Robot Basics: Representation, Rotation and Velocity

Jiawei Zhang   JIAWEI@IFMLAB.ORG

Founder and Director
Information Fusion and Mining Laboratory
(First Version: October 2022; Revision: November 2022.)

Abstract

In this article, we plan to provide an introduction about some basics about robots for readers. Several key topics of classic robotics will be introduced, including *robot representation*, *robot rotational motion*, *coordinates transformation* and *velocity transformation*. By now, classic rigid-body robot analysis is still the main-stream approach in robot controlling and motion planning. In this article, no data-driven or machine learning based methods will be introduced. Most of the materials covered in this article are based on the rigid-body kinematics that the readers probably have learned from the physics course at high-school or college. Meanwhile, these classic robot kinematics analyses will serve as the foundation for the latest intelligent robot control algorithms in modern robotics studies.

Keywords: Robotics; Robot Representation; Robot Configuration; Coordinate System; Rotational Motion; Linear Velocity; Angular Velocity

Contents

1. Introduction

According to the forecast report from UN (United Nations) [3], the world population aging will become one of the most challenging global problem of the 21st century. The working age population growth of the major developing countries (e.g., China, India) will gradually slow down and even start to decrease together with the major developed countries (e.g., UK, France, Germany, Japan and USA). To fulfill the "tremendous gap" between the supply and demand of working labor forces, various robots and automated machines has been (and will continue to be) developed and employed for massive production work globally. This is an irreversible trend for the 21st century. To accomplish such an objective, educating and training researchers and practitioner on robotics is imperative and critical at present.

1.1 This Article

In this tutorial article, we will provide a brief introduction about robotics for readers, which covers *robot representation*, *robot motion*, *coordinate transformation* and *velocity transformation* in the rotational motions. This tutorial article will also be organized according to the sequential order these topics. Some more advanced topics about *robot kinematics*, *trajectory generation*, *robot control*, *motion planning* and *manipulation* and *applications* will

be introduced in the follow-up tutorial articles instead. Besides these tutorial articles, for readers interested in robot control, we also have several textbooks recommended for you to read as well, like [1, 2].

This tutorial article will be math-heavy. Just like you, I don't like math either, but we have to use math to model the real world. Based on the equations we introduce in the article, we will be able to write the code to represent and control the robot. The bitter has to been swallowed first, then the sweet will come afterwards. Before we discuss about the aforementioned topics, we would like to briefly describe the notations to be used in this article as follows.

1.2 Basic Notations

In the sequel of this article, we will use the lower case letters (e.g., x) to represent scalars, lower case bold letters (e.g., $\mathbf{x}$) to denote column vectors, bold-face upper case letters (e.g., $\mathbf{X}$) to denote matrices. Given a vector $\mathbf{x}$, its length is denoted as $\|\mathbf{x}\|$. Given a matrix $\mathbf{X}$, we denote $\mathbf{X}(i,:)$ and $\mathbf{X}(:,j)$ as its i_{th} row and j_{th} column, respectively. The (i_{th}, j_{th}) entry of matrix $\mathbf{X}$ can be denoted as either $\mathbf{X}(i,j)$ or $\mathbf{X}_{i,j}$, which will be used interchangeably. We use $\mathbf{X}^\top$ and $\mathbf{x}^\top$ to represent the transpose of matrix $\mathbf{X}$ and vector $\mathbf{x}$. The cross product of vectors $\mathbf{x}$ and $\mathbf{y}$ is represented as $\mathbf{x} \times \mathbf{y}$. A coordinate system is denoted as Σ, and the vector $\mathbf{x}$ in coordinate system Σ can also be specified as $\mathbf{x}^{[\Sigma]}$. For a scalar x, vector $\mathbf{x}$ and matrix $\mathbf{X}$, we can also represent their first-order derivatives as $\dot{x}$, $\dot{\mathbf{x}}$ and $\dot{\mathbf{X}}$, and second-order derivatives as $\ddot{x}$, $\ddot{\mathbf{x}}$ and $\ddot{\mathbf{X}}$.

2. Robot Representation

In this tutorial article, we will focus on rigid-body robots with known shapes. Besides rigid-body robots, research on soft-body robots is also very popular nowadays, but it is out of the scope of this tutorial article.

2.1 Robot Structure

At the beginning, we would like to introduce the robot physical structure and its representation with certain data structures in program for readers.

2.1.1 Robot Physical Structure

As shown in Figure 1, we provide an example of the rigid-body robot structure, involving the trunk and a number of links connected together via the joints. For the representation and motion control simplicity, we prefer to attach the joints to the link that is further away from the trunk. In this robot structure, each link will have only one joint to control its motion, so the same algorithm can be applied to control all the links and joints in the robot.

Meanwhile, in the small plot shown at the right-hand side in Figure 1, we also provide another robot structure, where the joints are attached to the links closer to the trunk. In such a structure, the robot trunk will have multiple attached motors, e.g., 2 from leg, 2 from arm and 1 from neck. For such a robot structure, the control of the robot trunk with 5 attached joints will be very different from the control of the arms and legs, which may create extra difficulties in the robot motion and control.
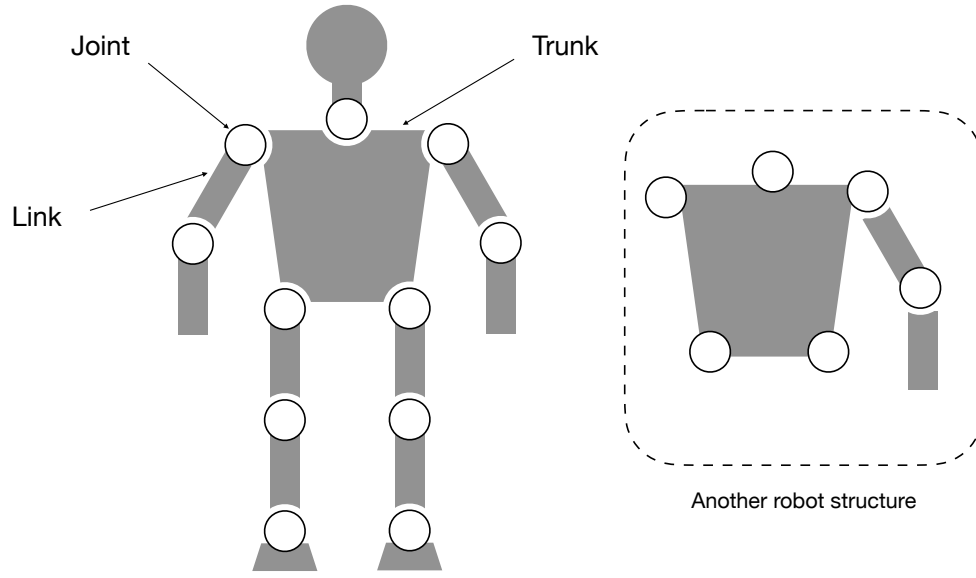
Figure 1: A Example of Rigid-Body Robot Structure (involving the trunk, joints and links).
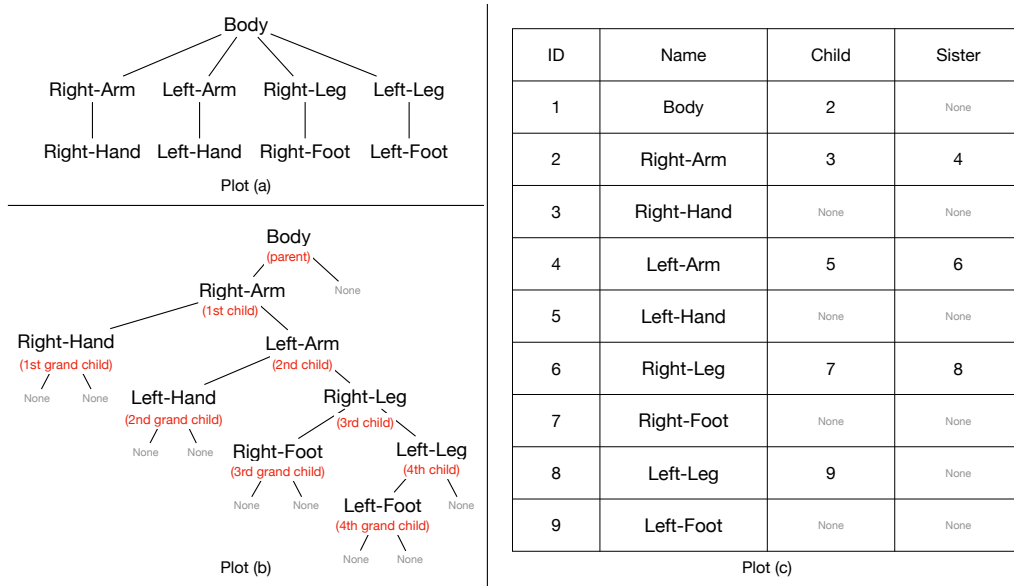

Figure 2: Data Structure for Representing Robot Body Structure: (a) balanced tree data structure, (b) binary tree data structure, (c) list of body parts.

2.1.2 DATA STRUCTURE

To represent the robot structure shown in Figure 1 in program, various data structures can be used to represent the robot body parts and connections among them. In robot control,

we are especially interested in the parts that are moveable. As illustrated in the Plot (a) of Figure 2, we can use a *tree* to represent the robot structure, involving the body and four limbs directly connected to the trunk body. Meanwhile, each of the four limbs is also connected to either a hand node or a foot node as their child, respectively.

For the nodes in Plot (a) of Figure 1, we observe that the tree nodes' out-degrees are quite different, *e.g.*, the body root node's out degree is 4, while the arm/leg nodes' out degrees are all 1 instead. Such node out-degree inconsistency may create potential problems in writing the code for robot control.

In Plot (b) of Figure 1, we illustrate another tree structured representation of the robot physical structure, where each internal node is connected to two other nodes, *i.e.*, one child and one sibling. For instance, the "Right-Arm" node is connected to both "Right-Hand" (its child node) and "Left-Arm" (its sibling node). For the nodes without children or siblings, we can just use the dummy "None" to fill the entries.

Compared with the balanced tree in Plot (a), the new tree structure in Plot (b) is skewed and inclined to the right-hand side. Meanwhile, the tree structure is a binary tree actually, as each node connects to two nodes at the lower level, programming based on which will be much easier. Actually, such a tree data structure can be implemented very easily as a list, where each node is represented as a record with unique IDs in the table, whose child and sibling nodes can be indicated as its attributes. An example to implement the binary tree structure as a list is also illustrated in Plot (c) of Figure 1.

2.1.3 Notation Table

Besides the basic notations introduced at the end of Section 1.2, in this part, we will also introduce some other notations that will be used in the following sections for describing the robot parts and their motions in the following Table 1.

Table 1: Robot Link Notation in Math and Program

| Notation Table in Math and Program Code		
Parameters of Robot Link	Math Notation	Code Notation
Link Self-ID	j	self_id
Child ID	N/A	child_id
Sibling ID	N/A	sibling_id
Parent ID	i	parent_id
Position Vector	$\mathbf{p}$	p
Rotation Matrix	$\mathbf{R}$	R
Homogeneous Transition Matrix	$\mathbf{T}$	T
Velocity in World Coordinate	$\mathbf{v}$	v
Angular Velocity in World Coordinate	$\boldsymbol{\omega}$	w
Joint Angle	q	q
Joint Rotation Velocity	$\dot{q}$	dq
Joint Rotation Acceleration	$\ddot{q}$	ddq
Mass	m	m
Center of Mass	$\mathbf{c}$	c
Moment of Inertia	$\mathbf{I}$	I
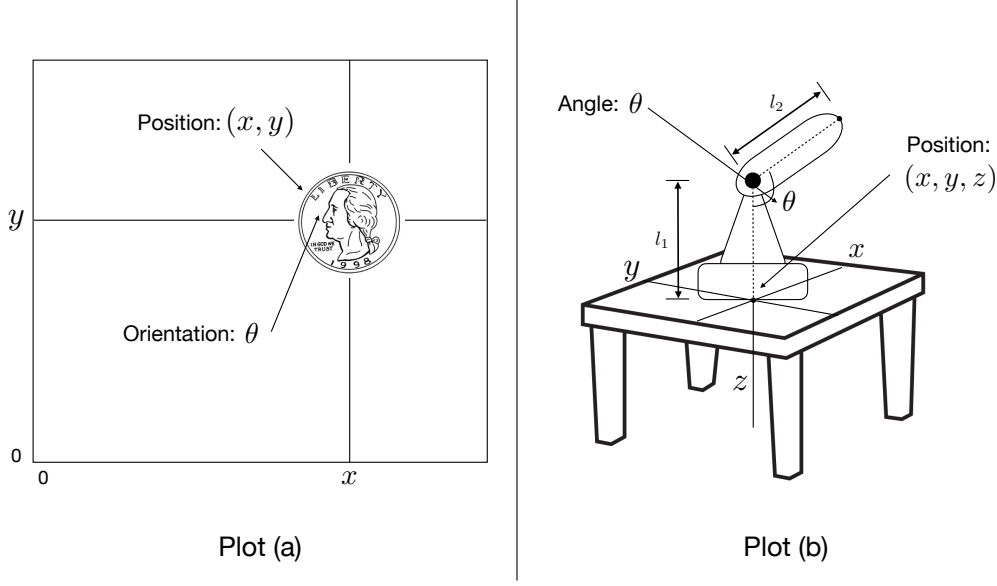
Figure 3: A Example of Rigid-Body Item Configuration.

2.2 Robot Configuration and Configuration Space

In this part, we will introduce the concepts of *robot configuration* and *configuration space*. Both *robot configuration* and *configuration space* play an important role in robot motion, control and kinematics. In the following sections, when we introduce the *robot motion, coordinate transformation* and *velocity transformation*, readers will observe that robot motion will involve a sequence of *robot configurations* (more like a configuration path) connecting from the starting configuration to the desired target configuration within the pre-defined robot *configuration space*.

2.2.1 ROBOT CONFIGURATION

For a rigid-body robot, its location and state can be uniquely specified by its *configuration*.

Definition 1 (***Robot Configuration***): *The robot configuration denotes a complete specification of the positions of every point in the robot system.*

In the real-world, we cannot enumerate all the points of a robot body to represent its *configuration*. For rigid-body robots, since all its composition parts are rigid and have pre-known shapes, actually only a few numbers will be sufficient to represent its *configuration*.

Example 1 *For instance, as shown in Plot (a) of Figure 3, given a coin lying heads up on a table (which can be represented as a 2D plane), the location and state of the coin can be specified with three parameters $((x, y), \theta)$: (1) the coordinate parameter pair (x, y) that denotes its location on the table, and (2) a third parameter θ that indicates the coin's orientation (head up or tail up).*

Example 2 *Besides the configuration in 2D space, 3D space is much more frequently used in robotics. As shown in Plot (b) of Figure 3, for a robot arm with one vertical link (of known length l_1) and a movable arm link (of known length l_2) connected by a joint, which forms an angle θ between these two links. The robot arm position and state can be represented with four parameters $((x, y, z), \theta)$, where (x, y, z) denotes the position of the base center and θ indicates the angle between these two arm links.*

2.2.2 ROBOT CONFIGURATION SPACE

Definition 2 *(**Configuration Space**): For a given robot, there usually exist different feasible configurations, and the set of all potential configurations of the robot defines the configuration space of the robot.*

In some textbooks, the robot *configuration space* is also called the *C-space* for abbreviation. It is also easy to observe that given the robot's *configuration space*, one of its specific *configuration* will actually be a point in the *configuration space*.

Example 3 *For instance, for the coin shown in Plot (a) of Figure 3, let $\mathbb{X}$ and $\mathbb{Y}$ denote the sets of potential x-coordinate and y-coordinate values of the coin's position. Then, the configuration space of the coin can be represented as $\mathbb{X} \times \mathbb{Y} \times \{Head, Tail\}$.*

Example 4 *When it comes to the robot arm shown in Plot (b) of Figure 3, let $\mathbb{X}$, $\mathbb{Y}$ and $\mathbb{Z}$ denote the sets of potential x, y and z coordinate values of the robot arm base center, the configuration space of the robot arm can be represented as $\mathbb{X} \times \mathbb{Y} \times \mathbb{Z} \times [0, 2\pi)$, where the arms' angle θ can be any value within the range of $[0, 2\pi)$.*

In the real world, there exist different ways to define robot configuration space, and we use the sphere as an example to illustrate different configuration spaces below.

Example 5 *For instance, a point on a sphere surface of radius $r \in \mathbb{R}$ can be uniquely represented with its "(latitude, longitude)" pair, just like the GPS localization on the earth. Meanwhile, the point on the sphere surface can also be represented with a (x, y, z) coordinate pair in the Euclidean space subject to the constraint $x^2 + y^2 + z^2 = r^2$.*

The former representation in the above example is normally called the *explicit representation*, while the latter one is called the *implicit representation*. Based on the *robot configuration* and *configuration space* concepts introduced above, next we will study robot representation and coordinate rotational transformation based on the *implicit representation* in Euclidean space by default.

2.3 Robot Coordinate System and Representation

Based on the above descriptions, we know that robot's position and state can be uniquely represented with its *configuration*, which denotes a group of numerical parameters. In this part, we will introduce the robot coordinate system to clearly specify such parameters. The coordinate systems to be used for representing robots include both the *world coordinates* and *local coordinates*.
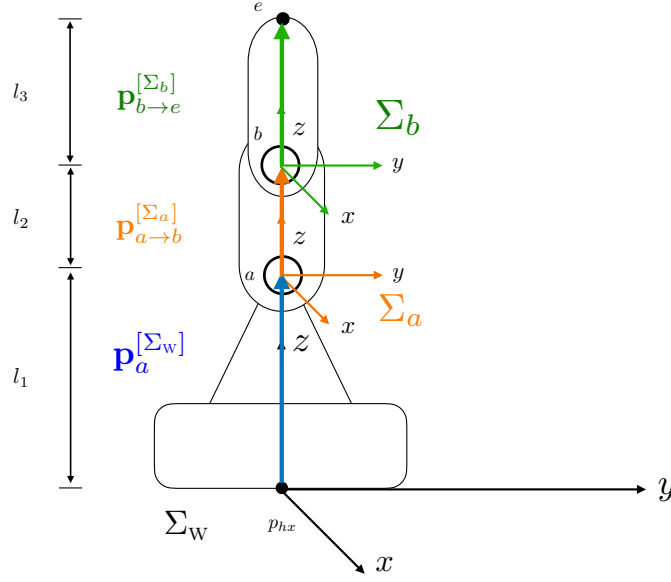
Figure 4: A Example of World Coordinate.

2.3.1 WORLD COORDINATE SYSTEM

To control robots in the real world, we need a fixed coordinate system to define the positions of all parts of the robot, *i.e.*, the *world coordinate*.

Definition 3 *(**World Coordinate**): Formally, the world coordinate denotes a coordinate system with a fixed origin and fixed orientations of the x, y and z axes. From the robot perspective, the orientation of the x axis faces forward, y axis to the left and z axis faces up. In this tutorial article, we can refer to the world coordinate with a default notation Σ_{W}.*

For the *world coordinate* notation Σ_{W}, the term Σ denotes the coordinate and the subscript w differentiates the *world coordinate* from other *local coordinates* to be introduced later.

Example 6 *As shown in Figure 4, we provide an example of the world coordinate Σ_{W}, whose origin is located at the center of the robot arm base. The robot arm has two links connected via the joints a and b, and the end point of the second arm link is denoted as e.*
Based on the world coordinate Σ_{W}, we can represent the vector point from the origin to the first joint a as a vector $\mathbf{p}_{o \to a}^{[\Sigma_{\mathrm{W}}]}$ (in the blue color), whose orientation faces upward. Similarly, we can also represent the vectors pointing from the world coordinate system origin to the joint b and end point e as $\mathbf{p}_{o \to b}^{[\Sigma_{\mathrm{W}}]}$ and $\mathbf{p}_{o \to e}^{[\Sigma_{\mathrm{W}}]}$, respectively.

In the above example, we observe the length of the arm links are l_1 (from the base to the first joint), l_2 (from the first joint to the second one) and l_3 (from the second joint to the end point). Therefore, the vectors $\mathbf{p}_{o \to a}^{[\Sigma_{\mathrm{W}}]}$, $\mathbf{p}_{o \to b}^{[\Sigma_{\mathrm{W}}]}$ and $\mathbf{p}_{o \to e}^{[\Sigma_{\mathrm{W}}]}$ can actually be represented as

$$\mathbf{p}_{o \to a}^{[\Sigma_{\mathrm{W}}]} = (0,0,l_1)^\top \ , \ \ \mathbf{p}_{o \to b}^{[\Sigma_{\mathrm{W}}]} = (0,0,l_1 + l_2)^\top \ , \ \ \mathbf{p}_{o \to e}^{[\Sigma_{\mathrm{W}}]} = (0,0,l_1 + l_2 + l_3)^\top, \tag{1}$$

where the subscript $o \rightarrow a$ denotes the vector pointing from the origin of the world coordinate to joint a.

When there exists one single world coordinate in controlling the robot, we can just omit the o in representing vectors pointing from the origin to certain points and vector representation $\mathbf{p}_{o \rightarrow a}^{[\Sigma_\mathrm{w}]}$ can be simplified as $\mathbf{p}_a^{[\Sigma_\mathrm{w}]}$ in the remaining parts of this tutorial. Similarly, vectors $\mathbf{p}_{o \rightarrow b}^{[\Sigma_\mathrm{w}]}$ and $\mathbf{p}_{o \rightarrow e}^{[\Sigma_\mathrm{w}]}$ can also be simplified as $\mathbf{p}_b^{[\Sigma_\mathrm{w}]}$ and $\mathbf{p}_e^{[\Sigma_\mathrm{w}]}$.

2.3.2 Local Coordinate Systems

Besides the *world coordinate*, we can also introduce various *local coordinate systems* attached to different parts of the robot rigid body. For the servo or motor driven robots, we normally prefer attach the origins of the *local coordinate systems* to the rotor centers the servo/motor.

Definition 4 (Local Coordinate): *By picking one point of the robot rigid body as the origin and specifying the x, y and z axes, we can construct a local coordinate system for a robot part. The local coordinate defines the position and state of the robot body parts from a local perspective.*

In the *local coordinate systems*, the representation of robot position and state will be very different from what it looks like from the world coordinate

Example 7 *As shown in Figure 4, by taking the robot arm joint a as the origin, we can also build a local coordinate system Σ_a (in the orange color). The orientation of the x, y and z axes are identical to the axis orientations of the world coordinate Σ_W. In a similar way, we can introduce the local coordinate system Σ_b (in the green color) with the identical axis orientations and have the origin attached to joint b.*

Within the local coordinate system Σ_a, we can represent the robot arm pointing from joint a to the second joint b as a vector

$$\mathbf{p}_{a \rightarrow b}^{[\Sigma_a]} = (0, 0, l_2)^\top, \tag{2}$$

where l_2 denotes the length of the corresponding arm link.

Similarly, in the local coordinate Σ_b, we can represent the robot arm pointing from joint b to the end point e as a vector

$$\mathbf{p}_{b \rightarrow e}^{[\Sigma_b]} = (0, 0, l_3)^\top, \tag{3}$$

where l_3 denotes the arm link length.

From the numbers of the vector representations of $\mathbf{p}_a^{[\Sigma_\mathrm{w}]}$, $\mathbf{p}_b^{[\Sigma_\mathrm{w}]}$ and $\mathbf{p}_e^{[\Sigma_\mathrm{w}]}$ in the world coordinate system, as well as $\mathbf{p}_b^{[\Sigma_a]}$ and $\mathbf{p}_e^{[\Sigma_b]}$ in the local coordinate systems, it seems that

$$\mathbf{p}_b^{[\Sigma_\mathrm{w}]} = \mathbf{p}_a^{[\Sigma_\mathrm{w}]} + \mathbf{p}_{a \rightarrow b}^{[\Sigma_a]}, \tag{4}$$

$$\mathbf{p}_e^{[\Sigma_\mathrm{w}]} = \mathbf{p}_b^{[\Sigma_\mathrm{w}]} + \mathbf{p}_{b \rightarrow e}^{[\Sigma_b]} = \mathbf{p}_a^{[\Sigma_\mathrm{w}]} + \mathbf{p}_{a \rightarrow b}^{[\Sigma_a]} + \mathbf{p}_{b \rightarrow e}^{[\Sigma_b]}. \tag{5}$$

Meanwhile, the above equations can hold as the x, y and z axes orientations of the local coordinate systems Σ_a and Σ_b are identical to those of the world coordinate Σ_W, i.e., $\mathbf{p}_{a \rightarrow b}^{[\Sigma_a]} = \mathbf{p}_{a \rightarrow b}^{[\Sigma_\mathrm{w}]}$ and $\mathbf{p}_{b \rightarrow e}^{[\Sigma_b]} = \mathbf{p}_{b \rightarrow e}^{[\Sigma_\mathrm{w}]}$. Meanwhile, when the arm links rotates, the local coordinate orientations will change and necessary transformation needs to be added to the above equations.
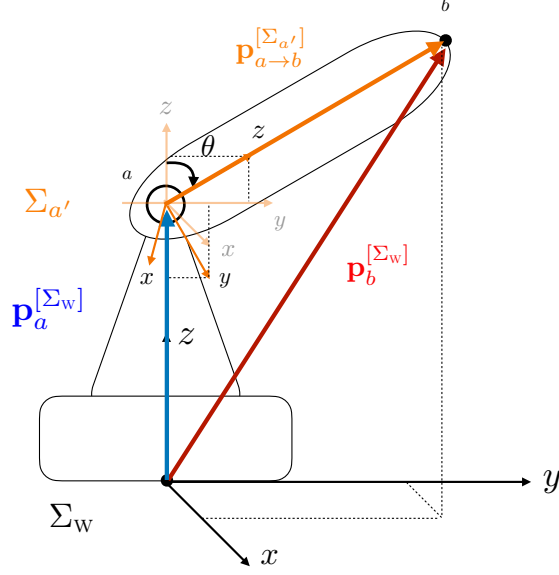
Figure 5: An Example of Robot Arm Local Coordinate After Rotation.

3. Robot End Point Position

Based on the above robot representation and notations, in this section, we will discuss about the robot end point position before and after the rotation of the robot arm. To interact with the world, robot end points play an important role. In this section, we are especially interested in the position, while the velocity of the end point will discussed in the following Section 5 instead.

3.1 Single-Link Robot End Point Position

In operation, robot links will rotate around the joint, and will change the position and state of the robot in the space. Such rotational movement can be clearly described with the *rotation matrix* and *homogeneous transformation matrix*. In this part, we will first introduce the *rotation matrix* that can denote the rotation transformation. After that, we will introduce the *homogeneous transformation matrix* that can project any vector from one coordinate system to another.

3.1.1 ROTATION MATRIX

Let's first discard the second robot arm link from the example shown in Figure 4 and only use the 1-arm robot illustrated in Figure 5 as an example.

Example 8 *As shown in Figure 5, we rotate the robot arm around joint a from the previous vertical position in Figure 4 in the clockwise direction with an angle of θ degrees to the current position. As the arm link rotates, the local coordinate attached to the robot arm joint will rotate clockwise with θ degrees as well, and the new local coordinate system after the rotation can be denoted as $\Sigma_{a'}$.*

Within the local coordinate system $\Sigma_{a'}$, the representations of robot arm link pointing from joint a to b after the rotation are not changed actually. In other words, in the new local coordinate $\Sigma_{a'}$, the arm link vector after rotation can still be represented as

$$\mathbf{p}_{a \to b}^{[\Sigma_{a'}]} = (0, 0, l_2)^\top. \tag{6}$$

Meanwhile, looking from the world coordinate Σ_W, we observe that the arm link vector pointing from joint a to b will be different from $\mathbf{p}_{a \to b}^{[\Sigma_{a'}]}$ actually, which can be represented as

$$\mathbf{p}_{a \to b}^{[\Sigma_W]} = (0, l_2 \sin\theta, l_2 \cos\theta)^\top. \tag{7}$$

To further clearly illustrate the rotations of the local coordinate systems from Σ_a to $\Sigma_{a'}$, we first need to talk about the relationship between the local coordinate systems Σ_a and $\Sigma_{a'}$ with the world coordinate Σ_W. Since the initial local coordinate system Σ_a axis orientations are identical to those of Σ_W, their conversion into each other is super easy and can be done just by shifting the origins to a new position without any rotation transformation. Next, we will focus on the rotation transformation from $\Sigma_{a'}$ to Σ_W, which will be identical to the rotation transformation from $\Sigma_{a'}$ to Σ_a.

By projecting the x, y and z axes of $\Sigma_{a'}$ to Σ_W, the unit vectors parallel to the x, y and z axes of $\Sigma_{a'}$ within the world coordinate Σ_W can be represented as

$$\mathbf{e}_x^{[\Sigma_{a'} \to \Sigma_W]} = (1, 0, 0)^\top \ , \ \mathbf{e}_y^{[\Sigma_{a'} \to \Sigma_W]} = (0, \cos\theta, -\sin\theta)^\top \ , \ \mathbf{e}_z^{[\Sigma_{a'} \to \Sigma_W]} = (0, \sin\theta, \cos\theta)^\top. \tag{8}$$

Definition 5 *(**Rotation Matrix**): Based on the above unit vectors projected from $\Sigma_{a'}$ to Σ_W, we can define the ration matrix from $\Sigma_{a'}$ to Σ_W as:*

$$\mathbf{R}^{[\Sigma_{a'} \to \Sigma_W]} = (\mathbf{e}_x^{[\Sigma_{a'} \to \Sigma_W]}, \mathbf{e}_y^{[\Sigma_{a'} \to \Sigma_W]}, \mathbf{e}_z^{[\Sigma_{a'} \to \Sigma_W]}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}. \tag{9}$$

With the *rotation matrix* $\mathbf{R}^{[\Sigma_a \to \Sigma_W]}$, we can actually project the arm link vector pointing from the joint a to joint b from the local coordinate $\Sigma_{a'}$ to its representation in the world coordinate Σ_W:

$$\underbrace{\begin{bmatrix} 0 \\ l_2 \sin\theta \\ l_2 \cos\theta \end{bmatrix}}_{\mathbf{p}_{a \to b}^{[\Sigma_W]}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}}_{\mathbf{R}^{[\Sigma_{a'} \to \Sigma_W]}} \underbrace{\begin{bmatrix} 0 \\ 0 \\ l_2 \end{bmatrix}}_{\mathbf{p}_{a \to b}^{[\Sigma_{a'}]}}. \tag{10}$$

3.1.2 Homogeneous Transformation Matrix

Next, let's have a look at the vector pointing from the world coordinate system origin to the robot arm joint b after the rotation. Looking from the world coordinate, we can represent the vector as

$$\mathbf{p}_b^{[\Sigma_W]} = \mathbf{p}_a^{[\Sigma_W]} + \mathbf{p}_{a \to b}^{[\Sigma_W]} \tag{11}$$

$$= \mathbf{p}_a^{[\Sigma_W]} + \mathbf{R}^{[\Sigma_{a'} \to \Sigma_W]} \mathbf{p}_{a \to b}^{[\Sigma_{a'}]}. \tag{12}$$

Linear algebra will be frequently used in representing robots in this article, and the above formula can also be rewritten as follows instead:

$$
\begin{bmatrix} \mathbf{p}_b^{[\Sigma_{\mathrm{w}}]} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{[\Sigma_{a'} \to \Sigma_{\mathrm{w}}]} & \mathbf{p}_a^{[\Sigma_{\mathrm{w}}]} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{a \to b}^{[\Sigma_{a'}]} \\ 1 \end{bmatrix}
\tag{13}
$$

Some extra entries with dummy values (0 or 1) are added to the vectors and matrices in the above formula just to make their dimensions match with each other.

Definition 6 *(**Homogeneous Transformation Matrix**): In the above equation, we introduce a new matrix representation composed of the position vector $\mathbf{p}_a$ and the rotation matrix $\mathbf{R}_{a'}$, which is also named as the homogeneous transformation matrix, i.e.,*

$$
\mathbf{T}^{[\Sigma_{a'} \to \Sigma_{\mathrm{w}}]} = \begin{bmatrix} \mathbf{R}^{[\Sigma_{a'} \to \Sigma_{\mathrm{w}}]} & \mathbf{p}_a^{[\Sigma_{\mathrm{w}}]} \\ \mathbf{0} & 1 \end{bmatrix}
\tag{14}
$$

The robot *homogeneous transformation matrix* $\mathbf{T}^{[\Sigma_{a'} \to \Sigma_{\mathrm{w}}]}$ will convert all points described in the robot arm's local coordinate to the world coordinate. Therefore, we can generally say that the *homogeneous transformation matrix* $\mathbf{T}^{[\Sigma_{a'} \to \Sigma_{\mathrm{w}}]}$ describes the position and state of the robot arm. Since the robot rotation transformation is very common, to simplify the notations, we can just use Σ_a to represent all the coordinate systems with origin at joint a (including Σ_a and $\Sigma_{a'}$ discussed before). As to Σ_a is the coordinate system before or after the rotation, readers can differentiate them from the context by yourself.

3.2 Multi-Link Robot End Point Position

Real-world robots are usually composed of multiple links connected to each other by the joints. At the end of the first section, we will take a more general robot with two links to study the local coordinate transformation via the chain rule.

3.2.1 MULTI-LINK ROBOTS WITH LOCAL TO LOCAL COORDINATE SYSTEMS

As illustrated in Figure 6, we rotates both of the two arm links of the robot shown in Figure 4 with certain degrees: the first robot arm rotates clockwise with an angle of θ degrees around joint a, while the second link rotates clockwise with an angle ϕ degrees around joint b. At these two joints, we introduce two local coordinate systems Σ_a and Σ_b, and these two arm links are defined as vectors $\mathbf{p}_{a \to b}^{[\Sigma_a]}$ and $\mathbf{p}_{b \to e}^{[\Sigma_b]}$ within these two local coordinate systems, respectively.

According to the analysis in the previous parts, by taking Σ_a as the "world coordinate", we can represent the rotation matrix to project coordinate Σ_b to Σ_a as

$$
\mathbf{R}^{[\Sigma_b \to \Sigma_a]} = (\mathbf{e}_x^{[\Sigma_b \to \Sigma_a]}, \mathbf{e}_y^{[\Sigma_b \to \Sigma_a]}, \mathbf{e}_z^{[\Sigma_b \to \Sigma_a]}),
\tag{15}
$$

where the unit vectors are defined as follows:

$$
\mathbf{e}_x^{[\Sigma_b \to \Sigma_a]} = (1,0,0)^\top; \ \mathbf{e}_y^{[\Sigma_b \to \Sigma_a]} = (0, \cos\phi, -\sin\phi)^\top; \ \mathbf{e}_z^{[\Sigma_b \to \Sigma_a]} = (0, \sin\phi, \cos\phi)^\top.
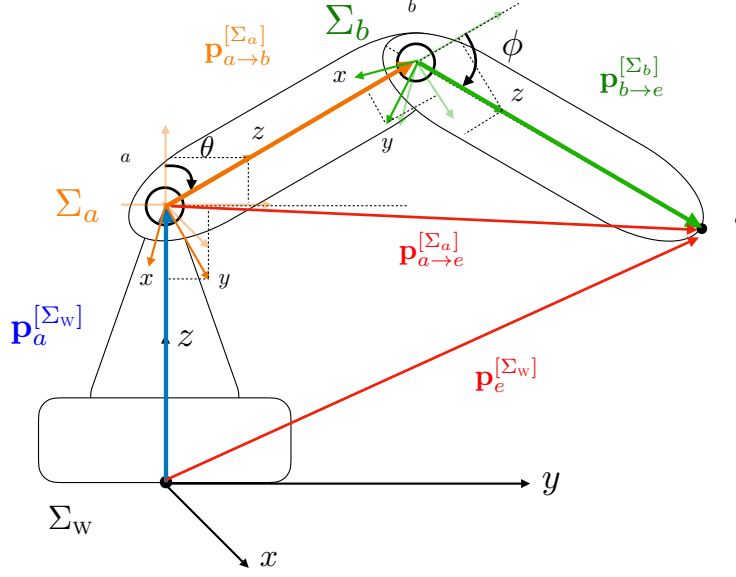\tag{16}
$$

Figure 6: A Example of Multi-Link Robot Arm Coordinate Transformation via Chain Rule.

Meanwhile, for the end point of the second arm link, we can represent its position in the coordinate Σ_a as $\mathbf{p}_{a \to e}^{[\Sigma_a]}$, whose representation after the rotation can be represented as

$$\begin{bmatrix} \mathbf{p}_{a \to e}^{[\Sigma_a]} \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} & \mathbf{p}_{a \to b}^{[\Sigma_a]} \\ \mathbf{0} & 1 \end{bmatrix}}_{\mathbf{T}^{[\Sigma_b \to \Sigma_a]}} \begin{bmatrix} \mathbf{p}_{b \to e}^{[\Sigma_b]} \\ 1 \end{bmatrix}. \tag{17}$$

The matrix that accomplishes the transformation denotes the *homogeneous transformation matrix* $\mathbf{T}^{[\Sigma_b \to \Sigma_a]}$ from coordinate Σ_b to Σ_a.

3.2.2 CHAIN RULE ON HOMOGENEOUS TRANSFORMATION

By combining Equation 13 and Equation 17, we can further derive the end point's position of the second link in the world coordinate to be

$$\begin{bmatrix} \mathbf{p}_e^{[\Sigma_w]} \\ 1 \end{bmatrix} = \mathbf{T}^{[\Sigma_a \to \Sigma_w]} \begin{bmatrix} \mathbf{p}_{a \to e}^{[\Sigma_a]} \\ 1 \end{bmatrix} = \mathbf{T}^{[\Sigma_a \to \Sigma_w]} \mathbf{T}^{[\Sigma_b \to \Sigma_a]} \begin{bmatrix} \mathbf{p}_{b \to e}^{[\Sigma_b]} \\ 1 \end{bmatrix}. \tag{18}$$

On the right-hand side of the equation, we multiply the two *homogeneous transformation matrices* $\mathbf{T}^{[\Sigma_a \to \Sigma_w]}$ and $\mathbf{T}^{[\Sigma_b \to \Sigma_a]}$ together to transform the local vector representation $\mathbf{p}_{b \to e}^{[\Sigma_b]}$ in coordinate Σ_b to the world coordinate $\mathbf{p}_e^{[\Sigma_w]}$. Formally, we can also represent the multiplication of $\mathbf{T}^{[\Sigma_a \to \Sigma_w]} \mathbf{T}^{[\Sigma_b \to \Sigma_a]}$ as the *homogeneous transformation matrix* from the local coordinate Σ_b to the world coordinate Σ_w:

$$\mathbf{T}^{[\Sigma_b \to \Sigma_w]} = \mathbf{T}^{[\Sigma_a \to \Sigma_w]} \mathbf{T}^{[\Sigma_b \to \Sigma_a]}, \tag{19}$$

and the Equation 18 can also be rewritten as

$$
\begin{bmatrix} \mathbf{p}_e^{[\Sigma_\mathrm{w}]} \\ 1 \end{bmatrix} = \mathbf{T}^{[\Sigma_b \to \Sigma_\mathrm{w}]} \begin{bmatrix} \mathbf{p}_{b \to e}^{[\Sigma_b]} \\ 1 \end{bmatrix}. \tag{20}
$$

For the robot arms with more than two links, the end point's position can be derived in a similar way with the chain rule on the multiple homogeneous transformation matrices between coordinate systems. By now, we should have provide a detailed introduction on multi-link rigid-body robot end point position and transformation across coordinate systems for the readers already. In the next section, we will discuss more about the robot rotational motion and its relationship with the robot velocity for the readers.

4. Robot Rotation

In the previous section, we have introduced the robot coordinate systems and representations. We have also briefly talk about the robot arm rotation, as well as the robot position and state changes due to the rotation movement. In this section, we will provide a more detailed analysis about the robot's general rotational motion and illustrate several other important properties of the rotation matrix defined before.

4.1 General Rotation

The rotation movements we introduce in the previous section are around one certain axis (e.g., the x axis) and the rotation direction is clockwise. In this part, we will introduce a more general robot rotation around any axes with either clockwise and counter-clockwise directions.

4.1.1 ROTATION AXIS

Depending on the orientation of the joints, robots can rotate around the joints not only along the x, y and z axes but also a combination of them, which will lead to the rotation in any directions in the 3D space.

Definition 7 (Roll, Pitch and Yaw): *Specifically, for the rotations around the x, y and z axes, we can give them different names to describe the rotation:*

- *Roll: The rotation movement along the x axis is also called roll.*

- *Pitch: The rotation movement along the y axis is also called pitch.*

- *Yaw: The rotation movement along the z axis is also called yaw.*

In the roll rotational motion, for all the points in the robot body, their x coordinate is not changed, while their y and z coordinates will change together with the rotation. Meanwhile, for the pitch rotation, all the x and z coordinates will change due to the rotation. The yaw rotation will change the x and y coordinates instead.

Readers may also wonder why the rotation along these axes are named as *roll*, *pitch* and *yaw*, respectively. These movement names are actually borrowed from the airplane control.
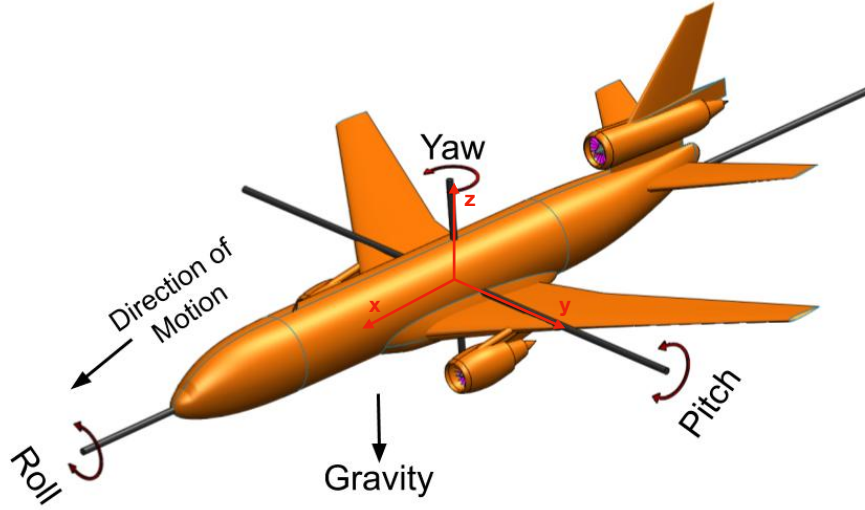
14

Figure 7: A Example of Roll, Pitch and Yaw Rotation Movements.

As illustrated in Figure 7, given a airplane, based on the coordinate system definition provided in Section 2.3.1, we can introduce a coordinate system along the plane body (x axis faces forward, y axis to the airplane left and z axis faces up).

If the readers having experiences in driving airplane, the two *ailerons* on the outer rear edge of each wing cause the plane to roll to the left or right. The *elevator* on the horizontal tail surface controls the pitch movement of the plane. Meanwhile, the rudder on the vertical tail fin controls swivels from side to side, which will push the tail in a left or right direction and control the *yaw* movement of the plane. Similar names of the motions are also used to describe the robot rotation here.

4.1.2 ROTATION MATRICES FOR ROLL, PITCH AND YAW

For the *roll*, *pitch* and *yaw* rotational motions, to differentiate them from each other, we can denote their rotation angles as ϕ, θ and ψ, respectively. Here, we assume the rotations are all in the counter-clockwise direction (i.e., the positive angles). Based on the degrees and the representations we learn from Section 3.1.1, regardless of the coordinate systems,

we can represent the corresponding rotation matrices for *roll*, *pitch* and *yaw* as follows:

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}, \tag{21}$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \tag{22}$$

$$\mathbf{R}_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\theta & 0 \\ \sin\psi & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{23}$$

$$\tag{24}$$

Let's assume, given a vector $\mathbf{p}$ pointing from the current joint to the end point, if we rotate the current joint with a ϕ-degree angle around the x axis, a θ-degree angle around the y axis and a ψ-degree angle around the z axis, we can represent the vector after those 3-way rotations as

$$\mathbf{p}' = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)\mathbf{p} = \mathbf{R}_{rpy}(\phi, \theta, \psi)\mathbf{p}, \tag{25}$$

where the *roll-pitch-yaw* rotation matrix $\mathbf{R}_{rpy}(\phi, \theta, \psi)$ can be represented as follows:

$$\mathbf{R}_{rpy}(\phi, \theta, \psi) = \tag{26}$$

$$\begin{bmatrix} \cos\psi\cos\theta & -\sin\psi\cos\phi + \cos\psi\sin\theta\sin\phi & \sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi \\ \sin\psi\cos\theta & \cos\psi\cos\phi + \sin\psi\sin\theta\sin\phi & -\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{27}$$

The readers may have notice the rotation matrix $\mathbf{R}_x(\phi)$ defined in Equation 21 is slightly different from the rotation matrix defined in Equation 9. The main differences lie at the rotation direction, which actually follows the right-hand rule. With the right hand index finger, middle finger, ring finger and little finger following the rotation direction, if the thumb orientation is the same as the axis positive direction, then the rotation angle will be positive; otherwise, the rotation angle will be negative.

According to the right-hand rule, for the rotations in the counter-clockwise direction, their rotation matrices can be simply represented as those in Equation 21. Meanwhile, when it comes to the clockwise rotation (just like Figure 5), we can add a negative mark to the angles before applying them in Equation 21 for defining the rotation matrices, which will lead to the rotation matrix representation in Equation 9.

With the *roll-pitch-yaw* rotation matrix $\mathbf{R}_{rpy}(\phi, \theta, \psi)$ defined above, arbitrary rotation can be achieved by the rotation angle pair (ϕ, θ, ψ) around the x, y and z axes, respectively.

4.2 Rotation Matrix Property

Theorem 8 *Given a rotation matrix $\mathbf{R}$, it will be orthogonal, i.e.,*

$$\mathbf{R}\mathbf{R}^\top = \mathbf{I}. \tag{28}$$

Proof When deriving the *rotation matrix* in Equation 9, we mention that the rotation matrix can be represented as a group of unit vectors corresponding to the x, y and z axes. For instance, the rotation matrix along the x axis in the counter-clockwise direction with an angle of ϕ degrees can be represented as

$$\mathbf{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} = (\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z). \tag{29}$$

We can observe that the product of $\mathbf{R}_x(\phi)^\top \mathbf{R}_x(\phi)$ will lead to an identity matrix, since

$$\mathbf{R}_x(\phi)^\top \mathbf{R}_x(\phi) = \begin{bmatrix} \mathbf{e}_x^\top \\ \mathbf{e}_y^\top \\ \mathbf{e}_z^\top \end{bmatrix} (\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z) \tag{30}$$

$$= \begin{bmatrix} \mathbf{e}_x^\top \mathbf{e}_x & \mathbf{e}_x^\top \mathbf{e}_y & \mathbf{e}_x^\top \mathbf{e}_z \\ \mathbf{e}_y^\top \mathbf{e}_x & \mathbf{e}_y^\top \mathbf{e}_y & \mathbf{e}_y^\top \mathbf{e}_z \\ \mathbf{e}_z^\top \mathbf{e}_x & \mathbf{e}_z^\top \mathbf{e}_y & \mathbf{e}_z^\top \mathbf{e}_z \end{bmatrix} \tag{31}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{32}$$

For the rotation matrices along the y and z axes, it is also easy to obtain that

$$\mathbf{R}_y(\theta)^\top \mathbf{R}_y(\theta) = \mathbf{I} \in \mathbb{R}^{3 \times 3}, \mathbf{R}_z(\psi)^\top \mathbf{R}_z(\psi) = \mathbf{I} \in \mathbb{R}^{3 \times 3}. \tag{33}$$

How about the *roll-pitch-yaw* rotation matrix $\mathbf{R}_{rpy}(\phi, \theta, \psi)$ introduced in Equation 26? Since $\mathbf{R}_{rpy}(\phi, \theta, \psi) = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)$, we can get

$$\mathbf{R}_{rpy}(\phi, \theta, \psi)^\top \mathbf{R}_{rpy}(\phi, \theta, \psi) \tag{34}$$

$$= (\mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi))^\top (\mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)) \tag{35}$$

$$= \mathbf{R}_x(\phi)^\top \mathbf{R}_y(\theta)^\top \mathbf{R}_z(\psi)^\top \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi) \tag{36}$$

$$= \mathbf{R}_x(\phi)^\top \left(\mathbf{R}_y(\theta)^\top \left(\mathbf{R}_z(\psi)^\top \mathbf{R}_z(\psi) \right) \mathbf{R}_y(\theta) \right) \mathbf{R}_x(\phi) \tag{37}$$

$$= \mathbf{I}. \tag{38}$$

Meanwhile, readers can also just calculate the product of $\mathbf{R}_{rpy}(\phi, \theta, \psi)^\top \mathbf{R}_{rpy}(\phi, \theta, \psi)$ based on the concrete representations shown in Equation 26 on a piece of paper, whose result should also be a identity matrix with value 1 on the diagonal. ∎

Here, if we assume the rotation matrix $\mathbf{R}$ is invertible, by multiplying both sides of equation $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$ with $\mathbf{R}^{-1}$, we can obtain

$$\mathbf{R}^\top = \mathbf{R}^{-1}. \tag{39}$$

4.3 Angular Velocity

Before we talk more about the rotation matrix, we would like to have a cut-in subsection here to talk about the robot rotation *angular velocity* first. More information about velocity and transformation will be provided in the following Section 5.
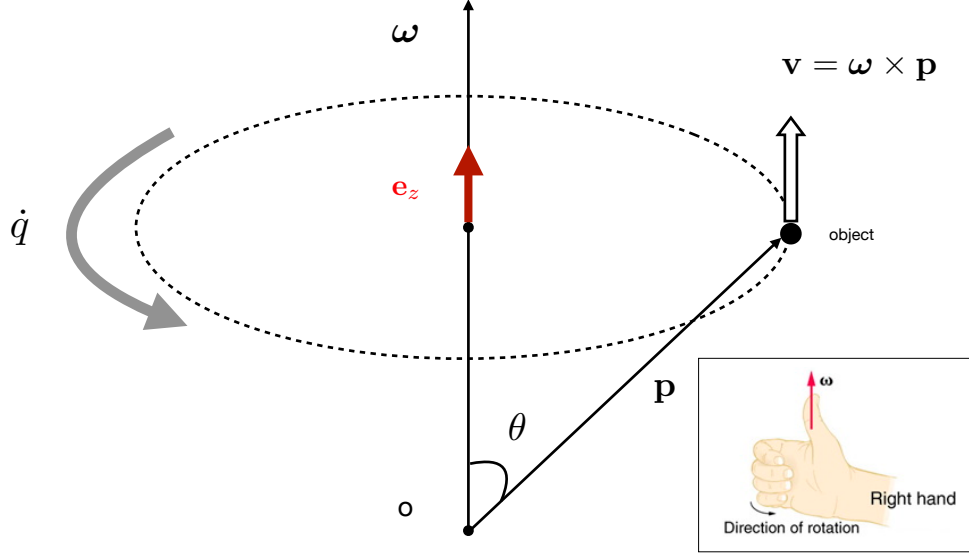
Figure 8: Angular Velocity Vector (Right Hand Rule).

4.3.1 ANGULAR VELOCITY VECTOR

As shown in Figure 8, given an object rotating in the counter clockwise direction around the z axis, according to the right-hand rule, we can represent the angular velocity vector orientation by the unit vector $\mathbf{e}_z$. Meanwhile, the rotation speed is denoted by a scalar $\dot{q} = 1$ rad/s. For the angular velocity unit "rad/s", it is equivalent to an ordinary frequency of $\frac{1}{2\pi}$ hertz or cycles per second. So, if an object rotation angular velocity is 2π rad/s, it will rotate 1 cycle per second.

Some Remarks: In this article, we like to use a regular scalar q or vector $\mathbf{p}$ to represent the angular and position of an object, and use $\dot{q}$ and $\dot{\mathbf{p}}$ to represent its first-order derivatives (*i.e.*, velocity), and $\ddot{q}$, $\ddot{\mathbf{p}}$ to represent the second-order derivatives (*i.e.*, acceleration).

In the 3D space, similar to the velocity and positions, the object rotation angular velocity can also be represented as a vector. For this example, based on the angular velocity direction unit vector $\mathbf{e}_z$ and rotation velocity $\dot{q}$, we can represent the object rotation angular velocity can be represented as:

$$\boldsymbol{\omega} = \dot{q}\mathbf{e}_z = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{40}$$

where each element has the unit of rad/s.

4.3.2 LINEAR VELOCITY VECTOR

In Figure 8, we also show the object position, which is denoted by a vector $\mathbf{p}$ from the origin pointing to its current position, and the angle between $\mathbf{p}$ and the rotation axis $\boldsymbol{\omega}$ is
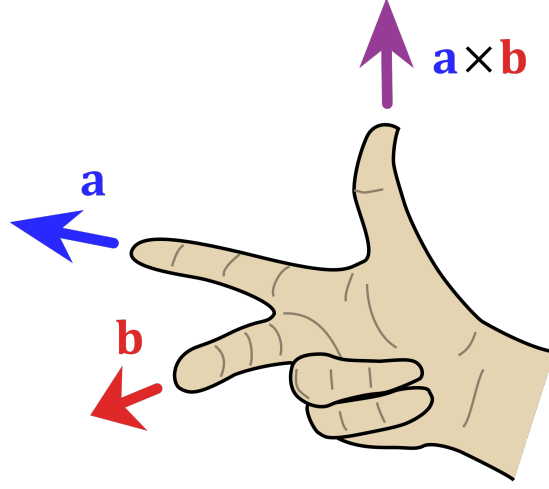
Figure 9: The Right-Hand Rule Indicating the Direction of the Cross Product Result Vector. (Given two vectors **a** and **b** represented by the index finger and middle finger, their cross-product result **a**×**b** direction is represented by the thumb finger, which is perpendicular to both **a** and **b**.)

θ. For the object in rotation, its *linear velocity* can be represented as a vector denoted as the *cross product* of its *angular velocity vector* and its current *position vector*.

For the object with angular velocity $\boldsymbol{\omega}$ and the position vector $\mathbf{p}$, we can represent its *linear velocity vector* as

$$\mathbf{v} = \boldsymbol{\omega} \times \mathbf{p}. \tag{41}$$

Background Knowledge: Given two vectors **a** and **b**, their cross product result $\mathbf{a} \times \mathbf{b}$ will also be a vector that it perpendicular to both **a** and **b**. To determine the direction of the result vector, as indicated in Figure 9, it also follows the right hand rule. We just need to place the index finger to vector **a** and middle finger to vector **b**, then the thumb finger will be the direction of $\mathbf{a} \times \mathbf{b}$. Following this rule, we can also observe that $\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$.

With the right hand rule introduced above, we can decide the direction of the *linear velocity vector* **v**, and observe that $\mathbf{v} \perp \boldsymbol{\omega}$ and $\mathbf{v} \perp \mathbf{p}$. As to the length of vector **v**, it can be represented to be

$$\|\mathbf{v}\| = \|\boldsymbol{\omega}\| \, \|\mathbf{p}\| \sin\theta. \tag{42}$$

If we know the elements of the vectors $\boldsymbol{\omega}$ and $\mathbf{p}$, we can also represent their cross product to be

$$\boldsymbol{\omega} \times \mathbf{p} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \omega_y p_z - \omega_z p_y \\ \omega_z p_x - \omega_x p_z \\ \omega_x p_y - \omega_y p_x \end{bmatrix}. \tag{43}$$

19

We observe that the result vector of $\boldsymbol{\omega} \times \mathbf{p}$ can actually also be obtained via a matrix-vector multiplication shown as follows:

$$
\begin{bmatrix} \omega_y p_z - \omega_z p_y \\ \omega_z p_x - \omega_x p_z \\ \omega_x p_y - \omega_y p_x \end{bmatrix} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}. \tag{44}
$$

In the following subsection, we will illustrate the physical meanings of the matrix multiplied with the vector $\mathbf{p}$, which can be build from the angular velocity vector directly.

4.3.3 Rotation Transformation on Velocity Vectors

If the origin in Figure 8 denotes a robot joint, which may rotate with time, the object's *angular velocity vector* together with the central axis will also change with the rotation. Here, we can represent the rotation as matrix $\mathbf{R}$. By multiplying both sides of Equation 40 with matrix $\mathbf{R}$, we can get

$$
\mathbf{R}\boldsymbol{\omega} = \mathbf{R}\dot{q}\mathbf{e}_z. \tag{45}
$$

If we introduce two new vectors $\boldsymbol{\omega}' = \mathbf{R}\boldsymbol{\omega}$ and $\mathbf{e}'_z = \mathbf{R}\mathbf{e}_z$, then the above equation can be rewritten as follows:

$$
\boldsymbol{\omega}' = \dot{q}\mathbf{e}'_z, \tag{46}
$$

where $\boldsymbol{\omega}'$ denotes the new angular velocity vector and $\mathbf{e}'_z$ is the new axis. The velocity scalar value is still $\dot{q}$ rad/s/. Similarly, by introducing another new vector $\mathbf{v}' = \mathbf{R}\mathbf{v}$ and a new position vector $\mathbf{p}' = \mathbf{R}\mathbf{p}$, we can also represent the object's linear velocity vector after the rotation as

$$
\mathbf{v}' = \mathbf{R}\mathbf{v} = \mathbf{R}(\boldsymbol{\omega} \times \mathbf{p}) = (\mathbf{R}\boldsymbol{\omega}) \times (\mathbf{R}\mathbf{p}) = \boldsymbol{\omega}' \times \mathbf{p}'. \tag{47}
$$

These above velocity transformation equations and analysis results will be frequently used in calculating the robot end point velocity in the following Section 5.

4.3.4 Rotation Matrix and Angular Velocity Vector

Before we have introduced that via the rotation matrix, we can map a robot arm link from a local coordinate system, *e.g.*, $\bar{\mathbf{p}}$, to the world coordinate system as follows:

$$
\mathbf{p} = \mathbf{R}\bar{\mathbf{p}}. \tag{48}
$$

For robot arm, its link vector representation $\bar{\mathbf{p}}$ within the local coordinate system is static and has a fixed representation, which can be denoted as

$$
\mathbf{R}^\top \mathbf{p} = \mathbf{R}^\top \mathbf{R}\bar{\mathbf{p}} \Rightarrow \bar{\mathbf{p}} = \mathbf{R}^\top \mathbf{p}. \tag{49}
$$

Meanwhile, both $\mathbf{p}$ and $\mathbf{R}$ will change with time. If we take the derivatives of $\mathbf{p}$ and $\mathbf{R}$ with regard to time, we can obtain

$$
\dot{\mathbf{p}} = \dot{\mathbf{R}}\bar{\mathbf{p}} = \dot{\mathbf{R}} \left(\mathbf{R}^\top \mathbf{p} \right) = \left(\dot{\mathbf{R}}\mathbf{R}^\top \right) \mathbf{p}, \tag{50}
$$

where $\dot{\mathbf{p}} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t}$ and $\dot{\mathbf{R}} = \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}t}$ represents the changing rate of the object's position and its rotation matrix.

Meanwhile, for an object, its position changing rate is actually its velocity, *i.e.*,

$$\dot{\mathbf{p}} = \mathbf{v} = \boldsymbol{\omega} \times \mathbf{p}, \tag{51}$$

which will lead to an important equation shown as follows

$$\boldsymbol{\omega} \times \mathbf{p} = \left(\dot{\mathbf{R}} \mathbf{R}^\top \right) \mathbf{p}. \tag{52}$$

The above equation correlates the object's *angular velocity vector* $\boldsymbol{\omega}$ and *position vector* $\mathbf{p}$ with its corresponding rotation matrix $\dot{\mathbf{R}} \mathbf{R}^\top$.

Theorem 9 *Matrix* $\dot{\mathbf{R}} \mathbf{R}^\top$ *is skew symmetric, i.e.,*

$$(\dot{\mathbf{R}} \mathbf{R}^\top)^\top = -\dot{\mathbf{R}} \mathbf{R}^\top. \tag{53}$$

Proof We can prove it based on the what we learn from the previous Section 4.2 that matrix $\mathbf{R}$ is *orthogonal*, *i.e.*, $\mathbf{R} \mathbf{R}^\top = \mathbf{I}$. By taking the derivative of both sides of $\mathbf{R}$ is *orthogonal*, *i.e.*, $\mathbf{R} \mathbf{R}^\top = \mathbf{I}$ with time, we can get

$$\dot{\mathbf{R}} \mathbf{R}^\top + \mathbf{R} \dot{\mathbf{R}}^\top = \mathbf{0} \Rightarrow (\dot{\mathbf{R}} \mathbf{R}^\top)^\top = -\dot{\mathbf{R}} \mathbf{R}^\top, \tag{54}$$

which concludes the proof that $\dot{\mathbf{R}} \mathbf{R}^\top$ is *skew symmetric*. ∎

The readers may also wonder how can we represent $\dot{\mathbf{R}} \mathbf{R}^\top$? Actually, in Equation 43, we have provided one potential representation of matrix $\dot{\mathbf{R}} \mathbf{R}^\top$ with elements form the *angular velocity vector* $\boldsymbol{\omega}$ already, *i.e.*,

$$\boldsymbol{\omega} \times \mathbf{p} = \underbrace{\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}}_{\dot{\mathbf{R}} \mathbf{R}^\top} \underbrace{\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}}_{\mathbf{p}}. \tag{55}$$

Definition 10 *(Projection Operators): To simplify the notations, we can also introduce two operators* $\wedge : \boldsymbol{\omega} \to \dot{\mathbf{R}} \mathbf{R}^\top$ *and* $\vee : \dot{\mathbf{R}} \mathbf{R}^\top \to \boldsymbol{\omega}$.

- *Given an angular velocity vector* $\boldsymbol{\omega}$*, we can represent the corresponding matrix* $\dot{\mathbf{R}} \mathbf{R}^\top$ *projected from it as*

$$\dot{\mathbf{R}} \mathbf{R}^\top = \boldsymbol{\omega}^\wedge, \text{ or } \dot{\mathbf{R}} \mathbf{R}^\top = \widehat{\boldsymbol{\omega}}. \tag{56}$$

- *Given a skew symmetric matrix* $\dot{\mathbf{R}} \mathbf{R}^\top$*, we can also represent the corresponding angular velocity vector* $\boldsymbol{\omega}$ *constructed from it as*

$$\boldsymbol{\omega} = (\dot{\mathbf{R}} \mathbf{R}^\top)^\vee. \tag{57}$$

Based on the above analysis, we can also rewrite Equation 52 as follows:

$$\boldsymbol{\omega} \times \mathbf{p} = \widehat{\boldsymbol{\omega}} \mathbf{p}, \tag{58}$$

which can calculate the vector cross-product with the regular matrix-vector multiplication.

4.4 Rotation Matrix Exponential and Logarithm

As robot arm rotates with angular velocity, it will change the arm's position with the rotation matrix. It seems both the angular velocity vector $\boldsymbol{\omega}$ and the rotation matrix $\mathbf{R}$ are describing the effects of the robot rotation motion. Then, given the angular velocity vector $\boldsymbol{\omega}$, can we directly calculate the corresponding rotation matrix $\mathbf{R}$? And given the rotation matrix $\mathbf{R}$, can we calculate the corresponding angular velocity vector $\boldsymbol{\omega}$?

To answer these two questions, at the end of this section, we will introduce two important operators defined on the rotation matrix $\mathbf{R}$, *i.e.*, the *rotation matrix exponential* and *rotation matrix logarithm*.

4.4.1 ROTATION MATRIX EXPONENTIAL

For Equation 56, if we multiply both sides with the rotation matrix $\mathbf{R}$, we can get

$$\widehat{\boldsymbol{\omega}}\mathbf{R} = \dot{\mathbf{R}}\underbrace{\mathbf{R}^\top \mathbf{R}}_{=\mathbf{I}} = \dot{\mathbf{R}}. \tag{59}$$

From the calculus course we learn at college, for a function $f(x)$, if $f(x)' = af(x)$, then the function $f(x)$ can be represented as

$$f(x) = \exp^{ax} f(0). \tag{60}$$

Similar representation can also be applied to represent the matrix $\mathbf{R}$ subject to Equation 59, we can denote the matrix $\mathbf{R}$ as

$$\mathbf{R}(t) = \exp^{\widehat{\boldsymbol{\omega}}t} \mathbf{R}(0). \tag{61}$$

In the above equation, we clearly specify the matrix $\mathbf{R}$ has variable t and $\mathbf{R}(0)$ denotes the representation of matrix $\mathbf{R}$ at timestamp $t = 0$. If we assume the initial condition $\mathbf{R}(0) = \mathbf{I}$, the above representation can be further simplified as

$$\mathbf{R}(t) = \exp^{\widehat{\boldsymbol{\omega}}t}. \tag{62}$$

Meanwhile, let's come back to the knowledge we learn from the calculus course, according to the Taylor series, the exponential term $\exp^{ax}$ we use to represent the $f(x)$ function can actually be calculated as the sum of a sequence of polynomial terms:

$$\exp^{ax} = 1 + ax + \frac{(ax)^2}{2!} + \frac{ax)^3}{3!} + \cdots \tag{63}$$

Therefore, we can further represent the matrix $\mathbf{R}(t)$ shown above as the sum of a sequence of matrix polynomial terms

$$\mathbf{R}(t) = \mathbf{I} + \widehat{\boldsymbol{\omega}}t + \frac{(\widehat{\boldsymbol{\omega}}t)^2}{2!} + \frac{(\widehat{\boldsymbol{\omega}}t)^3}{3!} + \cdots . \tag{64}$$

Theorem 11 *Given the angular velocity vector $\boldsymbol{\omega}$, it can be represented as $\boldsymbol{\omega} = \dot{q}\mathbf{e}$, where $\dot{q}$ is a scalar denotes the rotation velocity and $\mathbf{e}$ is a unit vector representing the angular velocity direction. Then we can have $\widehat{\boldsymbol{\omega}} = \dot{q}\widehat{\mathbf{e}}$. For the unit direction vector $\mathbf{e}$, subject to the $\wedge$ operator, we have*

$$\widehat{\mathbf{e}}^{2n+1} = (-1)^n \cdot \widehat{\mathbf{e}}, \tag{65}$$

$$\widehat{\mathbf{e}}^{2n+2} = (-1)^n \cdot \widehat{\mathbf{e}}^2. \tag{66}$$

We will leave the proof of the above theorem as an exercise for the readers, which can be done with the mathematical induction on n starting from $n = 0$.

Based on the above theorem, we can further simplify the Equation 64, the high-order power of the matrix $\widehat{\boldsymbol{\omega}}$ in the equation can be reduced to the sum of $\widehat{\mathbf{e}}$ and $\widehat{\mathbf{e}}^2$:

$$\mathbf{R}(t) = \mathbf{I} + \left((\dot{q}t) - \frac{(\dot{q}t)^3}{3!} + \frac{(\dot{q}t)^5}{5!} - \cdots \right) \widehat{\mathbf{e}} + \left(\frac{(\dot{q}t)^2}{2!} - \frac{(\dot{q}t)^4}{4!} + \frac{(\dot{q}t)^6}{6!} - \cdots \right) \widehat{\mathbf{e}}^2. \tag{67}$$

If the readers still remember the Taylor series of $\sin\theta$ and $\cos\theta$, you will observe that the above representation of matrix $\mathbf{R}(t)$ can be further simplified as follows:

$$\mathbf{R}(t) = \mathbf{I} + \widehat{\mathbf{e}}\sin\theta + \widehat{\mathbf{e}}^2(1 - \cos\theta), \tag{68}$$

where $\theta = \dot{q}t$ is introduced to further simplify the representations. This equation will be frequently used in robot control and kinematics to be introduced in the following articles.

4.4.2 Rotation Matrix Logarithm

Based on the analysis in the above subsection, given the object angular velocity vector $\widehat{\boldsymbol{\omega}}$, we can directly calculate its corresponding rotation matrix $\mathbf{R}$ with Equation 68. Meanwhile, when given a rotation matrix $\mathbf{R}$, can we also directly calculate its corresponding angular velocity vector or not? This is what we plan to introduce in this part.

According to Equation 62, the rotation matrix $\mathbf{R}$ can be represented as $\exp^{\widehat{\boldsymbol{\omega}}}$ (the time variable t is discarded). By defining the logarithm operator on the rotation matrix, we can calculate the angular velocity vector $\boldsymbol{\omega}$ to be

$$\boldsymbol{\omega} = (\widehat{\boldsymbol{\omega}})^\vee = (\ln\mathbf{R})^\vee. \tag{69}$$

Depending on the shape and contents of the rotation matrix $\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$, the actual calculation can be represented as

$$\boldsymbol{\omega} = (\ln\mathbf{R})^\vee = \begin{cases} (0, 0, 0)^\top & \text{if } \mathbf{R} = \mathbf{I} \\ \frac{\pi}{2}(r_{11} + 1, r_{22} + 1, r_{33} + 1,) & \text{if } \mathbf{R} \text{ is diagonal} \\ q\frac{\mathbf{a}}{\|\mathbf{a}\|} & \text{otherwise.} \end{cases} \tag{70}$$

In the above representation, the scalar q and vector $\mathbf{a}$ are defined based on the elements in matrix $\mathbf{R}$ as follows:

$$\mathbf{a} = (r_{32} - r_{23}, r_{13} - r_{31}, r_{21} - r_{12})^\top, q = \text{atan2}(\|\mathbf{a}\|, r_{11} + r_{22} + r_{33} - 1), \tag{71}$$
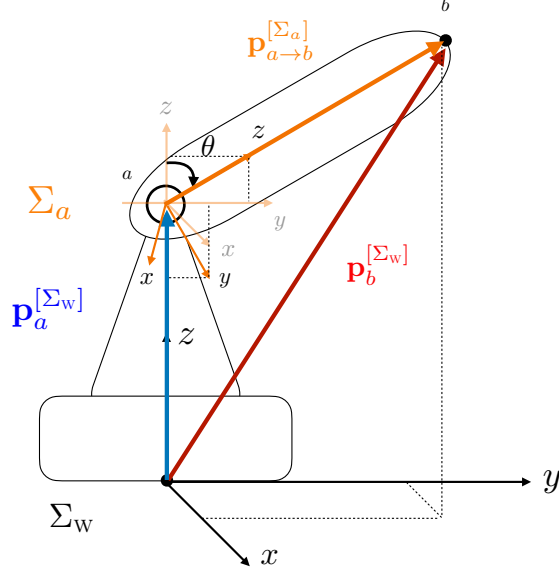
Figure 10: An Example of Single-Link Robot End Point Velocity.

where atan2(y, x) is the 2-argument arctangent function and it returns the angle in radians between the vector (x, y) (composed with the input arguments) and the x axis in the 2D Cartesian plane.

5. Robot End Point Velocity

In the previous Section 4.3, we have introduce the object rotation *angular velocity* and *linear velocity*. For an object rotating around the axis, we have illustrated their relationships and also discussed about the transformation on them by the rotation movement.

In this section, we will further discuss about the *angular velocity* and *linear velocity* about the end point of a robot during the rotational motion.

5.1 Single-Link Robot End Point Linear Velocity

We can borrow the example we use in the previous Section 2 here again. As shown in Figure 10, given the robot arm with one movable link, we can represent its end point in the world coordinate system as

$$\mathbf{p}_b^{[\Sigma_{\mathrm{w}}]} = \mathbf{p}_a^{[\Sigma_{\mathrm{w}}]} + \mathbf{R}^{[\Sigma_a \rightarrow \Sigma_{\mathrm{w}}]}\mathbf{p}_{a \rightarrow b}^{[\Sigma_a]}. \tag{72}$$

In the above equation, vector $\mathbf{p}_{a \rightarrow b}^{[\Sigma_a]}$ pointing from joint a to the end point b within the local coordinate system Σ_a is fixed and will not change with time. Meanwhile, the other vectors are all dynamic and will change as the robot moves. By calculating the derivatives of both sides of the above equation with time, we can obtain the *linear velocity* of the end
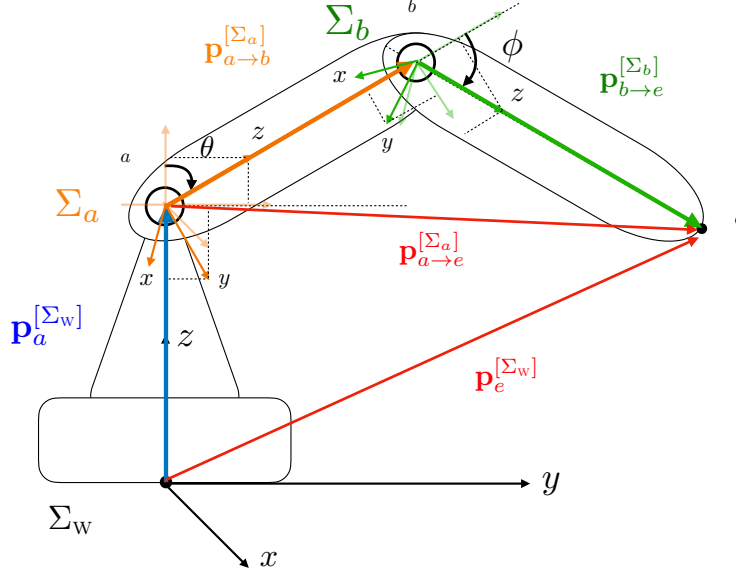
24

Figure 11: An Example of Multi-Link Robot End Point Velocity.

point b as

$$\mathbf{v}_b^{[\Sigma_\text{w}]} = \dot{\mathbf{p}}_b^{[\Sigma_\text{w}]} = \dot{\mathbf{p}}_a^{[\Sigma_\text{w}]} + \dot{\mathbf{R}}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} \tag{73}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \left(\widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \right) \mathbf{p}_{a \to b}^{[\Sigma_a]} \tag{74}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} \right) \tag{75}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \mathbf{p}_{a \to b}^{[\Sigma_\text{w}]} \tag{76}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \left(\mathbf{p}_b^{[\Sigma_\text{w}]} - \mathbf{p}_a^{[\Sigma_\text{w}]} \right). \tag{77}$$

In the above equation, we use many results obtained from the previous sections: (1) according to Equation 59, $\dot{\mathbf{R}} = \widehat{\boldsymbol{\omega}} \mathbf{R}$, (2) according to Equation 58, $\widehat{\boldsymbol{\omega}} \mathbf{p} = \boldsymbol{\omega} \times \mathbf{p}$, and (3) according to the above Equation 72, $\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} = \mathbf{p}_{a \to b}^{[\Sigma_\text{w}]} = \mathbf{p}_b^{[\Sigma_\text{w}]} - \mathbf{p}_a^{[\Sigma_\text{w}]}$.

Since there exist one single link that rotates, we can easily get that the *angular velocity* of the end point will be equal to the velocity of the joint a, *i.e.*,

$$\boldsymbol{\omega}_b^{\Sigma_\text{w}} = \boldsymbol{\omega}_a^{\Sigma_\text{w}}. \tag{78}$$

5.2 Multi-Link Robot End Point Velocity

For the robot with multiple (more than one) moveable links, the end point's velocity calculation will be more difficulty and challenging. We take the multi-link robot arm used before in this part to illustrate the calculation process for readers.

5.2.1 Multi-Link Robot Homogeneous Transition Matrix

As introduced in Section 3.2, for the robot arm involving multiple link, the *homogenous transformation matrices* between the local and world coordinate systems can be represented as follows:

$$\mathbf{T}^{[\Sigma_b \to \Sigma_a]} = \begin{bmatrix} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} & \mathbf{p}_{a \to b}^{[\Sigma_a]} \\ \mathbf{0} & 1 \end{bmatrix}, \text{ and } \mathbf{T}^{[\Sigma_a \to \Sigma_w]} = \begin{bmatrix} \mathbf{R}^{[\Sigma_a \to \Sigma_w]} & \mathbf{p}_a^{[\Sigma_w]} \\ \mathbf{0} & 1 \end{bmatrix} \tag{79}$$

Based on them, we can represent the *homogenous transformation matrices* from the second arm link's local coordinate Σ_b to the world coordinate Σ_w as follows:

$$\mathbf{T}^{[\Sigma_b \to \Sigma_w]} = \mathbf{T}^{[\Sigma_a \to \Sigma_w]}\mathbf{T}^{[\Sigma_b \to \Sigma_a]} \tag{80}$$

$$= \begin{bmatrix} \mathbf{R}^{[\Sigma_a \to \Sigma_w]} & \mathbf{p}_a^{[\Sigma_w]} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} & \mathbf{p}_{a \to b}^{[\Sigma_a]} \\ \mathbf{0} & 1 \end{bmatrix} \tag{81}$$

$$= \begin{bmatrix} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right) & \left(\mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{p}_{a \to b}^{[\Sigma_a]} + \mathbf{p}_a^{[\Sigma_w]} \right) \\ \mathbf{0} & 1 \end{bmatrix}. \tag{82}$$

According to the physical meanings of elements in the calculated *homogeneous transformation matrix* $\mathbf{T}^{[\Sigma_b \to \Sigma_w]}$, we know that the *rotation matrix* from local coordinate Σ_b at the second arm link to the world coordinate Σ_w and the arm end point position vector in the world coordinate Σ_w can be represented as

$$\mathbf{p}_b^{[\Sigma_w]} = \mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{p}_{a \to b}^{[\Sigma_a]} + \mathbf{p}_a^{[\Sigma_w]}, \tag{83}$$

$$\mathbf{R}^{[\Sigma_b \to \Sigma_w]} = \mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{R}^{[\Sigma_b \to \Sigma_a]}, \tag{84}$$

which will help calculate the *linear velocity* and *angular velocity* vectors of the end point in the world coordinate Σ_w.

5.2.2 Multi-Link Robot Linear Velocity

From Equation 1, by projecting the second arm link vector $\mathbf{p}_{b \to e}^{[\Sigma_b]}$ and adding to the position vector $\mathbf{p}_b^{[\Sigma_w]}$, we can calculate the end point *linear velocity* as

$$\mathbf{p}_e^{[\Sigma_w]} = \mathbf{p}_b^{[\Sigma_w]} + \mathbf{R}^{[\Sigma_b \to \Sigma_w]}\mathbf{p}_{b \to e}^{[\Sigma_b]} \tag{85}$$

$$= \mathbf{p}_a^{[\Sigma_w]} + \mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{p}_{a \to b}^{[\Sigma_a]} + \mathbf{R}^{[\Sigma_a \to \Sigma_w]}\mathbf{R}^{[\Sigma_b \to \Sigma_a]}\mathbf{p}_{b \to e}^{[\Sigma_b]}. \tag{86}$$

By taking the derivatives for both sides of Equation 83, we can obtain the end point *linear velocity* to be

$$\mathbf{v}_e^{[\Sigma_\text{w}]} = \dot{\mathbf{p}}_e^{[\Sigma_\text{w}]} \tag{87}$$

$$= \dot{\mathbf{p}}_a^{[\Sigma_\text{w}]} + \dot{\mathbf{R}}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} + \dot{\mathbf{R}}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \dot{\mathbf{R}}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} \tag{88}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} + \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} \tag{89}$$

$$+ \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} \tag{90}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{a \to b}^{[\Sigma_a]} \right) + \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} \right) \tag{91}$$

$$+ \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \left(\mathbf{R}^{[\Sigma_b \to \Sigma_a]} \mathbf{p}_{b \to e}^{[\Sigma_b]} \right) \tag{92}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \mathbf{p}_{a \to b}^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \mathbf{p}_{b \to e}^{[\Sigma_\text{w}]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \left(\boldsymbol{\omega}_b^{[\Sigma_a]} \times \mathbf{p}_{b \to e}^{[\Sigma_a]} \right) \tag{93}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \left(\mathbf{p}_{a \to b}^{[\Sigma_\text{w}]} + \mathbf{p}_{b \to e}^{[\Sigma_\text{w}]} \right) + \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \boldsymbol{\omega}_b^{[\Sigma_a]} \right) \times \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{p}_{b \to e}^{[\Sigma_a]} \right) \tag{94}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \mathbf{p}_{a \to e}^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_b^{[\Sigma_\text{w}]} \times \mathbf{p}_{b \to e}^{[\Sigma_\text{w}]} \tag{95}$$

$$= \mathbf{v}_a^{[\Sigma_\text{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\text{w}]} \times \left(\mathbf{p}_e^{[\Sigma_\text{w}]} - \mathbf{p}_a^{[\Sigma_\text{w}]} \right) + \boldsymbol{\omega}_b^{[\Sigma_\text{w}]} \times \left(\mathbf{p}_e^{[\Sigma_\text{w}]} - \mathbf{p}_b^{[\Sigma_\text{w}]} \right). \tag{96}$$

5.2.3 Multi-Link Robot Angular Velocity

Meanwhile, for the *angular velocity* of the end point, since the end point e is attached to the arm link in the coordinate system Σ_b with origin at joint b, we can calculate it based on the rotation matrix $\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]}$ introduced in Equation 84. As introduced before in Equation 56, we know that

$$\widehat{\boldsymbol{\omega}}_e^{[\Sigma_\text{w}]} = \dot{\mathbf{R}}^{[\Sigma_b \to \Sigma_\text{w}]} \left(\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]} \right)^\top \tag{97}$$

$$= \frac{d}{dt} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right) \left(\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]} \right)^\top \tag{98}$$

$$= \left(\dot{\mathbf{R}}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \dot{\mathbf{R}}^{[\Sigma_b \to \Sigma_a]} \right) \left(\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]} \right)^\top \tag{99}$$

$$= \left(\widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right) \left(\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]} \right)^\top \tag{100}$$

$$= \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right) \left(\mathbf{R}^{[\Sigma_b \to \Sigma_\text{w}]} \right)^\top \tag{101}$$

$$+ \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right)^\top \tag{102}$$

$$= \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \mathbf{R}^{[\Sigma_b \to \Sigma_a]} \left(\mathbf{R}^{[\Sigma_b \to \Sigma_a]} \right)^\top \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \right)^\top \tag{103}$$

$$= \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} + \mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \right)^\top \tag{104}$$

$$= \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} + \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\text{w}]} \boldsymbol{\omega}_b^{[\Sigma_a]} \right)^\wedge \tag{105}$$

$$= \widehat{\boldsymbol{\omega}}_a^{[\Sigma_\text{w}]} + \widehat{\boldsymbol{\omega}}_b^{[\Sigma_\text{w}]}. \tag{106}$$

By applying the $\vee$ operator to both sides of the above equation, we can represent the *angular velocity* of the end point as

$$\boldsymbol{\omega}_e^{[\Sigma_\mathrm{w}]} = \boldsymbol{\omega}_a^{[\Sigma_\mathrm{w}]} + \boldsymbol{\omega}_b^{[\Sigma_\mathrm{w}]}. \tag{107}$$

To derive the *angular velocity* and *linear velocity* of the end point, we use several equations to replace terms with their equivalent representations, like

$$\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \left(\boldsymbol{\omega}_b^{[\Sigma_a]} \times \mathbf{p}_{b \to e}^{[\Sigma_a]} \right) = \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \boldsymbol{\omega}_b^{[\Sigma_a]} \right) \times \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \mathbf{p}_{b \to e}^{[\Sigma_a]} \right) \tag{108}$$

and

$$\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \widehat{\boldsymbol{\omega}}_b^{[\Sigma_a]} \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \right)^\top = \left(\mathbf{R}^{[\Sigma_a \to \Sigma_\mathrm{w}]} \boldsymbol{\omega}_b^{[\Sigma_a]} \right)^\wedge. \tag{109}$$

We will leave the proof of the above two equations as an exercise for the readers.

5.3 A General Representation

Based on the above analysis, given the robot arm with multi-links, whose rotation joints are denoted as a, b, c, $\cdots$, we can represent the end point's *linear velocity* and *angular velocity* of the robot arm as follows:

$$\textbf{Angular Velocity: } \boldsymbol{\omega}_e^{[\Sigma_\mathrm{w}]} = \boldsymbol{\omega}_a^{[\Sigma_\mathrm{w}]} + \boldsymbol{\omega}_b^{[\Sigma_\mathrm{w}]} + \boldsymbol{\omega}_c^{[\Sigma_\mathrm{w}]} + \cdots, \tag{110}$$

$$\textbf{Linear Velocity: } \mathbf{v}_e^{[\Sigma_\mathrm{w}]} = \mathbf{v}_a^{[\Sigma_\mathrm{w}]} + \boldsymbol{\omega}_a^{[\Sigma_\mathrm{w}]} \times \left(\mathbf{p}_e^{[\Sigma_\mathrm{w}]} - \mathbf{p}_a^{[\Sigma_\mathrm{w}]} \right) \tag{111}$$

$$+ \boldsymbol{\omega}_b^{[\Sigma_\mathrm{w}]} \times \left(\mathbf{p}_e^{[\Sigma_\mathrm{w}]} - \mathbf{p}_b^{[\Sigma_\mathrm{w}]} \right) \tag{112}$$

$$+ \boldsymbol{\omega}_c^{[\Sigma_\mathrm{w}]} \times \left(\mathbf{p}_e^{[\Sigma_\mathrm{w}]} - \mathbf{p}_c^{[\Sigma_\mathrm{w}]} \right) + \cdots. \tag{113}$$

As to the *angular velocity* vectors, *e.g.*, $\boldsymbol{\omega}_a^{[\Sigma_\mathrm{w}]}$, $\boldsymbol{\omega}_b^{[\Sigma_\mathrm{w}]}$ and $\boldsymbol{\omega}_c^{[\Sigma_\mathrm{w}]}$, $\cdots$, and *position* vectors, *e.g.*, $\mathbf{p}_a^{[\Sigma_\mathrm{w}]}$, $\mathbf{p}_b^{[\Sigma_\mathrm{w}]}$ and $\mathbf{p}_e^{[\Sigma_\mathrm{w}]}$, $\cdots$, within the world coordinate system Σ_w can be obtained via the *rotation matrix* between the local coordinates and the world coordinate as we derive above.

6. What's Next?

By now, we have introduced the robot representations, robot rotation, and discuss about the robot end point velocity after the rotation. In the next article, we will further talk about several advanced topics about robot motion and control, including *forward kinematics*, *inverse kinematics*, *trajectory generation* and *motion planning*.

References

[1] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada, and Kazuhito Yokoi. *Introduction to Humanoid Robotics*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[2] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, USA, 1st edition, 2017.

[3] United Nations. Global issues: Ageing.