} A_{ii} \tag{3.2.46}$$

Also, the trace can be expressed in terms of the sum of the matrix's eigenvalues,

$$\text{Tr}(A) = \sum_{i=1} \lambda_i. \tag{3.2.47}$$

The trace of a matrix product is not dependent in the order of multiplication of the matrices:

$$\text{Tr}(AB) = \text{Tr}(BA). \tag{3.2.48}$$

3.2.14 Eigenvectors and Eigenvalues of a Matrix

A square $n \times n$ matrix A has n eigenvalues and n eigenvectors. If

$$A\boldsymbol{x} = \lambda \boldsymbol{x} \tag{3.2.49}$$

for a scalar λ and an $n \times 1$ vector $\boldsymbol{x}$ then we say the matrix A has eigenvalues λ and eigenvectors $\boldsymbol{x}$. Together, λ and $\boldsymbol{x}$ are called *eigendata*, the *characteristic roots*, *latent roots*, or *proper numbers and vectors* of the matrix.

Homework 14. If A has eigendata $(\lambda, \boldsymbol{x})$, then A^2 has eigendata $(\lambda^2, \boldsymbol{x})$.

Homework 15. Show that A^{-1} exists if and only if none of the eigenvalues of A are zero.

Homework 16. Show that the eigenvalues of A are real numbers if A is symmetric.

3.2.15 Other Matrix Properties

A *symmetric* $n \times n$ matrix A can be characterized as either positive definite, positive semidefinite, negative definite, negative semidefinite, or indefinite if matrix A is

- *Positive definite* if $\boldsymbol{x}^T A\boldsymbol{x} > 0$ for all nonzero $n \times 1$ vectors $\boldsymbol{x}$. That is, all the eigenvalues of A are positive real numbers. If A is positive definite, then so is A^{-1}.

- *Positive semidefinite* if $\boldsymbol{x}^T A\boldsymbol{x} \geq 0$ for all nonzero $n \times 1$ vectors $\boldsymbol{x}$. That is, all the eigenvalues of A are non-negative real numbers. A positive semidefinite matrices are sometimes called nonnegative definite.

- *Negative definite* if $\boldsymbol{x}^T A\boldsymbol{x} < 0$ for all nonzero $n \times 1$ vectors $\boldsymbol{x}$. That is, all the eigenvalues of A are negative real numbers. If A is negative definite, then so is A^{-1}.

- *Negative semidefinite* if $\boldsymbol{x}^T A\boldsymbol{x} \leq 0$ for all nonzero $n \times 1$ vectors $\boldsymbol{x}$. That is, all the eigenvalues of A are non-negative real numbers. A positive semidefinite matrices are sometimes called non positive definite.

- When some of the eigenvalues of A are positive and some are negative, then the matrix is said to be *indefinite*.

The singular values of matrix A are defined as

$$\begin{aligned}\sigma^2(A) &= \lambda(A^T A) \\ &= \lambda(A^T A)\end{aligned} \tag{3.2.50}$$

For an $n \times n$ matrix A, we have a $\min(n, m)$ singular values. If $n > m$, then AA^T will have the same eigenvalues as $A^T A$ and an additional $n - m$ zeroes. We do not consider the additional zeroes to be singular values of A because A always has $\min(n, m)$ singular values.

Quiz 2. If A is $n \times m$, what are number of eigenvalues of $A^T A$ and AA^T respectively?

3.2.16 The Matrix Inversion Lemma

This is sometimes called the *Woodbury matrix identity*, named after Max A. Woodbury., *Sherman-Morrison formula*, or the *modified matrices formula*. It a tool frequently used in statistics, system identification, state estimation and control theory. Assume we have a blockwise matrix $\begin{pmatrix} A & B \\ C & D \end{pmatrix}$ where A and D are invertible square matrices, and B and C are not necessarily square. We can define the following matrices

$$\begin{aligned}E &= D - CA^{-1}B \\ F &= A - BD^{-1}C.\end{aligned} \tag{3.2.51}$$

If E is invertible, it follows that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} A^{-1} + A^{-1}BE^{-1}CA^{-1} & -A^{-1}BE^{-1} \\ -E^{-1}CA^{-1} & E^{-1} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{3.2.52}$$

Also, if F were invertible, it follows that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} F^{-1} & -A^{-1}BE^{-1} \\ -D^{-1}CF^{-1} & E^{-1} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{3.2.53}$$

It follows that (3.2.52) and (3.2.53) are two expressions for the inverse of $\begin{pmatrix} A & B \\ C & D \end{pmatrix}$. We must therefore have the upper-left partitions of the two matrices equal so that

$$F^{-1} = A^{-1} + A^{-1}BE^{-1}CA^{-1} \tag{3.2.54}$$

and from the definition of F, we have

$$\left(A - BD^{-1}C\right)^{-1} = A^{-1} + A^{-1}B\left(D - CA^{-1}B\right)^{-1}CA^{-1} \tag{3.2.55}$$

An alternative statement of the matrix inversion lemma is

$$\left(A - +BD^{-1}C\right)^{-1} = A^{-1} + A^{-1}B\left(D + CA^{-1}B\right)^{-1}CA^{-1} \tag{3.2.56}$$

Quiz 3. Verify the expressions in (3.2.52) and (3.2.53).

Example 1. Suppose that at Brandeis, you took three courses in your Freshman year namely, RBOT 101, RBOT 103, and RBOT 105 where you got 90%, 85%, and 86% respectively. In your Sophomore year, you took RBOT 201, RBOT 203, and RBOT 205, where you got 65%, 68%, and 92% respectively, and in your junior year, you decide to retake your Sophomore classes, where your scores increased by 10%, 5%, on the first two courses and decreased by 8% in the last course. Your GPA each year increased by 4%, 3.5% and 2.5% respectively. Given your analytical prowess, you decide to model each year's GPA changes with the equation $z = au + bv + cw$, where u and v and w are the scores/grades you got as percentages and a, b, and c are unknown constants. To find the unknown constants, you figure you need to invert the matrix

$$A = \begin{bmatrix} 90 & 85 & 86 \\ 65 & 68 & 92 \\ 71.5 & 71.4 & 84.64 \end{bmatrix} \tag{3.2.57}$$

so that

$$A^{-1} = \begin{bmatrix} 23/20 & 155/104 & -145/52 \\ -207/136 & -569/274 & 6725/1768 \\ 5/16 & 205/416 & -175/208 \end{bmatrix} \tag{3.2.58}$$

It follows that the unknown constants are

$$X = A^{-1} \begin{pmatrix} 4 \\ 7/2 \\ 5/2 \end{pmatrix} = \begin{pmatrix} 2959/1040 \\ -2693/700 \\ 725/832 \end{pmatrix} \tag{3.2.59}$$

As a result, you are able to determine a model which allows you to predict future GPA changes based on how hard you work, sleep, engage in social activities. You can better allocate your time resource and improve your grades in the following years.

Suppose that in the aftermath of generating this model, you now realize that your grade in RBOT 201 the second year was 86% rather than 65%, this means that in order to find the constants, you want to invert

$$\bar{A} = \begin{bmatrix} 90 & 85 & 86 \\ 86 & 68 & 92 \\ 71.5 & 71.4 & 84.64 \end{bmatrix}. \tag{3.2.60}$$

Rather than invert all the matrix all over, you decide to apply a mathematical trick leveraging the inversion lemma. You write $\bar{A} = A + BD^{-1}C$, where

$$B = \begin{bmatrix} 0 & 21 & 3.44 \end{bmatrix}^T \; C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \; D = 1 \tag{3.2.61a}$$

so that

$$\bar{A}^{-1} = \left(A + BD^{-1}C\right)^{-1} \tag{3.2.62}$$

$$= A^{-1} - A^{-1}B\left(D + CA^{-1}B\right)^{-1} CA^{-1} \tag{3.2.63}$$

The $(D + CA^{-1}B)^{-1}$ term turns out to be a scalar so that

$$\bar{A}^{-1} = \begin{bmatrix} 0.0506 & 0.0656 & -0.1228 \\ 0.0239 & -0.073 & 0.0551 \\ -0.065 & 0.0035 & 0.0741 \end{bmatrix} \tag{3.2.64}$$

$$\begin{aligned} X &= \bar{A}^{-1} \begin{pmatrix} 4 \\ 7/2 \\ 5/2 \end{pmatrix} \\ &= \begin{pmatrix} 51/407 \\ -134/6037 \\ -148/2361 \end{pmatrix} \end{aligned} \tag{3.2.65}$$

Here, the matrix inversion lemma may not be necessary since the size of the matrix is small. However, if the matrix had a larger size, the computational savings of using the matrix inversion lemma becomes appreciated.

CHAPTER 4

REGISTRATION OF OBJECTS IN ROBOTICS.

In this chapter, we are concerned with the problem of optimally aligning two vectors, a model point/shape to a “sensed" or measured point/shape in space e.g. $\nu_1, \nu_2 \in \mathbb{R}^n$ to one another with the minimal amount of errors. To transform between two pints in the Cartesian coordinate system is akin to the problem of solving a rigid body motion problem where that yields a rotation and a translation. In addition, the scaling factor may be unknown. For translation, there are three degrees of freedom, while rotation has another three viz., the direction of the axis about which we are rotating, the angle of rotation itself, and the scaling. Three points in either coordinate systems give us nine constraints (with each contributing three coordinates), more than enough to find the seven unknowns. If we discard two of the constraints, we end up with seven equations in seven unknowns that can be developed to allow us to recover the parameters.

There exists many methods of solving this problem. Most of them leverage clever optimization methods and we will be looking into these in this chapter. We could follow the homogeneous transformation scheme we presented in Chapter 1, but we would not have an optimal solution. A popular technique in computer geometry and computer vision is to use the iterative closest point algorithm(ICP), an algorithm by Paul Besl and Neil McKay developed out of General Motors Laboratory in the 1990’s (Besl and McKay, 1992). This is more appropriate for 3D tasks and it describes a generic, representation method for the accurate and computationally efficient registration of three-dimensional (3-D) shapes. The ICP algorithm always converges monotonically to the nearest local minimum of a mean-square distance metric such as an l_2 distance, and this convergence rate is of the order of a few iterations. An important property of the ICP algorithm is that it can register data from unfixtured rigid objects with an ideal geometrical model prior to shape inspection. So, if we want to figure out that two geometric representations are congruent, estimate the motion between them in real-time where the correspondences are not known, ICP tends to be really good for such operations.

Now, suppose our dataset is not a complex geometric primitive[1], but rather a set of two vectors such that we are tasked with the problem of determining the best *unconstrained transformation* between the two sets of coordinates. We can formulate the problem into a constrained optimization problem and thereafter, through clever factorization, turn the problem into a simple one of factorizing the unconstrained transformation into a symmetric and orthogonal matrix by which we may solve for the optimal rotation and translation. The algorithm we shall be looking into will be the one that was invented in crystallography in 1976 and updated in 1978 by Wolfgang Kabsch, today dubbed the Kabsch algorithm (Kabsch, 1978). Kabsch showed that a direct solution was possible, irrespective of the non-linear character of the problem.

While other newer algorithms exist, these are the two popular algorithms that we shall be concerning ourselves with in this chapter.

4.1 Preliminaries

We will denote the real line by $\mathbb{R}$. An example of a **metric space** is the **Euclidean** n**-space** $\mathbb{R}^n$, which consists of $n-$tuples $x = (x_1, x_2, \ldots, x_n)$ where each $x_i \in \mathbb{R}$. We shall mean an $\mathbb{R}^n$ metric space to have the metric

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}. \tag{4.1.1}$$

If $n = 0$, then $\mathbb{R}^0$ is taken to be a single point $0 \in \mathbb{R}$.

A manifold is “locally" similar to one of the example metric spaces $\mathbb{R}^n$. Precisely, a **manifold** is a metric space $\boldsymbol{M}$ with the property that, *if $x \in M$., then there is some neighborhood U of x and some integer $n \geq 0$ such that U is homeomorphic*[2] *to* $\mathbb{R}^n$.

[1]We shall refer to a geometric primitive as a primitive 3D shape such as a cylinder, square, prism and the likes.

[2]A homeomorphic mapping means intrinsic topological equivalence between e.g. objects. Two objects are homeomorphic if they can be deformed into each other by a continuous, invertible mapping. Such a homeomorphism ignores the space in which surfaces are embedded, so the deformation can be completed in a higher dimensional space than the surface was originally embedded. Mirror images are homeomorphic, as are Möbius strip with an even number of half-twists, and Möbius strip with an odd number of half-twists (Weisstein, Weisstein).

A simple example of a manifold is $\mathbb{R}^n$: for each $x \in \mathbb{R}^n$ we can take U to be everything in $\mathbb{R}^n$.

Quiz 4. Suppose we supply $\mathbb{R}^n$ with an equivalent metric, which makes it homeomorphic to $\mathbb{R}^n$, would it also be a manifold?

Another example of a metric space is an open ball in $\mathbb{R}^n$, wherein one can take U to be the entire open ball since an open ball in $\mathbb{R}^n$ is homeomorphic to $\mathbb{R}^n$. Similarly, an open subset V of $\mathbb{R}^n$ is a manifold, $i.e.$ for each $x \in V$ we can choose U to be some open ball with $x \in U \subset V$.

The **Euclidean distance** $d(\boldsymbol{r}_1, \boldsymbol{r}_2)$ between two points $\boldsymbol{r}_1 = (x_1, y_1, z_1)$ and $\boldsymbol{r}_2 = (x_2, y_2, z_2)$ is given by

$$d(\boldsymbol{r}_1, \boldsymbol{r}_2) = \|\boldsymbol{r}_1 - \boldsymbol{r}_2\| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}. \tag{4.1.2}$$

Suppose that P is a point set with N_p points denoted as $\boldsymbol{p}_i : P = \{p_i\}$ for $i = 1, \ldots, N_p$. The distance between the point $\boldsymbol{q}$ and the point set P is

$$d(\boldsymbol{q}, P) = \min_{i \in \{1, \ldots, N_p\}} d(\boldsymbol{q}, \boldsymbol{p}_i). \tag{4.1.3}$$

We find that the closest point $\boldsymbol{p}_j$ of P satisfies $d(\boldsymbol{q}, \boldsymbol{p}_j) = d(\boldsymbol{q}, P)$.

Suppose that we have a **line segment** that connects the points, $\boldsymbol{r}_1, \boldsymbol{r}_2$, the distance between the point $\boldsymbol{r}$ and the line segment l is

$$d(\boldsymbol{p}, l) = \min_{x+y=1} \|x\boldsymbol{r}_1 + y\boldsymbol{r}_2 - \boldsymbol{p}\| \tag{4.1.4}$$

where $x, y \in [0, 1]$.

Homework 17. Find a closed-form expression for the solution to (4.1.4).

Now, if instead of a line segment, suppose we have a set of N_l line segments denoted l_i, and let $L = \{l_i\}$ for $i = 1, \ldots, N_l$. The **distance between the point $\boldsymbol{p}$ and the line segment set** L is

$$d(\boldsymbol{p}, L) = \min_{i \in \{1, \ldots, N_l\}} d(\boldsymbol{p}, l_i). \tag{4.1.5}$$

The closest point y_j on the line segment set L satisfies $d(\boldsymbol{p}, y_j) = d(\boldsymbol{p}, L)$. Let g be a triangle with the following coordinates $\boldsymbol{r} = (x_1, y_1, z_1), \boldsymbol{r}_2 = (x, y, z)$, and $\boldsymbol{r}_3 = (x_3, y_3, z_3)$. The **distance between the point $\boldsymbol{p}$ and the triangle** g is

$$d(\boldsymbol{p}, g) = \min_{x+y+z=1} \|x\boldsymbol{r}_1 + y\boldsymbol{r}_2 + z\boldsymbol{r}_3 - \boldsymbol{p}\| \tag{4.1.6}$$

where $x \in [0,1],\ y \in [0,1]$, and $z \in [0,1]$.

Homework 18. Find a closed-form expression for the problem in (4.1.6).

Now, if we have a collection of N_g triangles G, denoted by g_i such that $G = \{g_i\}$ for $i = 1, \ldots, N_g$. The **distance between the point $\boldsymbol{p}$ and the triangle set** G is

$$d(\boldsymbol{p}, G) = \min_{i \in \{1,\ldots,N_g\}} d(\boldsymbol{p}, g_i), \tag{4.1.7}$$

and the closest point y_j on the triangle set G satisfies the equality $d(\boldsymbol{p}, y_j) = d(\boldsymbol{p}, G)$.

4.1.1 Distance between a Point and a Parameterized Entity

We define a parametric curve and a parametric surface as single parametric entities $\boldsymbol{r}(\boldsymbol{u})$, where $\boldsymbol{u} = u \in \mathbb{R}^1$ denotes a parameterized curve, and $\boldsymbol{u} = (u, v) \in \mathbb{R}^2$ denotes parametric surfaces. We will evaluate a curve within an interval domain e.g. $[x, y]$ while the evaluation domain of a surface can be an arbitrarily closely-connected region in a plane.

We will take the distance from a given point $\boldsymbol{p}$ to a parametric entity E to be

$$d(\boldsymbol{p}, E) = \min_{\boldsymbol{r}(\boldsymbol{u}) \in E} d(\boldsymbol{p}, \boldsymbol{r}(\boldsymbol{u})) \tag{4.1.8}$$

To compute the point-to-curve and point-to-surface distances, let F be the set of N_e parametric entities denoted by E_i, and let $F = \{E_i\}$ for $i = 1, N_e$> The distance between a point $\boldsymbol{p}$ and the parametric entity set F is

$$d(\boldsymbol{p}, F) = \min_{i \in \{1,\ldots,N_e\}} d(\boldsymbol{p}, \boldsymbol{E}_i). \tag{4.1.9}$$

To find the distance from a point to a parametric entity, we can create a simplex-based approximation for e.g. a line segment or triangle. For a parametric space curve $C = \{\boldsymbol{r}(u)\}$, we can compute a polyline $L(C, \delta)$ such that the piecewise-linear approximation never deviates from the space curve by more than a prespecified distance δ. If we tag every point of the polyline with a corresponding u argument values of the parametric curve, we can obtain an estimate of the closest point from the line segment set.

In a similar vein, for a parametric surface $S = \{\boldsymbol{r}(u, v)\}$, one can compute a triangle set $G(S, \delta)$ such that the piecewise triangular approximation never deviates from the surface by more than a prespecified distance δ. If we tag each truangle vertex with the corresponding (u, v) argument values of the parametric surface, we can find the $U_a, v_a)$ of the argument values of the closest point from the triangle set. The initial value of $\boldsymbol{u}_a$ is assumed to be available such that $\boldsymbol{r}(\boldsymbol{u}_a)$ is very close to the closest point on the parametric entity.

We can employ a Newtonian minimization approach for solving the point to parametric entity problem when a reliable starting point $\boldsymbol{u}_a$ is available. The scalar objective function to be minimized is

$$f(\boldsymbol{u}) = \|\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}\|^2. \tag{4.1.10}$$

Suppose $\Delta = [\partial/\partial\boldsymbol{u}]^T$ is the vector differential gradient operator, the minimum of f must occur at $\Delta f = 0$. If we have a surface, then we must have $\Delta f = [f_u, f_v]^T$, with the 2-D Hessian matrix is given by

$$\Delta\Delta^T(f) = \begin{bmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{bmatrix} \tag{4.1.11}$$

where the partial derivatives of the objective function is

$$f_u(\boldsymbol{u}) = 2\boldsymbol{r}_u^T(\boldsymbol{u})(\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}) \tag{4.1.12a}$$

$$f_v(\boldsymbol{u}) = 2\boldsymbol{r}_v^T(\boldsymbol{u})(\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}) \tag{4.1.12b}$$

$$f_{uu}(\boldsymbol{u}) = 2\boldsymbol{r}_{uu}^T(\boldsymbol{u})(\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}) + 2\boldsymbol{r}_u^T(\boldsymbol{u})\boldsymbol{r}_u(\boldsymbol{u}) \tag{4.1.12c}$$

$$f_{vv}(\boldsymbol{u}) = 2\boldsymbol{r}_{vv}^T(\boldsymbol{u})(\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}) + 2\boldsymbol{r}_v^T(\boldsymbol{u})\boldsymbol{r}_v(\boldsymbol{u}) \tag{4.1.12d}$$

$$f_{uv}(\boldsymbol{u}) = 2\boldsymbol{r}_{uv}^T(\boldsymbol{u})(\boldsymbol{r}(\boldsymbol{u}) - \boldsymbol{p}) + 2\boldsymbol{r}_u^T(\boldsymbol{u})\boldsymbol{r}_v(\boldsymbol{u}). \tag{4.1.12e}$$

And the update relation for the curve and surface case is

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k - \left[\Delta\Delta^T(f)(\boldsymbol{u}_k)\right]^{-1} \Delta f(\boldsymbol{u}_k) \tag{4.1.13}$$

where $\boldsymbol{u}_0 = \boldsymbol{u}_a$.

4.1.2 Distance between a Point and an Implicit Entity

An implicit geometric entity is the zero set of a possibly vector-valued multivariate function $\boldsymbol{g}(\boldsymbol{r}) = 0$. Examples of this distance could be a point-to-curve or point-to-surface distance. The important thing to bear in mind is that the distance metric for an individual entity, once defined, makes the sets of implicit entities straightforward to implement. The distance from a given point $\boldsymbol{p}$ to an implicit entity I is given by

$$d(\boldsymbol{p}, I) = \min_{\boldsymbol{g}(\boldsymbol{r})=0} d(\boldsymbol{p}, \boldsymbol{r}) = \min_{\boldsymbol{g}(\boldsymbol{r})=0} \|\boldsymbol{r} - \boldsymbol{p}\|. \tag{4.1.14}$$

It is helpful to note that when computing the implicit entity distance from a point, the solution is never closed-form and are usually involved. Suppose that J is the set of N_I parametric entities, represented by I_k and $J = \{I_k\}$ for $k = 1, N_I$. The distance between a point $\boldsymbol{p}$ and the implicit entity set J is given by

$$d(\boldsymbol{p}, J) = \min_{k \in \{1,\ldots,N_I\}} d(\boldsymbol{p}, \boldsymbol{I}_k), \tag{4.1.15}$$

and the closest point $\boldsymbol{y}_j$ on the implicit entity I_j satisfies the equality $d(\boldsymbol{p}, \boldsymbol{y}_j) = d(\boldsymbol{p}, J)$. In order to compute the distance from a point to an implicit entity, we can create a simplex-based approximation such as line segments or triangles. The point-to-line or point-to-triangle set distance yields an approximate closest point $\boldsymbol{r}_a$ which can be used to compute the exact distance.

Typically, we must solve a constrained optimization problem when finding the closest point on an implicit entity, say $\boldsymbol{g}(\boldsymbol{r}) = 0$ to a point $\boldsymbol{p}$ in order to minimize a quadratic objective function that is subject to a nonlinear constraint

$$\min f(\boldsymbol{r}) = \|\boldsymbol{r} - \boldsymbol{p}\|^2 \tag{4.1.16}$$

where $\boldsymbol{g}(\boldsymbol{r}) = 0$ We can form the augmented Lagrange multiplier system of equations to solve the above, $i.e.$

$$\begin{aligned} \Delta f(\boldsymbol{r}) + \boldsymbol{\lambda}^T \Delta \boldsymbol{g}(\boldsymbol{r}) = 0 \\ \boldsymbol{g}(\boldsymbol{r}) = 0 \end{aligned} \tag{4.1.17}$$

where $\Delta - [\partial / \partial \boldsymbol{r}]^T$.

4.1.3 Quaternions

The unit quaternion is a four vector $\boldsymbol{q}_R = [q_0, q_1, q_2, q_3]^T$, where $q_0 \geq 0$, and $q_0^2 + q_1^2 + q_2^2 + q_3 = 1$, used to parameterize a rotation matrix. The 3×3 rotation matrix generated by a unit rotation quaternion is given by

$$R = \boldsymbol{q}_R^T \boldsymbol{q}_R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3 & 2(q_1 q_2 - q_0 q_3) & 2\left(q_1 q_3 + q_0 q_2\right) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 + q_1^2 - q_2^2 - q_3 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2 \end{bmatrix} \tag{4.1.18}$$

For more on unit quaternions, see § 6.5.3.

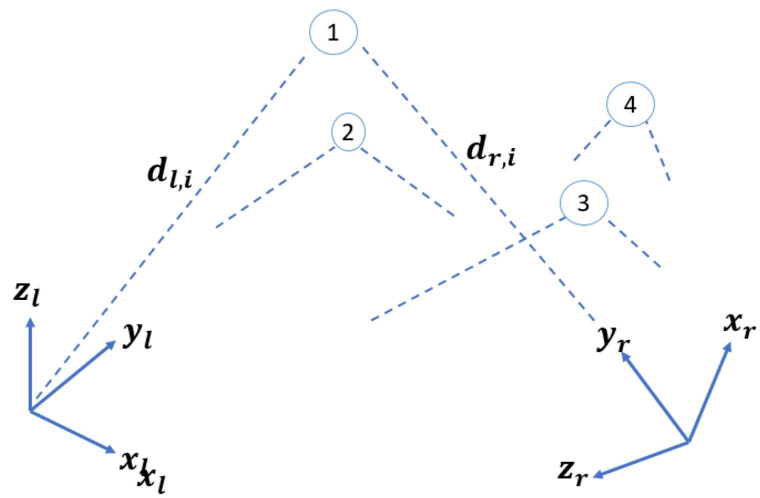

Figure 4.1: Given two coordinate systems, we measure a number of points in the two different coordinate systems. The goal is to find the transformation between the two points.

4.2 Closed-form Solution using Least Sum of Squares Errors

As we will see from our sensors, measurements are often inexact, which means we need a way to enforce greater accuracy when determining the transformation parameters. Therefore, we will need more than three points. One approach is to minimize the sum of squares of residual errors using various empirical, graphical, and numerical procedures. Because these are iterative in nature, they lead to an approximate solution and while the answer is better, it is imperfect.Iterative methods are repeatedly applied until the residual error is negligible.

There are closed-form solutions which present the absolute orientation in a single step with the best possible transformation given the measurements of the points in the two coordinate systems (Horn, 1987; Kabsch, 1978). With these closed-form least sum of squares methods, we do not need to find an initial good guess as is the case for iterative methods.

4.2.1 Kabsch Algorithm

Suppose we have two sets of vectors $\boldsymbol{x_n}$ and $\boldsymbol{y_n}$ where $n = 1, \ldots, N$, and weight w_n that corresponds to each pair $\boldsymbol{x_n}$ and $\boldsymbol{y_n}$. Our goal is to find an orthogonal matrix $\cup = (u_{ij})$ which minimizes the cost function

$$C = \frac{1}{2}\Sigma_n w_n \left(\cup \boldsymbol{x_n} - \boldsymbol{y_n}\right)^2 \tag{4.2.1}$$

subject to

$$\sum_k u_{ki}u_{kj} - \delta_{ij} = 0 \tag{4.2.2}$$

where δ_{ij} are the elements of a unit matrix. When there is a translation, we can find the centroid of the vector sets to the origin.

In order to solve the problem, we may introduce a symmetric Lagrangian matrix of multipliers, $L = (l_{ij})$ and an auxiliary function as follows

$$D = \frac{1}{2}\Sigma_{i,j} l_{ij} \left(\Sigma_k u_{kl} u_{kj} - \delta_{ij}\right) \tag{4.2.3}$$

so that we can form the Lagrangian, $E = C + D$. For each condition in eq. 4.2.2, we have an independent number l_{ij} so that the constrained minimum of C is part of the free minima of D. A free minimum of D can occur if

$$\frac{\partial E}{\partial u_{ij}} = \sum_k u_{ik} \left(\Sigma_n w_n x_{nk} x_{nj} + l_{k,j}\right) - \sum_n w_n y_{nl} x_{nj} = 0 \tag{4.2.4}$$

and

$$\frac{\partial^2 E}{\partial u_{mk} \partial u_{ij}} = \delta_{mi} \left(\Sigma_n w_n x_{nk} x_{nj} + l_{kj}\right) \tag{4.2.5}$$

are elements of a positive definite matrix x_{nk} and y_{nk} are the kth elements of $\boldsymbol{x_n}$ and $\boldsymbol{y_n}$. Now, suppose we have a matrix $R = (r_{ij})$ and a symmetric matrix $\boldsymbol{S} = (s_{ij})$, such that

$$r_{ij} = \sum_n w_n y_{ni} x_{nj} \tag{4.2.6}$$

and

$$s_{ij} = \sum_n w_n x_{ni} x_{nj}. \tag{4.2.7}$$

If the matrix (4.2.5) has 1 along its diagonal, we must have the minimum of the Lagrangian E to mean that $S + L$ is positive definite, and (4.2.4) translates to

$$U.(S + L) = R. \tag{4.2.8}$$

Our goal would be to find a matrix L of Lagrange multipliers so that $\cup$ is orthogonal. We can do this by multiplying both sides of (4.2.8) by their transposed matrices so that we can get rid of matrix $\cup$ as follows:

$$\begin{aligned} U(S + L)^T (S + L) &= (S + L)^T U^T U (S + L) \\ &= (S + L)(S + L) = R^T R. \end{aligned} \tag{4.2.9}$$

Now, we know that $R^T R$ is a symmetric positive definite matrix so that we can find the eigenvalues λ_k and eigenvectors $\boldsymbol{v}_k$ using standard procedures e.g. single value decomposition. Thus, since $S + L$ is symmetric and positive definite, it must have normalized eigenvectors, $\boldsymbol{v}_k$ and positive eigenvalues $\sqrt{\lambda_k}$ so that the Lagrange multipliers are

$$l_{ij} = \Sigma_k \sqrt{\lambda_k}; \qquad \boldsymbol{v}_{ki} \boldsymbol{v}_{ki} - s_{ij} \tag{4.2.10}$$

where $\boldsymbol{v}_{ki}$ signifies the ith component of $\boldsymbol{v}_k$ and the effect of the orthogonal matrix U on these eigenvectors $\boldsymbol{a}_k$ is determined from (4.2.8) which defines the unit vectors $\boldsymbol{q}_k$ as

$$\boldsymbol{q}_k = U.\boldsymbol{v}_k = \frac{1}{\sqrt{\lambda_k}} U(S + L)\boldsymbol{v}_k = \frac{1}{\sqrt{\lambda_k}} R\boldsymbol{v}_k. \tag{4.2.11}$$

The solution to find the constraint minimum of the minimum of the proposed cost function in (4.2.1) is then given by,

Kabsch's Optimal Rotation

$$u_{ij} = \Sigma_k b_{kl} a_{kj}. \tag{4.2.12}$$

4.2.2 Examples

There are clever ways of solving the optimal rotation between two vectors.

There is a jupyter notebook at the following link: Kabsch Algorithm and Implementation. For your convenience, it is included as a pdf file below.

Kabsch

December 15, 2020

0.1 Overview

Throughout this course, we will be leveraging Google's Colab Notebooks to reinforce the concepts we have been learning in class. For an introduction into how to use collab, in case you are not already familiar with it, have a go at this overview of Colaboratory features.

0.1.1 Kabsch's Algorithm

As stated in the course notes, the Kabsch algorithm is a very versatile tool for optimally aligning two vectors to one another. In this example, we are provided with two point sets - a model set and a point (measured) set, and our goal would be to compute the optimal rotation matrix U that allows us to efficiently rotate the point set into the model set.

0.1.2 Load the Measured Point Set

For the example we are interested in, we have measured the position of an object in 3D space using a Northern Digital Inc's Polaris Camera. The points are collected as a set of three-dimensional (3D) points in space, arranged in rows of `(x,y,z)` tuples and they are as given by the `measured_points_full` function below:

```
In [9]: # Here, we are importing all the libraries we will be using in these notebook
        import os
        import numpy as np
        from os.path import join, expanduser
        import scipy.linalg as LA
```

```
In [8]: def measured_points_full():
            # these are the (x,y,z) tuples
            pre_calib = {
                '0,0.0': [-369.88531494140625, 101.30087280273438, -1960.3780517578125],
                '200,0': [-369.8937683105469, 101.32111358642578, -1960.302734375],
                '0,0.1': [-369.8780212402344, 101.32646942138672, -1960.353271484375],
                '220,0': [-369.8780212402344, 101.32646942138672, -1960.353271484375],
                '0,0.2': [-367.74957275390625, 101.65080261230469, -1953.7960205078125],
                '240,0': [-370.8532409667969, 101.074951171875, -1942.255126953125],
                '0,0.3': [-366.7646484375, 101.17594909667969, -1949.628173828125],
                '255,0': [-381.33837890625, 97.10205078125, -1920.667236328125],
                '0,0.4': [-368.0609436035156, 100.83153533935547, -1953.857177734375],
                '0,220': [-382.8047790527344, 100.34918975830078, -1944.807373046875],
```

```
    '0,0.5': [-369.7981262207031, 100.01362609863281, -1958.6396484375],
    '0,240': [-382.71600341796875, 99.87244415283203, -1945.184326171875],
    '0,0.6': [-370.24237060546875, 98.66026306152344, -1957.2281494140625],
    '0,255': [-382.71600341796875, 99.87244415283203, -1945.184326171875],
    '0,0.7': [-370.1295166015625, 98.33242797851562, -1956.1732177734375],
    }
def exp(x):
    'This function expands the array along the second dimension so that '
    return np.expand_dims(x, 1)

# sort pre-recorded points in the order.
measured_calib = np.array(([[
                            pre_calib['0,0.0'],
                            pre_calib['0,220'],
                            pre_calib['0,0.1'],
                            pre_calib['0,240'],
                            pre_calib['0,0.2'],
                            pre_calib['0,255'],
                            pre_calib['0,0.3'],
                            pre_calib['200,0'],
                            pre_calib['0,0.4'],
                            pre_calib['220,0'],
                            pre_calib['0,0.5'],
                            pre_calib['240,0'],
                            pre_calib['0,0.6'],
                            pre_calib['255,0'],
                            pre_calib['0,0.7'],
                        ]]))
"""
 As it is currently, our array has 3 dimensions. We need to reduce the size of the
 array alongthe singleton dimention for efficient matrix manipulations, hence why we
 are squeezing the matrix
"""
measured_calib_zero_centered = np.array(([0, 0, 0]))
for i in range(len(measured_calib)):
    ' find the centroid of the points '
    centered = measured_calib[i] - np.min(measured_calib, 0)
    measured_calib_zero_centered = np.vstack((measured_calib_zero_centered, centered
measured_calib_zero_centered = measured_calib_zero_centered[1:]

return measured_calib_zero_centered
```

0.1.3 Load the model set

It now behooves us to load the model set so we can begin our Kabsch computation. For this, we have them saved in a numpy array. Therefore, we will import numpy as well as associated and needed libraries necessary for our computation.

```
In [12]: model_points = np.array((
                                  [[-1755.87720294,   866.87898685,   283.0353811 ],
                                   [-1755.76266696,   866.8540598,    282.9782946 ],
                                   [-1758.9453555,    857.8363267,    296.13326449],
                                   [-1759.02853104,   865.92774874,   283.52951211],
                                   [-1777.42772925,   826.8692224,    293.38292356],
                                   [-1784.34737705,   836.7521396,    281.74652354],
                                   [-1777.77335781,   826.96331701,   292.88727602],
                                   [-1783.56649649,   836.45510137,   281.56390533],
                                   [-1783.53245361,   836.46510174,   281.54257437],
                                   [-1783.6947516,    836.55773878,   281.52364873],
                                   [-1783.58522171,   836.46979064,   281.55684051],
                                   [-1783.66230977,   836.54098015,   281.50709046],
                                   [-1783.52724697,   836.44927943,   281.56064662],
                                   [-1783.59681243,   836.52118858,   281.52347799],
                                   [-1783.44129296,   836.40624764,   281.5671847 ]]
         ))
```

0.1.4 Get the point set from the function above.

```
In [13]: point_set = measured_points_full()
```

0.2 Now, let us calculate the transformation as we described in our notes

```
In [23]: def Kabsch(P=None, Q=None, augment_Q=True, center=True):
             '''P and Q must be nX3. This rotation is accurate.
              Rotates points in P optimally to measured reference points in Q

              Params
              ======
              Q: Points to be rotated into
              augment_Q: Whether Q was recorded without the zero/home points embedded between su

              '''
             if not isinstance(P, np.ndarray) or not isinstance(Q, np.ndarray):
                 P, Q = prepro()

             # calculate the centroids
             if center:
                 'This only for computed old points'
                 q0 = np.mean(Q, 1)
                 p0 = np.mean(P, 1)

                 Q_ctr = Q - np.expand_dims(q0, 1)
                 P_ctr = P - np.expand_dims(p0, 1)
             else:
                 Q_ctr, P_ctr = Q, P
```

```
    # add the zero points to preconputed control points
    if augment_Q:
        'This only for computed old points'
        Q_aug = np.array(([[0,0,0]]))
        for i in range(Q_ctr.shape[0]-1):
                Q_aug = np.append(Q_aug, np.expand_dims(Q_ctr[i+1], 0),0)
                Q_aug=  np.append(Q_aug, np.expand_dims(Q_ctr[0], 0),0)
        Q_ctr = Q_aug[1:]


    Hmat = P_ctr.T@Q_ctr
    U, S, V = LA.svd(Hmat)
    d = np.sign(np.linalg.det(V@U.T))
    M = np.eye(3); M[-1][-1] =d
    opt_rot = V@M@U.T
    opt_trans = Q_ctr.T- opt_rot@P.T

    return opt_rot, np.mean(opt_trans, 1)
```

0.2.1 Test the algorithm

Remember that we are rotating the points in `point_set` into `model_points`. So we would go ahead and call the Kabsch function above as follows:

```
In [24]: Rot, Trans = Kabsch(model_points, point_set, augment_Q=False, center=False)

In [25]: print(Rot)

[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]


In [26]: print(Trans)

[1775.85125374 -842.66314863 -284.40256961]
```

Homework 19. For the following model points P and measured poinyts Q, compute the optimal rotation matrices for moving points Q into point P. For the three assignments below, report your results within a colab notebook, download the colab notebook as a pdf and upload on Latte.

1.
$$P = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \qquad Q = \begin{bmatrix} 0 & -1 & -1 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{4.2.13}$$

2.
$$P = \begin{bmatrix} 3172.79468418 & 727.52462347 & 7122.70450243 \\ 165.28953155 & -3552.32467068 & -2045.15346584 \\ 5292.45250241 & -1748.52037006 & -6181.40300009 \\ 1893.07584225 & 5897.19719625 & 3130.41287776 \end{bmatrix}, \tag{4.2.14}$$
$$Q = \begin{bmatrix} 1774.11606309 & -4241.11341178 & 5259.04277742 \\ 6079.70499031 & -98.14197972 & -3442.0914569 \\ 813.07069876 & 3334.26289147 & -6112.55652513 \\ 1856.72080823 & 2328.86927901 & 6322.16611888 \end{bmatrix}$$

3. For a toy problem, measure the coordinates of an object in the world using your favorite measuring instrument (a 3D camera sensor, iPhone app (e.g. ArkIt), android app e.t.c.). Be sure to record the position of the object at multiple points in world coordinates and make sure that the physical locations of these points are known (these are your model points). Then compute the optimal rotation and translation between the model and measured points.

4.2.3 Corresponding Point Set Registration with Quaternions

While the Kabsch algorithm does yield an optimal solution for the rotation of two sets of points that correspond to one another, it leverages the orthonormal rotation matrix with positive determinant in its computations. This suffers from the non-uniqueness of solutions that arise from reflections. Using matrices straightforward is problematic because we need six nonlinear constraints to guarantee the orthonormality of the rotation matrix. To yield a least squares rotation, and translation, we will generally avoid singular value decomposition (SVD) methods in two and three dimensions since we generally do not want reflections. For $n > 3$ in any n-dimensional application, the SVD approach, based on the cross-covariance matrix of two point distributions, does generalize easily to n dimensions.

Let $\boldsymbol{t} = [t_x, t_y, t_z]^T$ denote the translation vector and $\boldsymbol{q}_R = [q_0, q_1, q_2, q_3]^T$ denote the unit quaternion. Suppose further that the complete registration set of vectors is $\boldsymbol{H} = [\boldsymbol{q}_R|\boldsymbol{t}]^T$. Now, let $D_l = \{\boldsymbol{d}_{l_i}\}$ be the measured set of points which we want to align with the model point set $D_r = \{\boldsymbol{d}_{r_i}\}$, where the cardinality, N_l of D_l is same as that of D_r, N_r, and where each point $\boldsymbol{d}_{l_i}$ corresponds to point $\boldsymbol{d}_{r_i}$ with the same index. We are looking for a transformation of the form

$$D_r = a\boldsymbol{R}(D_l) + \boldsymbol{t} \tag{4.2.15}$$

from the left to the right coordinate system as shown in Fig. 6.4, where a is a scale factor, and $\boldsymbol{t}$ is the translation vector offset. $R(D_l)$ denotes the rotated version of D_l. Since we do not expect to have a perfect data, it will be difficult to find a scale factor, a translation and a rotation so that the transformation equation is satisfied for every point. Thus, there will be a residual error,

$$\boldsymbol{e}_i = \boldsymbol{d}_{r,i} - a\boldsymbol{R}(\boldsymbol{d}_{l,i}) - \boldsymbol{t} \tag{4.2.16}$$

and the cost function will minimize the sum of squares is given as,

$$f(\boldsymbol{s}) = \min \|\boldsymbol{e}_i\|^2. \tag{4.2.17}$$

Finding Translation

We can find the translation, scale and finally rotation by systematically varying the total error.

Consider the centroids of the measured and point sets,

$$\bar{D}_l = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{d}_{l,i}, \qquad \bar{D}_r = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{d}_{r,i}, \tag{4.2.18}$$

so that the new coordinates are

$$\boldsymbol{d}'_{l,i} = \boldsymbol{d}_{l,i} - \bar{\boldsymbol{d}}_l, \qquad \boldsymbol{d}'_{r,i} = \boldsymbol{d}_{r,i} - \bar{\boldsymbol{d}}_r. \tag{4.2.19}$$

If we write $\boldsymbol{t}' = \boldsymbol{t} - \bar{\boldsymbol{t}} + a\boldsymbol{R}(\boldsymbol{d}_l)$, it follows that we can write the error as

$$\boldsymbol{e}_i = \boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i}) - \boldsymbol{t}' \tag{4.2.20}$$

and the sum of squares of errors becomes

$$\sum_{i=1}^{n} \|\boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i}) - \boldsymbol{t}'\|^2 \equiv \sum_{i=1}^{n} \|\boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i})\|^2 - 2\boldsymbol{t}'.\sum_{i=1}^{n} \left[\boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i})\right] + n\|\boldsymbol{t}'\|^2. \tag{4.2.21}$$

The middle term on the right hand side vanishes since the measurements are referred to the centroid and we are left with the first and the third terms. The first term is independent of $\boldsymbol{t}'$ and the last term cannot be negative given the squared norm. Thus, the total error to be minimized with $\boldsymbol{t}' = 0$ is

Optimal Translation

$$\boldsymbol{t} = \bar{\boldsymbol{d}}_r - a\ R(\bar{\boldsymbol{d}}_l) \tag{4.2.22}$$

In other words, *the translation is the difference between the right centroid and the scaled and rotated left centroid.*

We can now rewrite the error term from (4.2.20) as

$$\boldsymbol{e}_i = \boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i}) \tag{4.2.23}$$

since $\boldsymbol{t}' = 0$. So the total error to be minimized is

$$\sum_{i=1}^{n} \|\boldsymbol{d}'_{r,i} - a\boldsymbol{R}(\boldsymbol{d}'_{l,i})\|^2. \tag{4.2.24}$$

Finding Scale

Expanding (4.2.24), we find that

$$\sum_{i=1}^{n} \|\boldsymbol{d}'_{r,i}\|^2 - 2a \sum_{i=1}^{n} \boldsymbol{d}'_{r,i}.R(\boldsymbol{d}'_{l,i}) + s^2 \sum_{i=1}^{n} \|\boldsymbol{d}'_{l,i}\|^2, \tag{4.2.25}$$

and since rotation preserves distances, $\|R(\boldsymbol{d}'_{l,i})\|^2 = \|\boldsymbol{d}'_{l,i}\|^2$, we can write the foregoing as $S_r - 2sD + s^2 S_l$, where S_r and S_l are the sums of the squares of the measurement vectors (relative to their centroids), while D is the sum of the dot products of corresponding coordinates in the right system with the rotated coordinates in the left system. Completing the square in s, we find that

$$\left(a\sqrt{S_l} - D/\sqrt{S_l}\right)^2 + \left(S_r S_l - D^2\right)/S_l. \tag{4.2.26}$$

If we minimize with respect to scale a when the first term is 0 or $a = D/S_l$, we find that

$$s = \frac{\sum_{i=1}^{n} \boldsymbol{d}'_{r,i} \cdot R(\boldsymbol{d}'_{l,i})}{\sum_{i=1}^{n} \|\boldsymbol{d}'_{l,i}.\|^2.} \tag{4.2.27}$$

Finding rotation

To find the optimal rotation, we note that the cross-covariance matrix Σ_{lr} between the sets D_l and D_r is given by

$$\Sigma_{lr} = \frac{1}{N_l} \sum_{i=1}^{N_l} \left[(\boldsymbol{d}_{l,i} - \bar{\boldsymbol{d}}_l)(\boldsymbol{d}_{r,i} - \bar{\boldsymbol{d}}_r)^T\right] \tag{4.2.28}$$

$$= \frac{1}{N_l} \sum_{i=1}^{N_l} \left[\boldsymbol{d}_{l,i}\boldsymbol{d}_{r,i}^T\right] - \bar{\boldsymbol{d}}_l \bar{\boldsymbol{d}}_r^T. \tag{4.2.29}$$

The cyclic components of the skew symmetric matrix $Q_{ij} = \left(\Sigma_{lr} - \Sigma_{lr}^T\right)_{ij}$ are used to construct the column vector $\Delta = [Q_{23} \quad Q_{31} \quad Q_{12}]^T$, so that the vector is then used to form the symmetric matrix

$$Q(\Sigma_{lr}) = \begin{bmatrix} \mathrm{tr}(\Sigma_{lr}) & \Delta^T \\ \Delta & \Sigma_{lr} + \Sigma_{lr}^T - \mathrm{tr}(\Sigma_{lr})\boldsymbol{I}_3 \end{bmatrix} \tag{4.2.30}$$

where $\boldsymbol{I}_3$ is the 3×3 identity matrix and the unit eigenvector $\boldsymbol{q}_R = [q_0 \quad q_1 \quad q_2 \quad q_3]^T$ that corresponds to the maximum eigenvalue of $Q(\Sigma_{lr})$ is chosen as the optimal rotation.

4.3 Iterative Closest Point

The Iterative Closest Point (ICP) algorithm applies to the following sets of problems (i) sets of points, (ii) sets of line segments, (iii) sets of parametric curves, (iv) sets of implicit curves, (v) sets of triangles, (vi) sets of parametric surfaces, and (vii) sets of implicit surfaces. To properly describe the algorithm, we choose a data, P, which is to be moved or registered/positioned to best align with a "model" data X. It is best if the data and model shape are decomposed into a point set if they are not already in point set form. For triangles and line segments, we use their vertices and endpoints respectively; while for curves and surfaces, an approximation to the vertices and endpoints of triangles and lines are used. Suppose we denote, as before, the number of points in the data shape as N_p and N_x as the number of points, line segments, or triangles in the model shape. The distance metric d between an individual data point $\boldsymbol{p}$ and a model shape X will be denoted

$$d(\boldsymbol{p}, X) = \min_{\boldsymbol{x}(X)} \|\boldsymbol{x} - \boldsymbol{p}\|. \tag{4.3.1}$$

The closest point in X that yields the minimum distance is denoted $\boldsymbol{y}$ such that $d(\boldsymbol{p}, \boldsymbol{y}) = d(\boldsymbol{p}, X)$, where $\boldsymbol{y} \in X$.

Quiz 5. 1. What is the worst case asymptotic computation for the closest point in X and why?

2. What is the expected worst case computation time?

When the closest point computation from $\boldsymbol{p}$ to X is performed for each point P, that process is worst case $O(N_p, N_x)$. Let Y denote the resulting set of closest points, and $\mathcal{C}$ the closest point operator, $i.e.$

$$Y = \mathcal{C}(P, X). \tag{4.3.2}$$

For the resultant corresponding point set Y, the least squares registration can be computed as

$$(\boldsymbol{q}, d) = \mathcal{Q}\left(P, Y\right). \tag{4.3.3}$$

and the positions of the data shape point set are] then updated via $P = \boldsymbol{q}(P)$.

Algorithm 1 ICP Algorithm

```
Given point set $P$ with $N_p$ points $\{\boldsymbol{p}_i\}$ from the data shape and the model shape $X$ with $N_x$ supporting geometric primitives: points, lines, or triangles
Start the iteration with $P_0$ set to $P$, $\boldsymbol{q}_0 = [1, 0, 0, 0, 0, 0, 0]^T$ and $k = 0$ and define the registration vector relative to the initial data set $P_0$ so that the final registration denotes the complete transformation.
Given a mean-square error with preset threshold $\tau > 0$, and a desired registration accuracy, $d$
while $\tau > d_k - d_{k+1}$ do
    Compute the closest points, $Y_k = \mathcal{C}(P_k, X)$ (cost: $O(N_o, N_x)$, worst-case: $O(N_p \log N_x)$ average).
    Compute the registration: $(\boldsymbol{q}_k, d_k) = \mathcal{Q}(P_0, Y_k)$ (cost: $O(N_p)$).
    Apply the registration, $P_{k+1} = \boldsymbol{q}(P_0)$ (cost: $O(N_p)$).
end while
```

CHAPTER 5

RIGID BODY MOTIONS: INTRODUCTION

We are chiefly concerned with *rigid bodies* and occasionally *semi-rigid* or *soft* bodies connected together by *joints*. In general, we take robots to be *mechanisms* that are made up of *links* connected to one another by *joints*. Typically, the joints connect two or more links and are formed by simple contact with adjacent bodies. Sometimes, the joints may be flexible – whether by belt, band, spring or some kind of elastic component such as bellows, diaphragms, tendons (Bern et al., 2017), fiber-reinfoced elastomers (Bishop-Moser et al., 2012), resilient pads, strip, or bush. The assembly formed after the various connections between links and joints are called a *kinematic chain*.

A kinematic chain is a form of a *mechanism*. A "*mechanism can be interpreted as a means of transmitting, controlling, or constraining relative movement*" (Hunt, 1977). The term *kinematics* is generally used to describe the motion of a rigid body in space. It also applies to the motion of a semi-rigid or a completely soft robot in space. The rigid, semi-rigid or completely soft kinematic systems that allow a body to exhibit motion under a/some controlled motion of its freedom in space with respect to a fixed base frame are what we describe in this module.

Figure 5.1: Early serial kinematic chain robot manipulators. *Left:* The Stanford Arm Serial Manipulator, 1969. ©Infolab, Stanford University. *Right:* The PUMA robot arm. Reprinted from Wikipedia.

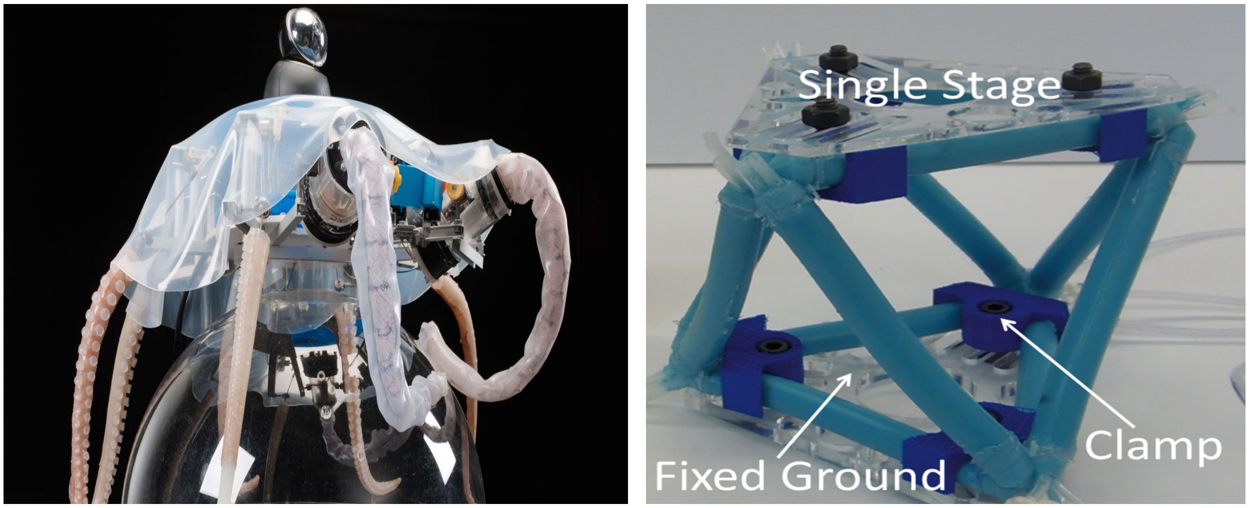

Figure 5.2: Soft Robot Manipulators *Left:* An octopus inspired soft robot. Robots classified under muscular hydrostats. ©Cecilia Laschi. *Right:* A soft parallel robot manipulator (Hopkins et al., 2015).

5.1 Robots Components

The joints of a robot can be *revolute* or *prismatic*. Revolute joints allow rotation about a fixed axis between two or more links while prismatic joints allow a linear motion about an *axis* – which is between two links. Symbolically, the revolute joint is denoted by R while the prismatic joint is denoted by P. By this logic, it follows that a robot made up of three links that are connected by a prismatic joint would be characterized as a PPP arm, while a three link arm whose links are all connected by three rotary joints would be denoted as an RRR arm.

A mathematical way to abstract the interconnection between the links of a robot arm is to denote the axis of rotation or the axis of translation by z_i for the joint that connects link i to link $i+1$. When we characterize the robot's motion, it often helps to work in a so-called *generalized coordinate*. Now consider a rigid body such as a link of a robot arm for example. This link contains myriads of particles, with each particle having its own position and orientation in the world. How can we describe the motion of this rod given its infinitely many particulate matter? For rigid bodies, classical mechanics comes to the rescue. As such, for a revolute joint of a rigid robot manipulator for example, the generalized coordinate is often the *joint angle* θ, while for a prismatic joint it is the link length d – representing the relative displacement between adjacent links. When the body we speak of is a continuum, the motion of each particle within a link needs to be accounted for. In such scenarios, researchers often work with a so-called a Frenet-Serret frame that models a curve on the soft robot's surface – with or without torsion. This simplifies the dynamics and kinematics of the system and the specification of these generalized coordinates uniquely determines the position of all the particles (in a material sense for a rigid system) that make up the robot. For more examples on parameterizing soft robots, please look through some of the following references: (Ogunmolu et al., 2015), (Hannan and Walker, 2000) (Ogunmolu et al., 2016), (Renda and Seneviratne, 2018), (Shepherd et al., 2011), (Hannan and Walker, 2003), (Ogunmolu et al., 2017), (Giorelli et al., 2015), (Ogunmolu et al., 2020), (Renda et al., 2014), (Ogunmolu, 2019a), (Ogunmolu, 2019b).

5.2 Robot Spaces, Geometries, and Classifications

The location of a robot in the world is described by its *configuration*. This configuration specifies the positions of all points of the robot. While the coordinates of these points may be multiple over a continuous range of real numbers, the smallest number of independent variables of these real-valued coordinates needed to represent the robot configuration are the *degrees of freedom* of the system. In general, a *free rigid body* has six degrees of freedom. The motion of the robot could be about a translation or rotary axis. The n-dimensional space that contains all the possible configurations of the robot is the *configuration space* (or *C-space*) of the robot. The configuration of the robot is expressed by a point in its C-space.

A manipulator with more than 6-DOF is said to be *kinematically redundant*. The set of variables that together with a description of the manipulator's dynamics and future inputs that determines the future transient response of the manipulator is the *state* of the robot, while the set of all possible states is referred to as the *state space*. For a rigid robot manipulator for example, the material and referential description alone is sufficient to describe the dynamics, which are Newtonian in nature. For such systems, the state may be specified by the joint variables $\{q, \dot{q}\}$ signifying the joint angles and joint velocities. For continuum systems, such representations are often inadequate. A popular method is to introduce a Frenet-Serret (you can read more about F-S frames in the link included earlier) frame on the body of the soft robot (SoRo) and characterize the state space based on the three parameters $i.e.$ $\{\mathcal{K}, l, \alpha\}$ that characterize the kinematic motion of the body viz., curvature of an arc projected on the SoRo's body, $\mathcal{K}$, the arc's length, l, and the angle subtended by a tangent along that arc, α.

5.3 Characterization of Kinematic Geometry

As mentioned earlier, disassociated from the dynamics ($i.e.$ forces, torques and such) of a body, that which deals with the displacements, or movements of a body relative to another within a mechanism is termed the *kinematics*. The displacement could be linear, angular – and in combination with the derivatives with respect to time of such displacements viz., velocities, accelerations, and hyper-accelerations all are part of the kinematics of a body. The other branch of *dynamics*, called *kinetics*, determines the forces, torques, energy, momenta, inertia, equilibrium and dynamic stability of the system and that is treated in a separate module in this course.

5.3.1 Kinematic Geometries

Kinematic geometry deals only with the *displacements* of a system in the first and simplest segment of kinematics. In general, the use of time as a variable is shunned upon since the displacements we intend to carry out may be performed as fast as user wishes in their implementation. For the rest of this subsection, by kinematics, we shall mean the displacement of rigid bodies or “the solid geometry of relatively moving bodies".

The majority of robot arms (at least today) have their actuators (or joints) connected in series along an *anthropomorphic* arm, with each joint being either at or associated with a single DOF in the robot arm. Fig. 5.3 describes the two major ways rigid links are actuated today. From a geometric point of view, both types of actuation in Fig. 5.3 serve the same purpose, which is to control the

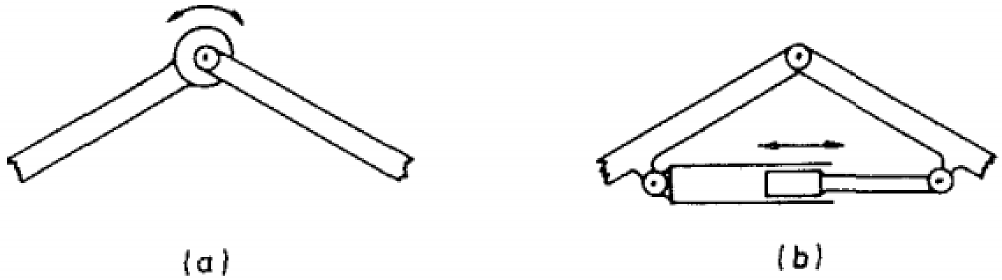

Figure 5.3: Schematic description of in-series-actuated robot arms: (a) rotary joint actuated “about" the hinge (b) prismatic joint actuated “across" a hinge. Reprinted from (Hunt, 1983).

rotation of the joint. However, when we are interested in controlling “pure translation" such as sliding between links, non-pivoted linear actuators are used on their own.

5.3.2 Open Kinematic Chains

When the links are serially connected to one another and there is no link that connects the base link to the tool frame, the robot is said to be in-series actuated. Series-jointed links have a major disadvantage, namely that they accumulate errors from shoulder out to end-effector and they suffer from a lack of rigidity. In order to eliminate their inherent load-dependent error, manufacturers typically stiffen them. However, this stiffening process increases the mass of the arm such that greater demand is placed on the actuation system. Some manufacturers employ a compensated actuation or sophisticated control techniques in order to overcome the load-dependent errors.

Examples of in-series actuated mechanisms is the *spherical robot* (see Fig. 5.5), where a succession of the robot segments are linked to the predecessors by revolute joints. By actuating each of the n joints we can control the n degrees of freedom of the end-effector. The SCARA robot, shown in Fig. 5.5 is an example that allows the control of the end-effector based on the geometry of the previously connected links.

It is ideally desirable to keep the load-to-mass ratio of a robot as minimum as possible whilst preserving positioning accuracy. This is so as to preserve two important properties of the robot:

Figure 5.4: The Staubli 6-DOF Arm is an example of a Spherical Manipulator. Reprinted from DirectIndustry's Webpage.

repeatability ($i.e.$ the maximum distance between two positions of the end-effector reached for the same desired pose from different starting positions), and absolute accuracy ($i.e.$ distance between the desired and actual position of the end-effector) respectively. Positioning accuracy is a function of the deformation of flexure – typically not accounted for by the robot's internal sensors – and the

Figure 5.5: The SCARA Arm. ©Fanuc America.

absolute accuracy. The absolute accuracy is a function of the sensors at the manipulator joints, the clearance for the drive, flexure of links, and geometric accuracy of link orientations inter alia. When geometric constraints are violated (take a small error in the perpendicularity between successive links of a spherical arm, for example), large errors are bound to occur in the vertical motions so that such errors must be accounted for during manufacture. The conventional approach to overcoming this currently is for manufacturers to stiffen the robot's links. However, stiffening the links is tantamount to overall heavier mass, which in turn means the manipulator experiences higher inertia, centrifugal and Coriolis forces – these complicate motion planning for complex trajectories.

As we close this part of the module, bear in mind that open kinematic chains are governed by inertia and centrifugal forces – forces that exist on different scales. For example, inertia forces are a function of the square of the link lengths while frictional forces are not affected by the such dimensions. As such, serial robots cannot be scaled down to a micro level as inertia forces would be reduced while frictional forces would remain unchanged. It is not surprising that these attributes listed make serial manipulators exhibit poor positioning accuracy.

5.3.3 Closed-loop Kinematic Chains

When the number of links connected to a joint (the *connection degree*) is more than 2, we have a closed-loop kinematic chain. Closed-loop kinematic chains resolve the accuracy problems of open-loop chains by mechanically distributing the load on the links: they link the end-effector to the ground by a set of links that support only a fraction of the load. While theoretical works that envisioned parallel mechanisms have been in existence since 1645 (Merlet, 2015), the Stewart (Stewart, 1965) and Gough (Gough, 1957) platforms are the original designs of closed-loop chains on record. The mechanical arrangement of the links of the platform help with the load-to-mass ratio: the manipulator mass is reduced, and the disturbing effects of the Coriolis forces decreases. Albeit, these manipulators typically have a smaller workspace. Formally, we define a *generalized parallel manipulator* as a closed-loop kinematic chain mechanism whose end-effector is linked to the base by many independent kinematic chains (Merlet, 2015).

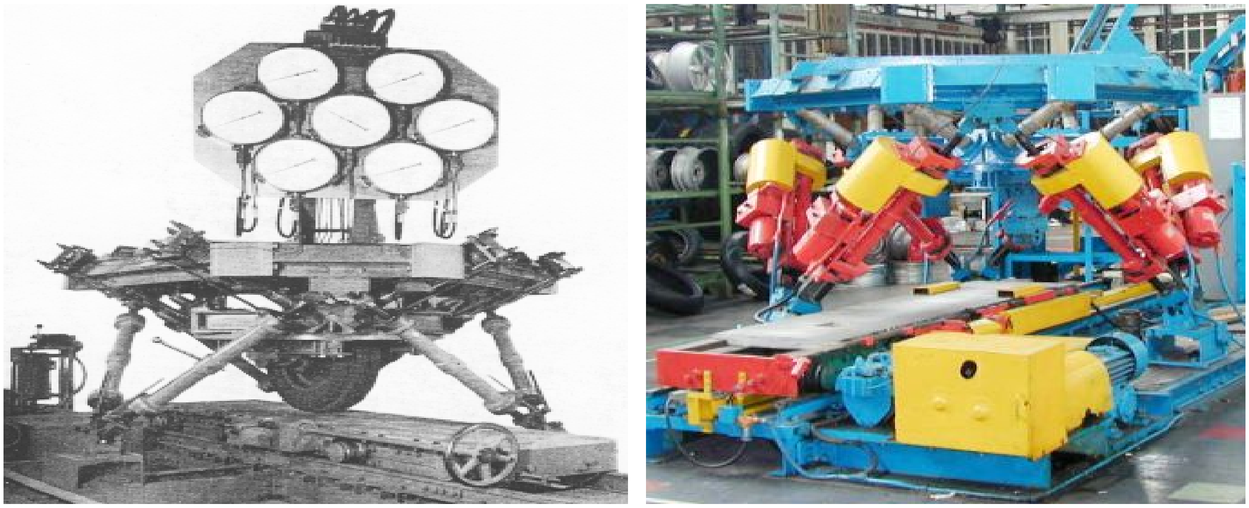

Figure 5.6: Stewart-Gough Platforms *Left:* The 1954 Octahedral hexapod of the original Gough platform. *Right:* The retired platform from Dunlop tires in 2000. Picture courtesy of Parallemic.org.

5.4 Freedom and Structure in Mechanisms

Early on, we briefly defined what constitutes the degrees of freedom of a rigid body. We will now systematically analyze how to determine the mobility of a mechanism given its *kinematic pairs*, linkages and freedoms.

5.4.1 Freedom, Connectivity and Mobility

For every *kinematic pair*[1], there exists a characteristic number of the degrees of freedom that characterize its mobility. If a kinematic pair has elements that touch at a single point, we would have five degrees of freedoms – two of which would be translatory and three rotary. A kinematic pair whose elements always touch along a line or a curve has four or fewer degrees of freedom, or for short, freedom (see Fig. 5.7).

However, in mechanisms, we are more concerned about the relative motion between objects in the mechanisms that do not directly touch one the other. In such systems, what we do is consider the freedom that is contributed by each of the several kinematic pairs that connect them. For the popular

[1] Bonus homework: Research what is a kinematic pair and produce a one-page written report.

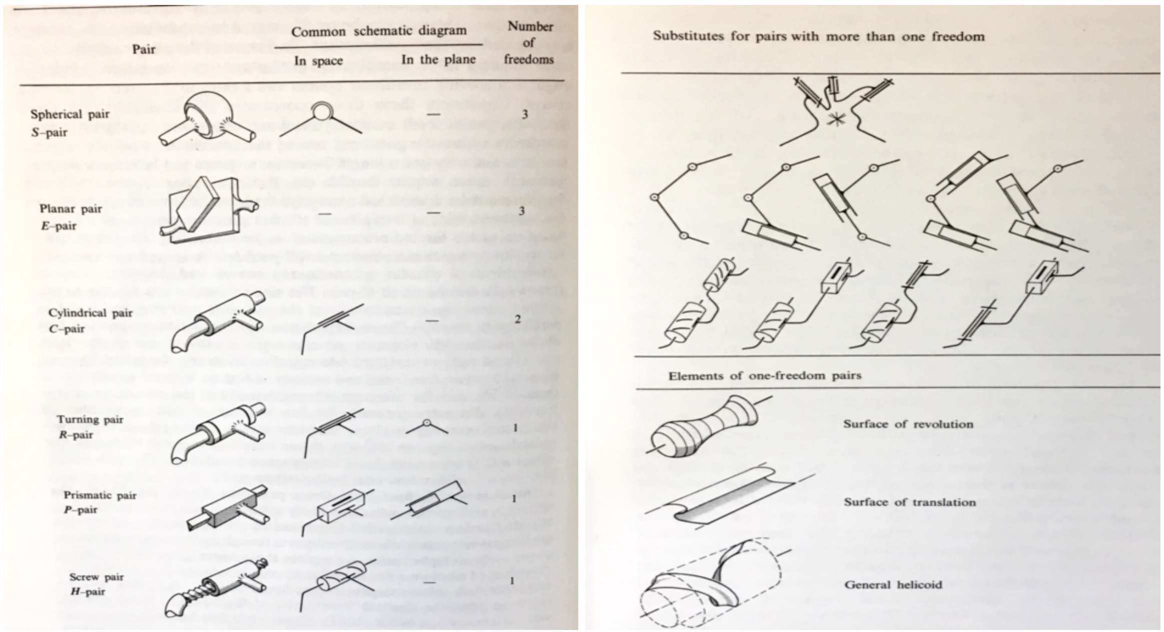

Figure 5.7: The Lower Kinematic Pairs. Reprinted from (Hunt, 1977).

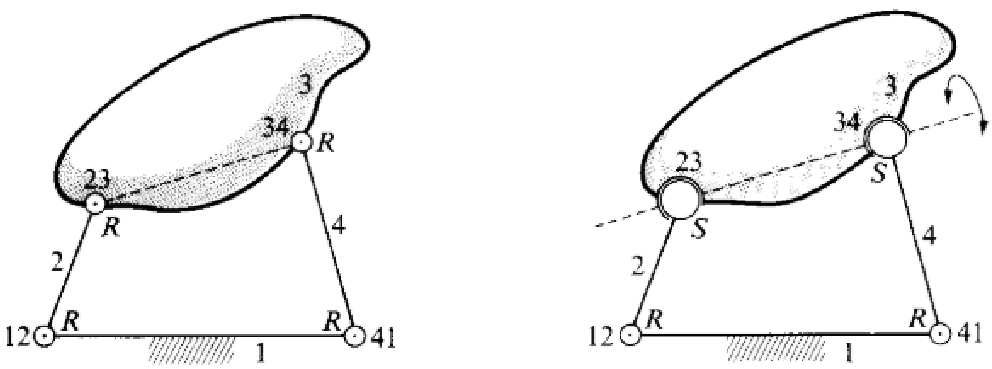

Figure 5.8: The planar RRR linkage, (*left*) is modified in (*right*) to allow spatial spin-movement of the coupler 3; the connectivity $\mathscr{C}_{13} = 2$). Reprinted from (Hunt, 1977).

four bar linkage, for example (see (Murray, 1994)), the four R-pairs on the two connectivity-links 2 and 4 complete a *coupling* or *connection* between 1 and 3. This is said to have a connectivity of 1, typically written as $\mathscr{C}_{13} = 1 = \mathscr{C}_{31}$. Note that here, all connectivities are the same , and $\mathscr{C}_{ij} = 1$, for $i \neq j$ and $i, j = 1, 2, 3, 4$. Now, suppose that the R pairs are replaced by spherical, or for short, S pairs, then we have an additional so-called *spin-freedom* about the SS axis (see figure Fig. 5.8) so that $\mathscr{C}_{13} = 2$. In any case, $\mathscr{C}_{12} = \mathscr{C}_{13} = \mathscr{C}_{43} = 2$.

Related to the concept of connectivity and more popular in literature is the concept of *relative mobility* or simply *mobility*. The mobility, $\mathfrak{M}$, is usually referred to as the number of *degrees of freedom* of a mechanism. To specify the independent variables needed to determine all the relative locations of the complete members of a mechanism with respect to one another, we typically use the *mobility criterion*. As such, we would have the kinematic chain of the left inset of Fig. 5.8 having $\mathfrak{M} = 1$ since only an angle between the elements of any of the four kinematic pairs is required so as to prevent all relative movement. However, for the $RSSR$ mechanism on the right of Fig. 5.8, the mobility would be $\mathfrak{M} = 2$ since member 3 has a spin-freedom.

Determining Degrees of Freedom

Screw Coordinates are better suited – from a kinematic standpoint, at any rate – to determine the general and impartial location of a rigid body.

5.4.2 Constraints and Freedoms

In rigid body kinematics, it is a given that the freedoms of a *free rigid body* is 6. But how is this determined? Suppose we have six homogeneous *screw coordinates* (to be introduced shortly) $x_1, x_2, \ldots, x_6$, we find that the body is fixed when values are attached to all six of them. However, the physical constraining system to which the body is subjected is not likely to have for every constraint exactly one corresponding independent screw coordinate. For every constraint in the system, there is an influence on each of the six coordinates. For every independent constraint, a freedom of the body is suppressed so that the six independent constraint-equations $f_i(x_1, x_2, \ldots, x_6) = 0$, $i = 1, 2, \ldots, 6$ must all be simultaneously satisfied. We define the *condition* as the constraint equation that describes a particular constraint algebraically. As the constraints are progressively relaxed, the corresponding constraint-equations are struck out one after another, so that the body acquires one, two, . . ., degrees of freedom, until the body is completely free with no constraints or equations and we end up with six freedoms. Suppose the number of freedoms is f and the number of *unfreedoms* or constraints is

u, it turns out that

$$u + f = 6. \tag{5.4.1}$$

5.4.3 The Mobility Criterion

For g working joints between a total of n bodies, the number of relative degrees of freedom can be identified with the relative mobility of the system of bodies, described as

$$\mathfrak{M} = 6(n-1) - \sum_{i=1}^{g} u_i \tag{5.4.2}$$

where the summation term aggregates over all individual unfreedoms. Plugging (5.4.1) into (5.4.2), we have

$$\mathfrak{M} = 6(n-1) - \sum_{i=1}^{g} 6 + \sum_{i=1}^{g} f_i \tag{5.4.3}$$

or

$$\mathfrak{M} = 6(n-g-1) + \sum_{i=1}^{g} f_i. \tag{5.4.4}$$

Equation (5.4.4) is termed the general mobility criterion. It is attributed to Grübler (1908 and 1917), and independently to Kutzbatch (1929). In this module, we will generally call it the Grübler-Kutzbach's mobility criterion. When there are independent kinematic chains within the body of concern, it is sometimes more convenient to write (5.4.4) as

$$\mathfrak{M} = \sum_{i=1}^{g} f_i - 6c \tag{5.4.5}$$

with c being the number of independent chains.

There are exceptions to the Grübler-Kutzbach's mobility criterion such as when we have planar or spherical mechanisms. Take the four-bar linkage for example. There are four freedoms to the kinematic pairs, yet patently it has a mobility of $\mathfrak{M} = 1$, and not $\mathfrak{M} = -2$ as (5.4.5) would have us so determine it. This is because for planar and spherical mechanisms, the number of freedoms is not six as in *general* space, but three. Therefore, we write out the special version of (5.4.4) as

$$\mathfrak{M} = 3(n - g - 1) + \sum_{i=1}^{g} f_i. \tag{5.4.6}$$

This special nature of (5.4.6) arises because the joints' freedoms are not independent in planar motion, because the axes of the turning pairs are all directed parallel to one another, any prismatic pair being perpendicular to them. For spherical motions, the turning axes co-intersect at a single point.

Homework 20. Read up on the common kinematic arrangements in Section 1.3 of (Spong et al., 2006) and produce a 2-page single-spaced summary report. Your report must not contain diagrams but feel free to do as much analysis of the configurations of the various kinematic arrangments that are mentioned.

- Using the mobility condition, determine and explain why the SCARA robot of Fig. 5.5 has the number degrees of freedom that you find.
- For the mobile manipulator we are using in this course, analyze the connectivities and freedoms of the kinematic pairs in the mechanism. In addition, determine the freedom of the overall mechanism and write out the mobility criterion.
- With the *Grübler-Kutzbach's mobility condition* that we have learned, analyze the mobility criteria of the mechanism of Fig. 5.9. Hint: This mechanism is made up of two chains: chains $A_3\, B_3\, B_1\, A_1$ and $A_2\, B_2\, B_4\, A_4$, and there is a fixed distance between the U-joints, $A_2\, A_3$ as well as A_1, A_4.

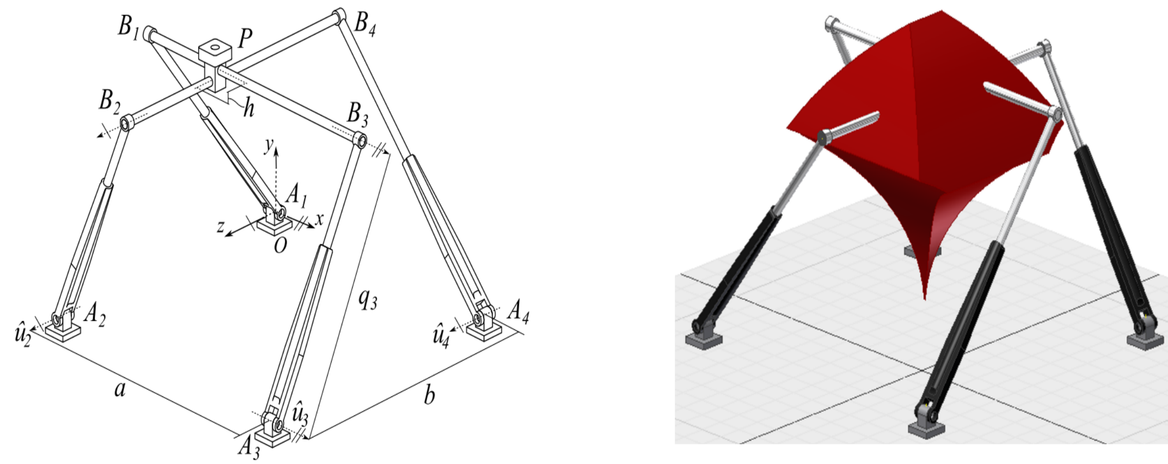

Figure 5.9: *Left*: A Parallel Planar Robot Mechanism. *Right:* Workspace of the mechanism.

CHAPTER 6

MOTION OF RIGID BODIES

In this chapter, we shall lay the foundations for analyzing the kinematics of a rigid or soft body in general space. Our goal is to present a geometric view of the translational and rotational rigid body motions. We take the classical screw theory approach owing to its simplicity in analyzing the geometry of kinematic motions compared against the common Denavit-Hartenberg (DH) conventions. We leave the treatment of the DH convention as an exercise for the reader. While the materials presented here may appear abstract, the applications are practical and useful. Therefore in reading the materials presented forthwith, it is recommended to the reader to ask what is it to the kinematic geometry of mechanisms that we have labored upon in chapter 1. The recommended texts for this chapter are

- Murray, R. M., Li, Z., & Sastry, S. S. (1994). A Mathematical Introduction to Robotic Manipulation . Book (Vol. 29), Chapter 2
- A treatise on the theory of screws, Sir R.S. Ball, Chapter 1
- Screw theory for robotics, Jose M. Pardos-Gotors, IROS2018 tutorial; (Email the instructor for a copy).

6.1 Screws

Simply put, *a screw is a line (called an axis) by which a definite linear magnitude (termed the pitch, p) is associated.* A screw may be defined by a 6-vector of *screw coordinates*, $\underline{s} = (S_1, S_2, S_3, S_4, S_5, S_6)$ which has the following interpretations in terms of the Plücker line coordinates of the axis

$$L = S_1, \quad M = S_2, \quad N = S_3, \tag{6.1.1a}$$

$$P = S_4 - pS_1, \quad Q = S_4 - pS_2, \quad R = S_6 - pS_3 \tag{6.1.1b}$$

where L, M, N are proportional to the direction cosines of the line forming the axis while P, Q, R are proportional to the moment of the line about the origin of the reference frame. If we scale the Plucker coordinates by $(L^2 + M^2 + N^2)^{1/2}$, the first three would represent the direction cosines of the line while the last three would represent the moments. We have the moment of the line as the cross product of a vector from the reference frame's origin to any point on the line with the unit vector in the line direction. Thus, we have the pitch of the screw defined as

$$p = \frac{S_1\, S_4 + S_2 S_5 + S_3 S_6}{S_1^2 + S_2^2 + S_3^2} \tag{6.1.2}$$

whose magnitude is

$$\|\ell\|^2 = S_1^2 + S_2^2 + S_3^2 \tag{6.1.3}$$

for a finite pitch, otherwise, it is

$$\|\ell\|^2 = S_4^2 + S_5^2 + S_6^2 \tag{6.1.4}$$

for an infinite pitch. Note that for infinitesimal screws, scalar multiplication and vector addition are valid so that two screws, $\underline{s}_1$ and $\underline{s}_2$ are considered linearly dependent if there exists non-zero scalars, c_1 and c_2 $i.e.$ $c_1\underline{s}_1 + c_2\underline{s}_2 = 0$.

6.1.1 Motion screws: Twist's Pitch, Axis, and Magnitude

The unique line about which a body in space rotates or translates for an infinitesimal motion or velocity of the body is called the *twist axis*. Formally, we say *a body receives a twist about a screw when it is uniformly rotated about the screw, while it is translated uniformly parallel to the screw, through a distance equal to the product of the pitch and the circular measure of the angle of rotation* (Ball, 1908).

Similar to the screw, the *twist* is characterized by a six-vector of coordinates, $\underline{t} = (T_1, T_2, T_3, T_4, T_5, T_6)$ so that the components of the angular velocity of the body are denoted by $\underline{w} = (T_1, T_2, T_3)$ while the components of the linear velocity of a fixed point in the body lying at the origin of the coordinate system is $\underline{v} \equiv (T_4, T_5, T_6)$.

An important thing to note is that the a twist is characterized by the attributes *pitch, magnitude and an axis*. Hence, we have the Plücker coordinates of the *twist axis* given as

Definition 1 (Twist Axis)**.**

$$L = T_1, \quad M = T_2, \quad N = T_3 \tag{6.1.5a}$$

$$P = T_4 - pT_1, \quad Q = T_5 - pT_2, \quad R = T_6 - pT_3. \tag{6.1.5b}$$

Definition 2 (Pitch of a Twist)**.** The pitch of the twist, $\xi = \begin{pmatrix} \omega & v \end{pmatrix}^T$ is

$$p = \begin{cases} \dfrac{T_1T_4 + T_2T_5 + T_3T_6}{T_1^2 + T_2^2 + T_3^2} = \dfrac{\underline{w}^T \cdot \underline{v}}{\underline{w} \cdot \underline{w}} \text{ if } \underline{\omega} \neq 0 \\ \infty \text{ otherwise} \end{cases} . \tag{6.1.6}$$

Definition 3 (Magnitude of a Twist)**.** The magnitude of the twist, $\xi = \begin{pmatrix} \omega & v \end{pmatrix}^T$ is

$$\|\xi\| = \begin{cases} \|\underline{\omega}\| \text{ if } \underline{\omega} \neq 0 \\ \|\underline{v}\| \text{ otherwise} \end{cases} . \tag{6.1.7}$$

In screw theory, it is typical to concern ourselves only with the small displacements of a system's motion. Whenever a body admits an indefinitely small movement of a continuous nature, it is capable of executing that kind of movement denoted by a twist about a screw. When the body receives twists in several succession, the position that is finally attained is called the *resultant twist*.

Homework 21. What is the geometric meaning of (6.1.6) on a twist axis to you. Define *pure rotation* and a *pure translation* in terms of (6.1.6)[1].

[1] Figure out how they correspond to zero pitch and infinite pitch twists.

6.1.2 Dynamics Screws: Twist's Pitch, Axis, and Magnitude

For a set of forces and moments applied to a rigid body, there is a *wrench axis*, a unique line, which has associated with it a pitch, p, and a magnitude. We may consider the set of forces and moments that act on the body to be a single force along the wrench axis and a a moment about exerted about the axis. This force-moment pair is termed a *wrench*, and it is characterized by the 6-vector $\underline{w} = (W_1, W_2, W_3, W_4, W_5, W_6)$. We consider the first three components of $\underline{w}$ to be the net force components, $\underline{f}$, exerted on the body and $\underline{m} = \{W_4, W_5, W_6\}$ to be the components of the net moment resolved at the origin of the reference frame. We define the Plücker coordinates of the *wrench axis* as

$$L = W_1, \quad M = W_2, \quad N = W_3 \tag{6.1.8a}$$

$$P = W_4 - pW_1, \quad Q = W_5 - pW_2, \quad R = W_6 - pW_3 \tag{6.1.8b}$$

with pitch defined as

$$p = \frac{W_1 W_4 + W_2 W_5 + W_3 W_6}{W_1^2 + W_2^2 + W_3^2} = \frac{\underline{f} \cdot \underline{m}}{\underline{f} \cdot \underline{f}}. \tag{6.1.9}$$

Equation (6.1.9) signifies that the pitch of the wrench is the ratio of the magnitude of the applied moment about a point to the magnitude of the applied force along a wrench axis. A *zero pitch wrench* would therefore correspond to pure force while an infinite pitch wrench would be a pure moment. We define the magnitude of the wrench as $\|\underline{f}\| = \sqrt{W_1^2 + W_2^2 + W_3^2}$ when we have a finite pitch. If the pitch in infinite, the magnitude is $\|\underline{m}\| = \sqrt{W_4^2 + W_5^2 + W_6^2}$.

Homework 22. A unit screw, twist or wrench is one where the magnitude of the screw, twist or wrench is 1.

(1). What is the geometric meaning of a unit screw to you? (2) Consult the identified reference materials and explain what a reciprocal screw is in no more than five sentences.

6.1.3 Screw Motions

A *screw motion* is a rotation about the axis, l by an amount $\alpha = \ell$ followed by a translation by an amount $h\alpha$ that is parallel to the axis l. A pure translation occurs when $h = \infty$ so that the screw motion is composed of a *pure translation* along the axis of the screw by a distance ℓ.

6.2 Rodrigues' Formula

The matrix exponential

$$e^{\hat{\omega}t} = I + \hat{\omega}t + \frac{(\hat{\omega}t)^2}{2!} + \frac{(\hat{\omega}t)^3}{3!} + \cdots \tag{6.2.1}$$

is instrumental in defining the rotation about an axis, ω:

$$R(\omega, \theta) = e^{\hat{\omega}\theta}. \tag{6.2.2}$$

Thus, for a matrix $\hat{\omega}$ in the Lie algebra $so(3)$, a unit vector $\|\omega\| = 1$, and a real number $\theta \in \mathbb{R}$, we write the exponential of $\hat{\omega}\theta$ as

$$e^{\hat{\omega}\theta} = I + \theta\hat{\omega} + \frac{\theta^2}{2!}\hat{\omega}^2 + \frac{\theta^3}{3!}\hat{\omega}^3 + \cdots \tag{6.2.3}$$

Homework 23. Given a matrix $\hat{m} \in so(3)$, suppose that the following relation holds,

$$\hat{m}^2 = mm^T - \|m\|^2 I \tag{6.2.4}$$

$$\hat{m}^3 = -\|m\|^2 \hat{m} \tag{6.2.5}$$

with the fact that higher powers of $\hat{m}$ can be recursively found. Therefore, utilizing this lemma with $m = \omega\theta$, $\|\omega\| = 1$, show that

$$e^{\hat{\omega}\theta} = I + \hat{\omega}\sin\theta + \hat{\omega}^2(1 - \cos\theta). \tag{6.2.6}$$

Equation (6.2.6) is *Rodrigues' formula*.

6.3 The Matrix Exponential, The Lie Group and Lie Algebra

For any matrix

$$g = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \in SE(3) \text{ such that } R \in SO(3), d \in \mathbb{R}^3, \text{ and } RR^T = I \tag{6.3.1}$$

there exists a matrix

$$N = \begin{bmatrix} S & x \\ 0 & o \end{bmatrix} \text{ such that } S = -S^T \tag{6.3.2}$$

such that $\exp(N) = g$ since we can write

$$\begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & d \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & d \\ 0 & 1 \end{bmatrix} \exp \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix}. \tag{6.3.3}$$

We see that the exponential map is surjective onto $\mathbb{E}(3)$. $SE(3)$ is Lie Group whose isomorphism is the *lie algebra* $\mathfrak{se}(3)$. Generally, *we define the Lie Group SE(n) on the configuration of a rigid body that consists of the pair* (p_{ij}, R_{ij}) *to be the product space of* $\mathbb{R}^n$ *with SO(n, called the* ***special Euclidean group*** $i.e.$

$$SE(n) = \{(p, R) : p \in \mathbb{R}^n, R \in SO(n)\} = \mathbb{R}^n \times SO(n) \tag{6.3.4}$$

An element of $\mathfrak{se}(3)$[2] is the *twist* earlier introduced, which is the infinitesimal generator of the Euclidean group. Formally, we define $\xi = (\omega, v) \in \mathbb{R}^6$ as the twist coordinates of $\hat{\xi}$. Equation (6.3.3) is basically a re-statement of *Euler's theorem, that is that a rigid body transformation is basically a rotation about a line passing through a preassigned fixed point*. Note that S is the anti-symmetric matrix or skew-symmetric matrix with the following special property

$$S(\overrightarrow{\omega}) = \hat{\omega} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} \tag{6.3.5}$$

[2]We typically write the Lie algebra in lowercase with the math frak style.

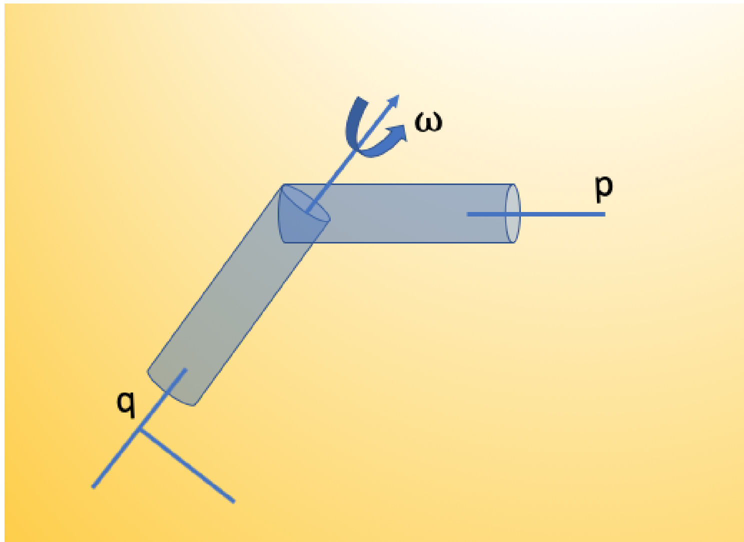

Figure 6.1: Illustration of rotation of interconnecting link of two spherical links.

that is $\omega_x, \omega_y, \omega_z$ are the component of the vector $\overrightarrow{\omega}$ such that

$$s_{ij} = \begin{cases} 0, & \text{if } i = j \\ -s_{ji} & \text{if } i \neq j. \end{cases}$$

It is a common convention in robotics texts to denote the skew symmetric matrix by $\hat{\omega}$ and we will adopt that convention for the rest of these notes. In particular, we define the *homogeneous coordinates* for a point $q \in \mathbb{R}^3$ that is rotating about an axis $\overrightarrow{\omega} \in \mathbb{R}^3$ such that $\|\overrightarrow{\omega}\| = 1$ (see Fig. 6.1) as

$$\hat{\xi} = g^{-1}\dot{g} = \begin{pmatrix} \hat{\omega} & v \\ 0 & 0 \end{pmatrix} \in \mathfrak{se}(3) \tag{6.3.6}$$

where $v = -\omega \times q$. Equation (6.3.6) is the object's velocity in the body frame; essentially the *Lie algebra element* and we can obtain the twist coordinates from it via the so-called *wedge operator*

$$\begin{pmatrix} \hat{\omega} \\ v \end{pmatrix}^{\wedge} = \begin{pmatrix} \hat{\omega} & v \\ 0 & 0 \end{pmatrix}. \tag{6.3.7}$$

If the link of Fig. 6.1 moves at a unit velocity, we can write the velocity at the tip point as

$$\dot{p} = \omega \times (p(t) - q(t)) \tag{6.3.8}$$

where $p(t), q(t)$ are the paths traced out by points p and q respectively during the motion.

We define the exponential for the matrix $A \in SO(n)$, which is a component of the solution to the ordinary differential equation, $\dot{x}(t) = Ax(t)$ where $x(t) \in \mathbb{R}^n$ as

The Matrix Exponential

$$e^{At} = I + At + \frac{A^2t^2}{2!} + \frac{A^3t^3}{3!} + \dots \tag{6.3.9}$$

For the matrix exponential of (6.3.9), we would like to write the equation in a closed-form expression since we are interested in definite solutions for our forward kinematics problems. For suppose that some matrix $D \in \mathbb{R}^{n\times n}$ and $P \in \mathbb{R}^{n\times n}$ are available, then we find that

$$\begin{aligned} e^{At} &= I + (PDP^{-1})t + (PDP^{-1})(PDP^{-1})\frac{t^2}{2!} + \dots \\ &= P\left(IDt + \frac{(Dt)^2}{2!} + \dots\right)P^{-1} && (6.3.10) \\ &= Pe^{Dt}P^{-1}. && (6.3.11) \end{aligned}$$

Definition 4 (Exponential Map Properties)**.** We note the following properties of the matrix exponential:

(1) $d(e^{At}) = Ae^{At} = e^{At}A$.

(2) For $A = PDP^{-1}$ for some diagonal $D \in \mathbb{R}^{n\times n}$ and an invertible $P \in \mathbb{R}^{n\times n}$, $e^{At} = Pe^{Dt}P^{-1}$.

(3) For $AB = BA$, we have $e^Ae^B = e^{A+B}$.

(4) $(e^A)^{-1} = e^{-A}$

For the mapping from the twist map in the Lie algebra to the Lie group notation, we have the following:

Exponential map from $\mathfrak{se}(3)$ to $SE(3)$

$$e^{\hat{\xi}\theta} = \begin{pmatrix} e^{\hat{\omega}\theta} & (I - e^{\hat{\omega}\theta})(w \times v) + \omega\omega^T v\theta \\ 0 & I \end{pmatrix} \in SE(3) \tag{6.3.12}$$

6.3.1 The Lie Group Notations

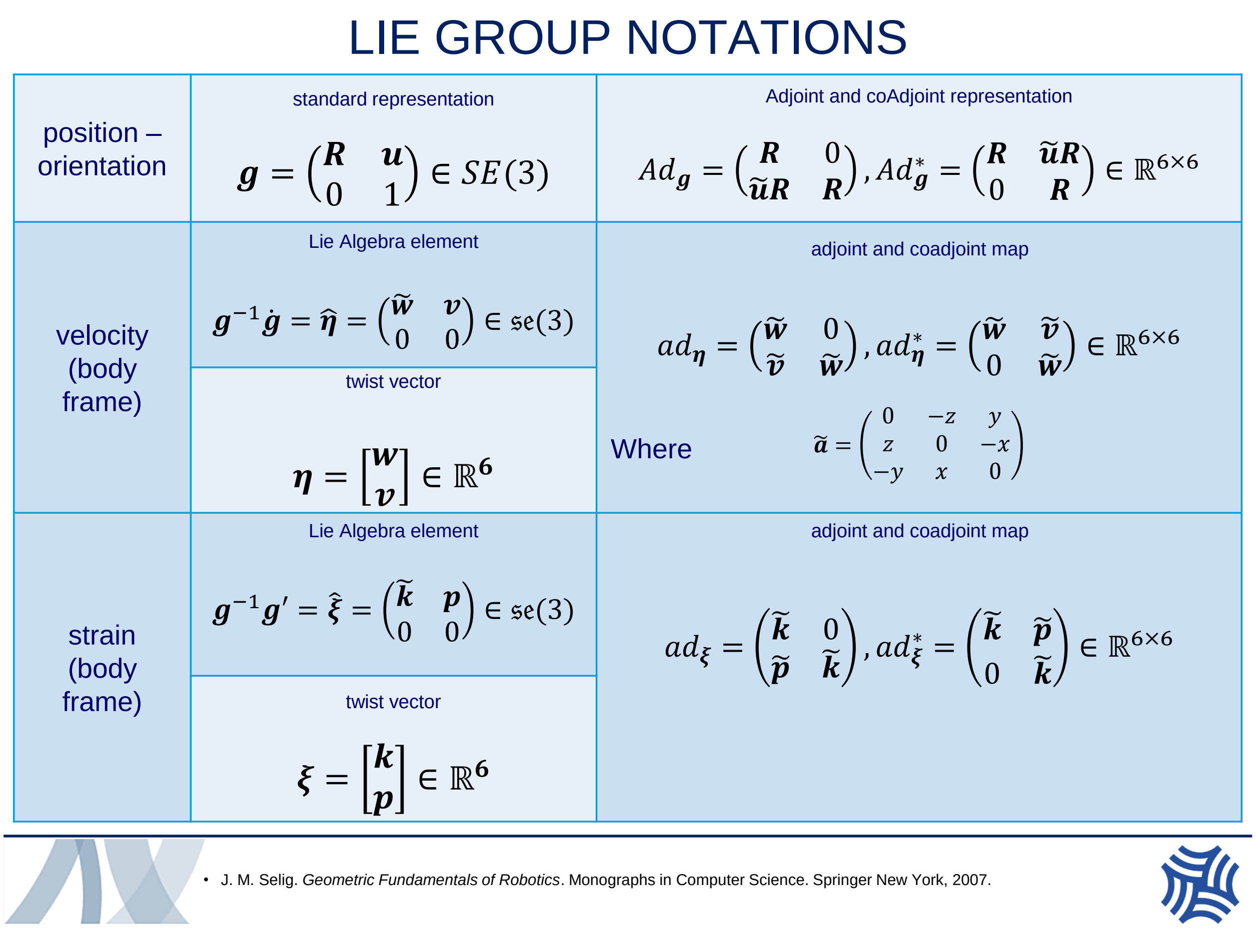

Figure 6.2: Lie Group and their Notations. Reprinted with permission from Federico Renda. IROS 2018, Screw Theory Tutorial.

6.3.2 Exponential Map and Kinematic Chains

Now consider a robot manipulator of the form shown in Fig. 5.4. The reader may imagine that there are right-handed triads of orthogonal vectors at the tip of each link of the chain so that the Euclidean transformation that describes the position and orientation of the $(i+1)'th$ link with the $i'th$ link is

$$\begin{pmatrix} R_i & d_i \\ 0 & 1 \end{pmatrix} \exp \begin{pmatrix} S_i & 0 \\ 0 & 0 \end{pmatrix} \theta_i = g_i \exp\left(\hat{\omega}_i \theta_i\right) \tag{6.3.13}$$

and we may imagine the triad that is fixed at the end-effector to be related to the triad at the base of the robot by the following relation

$$H(\theta_1, \theta_2, \cdots, \theta_n) = R_1\, e^{\theta_1 \hat{\omega}_1}\, R_2\, e^{\theta_2 \hat{\omega}_2}, \cdots, R_n\, e^{\theta_n \hat{\omega}_n} \tag{6.3.14}$$

and since $Pexp(M)P^{-1} = exp(PMP^{-1})$, we can write the *forward kinematic map*, $g_{st} : Q \to SE(3)$ as

$$g_{st}(\theta) = e^{\theta_1 \hat{\omega}_1}\; e^{\theta_2 \hat{\omega}_2} \cdots,\; e^{\theta_n \hat{\omega}_n}\, g_{st}(0) \tag{6.3.15}$$

using the identity repeatedly.

6.4 Rigid Body Transformations

A rigid body motion is one that preserves the distance between points. In classical mechanics, we are concerned about the material description of a rigid body whereby we consider all the particles that make up the rigid body as a whole rather than treat them as a continuum as it is typical in continuum mechanics. Therefore by this logic, *a rigid body is a collection of particles by which the distance between any two particles remain fixed, irrespective of the motions of the body or forces exerted on that body (Murray, 1994).*

Suppose we have two points a and b on a rigid body, we must have

$$\|a(t_f) - b(t_f)\| = \|a(t_i) - b(t_i)\| \tag{6.4.1}$$

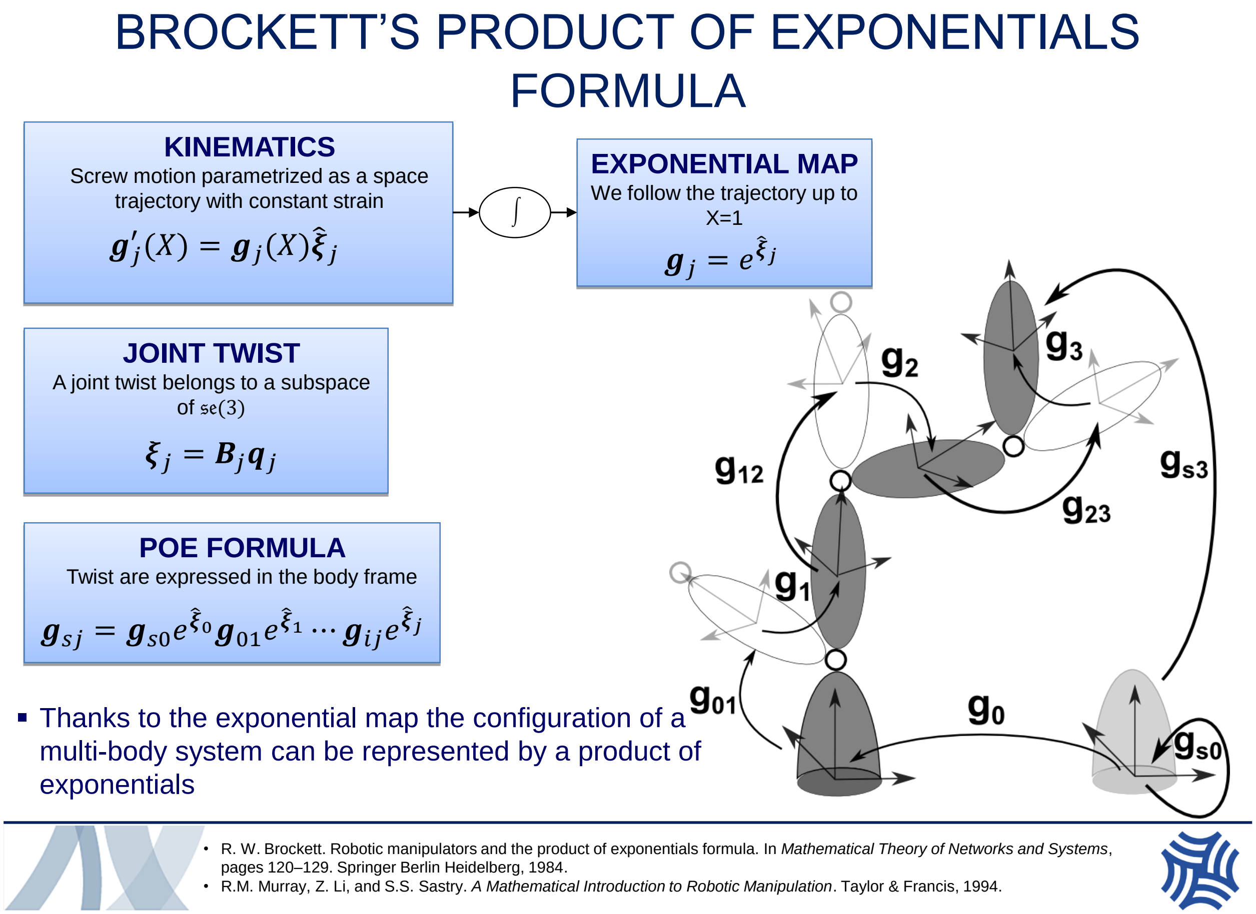

Figure 6.3: An illustration of the exponential map for a multi-body rigid system. Reprinted with permission from Federico Renda. IROS 2018, Screw Theory Tutorial.

where t_i and t_f are two instants of time that the two points $i.e.$ a and b are observed on the rigid body. We thus see that distance is preserved irrespective of observer placement in rigid body transformations. For continuum-based systems such as soft robots, this is not so and we often need to come up with clever mechanisms of characterizing the motion of its particles.

Definition: A mapping $g : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is a *rigid body transformation* if it satisfies the following properties:

- Length is preserved $i.e.$ $\|g(a) - g(b)\| = \|a - b\|$ for all point $a, b \in \mathbb{R}^3$.

- The cross product for all vectors is preserved $i.e.$ $g(p \times q) = g(p) \times g(q)$ where $p, q \in \mathbb{R}^3$.

This means that the inner product is preserved so that we have,

$$p^T q = g(p)^T \times g(q). \tag{6.4.2}$$

It follows that orthogonal vectors are transformed to orthogonal vectors and since cross product is naturally preserved, rigid body transformations map orthonormal coordinate frames to orthonormal coordinate frames. Note that it is possible to have rotation of particles despite the two strong forms presented above. To track the location of a rigid body in space, it therefore follows that we need to keep track of the motion of any one point as well as the rotation of the rigid body about this point. Therefore, the *configuration* of the rigid body is found by attaching a Cartesian coordinate frame to a point on the rigid body and keeping track of the motion of the body coordinate frame with respect to a fixed frame. To ease representation, we typically require that all coordinate frames be *right-handed*: given three Cartesian orthonormal vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^3$ which characterize the motion of a coordinate frame, they must satisfy $\boldsymbol{z} = \boldsymbol{x} \times \boldsymbol{y}$.

Going by the definitions of twist in the foregoing, it follows that a twist is to a rigid body what a vector is to a point. They both express the relation needed to transfer an object from one given position to another. When a body twists at an instant, the screw about which it twists is referred to as the *instantaneous screw*.

6.4.1 Translation in $\mathbb{R}^3$

For the two coordinate frames shown in Fig. 6.4, say we choose the $o_0x_0y_0$ frame as the reference and the $o_1x_1y_1$ as the moving coordinate frame, the way we would characterize the translation motion of the the point q would be to represent its translation from the reference frame by the Cartesian displacement along x and y so that we have

$$q^0 = \begin{pmatrix} q_x^0 \\ q_y^0 \end{pmatrix}, \quad q^1 = \begin{pmatrix} q_x^1 \\ q_y^1 \end{pmatrix} \tag{6.4.3}$$

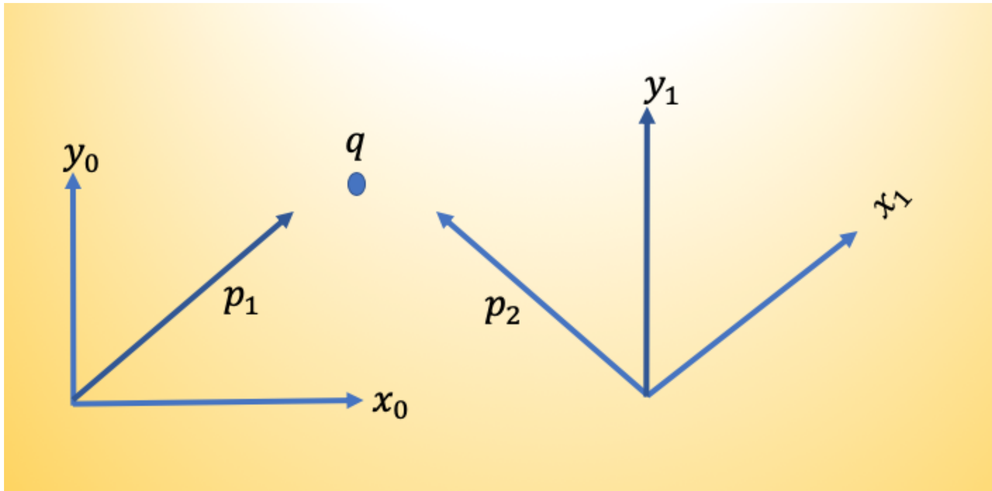

Figure 6.4: A point q in space with respect to two coordinate frames.

where the superscript denotes the reference frame and the subscript denotes the coordinate of the point q along an axis in either Cartesian frame. The origin of the two frames are both points in space; therefore, the we can assign the coordinates that denote the position of the origin of one coordinate frame with respect to another as

$$o_1^0 = \begin{pmatrix} o_x^0 \\ o_y^0 \end{pmatrix}, \quad o_0^1 = \begin{pmatrix} o_x^1 \\ o_y^1 \end{pmatrix}. \tag{6.4.4}$$

For vectors such as twists or wrenches, we use a similar notation as those used for points. Thus if p_1, p_2 are two vectors that are invariant with respect to the choice of coordinate frames, we would have

$$p_1^0 = \begin{pmatrix} o_x^0 \\ o_y^0 \end{pmatrix}^T, \quad p_1^1 = R(-\theta)q^0, \quad p_2^0 = R(\theta)q^0, \quad p_2^1 = \begin{pmatrix} o_x^1 \\ o_y^1 \end{pmatrix}^T \tag{6.4.5}$$

where θ is the angle that coordinate frame 1 makes with respect to coordinate frame 0. In robotics, the standard way to apply a rotation is counterclockwise. This is the reason we negate the angle of rotation when finding the vector p_1 in frame 1. We will introduce the definition of the rotation matrix R shortly. Performing frame transformations is a fundamental step to getting a robot work as envisioned. We must ensure that all coordinate vectors are defined with respect to the same coordinate frame. We say two vectors are "equal" when they have the same magnitude and direction. Therefore, for vectors that are not constrained to be located at the same point in space, we require that they defined with respect to frames whose coordinates are parallel, given that absolute locations are not consequential, but the magnitude and direction of the vector.

6.4.2 Rotations in $\mathbb{R}^3$

We would like to establish a convention that all coordinate frames will be right-handed. Our goal is to establish the orientation of an object by specifying the local coordinate on the body; we then describe the body's *relative orientation* between a coordinate frame attached to the body and a fixed or an inertial coordinate frame. Suppose that we have two frames I and J, where I is the inertial frame, while J is the body frame as shown in Fig. 6.5. Let $\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij}, \boldsymbol{z}_{ij} \in \mathbb{R}^3$ be the coordinates of the principal axes of J relative to I so that we have the following matrix as a result of composing the respective coordinate vectors

$$R_{ij} = \begin{bmatrix} \boldsymbol{x}_{ij} & \boldsymbol{y}_{ij} & \boldsymbol{z}_{ij} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{6.4.6}$$

The resulting matrix in (6.4.6) is the *rotation matrix*. This matrix can be rewritten by noting that the components of a vector are the projections of that vector onto the unit directions of its reference frame. Thus, the components if R_{ij} in (6.4.6) can be written as the dot product of a pair of unit vectors:

$$R_{ij} = \begin{bmatrix} \boldsymbol{x}_j \cdot \boldsymbol{x}_i & \boldsymbol{y}_j \cdot \boldsymbol{x}_i & \boldsymbol{z}_j \cdot \boldsymbol{x}_i \\ \boldsymbol{x}_j \cdot \boldsymbol{y}_i & \boldsymbol{y}_j \cdot \boldsymbol{y}_i & \boldsymbol{z}_j \cdot \boldsymbol{y}_i \\ \boldsymbol{x}_j \cdot \boldsymbol{z}_i & \boldsymbol{y}_j \cdot \boldsymbol{z}_i & \boldsymbol{z}_j \cdot \boldsymbol{z}_i \end{bmatrix}. \tag{6.4.7}$$

The components of the rotation matrix (6.4.6) are sometimes called direction cosines since the dot product of two unit vectors give the cosine of the angle between them as shown in (6.4.7). If we

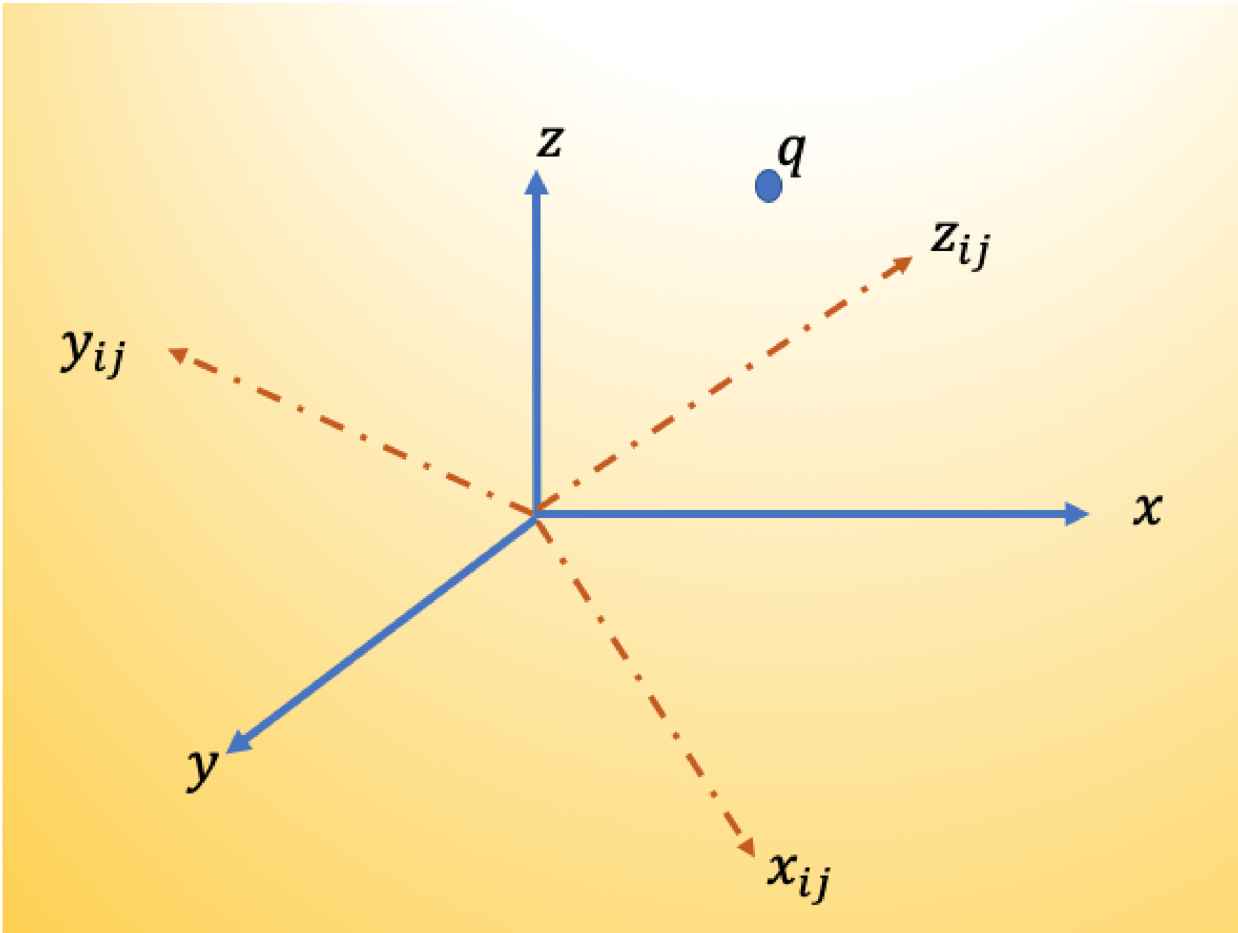

Figure 6.5: An illustration of the relative orientation of a rigid object q between an inertial frame I and a body frame J.

examine the rows of (6.4.7), we see that the rows of R_{ij} are the unit vectors coordinates of I in the frame J so that we have

$$R_{ij} = R_{ji}^T \tag{6.4.8}$$

which is to say that the inverse of the rotation matrix is equal to its transpose. Noting that $\|\boldsymbol{r}\|^2 = x_i^2 + y_j^2 + z_i^2 = 1$, $r_i \cdot r_j = 0$ when $i \neq j$, and $r_i \cdot r_i = 0$, this can be thus verified as

$$R_{ij}^T \cdot R_{ji} = \begin{bmatrix} \boldsymbol{x}_j \cdot \boldsymbol{x}_i & \boldsymbol{x}_j \cdot \boldsymbol{y}_i & \boldsymbol{x}_j \cdot \boldsymbol{z}_i \\ \boldsymbol{y}_j \cdot \boldsymbol{x}_i & \boldsymbol{y}_j \cdot \boldsymbol{y}_i & \boldsymbol{y}_j \cdot \boldsymbol{z}_i \\ \boldsymbol{z}_j \cdot \boldsymbol{x}_i & \boldsymbol{z}_j \cdot \boldsymbol{y}_i & \boldsymbol{z}_j \cdot \boldsymbol{z}_i \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x}_j \cdot \boldsymbol{x}_i & \boldsymbol{y}_j \cdot \boldsymbol{x}_i & \boldsymbol{z}_j \cdot \boldsymbol{x}_i \\ \boldsymbol{x}_j \cdot \boldsymbol{y}_i & \boldsymbol{y}_j \cdot \boldsymbol{y}_i & \boldsymbol{z}_j \cdot \boldsymbol{y}_i \\ \boldsymbol{x}_j \cdot \boldsymbol{z}_i & \boldsymbol{y}_j \cdot \boldsymbol{z}_i & \boldsymbol{z}_j \cdot \boldsymbol{z}_i \end{bmatrix} \tag{6.4.9}$$

$$= \begin{bmatrix} \boldsymbol{x}_j \cdot \boldsymbol{x}_j \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{x}_j \cdot \boldsymbol{y}_j \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{x}_j \cdot \boldsymbol{z}_j \cdot \|\boldsymbol{r}\|^2 \\ \boldsymbol{y}_j \cdot \boldsymbol{x}_j \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{y}_j \cdot \boldsymbol{y}_j \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{y}_j \cdot \boldsymbol{z}_j \cdot \|\boldsymbol{r}\|^2 \\ \boldsymbol{z}_j \cdot \boldsymbol{x}_j \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{z}_j \cdot \boldsymbol{y}_i \cdot \|\boldsymbol{r}\|^2 & \boldsymbol{z}_j \cdot \boldsymbol{z}_j \cdot \|\boldsymbol{r}\|^2 \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6.4.10}$$

$$= \boldsymbol{I}_3 \tag{6.4.11}$$

where $\boldsymbol{I}_3$ is the 3×3 identity matrix.

Homework 24. Verify that $R_{ij} = R_{ji}^{-1} \equiv R_{ji}^T$. Furthermore, verify that the determinant of the rotation matrix is ± 1 $i.e.$ $detR = \pm 1$.

The determinant of the R matrix is written as

$$\det R = r_1^T \left(r_2 \times r_3 \right). \tag{6.4.12}$$

For right-handed coordinate frames, we have that $r_2 \times r_3 = r_1$ so that $r_1^T r_1 = 1$ for a coordinate frame that aligns with the right-hand orthonormal frame representation. This special property that a 3×3 matrix satisfies $r_2 \times r_3 = r_1$ and that $\det R = r_1^T r_1 = 1$ is called the special orthogonal property, denoted *SO(3)*. Special orthogonal means $\det R = +1$. The set of all SO matrices in $\mathbb{R}^{n \times n}$ is defined by

$$SO(n) = \{R \in \mathbb{R}^{n \times n} : RR^T = \boldsymbol{I}, \det R = +1\}. \tag{6.4.13}$$

6.4.3 Rotation Matrices as Transformations

Suppose we are tasked with transforming a point q from one coordinate frame J to a frame I based on Fig. 6.5. Suppose that $q_j = (x_j, y_j, z_j)$ are the coordinates of q with respect to the frame J. We may reason that $x_j, y_j, z_j \in \mathbb{R}^3$ are the projections of q onto the coordinate axes of B, which in turn, have coordinates $\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij}, \boldsymbol{z}_{ij} \in \mathbb{R}^3$ with respect to coordinate frame I; then it follows that the coordinates of q relative to frame I is

$$q_i = \boldsymbol{x}_{ij} x_j + \boldsymbol{y}_{ij} y_j + \boldsymbol{z}_{ij} z_j. \tag{6.4.14}$$

which in vectorized form is

$$q_i = \begin{pmatrix} \boldsymbol{x}_{ij} & \boldsymbol{y}_{ij} & \boldsymbol{z}_{ij} \end{pmatrix} \begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix} \qquad = R_{ij} q_j \tag{6.4.15}$$

where the last part of the above equation follows from (6.4.7).

Just as a rotation matrix can act on points to transform them in the world, so can rotation matrices act on vectors. Say we have another point p_j on the frame J in Fig. 6.5, then the vector that connects a point q_j in the frame J to p_j is $v_j = q_j - p_j$ so that the action of the rotation matrix on v_j is

$$R_{ij}(v_j) := R_{ij} q_j - R_{ij} p_j = q_i - p_i = v_i. \tag{6.4.16}$$

6.4.4 Planar Rotations

We now revisit the relative orientation between two coordinate frames as shown in Fig. 6.5. Suppose now that the angle of rotation between the two coordinate frames is θ as shown in Fig. 6.6, it follows that the composition of the rotation allows us to write

$$R_1^0 = \begin{pmatrix} x_1^0 \mid & y_1^0 \end{pmatrix} \tag{6.4.17}$$

whereupon x_1^0 and y_1^0 have the usual meanings as before and are expressed as

$$x_1^0 = \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \text{ and } y_1^0 = \begin{pmatrix} -\sin\alpha \\ \cos\alpha \end{pmatrix} \tag{6.4.18}$$

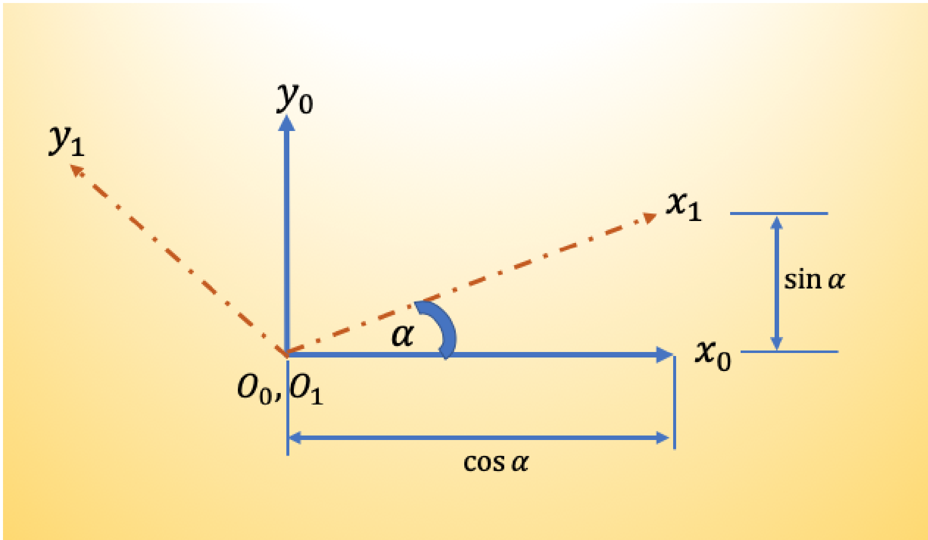

Figure 6.6: Illustration of rotation between two frames in a plane.

so that composing the rotation matrix, we have

$$R = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \tag{6.4.19}$$

While you might find the above approach rather daunting, an easier geometric way to visualize the transformation of matrices is to recall that the columns of the rotation matrix are the direction cosines of the coordinate axes of $o_1x_1y_1$ relative to the coordinates of $o_0x_0y_0$ c.f. (6.4.7). For a planar rotation, we could extract the first 2×2 block of (6.4.7) so that

$$R_1^0 = \begin{bmatrix} \boldsymbol{x}_0 \cdot \boldsymbol{x}_1 & \boldsymbol{y}_1 \cdot \boldsymbol{x}_0 \\ \boldsymbol{x}_0 \cdot \boldsymbol{y}_1 & \boldsymbol{y}_1 \cdot \boldsymbol{y}_0 \end{bmatrix} \tag{6.4.20}$$

And since the dot product of two vectors is basically the cosine of the angles between them, we have

$$R_1^0 = \begin{bmatrix} cos\alpha & -cos(\pi/2 - \alpha) \\ cos(\pi/2 - \alpha) & cos\alpha \end{bmatrix} = \begin{bmatrix} cos\alpha & -\sin\alpha \\ \sin\alpha & cos\alpha \end{bmatrix} \tag{6.4.21}$$

Note the way the negative signs have entered the matrix due to the counterclockwise direction of rotation that we have chosen so as to preserve the positiveness of the determinant of R. In particular, the projection of y_1 on x_0 is negative because of our right-handed frame.

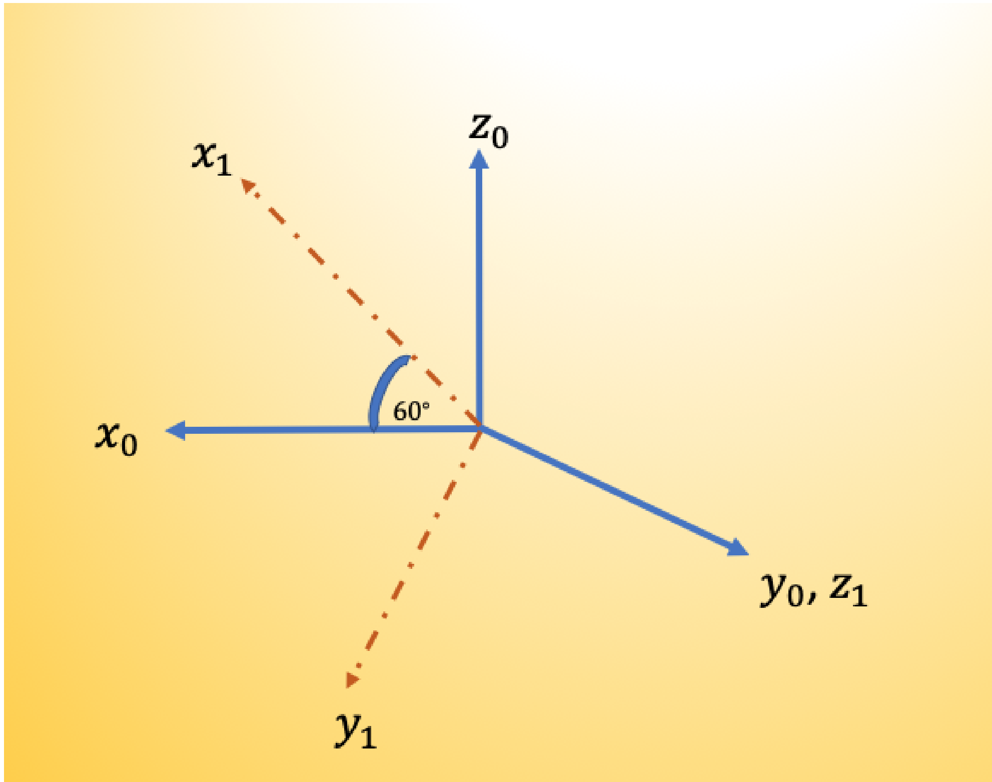

Figure 6.7: Relative orientation between two frames.

Homework 25. Compose the rotation matrix in three dimensions where all axes of the inertial frame are rotated by an angle β around each of the x_0, y_0 and z_0 axes respectively using the foregoing logic. In addition, for each transformation, verify that (1) $R_{e,\beta} = I$ where e is the axes about which we are rotating and β is the angle of rotation, (2) the composition of rotations about the angles β and α in a successive manner implies that $R_{z,\beta}, R_{z,\alpha} = R_{z,\beta+\alpha}$, and (3) $(R_{z,\beta})^{-1} = R_{z,-\beta}$. Bonus points will be awarded for cool 3D visualizations.

Homework 26. For the two frames shown in Fig. 6.7, determine the rotation matrix between them. In addition, explain the difference between rotating about a *current frame* and rotating about a *fixed frame*[3]. In particular, when is it necessary to carry out a *pre-multiplication* and when is it necessary to carry out a *post-multiplication* when transforming points or vectors about coordinate frames?

[3]See sections 2.4.1 and 2.4.2 of Spong's book.

Order of rotations

A rotation about a fixed axis necessarily means a **pre-multiplication** while a rotation about a current axis necessitates a *post-multiplication*.

6.4.5 Composition of Rotations

Rotations matrices have this desirable property that they can be composed together to form transform a point between successive frames. Take for example a frame K whose orientation relative to frame J is R_{jk}, and frame J whose orientation relative to frame I is R_{ij}, we can write out the orientation of frame K relative to I as

$$R_{ik} = R_{ij} R_{jk}. \tag{6.4.22}$$

The rotation described above would be equivalent to rotating frame K relative to frame I according to R_{ij}, then aligning frame J to K, we rotate K relative to I according to R_{jk}. The resulting frame K has orientation with respect to I given by $R_{ij}R_{jk}$. This frame relative to which rotation occurs is termed the *current frame*.

Homework 27. For the robot manipulator we are using in this class, suppose that you have the following point in the base frame of the robot, $q_o = [-2, 3, 1]$. Furthermore, suppose that the joint angles for all six joints are respectively $\{-90, 60, 30, 45, 90, 125\}$, transform the point q_0 in the base frame to a coordinate frame on the sixth joint.

Properties of Rigid Body Rotation Matrices

Rigid body rotations preserve

- distance: $\|Rq - Rp\| = \|q - p\|$ for all $q, p \in R^3$
- orientations: $R(i \times j) = Ri \times Rj$ for all $i, j \in \mathbb{R}^3$.

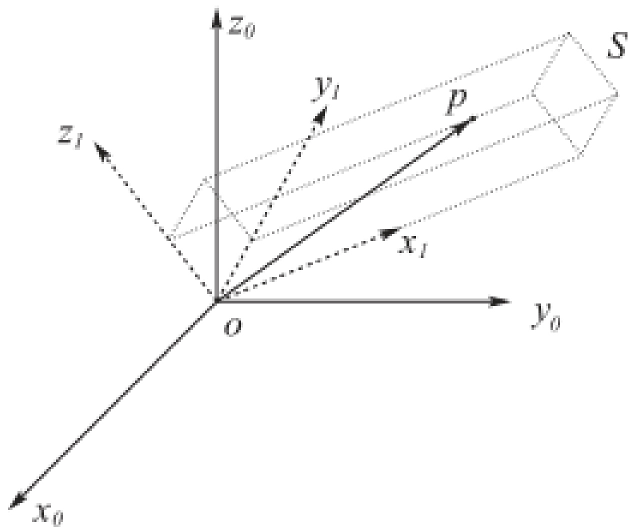

Figure 6.8: Coordinate frames on a body. Reprinted from (Spong et al., 2006)

.

What follows is adapted from Figure 2.5 from Spong and Vidyasagar's book. It is meant to illustrate the rotation applied to a point from one frame to another.

For consider Fig. 6.8 whereupon the rigid body S has a coordinate frame $o_1 x_1 y_1$ attached. For the coordinates of the point p with respect to frame $o_1 x_1 y_1$ or p^1, we are tasked with finding the coordinates of p relative to a fixed reference frame $o_0 x_0 y_0$. We note that $p^1 = [u, v, w]^T$ satisfies

$$p = u x_1 + v y_1 + w z_1. \tag{6.4.23}$$

Matter-of-factly, the coordinates of p^0 can be obtained based on the projection trick we introduced earlier by projecting p onto the coordinate axes of $o_0 x_0 y_0 z_0$ so that we have

$$p^0 = \begin{pmatrix} p \cdot x_0 \\ p \cdot y_0 \\ p \cdot z_0 \end{pmatrix} \tag{6.4.24}$$

so that substituting (6.4.23) into (6.4.24), we have

$$p^0 = \begin{pmatrix} (ux_1 + vy_1 + wz_1) \cdot x_0 \\ (ux_1 + vy_1 + wz_1) \cdot y_0 \\ (ux_1 + vy_1 + wz_1) \cdot z_0 \end{pmatrix} = \begin{pmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 & z_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 & z_1 \cdot y_0 \\ x_1 \cdot z_0 & z_1 \cdot z_0 & z_1 \cdot z_0 \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{6.4.25}$$

which upon inspection turns out to be a multiplication of the rotation matrix that transforms point p^1 to point p^0 $i.e.$

$$p^0 = R_1^0 p^1, \tag{6.4.26}$$

implying that the rotation matrix not only serves to represent the relative orientation of coordinate frames with respect to one another but to transform coordinates of a point from one frame to another.

Finally, we define a similarity transformation *as the matrix representation of a general linear transformation that is transformed from one frame to another*. So if I is the matrix representation of a linear transformation in a frame $o_0x_0y_0z_0$ and J is the equivalent transformation in $o_1x_1y_1z_1$ then I and J are related by

$$J = (R_1^0)^{-1} A R_1^0 \tag{6.4.27}$$

where R_1^0 is the coordinate transformation between frames $o_1x_1y_1z_1$ and $o_0x_0y_0z_0$.

Lab Exercise: Carry out a similarity transformation in the elbow frame ($o_1x_1y_1z_1$) of the manipulator with respect to the shoulder frame ($o_0x_0y_0z_0$) when the two frames are related by the rotation

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{6.4.28}$$

Consider the composition of rotations of Fig. 6.9 where we first rotate by an angle θ about the x axis and then rotate about an angle ψ about the z axis. The rotation matrix can be composed as

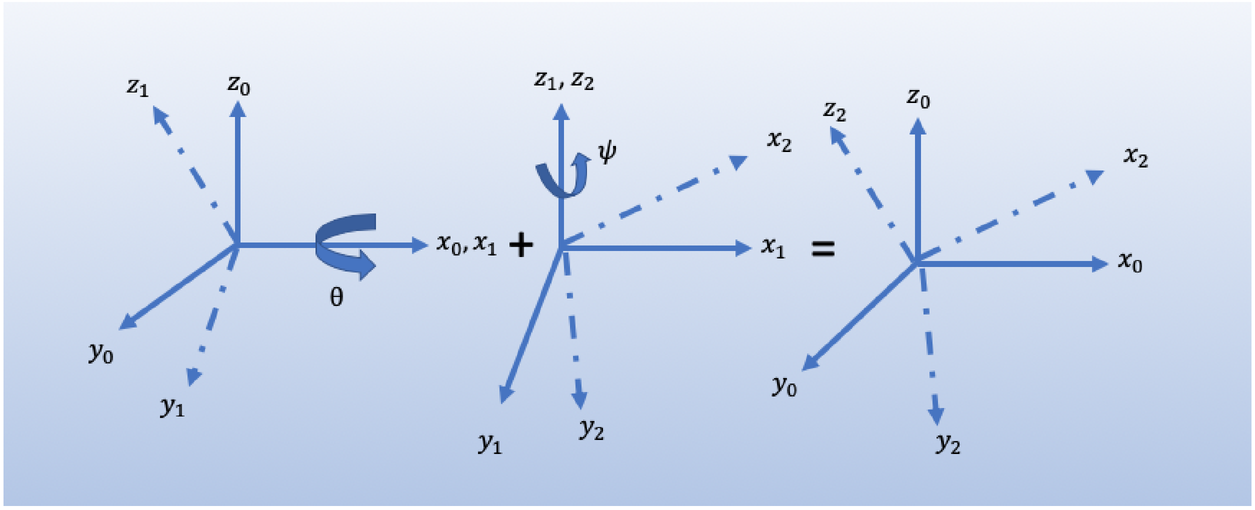

Figure 6.9: Illustration of composition of rotations about a **current axis**.

$$R = R_{x,\theta} R_{z,\psi} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\theta & -s_\theta \\ 0 & s_\theta & c_\theta \end{pmatrix} \cdot \begin{pmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} c_\psi & -s_\psi & 0 \\ 0 & c_\theta c_\psi & -s_\theta \\ s_\theta s_\psi & s_\theta c_\psi & c_\theta \end{pmatrix} \tag{6.4.29}$$

Notice how the order of multiplication is carried out, owing to the axis about which we are making the transformation.

Homework 28. Carry out the transformation above in reverse order. What do you notice?

6.5 Parameterization of Rotations $\in SO(3)$

6.5.1 Axis-Angle Parameterizations

The exponential coordinates are the *canonical* coordinates of the rotation group. A rigid body has at most three rotational degrees of freedom so that we need at most three variables to denote its orientation in the world. For the matrix

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{6.5.1}$$

so that equating the foregoing with the exponential map for a rotation about θ around an axis ω as yields

$$e^{\hat{\omega}\theta} = I + \hat{\omega}\sin\theta + \hat{\omega}^2(1-\cos\theta) \tag{6.5.2}$$

$$= \begin{bmatrix} \omega_y^2 v_\theta + c_\theta & \omega_x\omega_y v_\theta - \omega_z s_\theta & \omega_x\omega_z v_\theta + \omega_y\, s_\theta \\ \omega_x\omega_y v_\theta + \omega_z s_\theta & \omega_y^2 v_\theta + c_\theta & \omega_y\omega_z v_\theta - \omega_x s_\theta \\ \omega_x\omega_z v_\theta - \omega_y s_\theta & \omega_y\omega_z v_\theta + \omega_x s_\theta & \omega_z^2 v_\theta + c_\theta \end{bmatrix}, \tag{6.5.3}$$

where we have used $v_\theta = 1 - \cos\theta$. Therefore, we have

$$trace(R) = r_{11} + r_{22} + r_{33} = 1 + 2c_\theta \tag{6.5.4}$$

Inspecting the matrix of (6.5.3), we have the angle of rotation in terms of the three components of the rotation matrix as

$$\theta = \cos^{-1}\left(\frac{trace(R) - 1}{2}\right) \tag{6.5.5}$$

where θ is defined under the constraint, $-2\pi n \leq \theta \leq 2\pi n$ owing to the special property of R.[4].

Homework 29. Verify by equating the off-diagonal terms that

$$\omega = \frac{1}{2\sin\theta}\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix} \tag{6.5.6}$$

suppose $\theta \neq 0$. Equation (6.5.5) together with (6.5.6) are what we call the *axis-angle representation*.

6.5.2 Euler Angles

The *Euler Angles* are particularly useful for specifying the orientation of a a coordinate frame J with respect to I. To do this, we start as follows (see Fig. 6.10 for reference):

[4]Since R preserves lengths and $\det R = +1$, the eigenvalues of R have a unit magnitude and occur in complex conjugate pairs. This implies $-1 \leq trace(R) \leq 3$

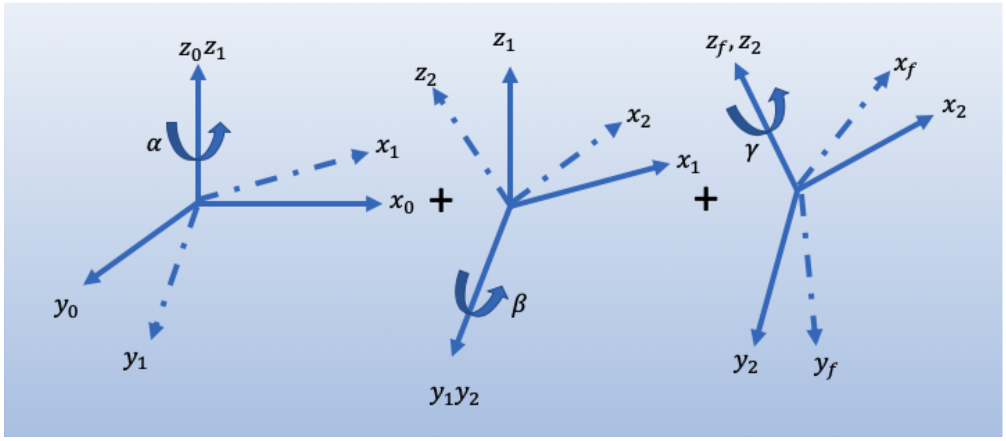

Figure 6.10: An illustration of the ZYZ rotation angles

- we orientate frame J with frame I so that the two frames are coincident by rotating by the z axis about an angle α;

- we then rotate about the *new* y-axis of J by an angle β;

- lastly, we rotate about the new z-axis of rge frame J by an angle γ

so that altogether, we have a rotation $R_{ij}(\alpha, \beta, \gamma)$ given as

$$
\begin{aligned}
R_{ij}(\alpha, \beta, \gamma) &= R_z(\alpha) R_y(\beta) R_z(\gamma) \qquad (6.5.7)\\
&= \begin{bmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \\
&= \begin{bmatrix} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\alpha c_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix} \qquad (6.5.8)
\end{aligned}
$$

where as before the abbreviations c_α, s_α are abbreviations for $\cos\alpha$ and $\sin\alpha$. Euler angles arise when we need to recover the α, β, γ angles from (6.5.8). Suppose that r_{13} and r_{23} are both $\neq 0$, we find that $c_\beta \neq \pm 1$ and thus with the knowledge that $\sin\beta > 0$ given (6.5.9a), we find that ,

$$\beta = \arctan 2(r_{33}, \sqrt{1 - r_{33}^2}) \tag{6.5.9a}$$

$$\alpha = \arctan 2(r_{23}/\sin\beta, r_{13}/\sin\beta) \tag{6.5.9b}$$

$$\gamma = \arctan 2(r_{32}/\sin\beta, -r_{31}/\sin\beta) \tag{6.5.9c}$$

where $\arctan 2(y, x)$ computes $\tan^{-1}(y/x)$, determining the quadrant of the angle based on the sign of x and y. When $\sin\beta < 0$, we find that

$$\beta = \arctan 2(r_{33}, -\sqrt{1 - r_{33}^2}) \tag{6.5.10a}$$

$$\alpha = \arctan 2(-r_{23}/\sin\beta, -r_{13}/\sin\beta) \tag{6.5.10b}$$

$$\gamma = \arctan 2(-r_{32}/\sin\beta, r_{31}/\sin\beta) \tag{6.5.10c}$$

implying that *the Euler angles are not unique* owing to the sign of the angle about which the y axis rotates.

Homework 30. What happens when $r_{13} = r_{23} = 0$? Can you write out the rotation matrix as well as the euler angles for this situation?

In the scenario that results from homework viii, there will be infinitely many solution owing to the fact that only $\alpha + \beta$ only can be determined. This infinite solutions occur wehn $R = I$ and an example scenario of when this occurs is when $(\alpha, 0, -\alpha)$.

Using the ZYZ angles are not the only way of parameterizing the rotation matrix. We could permute the order of rotation or rotate successively about different axis. Examples include ZYX axes rotations (or Fick angles) and the YZX axes parameterization or Helmholtz angles. In general robotics speak, note that the *ZYX angles are otherwise referred to as the yaw, pitch and roll angles, wherein the rotation matrix R_{ij} is defined by rotating about the $x-$axis in the body frame (roll), then the $y-$axis in the body frame (pitch), and finally the $z-$axis in the body frame (yaw).* An advantage of the Fick angle and Helmholtz angle parameterization is that they avoid singularity at $R = I$ though they do contain singularities at other configurations.

6.5.3 Quaternions

Rather than use rotation matrices to represent orientations, a more effective approach of representing orientations are *quaternions*. Intead of locally parameterizing the *SO(3)* Lie group, quaternions, unlike rotation matrices, globally parameterize the *SO(3)* Lie Group. Formally, we represent a quaternion as follows:

$$Q = q_0 + q_x\boldsymbol{i} + q_y\boldsymbol{j} + q_z\boldsymbol{k}, \quad q_{i\{i=0,\cdots,3\}} \in \mathbb{R} \tag{6.5.11}$$

where q_0 is the scalar component of Q and $\boldsymbol{q} = (q_x, q_y, q_z)$ is the vector component.

The *unit quaternions* are the subset of all $Q \in \mathbb{Q}$ such that $\|Q\| = 1$; for a rotation matrix $R = \exp(\hat{\omega}\theta)$, we have the unit quaternion as

$$Q = (\cos(\theta/2), \omega \sin(\theta/2)), \tag{6.5.12}$$

where $\omega \in \mathbb{R}^3$ is the axis of orientation and $\theta \in \mathbb{R}$ is the angle of rotation.

Summary of Parameterizations

Rotation matrices can be parameterized in one of many ways depending on our use case. The common examples of parameterizations are

(1) Axis-Angle representation;

(2) Euler angles (ZYZ) representation;

(3) Fick angles ($i.e.$ ZYX or yaw, pitch and roll) representation;

(4) Helmholtz angles (or YZX) angles representation; and

(5) Quaternions.

6.6 Homogeneous Coordinates

For a point $q \in \mathbb{R}^3$, we represent the *homogeneous coordinates* of point q as

$$\bar{q} = \begin{pmatrix} q_x & q_y & q_z & 1 \end{pmatrix}^T \tag{6.6.1}$$

whose origin has the following coordinates

$$\bar{O} = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}^T. \tag{6.6.2}$$

For vectors, it would suffice for us to write

$$\bar{v} = \begin{pmatrix} v_x & v_y & v_z & 0 \end{pmatrix}^T \tag{6.6.3}$$

where the last element is zero because a vector is the difference of two points.

Vectors and Points

- The sum or difference of two vectors results in a vector
- The difference between two points is a vector
- The sum of two points do not exist

We now define the homogeneous transformation of a point q_j in frame J with respect to a coordinate frame I with p_{ij} being the distance between $q_i,$ and q_j, and R_{ij} being the rotation matrix that transforms points in frame J to points in frame I. We write

$$q_i = p_{ij} + R_{ij} q_j \tag{6.6.4}$$

or more appropriately

$$\begin{pmatrix} q_i \\ 1 \end{pmatrix} = \begin{pmatrix} R_{ij} & p_{ij} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} q_j \\ 1 \end{pmatrix} \tag{6.6.5}$$

which implies that $\bar{q} = \bar{g}_{ij}\bar{q}_j$. We say $\bar{g}_{ij}$ is the *homogeneous representation of* $g_{ij} \in SE(3)$.

Similar to rotation matrices, rigid body transformation matrices can be composed to form new rigid body transformations. So, suppose we have the g_{jk} which is the transformation of a body K relative to body J and g_{ij} which is the transformation of body J relative to body I, then we can find the configuration of K relative to I as follows

$$\bar{g}_{ik} = \bar{g}_{ij}\bar{g}_{jk} = \begin{pmatrix} R_{ij} & p_{ij} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R_{jk} & p_{jk} \\ 0 & 1 \end{pmatrix} \tag{6.6.6}$$

$$= \begin{pmatrix} R_{ij}R_{jk} & R_{ij}p_{jk} + p_{ij} \\ 0 & 1 \end{pmatrix}. \tag{6.6.7}$$

Since (6.6.7) is a form of (6.3.4), we conclude that the composition of rigid body transformations and $I_{4\times 4}$ is in the special Euclidean group as well. In addition,

$$g^{-1} = \left(R^T p, R^T\right). \tag{6.6.8}$$

6.7 Rigid Body Velocities

We are concerned with the velocity of a rigid body with motion described by a time-parameterized curve $g(t) \in SE(3)$. For a start, we will consider the trajectory motion of an object frame J whose origin is at a frame I and it is rotating relative to the fixed frame I. We call frame I the *spatial frame* and the frame J, the *body frame*.

6.7.1 Rotational Velocities

For a point q whose coordinates q_j are fixed in the body frame, a path in spatial coordinates is followed given by

$$q_i(t) = R_{ij}(t)q_j. \tag{6.7.1}$$

The velocity in spatial coordinates is

$$v_{q_i}(t) = \frac{\partial}{\partial t}q_i(t) = \dot{R}_{ij}(t)q_j. \tag{6.7.2}$$

We see that $\dot{R}_{ij}$ maps the body coordinates, q_j to the spatial velocity of q. We would like to develop a more compact representation of the orientation of the point q relative to the spatial frame I by exploiting the special structure of $\dot{R}_{ij}$. Thus, we write

$$v_{q_i}(t) = \dot{R}_{ij}(t)R_{ij}^{-1}(t)R_{ij}(t)q_j. \tag{6.7.3}$$

We define the *instantaneous spatial angular velocity* $\hat{\omega}_{ij}^s \in \mathbb{R}^3$ as seen in the spatial frame I as

$$\hat{\omega}_{ij}^s(t) = \dot{R}_{ij}(t)R_{ij}^{-1}(t) \tag{6.7.4}$$

and we define the *instantaneous body angular velocity* as seen in the body frame J as

$$\hat{\omega}_{ij}^b(t) = R_{ij}^{-1}(t)\dot{R}_{ij}(t) \tag{6.7.5}$$

We define the body angular velocity as viewed from the instantaneous body frame J as

$$\hat{\omega}_{q_b}^b = R_{ij}^{-1}\hat{\omega}_{ij}^s R_{ij}, \text{ or } \hat{\omega}_{ij}^b = R_{ij}^{-1}\hat{\omega}_{ij}^s, \tag{6.7.6}$$

so that the body angular velocity can be recovered from the spatial angular velocity by rotating the angular velocity vector into the instantaneous body frame. Thus, (6.7.3) becomes

$$v_{q_i}(t) = \hat{\omega}_{ij}^s(t)R_{ij}(t)q_j = \hat{\omega}_{ij}^s(t) \times q_i(t) \tag{6.7.7}$$

Similarly, the velocity in the frame J can be derived from (6.7.1) as

$$q_j = R_{ij}^{-1}(t)q_i(t) \tag{6.7.8a}$$

$$v_{q_j}(t) = \dot{R}_{ij}^T(t)\, v_{q_i}(t) = \omega_{ij}^b(t) \times q_j. \tag{6.7.8b}$$

Thus we have the compact description of the rigid body particles' velocities in both the body and spatial angular frames as ω_{ij}^b and ω_{ij}^s.

Homework 31. Consider the motion of a point body about the x axis of an orthogonal triad with respect to a certain spatial frame located at $O(0, 0, 0)$. Determine the body and spatial velocities of the point with respect to the origin, O.

6.7.2 Rigid Body Velocity

We shall denote the rigid body motion of a frame J attached to a body so that is it fixed relative to a frame I by $g_{ij}(t)$. We write the velocity in the spatial frame as $\dot{g}_{ij}g_{ij}^{-1}$ which is symbolically $\left(\dot{p}_{ij}, \dot{R}_{ij}\right) \cdot \left(R_{ij}^T, -R_{ij}^T p_{ij}\right)$ following the notation we introduced in (6.6.8). This has an isomorphism of a twist and it is given by

Twist in Spatial Frame

$$\eta_{ij}^s = \dot{g}_{ij}g_{ij}^{-1} = \begin{bmatrix} \omega_{ij}^s & v_{ij}^s \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \dot{R}_{ij}R_{ij}^T & -\dot{R}_{ij}R_{ij}^T p_{ij} + \dot{p}_{ij} \\ 0 & 0 \end{bmatrix} \in \mathfrak{se}(3) \tag{6.7.9}$$

whereupon the twist coordinates $\xi^s \in \mathbb{R}^6$ in the body frame can be recovered as

$$\begin{pmatrix} v_{ij}^s \\ \omega_{ij}^s \end{pmatrix} = \begin{pmatrix} -\dot{R}_{ij}R_{ij}^T p_{ij} + \dot{p}_{ij} \\ (\dot{R}_{ij}R_{ij}^T)^\vee \end{pmatrix} \tag{6.7.10}$$

Similarly, in body coordinates, we find that the twist is given by

Twist in Body Frame

$$\eta_{ij}^b = g_{ij}^{-1}\dot{g}_{ij} = \begin{bmatrix} \omega_{ij}^b & v_{ij}^b \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \dot{R}_{ij}R_{ij}^T & R_{ij}^T p_{ij}\dot{p}_{ij} \\ 0 & 0 \end{bmatrix} \in \mathfrak{se}(3) \tag{6.7.11}$$

whereupon the twist coordinates $\xi^b \in \mathbb{R}^6$ in the body frame can be recovered as

$$\begin{pmatrix} v_{ij}^b \\ \omega_{ij}^b \end{pmatrix} = \begin{pmatrix} -\dot{R}_{ij}^T \dot{p}_{ij} \\ (R_{ij}^T \dot{R}_{ij})^\vee \end{pmatrix} \tag{6.7.12}$$

Furthermore, we have that

$$\eta_{ij}^s = \begin{pmatrix} -\dot{R}_{ij} & \hat{p}_{ij}R_{ij} \\ 0 & R_{ij} \end{pmatrix} \begin{pmatrix} v_{ij}^b \\ \omega_{ij}^b \end{pmatrix}. \tag{6.7.13}$$

Homework 32. Confirm equations (6.7.9), (6.7.11) and (6.7.13).

The matrix which transforms from the body coordinate frame to the spatial velocity frame is the so-called *adjoint transformation* of g (c.f. Fig. 6.2), defined as

$$Ad_g = \begin{pmatrix} R & \hat{p}R \\ 0 & R \end{pmatrix} \tag{6.7.14}$$

and whose inverse is given by

$$Ad_g^{-1} = \begin{pmatrix} R^T & -(R^Tp)^\wedge R^T \\ 0 & R^T \end{pmatrix} = \begin{pmatrix} R^T & -R^T\hat{p} \\ 0 & R^T \end{pmatrix} = Ad_{g^{-1}} \tag{6.7.15}$$

CHAPTER 7

STATE ESTIMATION

The next few topics in this course shall involve the quantification of uncertainty in order to enable a robot navigate, move, or understand its environment via visual or audio sensors. In order to do justice to this topic, we shall soon find out that the concept of putting a value or percentage on how sure we are about a robot's environment shall be very helpful in effective control of our robots. Thus the concept of probability shall greatly aid us in quantifying uncertainty. Even so, we introduce the concept of states, grounded in a mathematical theory that allows the engineer to implement a state through discrete-time systems (since we assume that most implementations shall be done on digital computers). By the *state* of a system, we shall loosely mean "those variables that provide a complete representation of the internal condition ot status of the system at a given time instant." In this sentiment, the states of a motor system may mean currents that flow through the inductive coils, the position and speed of motor shaft, or the voltage across the coils of a solenoid valve. The states of a military power may include the number of its aircraft carriers, the size and horsepower of its nuclear submarines, the number of enlisted servicemen in its forces e.t.c. For a biological system, the states might include blood sugar levels, heart and respiration rates, or body temperature.

Robot systems may include mobile platforms for extraterrestrial navigation, robotics arms in assembly lines, autonomous cars, or actuated surgical devices that assist surgeons. Our goal is to treat uncertainty. Uncertainty occurs if the robot lacks important information that hinders it from carrying out assigned tasks. We may classify this uncertainty into five different factors, viz.,

1. **Environments.** The physical worlds is inherently unpredictable. While the degree of uncertainty in well-structured environments such assembly lines is small, environments such as highways and private homes are highly dynamic and unpredictable.

2. **Sensors.** Most sensors have limitations in their perceptual ability arising from noise and the range and the resolution of the sensors. For example, environmental disturbances, weather,

lighting conditions limit the information that can be extracted from sensors. Secondly, as to range and resolution, cameras cannot see through walls despite the perceptual range that the spatial resolution of the camera is limited.

3. **Models.** In general, models are at best an approximation or a mathematical representation or abstraction of the physical world. As such, model errors are a source of uncertainty that need to be incorporated in modeling robotics problems.

4. **Computation.** Being real-time systems, robots require a lot of computation in order to be able to achieve timely-response through sacrificing accuracy.

We will estimate states as they shall represent latent or underlying variables that influence the physical or chemical or financial properties of the system. And in motivating the study of a system's state, we can resolve to many weapons in our estimation arsenal which may include linear state filtering (the simple Kalman filter), nonlinear state filters (the extended Kalman filter, unscented Kalman filter e.t.c.), Bayesian estimation, and *frequentist/classical* estimation approaches. In general, state estimation is an important topic to the engineer because:

- We may need to implement a feedback controller in order to regulate a system's behavior. If the application was for a surgeon to regulate blood pH levels, we may need to estimate the system's state. Or if the challenge is to adequately position a patient's head to a position in 3D space during cancer stereotactic radiosurgery, we may need to estimate the position and orientation of the patient's head and neck in the inertial frame.

- If the states in question are curious enough, we may want to measure these states to understand the faults tolerance of the system in order to perform a good fault identification and prognosis. For example, we might want to estimate the internal states of an aircraft system in flight such that if an aircraft engine fails during flight, we can safely monitor system states in real-time in order to determine how long we can continue flying the aircraft or if we should quickly find a near-by airport where we could land the aircraft for maintenance.

In our treatment, therefore, we shall give a brief introduction to linear systems theory, touch upon standard linear filters and then proceed to treat probability theory before we treat frequentist and Bayes inference for classification, and decision-making.

7.1 Linear Systems

State-space systems are very important in engineering systems because they allow us to gain insight into the characteristics of the system, be able to predict future behaviors of the system, as well as identify the controllable and observable states of the system. The mathematical model of the process allows us to infer the information about the process. State-space models can be classified into linear and nonlinear systems. While most real-world systems are nonlinear, the tools that exist for analyzing and synthesizing nonlinear systems are well-developed and sophisticated that most nonlinear systems can be approximated by linear systems in order to exercise good control and estimation for real-world applications.

A continuous-time, deterministic linear system can be described by the equations

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{7.1.1}$$

where x is the *state vector* in $\mathbb{R}^n \times 1$, u is the *control vector* in $\mathbb{R}^p \times 1$, and y is an $\mathbb{R}^n \times 1$ vector. Matrices A, B, and C are respectively $n \times n$, $n \times p$ and $n \times 1$ in dimension. The matrix A is often called the system matrix, B the input or control matrix, while C is often called the output matrix. A, B, and C can be time-varying matrices, in which case the system is linear. Otherwise, the solution to the linear system of equations above is

$$x(t) = e^{A(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{A(t-\tau)}Bu(\tau)d\tau \tag{7.1.2}$$

$$y(t) = Cx(t) \tag{7.1.3}$$

where t_0 is the initial time of the system. If the input control law is zero, then we have

$$x(t) = e^{A(t-t_0)}x(t_0) \tag{7.1.4}$$

and because of this, e^{AT} is called the state-transition matrix $i.e.$ it describes how the state moves between transitions at different times regardless of external inputs. At $t = t_0$, we have that

$$e^{A\,0} = I, \tag{7.1.5}$$

which is similar to the scalar exponential of a zero. What happens if x is an n-element vector? The solution in (7.1.3) still remains valid but we must note that the exponential of the matrix becomes interpreted as

$$\begin{aligned} e^{At} &= \sum_{j=0}^{\infty} \frac{(At)^j}{j!} \\ &= \mathcal{L}^{-1}\left[sI - A\right]^{-1} = Qe^{\hat{A}t}Q^{-1} \end{aligned} \tag{7.1.6}$$

where the symbol $\mathcal{L}^{-1}$ is the symbol for the inverse Laplace transform and "s" is the Laplace operator. We see that A must be square in order for e^{At} to exist. Q contains the eigenvectors of A and $\hat{A}$ are the Jordan form of A.

Quiz 6. Write a note about the Jordan form. Also, explain how it can be determined from (7.1.6).

Quiz 7. Does the matrix A commute with its exponential i.e. does $A\,e^{At} = e^{At}\,A$?

The matrix $\hat{A}$ is often diagonal, so that case $e^{\hat{A}t}$ can be computed as

$$\hat{A} = \begin{bmatrix} \hat{A}_{11} & 0 & \dots & 0 \\ 0 & \hat{A}_{22} & \dots 0 & \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & \hat{A}_{nn} \end{bmatrix} \quad e^{\hat{A}t} = \begin{bmatrix} e^{\hat{A}_{11}} & 0 & \dots & 0 \\ 0 & e^{\hat{A}_{22}} & \dots 0 & \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & e^{\hat{A}_{nn}} \end{bmatrix} \tag{7.1.7}$$

From (7.1.6), we can write

$$\left[e^{At}\right]^{-1} = e^{-At} = Qe^{-\hat{A}t}Q^{-1} \tag{7.1.8}$$

Since A and $-A$ have eigenvalues that are negative of each other, e^{At} is always invertible.

Example 2. Suppose we are controlling angular heading of a mobile robot (for example, using voltage applied to its wheels' rotor windings in order to generate command velocity along the x, y, and z heading, i.e. θ, ω and α respectively). The derivative of the angular velocity vector can be written as

$$\begin{aligned}\dot{\theta} &= \omega + \alpha + 3.5\omega_1 + 6\theta_2 \\ \dot{\omega} &= u + 0.1\theta + 2.5\alpha + \omega_1 + \omega_2^2 \\ \dot{\alpha} &= \theta_1 + 2u \end{aligned} \tag{7.1.9}$$

The scalars ω_1, ω_2, θ_1 and θ_2 are acceleration noise terms such as gear backlash, friction, and modeling errors. If our measurement consists of the θ and ω states, it follows that we can write the state space equation as

$$\begin{aligned}\begin{bmatrix}\dot{\theta} \\ \dot{\omega} \\ \dot{\alpha}\end{bmatrix} &= \begin{bmatrix}0 & 1 & 1 \\ 1 & 0 & 2.5 \\ 0 & 0 & 0\end{bmatrix} + \begin{bmatrix}0 \\ 1 \\ 2\end{bmatrix} u + \begin{bmatrix}3.5\omega_1 + 6\theta_2 \\ \omega_1 + \omega_2^2 \\ \theta_1\end{bmatrix} \\ y &= \begin{bmatrix}1 & 1 & 0\end{bmatrix} + \begin{bmatrix}\theta \\ \omega \\ \alpha\end{bmatrix} + \begin{bmatrix}v_x \\ v_y \\ v_z\end{bmatrix}\end{aligned} \tag{7.1.10}$$

where $v = [v_x, v_y, v_z]^T$ is the linear velocity vector for the robot.

Example 3. Suppose that

$$A = \begin{bmatrix}0 & 1 \\ 0 & 0\end{bmatrix} \tag{7.1.11}$$

It follows that

$$\begin{aligned}e^{At} &= \sum_{j=0}^{\infty} \frac{(At)^j}{j!} \\ &= (At)^0 + (At)^1 + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \dots \\ &= I + At\end{aligned} \tag{7.1.12}$$

where the last term follows from the fact that $A^k = 0$ for $k > 1$ so that

$$e^{At} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & t \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \tag{7.1.13}$$

Using the expression for the inverse Laplace transform earlier, we have

$$\begin{aligned} e^{At} &= \mathcal{L}^{-1}\left[(sI - A)^{-1}\right] \\ &= \mathcal{L}^{-1}\left(\begin{bmatrix} s & -1 \\ 0 & s \end{bmatrix}^{-1}\right) \\ &= \mathcal{L}^{-1}\begin{bmatrix} 1/s & 1/s^2 \\ 0 & 1/s \end{bmatrix} \\ &= \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \end{aligned} \tag{7.1.14}$$

Homework 33. Find the eigendata of the matrix A in (7.1.14). Then determine the following terms using the eigenvector and eigenvalue that you may find: $\hat{A}, Q$ and e^{At}.

Homework 34. Produce a one-page report on a control system transfer function.

7.2 State Space Standard Forms

For a linear system, there are many possible state space models that can result in the same *transfer function dynamics*. Therefore, standardizing the state space model structures is relevant for solving problems in a conformal way. For consider the following input-output system linear difference equation

$$y_n + a_1 y_{n-1} + \ldots + a_{n-1} y_1 + a_n y = b_0 u_n + b_1 u_{n-1} + \ldots + b_{n-1} u_1 + b_n u \tag{7.2.1}$$

with u and y serving respectively as the input and output, and y_n serving as the nth derivative of y with respect to time. If we take the Laplace transform of both sides, we have

$$Y(s)\left(s^n + a_1 s^{n-1} + \ldots + a_{n-1} s + a_n\right) = U(s)\left(b_0 s^n + b_1 s^{n-1} + \ldots + b_{n-1} s + b_n\right) \tag{7.2.2}$$

so that the transfer function from the input u to the output y can be written as

$$\frac{Y(s)}{U(s)} = \frac{b_0 s^n + b_1 s^{n-1} + \ldots + b_{n-1}s + b_n}{s^n + a_1 s^{n-1} + \ldots + a_{n-1}s + a_n} \tag{7.2.3}$$

7.2.1 Companion form

In *companion form* representation, the coefficients of the transfer function in (7.2.3) are arranged along its far rows or columns. An example would be

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & 0 & \cdots & 0 & -a_2 \\ 0 & 0 & 1 & \cdots & 0 & -a_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -a_{n-1} \end{bmatrix} \tag{7.2.4}$$

or

$$\begin{bmatrix} -a_{n-1} & -a_{n-2} & -a_{n-3} & \cdots & -a_1 & -a_0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{7.2.5}$$

In general, we use the convenient *observable* and *controllable* canonical forms in control theory. They are exactly the transpose of one another and using either for control design simplifies the system structure so that it can be readily manipulated for a desired control.

7.2.2 Modal Form

The modal form is the dual to the companion form. In the modal form, the state matrix is a diagonal matrix with non-repeating eigenvalues such that the control has a unitary influence on each eigenspace, and the output is a linear combination of the contributions from the eigenspaces. That is,

$$A = \begin{bmatrix} -p_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -p_2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -p_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & -p_n \end{bmatrix} \tag{7.2.6a}$$

$$B = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix} \tag{7.2.6b}$$

Homework 35. Write out the solution to eq. 36 in modal form.

7.2.3 Controllable Canonical Form

When we want to design a controller that leverages the full state of the system (assuming this is known), often the *controllable canonical form* will come in handy. It is expressed as follows:

$$A = \begin{bmatrix} -a_1 & -a_2 & -a_3 & \cdots & -a_{n-1} & -a_n \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{7.2.7a}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad C = \begin{bmatrix} b_1 & b_2 & b_3 & \cdots & b_n \end{bmatrix} \quad D = \begin{bmatrix} b_0 \end{bmatrix} \tag{7.2.7b}$$

Example 4. For the system

$$\frac{Y(s)}{U(s)} = \frac{5s^2 - s + 8}{s^2 + 4s - 2} \tag{7.2.8}$$

we can realize the state space representation in canonical form as follows:

1. Observe that n from (7.2.3) is 3, $i.e.$ the highest s exponent in the given transfer function.

2. It follows that we have $a_0 = -2$ and $a_1 = 4$; and $b_0 = 1$, $b_1 = -21$, $b_2 = 102$, so that we can write the state space model as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} u_1 & u_2 \end{bmatrix}$$
$$y = \begin{bmatrix} -21 & 102 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \boldsymbol{u} \tag{7.2.9}$$

Homework 36. Derive the companion form for the system:

$$\frac{Y(s)}{U(s)} = \frac{3s^2 - 2s + 1}{s^2 - 8s + 5} \tag{7.2.10}$$

The controllable canonical form is helpful in when using the pole placement method for controller design. However, the system's transformation to companion form is based on the controllability matrix which is almost always numerically singular for mid-range orders. It should be avoided for computation when possible.

7.2.4 Observable Canonical Form

In observable canonical form, the transfer function coefficients of (7.2.3) are written in the rightmost column of the A matrix similar to the companion canonical form but the B matrix takes a different form. It is given as follows:

$$A = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & 0 & \cdots & 0 & -a_2 \\ 0 & 0 & 1 & \cdots & 0 & -a_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -a_{n-1} \end{bmatrix} \tag{7.2.11}$$

$$B = \begin{bmatrix} b_n - a_n b_0 \\ b_{n-1} - a_{n-1} b_0 \\ b_{n-2} - a_{n-2} b_0 \\ \vdots \\ b_1 - a_1 b_0 \end{bmatrix} \quad C = \begin{bmatrix} 0 & 0 & \ldots & 1 \end{bmatrix}, \quad D = b_0 \tag{7.2.12}$$

This observable canonical form is ill-conditioned for most state-space computation. It should be avoided for computation when possible as its controllability matrix is almost always numerically singular for mid-range orders.

The observable and controllable canonical forms' matrices are respectively transposes of one another.

Homework 37. Transform the exercise of 36 to observable canonical form.

7.3 Nonlinear Systems

All the world is a nonlinear system. He linearized to the right. He linearized to the left. Till nothing was right. And nothing was left. – Stephen Billings.

Our treatment of dynamical so far has involved linear systems. These are optimistic models of the real world as in the reality, nothing is really linear. In general, a nonlinear system is a system which is not linear $i.e.$, does not satisfy the *principle of superposition*. Even a simple resistor exhibits nonlinearity. However, we utilize Ohm's law in approximating the dynamics of a resistor. This is because the equation is valid over a wide enough operating range. In this light, while we may say linear systems do not exist in real life, linear systems is a useful tool for describing nonlinear systems. We will write a general nonlinear system with the equation

$$\begin{aligned} \dot{x} &= f(x, u, w) \\ y &= h(x, v) \end{aligned} \tag{7.3.1}$$

where $f(\cdot)$ and $h(\cdot)$ are arbitrary vector valued functions, w denotes the process noise, and v denotes the measurement noise. We have a time-varying system if $f(\cdot)$ and $h(\cdot)$ are explicit functions of t, otherwise, the system is termed time-invariant. Suppose that $f(x, u, w) = Ax + Bu + w$ and $h(x, v) = Hx + v$, then the system is linear. Otherwise, the system is nonlinear.

Often, we will need to linearize a nonlinear system in order to properly analyze its stability properties or synthesize its parameters for example, a particular control application. Suppose we have a nonlinear vector function $f(\cdot)$ of a scalar x, we can expand $f(x)$ in a Taylor series around some nominal operating point, $x = \bar{x}$ so that

$$f(x) = f(\bar{x}) + \left.\frac{\partial f}{\partial x}\right|_{\tilde{x}} \tilde{x} + \frac{1}{2!}\left.\frac{\partial^2 f}{\partial x^2}\right|_{\tilde{x}} \tilde{x}^2 + \frac{1}{3!}\left.\frac{\partial^3 f}{\partial x^3}\right|_{\tilde{x}} \tilde{x}^3 + \dots \tag{7.3.2}$$

where $\tilde{x} = x - \bar{x}$. For a 2×1 vector x, we can write $f(x)$ as follows:

$$\begin{aligned} f(x) = f(\bar{x}) + \left.\frac{\partial f}{\partial x_1}\right|_{\tilde{x}} \tilde{x}_1 + \left.\frac{\partial f}{\partial x_2}\right|_{\tilde{x}} \tilde{x}_2 + \frac{1}{2!}\left(\left.\frac{\partial^2 f}{\partial x_1^2}\right|_{\tilde{x}} \tilde{x}_1^2 + \left.\frac{\partial^2 f}{\partial x_2^2}\right|_{\tilde{x}} \tilde{x}_2^2 + 2\left.\frac{\partial^2 f}{\partial x_1 x_2}\right|_{\tilde{x}} \tilde{x}_1\tilde{x}_2\right) + \\ \frac{1}{3!}\left(\left.\frac{\partial^3 f}{\partial x_1^3}\right|_{\tilde{x}} \tilde{x}_1^3 + \left.\frac{\partial^3 f}{\partial x_2^3}\right|_{\tilde{x}} \tilde{x}_2^3 + 3\left.\frac{\partial^3 f}{\partial x_1^2 x_2}\right|_{\tilde{x}} \tilde{x}_1^2\tilde{x}_2 + 3\left.\frac{\partial^3 f}{\partial x_1 x_2^2}\right|_{\tilde{x}} \tilde{x}_1\tilde{x}_2^2\right) + \dots \end{aligned} \tag{7.3.3}$$

which can be compactly written as

$$f(x) = f(\tilde{x}) + \tag{7.3.4}$$

REFERENCES

Ball, R. S. (1908). *A Treatise on the Theory of Screws*. Cambridge university press. 63

Bern, J. M., K.-H. Chang, and S. Coros (2017). Interactive design of animated plushies. *ACM Transactions on Graphics (TOG) 36*(4), 80. 49

Besl, P. J. and N. D. McKay (1992). Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, Volume 1611, pp. 586–606. International Society for Optics and Photonics. 28

Bishop-Moser, J., G. Krishnan, C. Kim, and S. Kota (2012). Design of soft robotic actuators using fluid-filled fiber-reinforced elastomeric enclosures in parallel combinations. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4264–4269. IEEE. 49

Giorelli, M., F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi (2015). Neural Network And Jacobian Method For Solving The Inverse Statics Of A Cable-driven Soft Arm With Nonconstant Curvature. *IEEE Transactions on Robotics 31*(4), 823–834. 51

Gough, V. (1957). Contribution to discussion of papers on research in automobile stability, control and tyre performance. *Proc. of Auto Div. Inst. Mech. Eng. 171*, 392–395. 55

Hannan, M. W. and I. D. Walker (2000). Novel Kinematics for Continuum Robots. pp. 227–238. Advances in Robot Kinematics. 51

Hannan, M. W. and I. D. Walker (2003). Kinematics and the Implementation of an Elephant's Trunk Manipulator and Other Continuum Style Robots. *Journal of Robotic Systems 20*(2), 45–63. 51

Hopkins, J. B., J. Rivera, C. Kim, and G. Krishnan (2015). Synthesis and analysis of soft parallel robots comprised of active constraints. *Journal of Mechanisms and Robotics 7*(1), 011002. vi, 50

Horn, B. K. (1987). Closed-form solution of absolute orientation using unit quaternions. *Josa a 4*(4), 629–642. 35

Hunt, K. H. (1977). *Kinematic Geometry of Mechanisms*. Oxford University Press. vi, 49, 57

Hunt, K. H. (1983). Structural Kinematics of In- Parallel-Actuated Robot-Arms. *105*(December 1983), 705–712. vi, 53

Kabsch, W. (1978). A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A 34*(5), 827–828. 29, 35

Merlet, J. (2015). *Parallel robots*. Springer. 55

Murray, R. M. (1994). *A Mathematical Introduction To Robotic Manipulation*. CRC press. 57, 71

Ogunmolu, O. (2019a). Kinematics and Dynamics of an In-Parallel-Actuated Soft Robot Manipulator: Application to Cranial Cancer Radiosurgery. 51

Ogunmolu, O., A. Kulkarni, Y. Tadesse, X. Gu, S. Jiang, and N. Gans (2017). Soft-NeuroAdapt: A 3-DOF Neuro-Adaptive Patient Pose Correction System for Frameless and Maskless Cancer Radiotherapy. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, CA*, pp. 3661–3668. IEEE. 51

Ogunmolu, O., X. Liu, N. Gans, and R. Wiersma (2020). Mechanism and Constitutive Model of a Continuum Robot for Head and Neck Cancer Radiotherapy. In *IEEE International Conference on Robotics and Automation (ICRA), Paris, France*. 51

Ogunmolu, O. P. (2019b). *A Multi-DOF Soft Robot Mechanism for Patient Motion Correction and Beam Orientation Selection in Cancer Radiation Therapy.* Ph. D. thesis, The University of Texas at Dallas; UT Southwestern Medical Center. 51

Ogunmolu, O. P., X. Gu, S. Jiang, and N. R. Gans (2015). A Real-Time, Soft Robotic Patient Positioning System for Maskless Head and Neck Cancer Radiotherapy: An Initial Investigation. In *Automation Science and Engineering (CASE), 2015 IEEE International Conference on, Gothenburg, Sweden*, pp. 1539–1545. IEEE. 51

Ogunmolu, O. P., X. Gu, S. Jiang, and N. R. Gans (2016). Vision-based Control of a Soft Robot for Maskless Head and Neck Cancer Radiotherapy. In *Automation Science and Engineering (CASE), 2016 IEEE International Conference on, Fort Worth, Texas*, pp. 180–187. IEEE. 51

Renda, F., M. Giorelli, M. Calisti, M. Cianchetti, and C. Laschi (2014). Dynamic model of a multibending soft robot arm driven by cables. *IEEE Transactions on Robotics 30*(5), 1109–1122. 51

Renda, F. and L. Seneviratne (2018). A Geometric and Unified Approach for Modeling Soft-Rigid Multi-body Systems with Lumped and Distributed Degrees of Freedom. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 1567 – 1574. 51

Shepherd, R. F., F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides (2011). Multigait soft robot. *Proceedings of the National Academy of Sciences 108*(51), 20400–20403. 51

Spong, M. W., S. A. Hutchinson, and M. Vidyasagar (2006). Robot modeling and control. *IEEE Control Systems 26*(6), 113–115. vi, 60, 82

Stewart, D. (1965). A platform with six degrees of freedom. *Proceedings of the institution of mechanical engineers 180*(1), 371–386. 55

Weisstein, E. W. Homeomorphic. 29