Mobile robotics: Inertial
https://www.ensta-bretagne.fr/inmooc/

Luc Jaulin

September 12, 2022

Contents

Introduction

A *mobile robot* can be defined as a mechanical system capable of moving in its environment in an autonomous manner. For that purpose, it must be equipped with:

- *sensors* that will collect knowledge of its surroundings (which it is more or less aware of) and determine its location ;
- *actuators* which will allow it to move ;
- an *intelligence* (or algorithm, regulator), which will allow it to compute, based on the data gathered by the sensors, the commands to send to the actuators in order to perform a given task.

Finally, to this we must add the *surroundings* of the robot which correspond to the world in which it evolves and its *mission* which is the task it has to accomplish. Mobile robots are constantly evolving, mainly from the beginning of the 2000s, in military domains (airborne drones [1], underwater robots [2], etc.), and even in medical and agricultural fields. They are in particularly high demand for performing tasks considered to be painful or dangerous to humans. This is the case for instance in mine-clearing operations, the search for black boxes of damaged aircraft on the ocean bed and planetary exploration. Artificial satellites, launchers (such as Ariane V), driverless subways and elevators are examples of mobile robots. Airliners, trains and cars evolve in a continuous fashion towards more and more autonomous systems and will very probably become mobile robots in the following decades.

Mobile robotics is the discipline which looks at the design of mobile robots [3]. It is based on other disciplines such as automatic control, signal processing, mechanics, computing and electronics. The aim of this book is to give an overview of the tools and methods of robotics which will aid in the design of mobile robots. The robots will be modeled by *state equations*, *i.e.*, first order (mostly non-linear) differential equations. These state equations can be obtained by using the laws of mechanics. It is not in our objectives to teach, in detail, the methods of robot modeling (refer to [4] and [5] for more information on the subject), merely to recall its principles. By *modeling*, we mean obtaining the state equations. This step is essential for simulating robots as well as designing controllers. In this book, we will provide the principle and the tools for three-dimensional modeling of a solid (non-articulated) robot modeling in three-dimension such as an airplane, a quadcopter, a submarine, and so forth.

Through this modeling we will introduce a number of fundamental concepts in robotics such as state representation, rotation matrices and Euler angles. The robots we will consider are assumed to be put into a state representation form:

$$\left\{ \begin{array}{rcl} \dot{\mathbf{x}}(t) & = & \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) & = & \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) \end{array} \right.$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ the input vector and $\mathbf{y}$ the vector of measurements [4]. We will call *modeling* the step which consists of finding a more or less accurate state representation of the robot. In general, constant parameters may appear in the state equations (such as the mass and the moment of inertia of a body, viscosity, etc.). In such cases, an identification step might prove to be necessary. We will assume that all of the parameters are known. Of course, there is no systematic methodology that can be applied for modeling a mobile robot. We will present the tools which allow to reach a state representation of three-dimensional solid robots in order for the reader to acquire a certain experience which will be helpful when modeling his/her own robots. This modeling will also allow us to recall a number of important concepts in Euclidean geometry, which are fundamental in mobile robotics.

Chapter 1

Set a rigid body in a 3D space

This chapter recalls a number of important concepts in kinematics which will be useful for the modeling in three dimension.

1.1 Rotation matrices

For three-dimensional modeling, it is essential to have a good understanding of the concepts related to rotation matrices, which are recalled in this section. It is by using this tool that we will perform our coordinate system transformations and position our objects in space.

Let us recall that the j^{th} column of the matrix of a linear application of $\mathbb{R}^n \to \mathbb{R}^n$ represents the image of the j^{th} vector $\mathbf{e}_j$ of the standard basis (see Figure 1.1). Thus, the expression of a rotation matrix of angle θ in the plane $\mathbb{R}^2$ is given by:

$$\mathbf{R} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}.$$

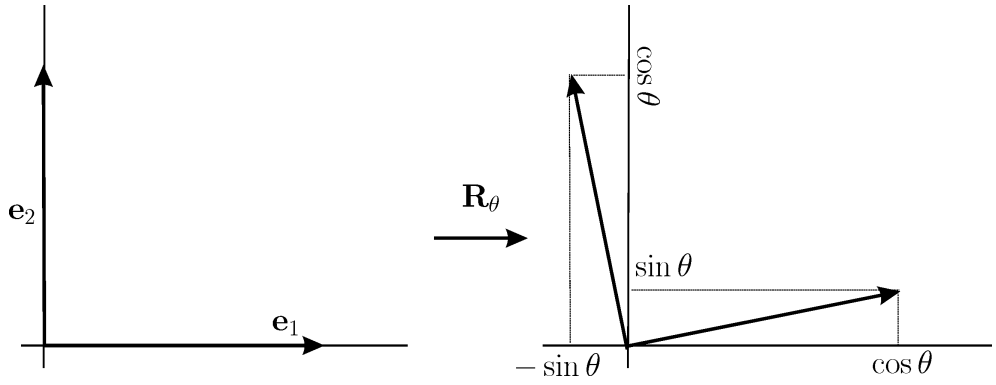

Figure 1.1: Rotation of angle θ in a plane

Concerning rotations in the space $\mathbb{R}^3$ (see Figure 1.2), it is important to specify the axis of rotation. We distinguish three main rotations: the rotation around the Ox axis, the one around the Oy axis and the one around the Oz axis.

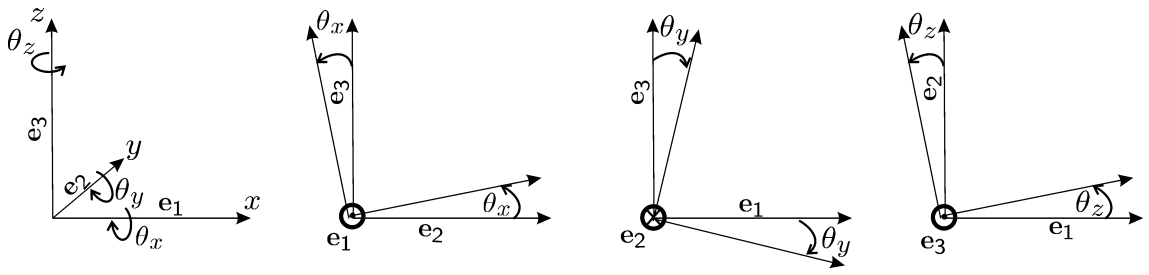

Figure 1.2: Rotations in $\mathbb{R}^3$ following various viewing angles

The associated matrices are respectively given by:

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{pmatrix}, \ \mathbf{R}_y = \begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{pmatrix}, \mathbf{R}_z = \begin{pmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Let us recall the formal definition of a rotation. A *rotation* is a linear application which is an isometry (*i.e.*, it preserves the scalar product) which is direct (it does not change the orientation in space).

Theorem 1. *A matrix* $\mathbf{R}$ *is a rotation matrix if and only if :*

$$\mathbf{R}^T \cdot \mathbf{R} = \mathbf{I} \text{ and } \det \mathbf{R} = 1.$$

Proof. The scalar product is preserved by $\mathbf{R}$ if, for any $\mathbf{u}$ and $\mathbf{v}$ in $\mathbb{R}^n$, we have :

$$(\mathbf{R}\mathbf{u})^{\mathrm{T}} \cdot (\mathbf{R}\mathbf{v}) = \mathbf{u}^{\mathrm{T}} \cdot \mathbf{R}^{\mathrm{T}}\mathbf{R} \cdot \mathbf{v} = \mathbf{u}^{\mathrm{T}}\mathbf{v}.$$

Therefore $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$. The symmetries relative to a plane, as well as all the other improper isometries (isometries that change the orientation of space, such as a mirror), also verify the property $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$. The condition $\det \mathbf{R} = 1$ allows us to be limited to the isometries which are direct. □

1.2 Lie group

The set of rotation matrices of $\mathbb{R}^n$ forms a group with respect to the multiplication [6]. It is referred to as a *special orthogonal group* (*special* because $\det \mathbf{R} = 1$, *orthogonal* because $\mathbf{R}^{\mathrm{T}} \cdot \mathbf{R} = \mathbf{I}$) and denoted by $SO(n)$. It is trivial to check that $(SO(n), \cdot)$ is a group where $\mathbf{I}$ is the neutral element. Moreover, the multiplication and the inversion are both smooth. This makes $SO(n)$ a *Lie group* which is a manifold of the set of matrices $\mathbb{R}^{n \times n}$.

The set $\mathbb{R}^{n \times n}$ of $n \times n$-matrices is of dimension n^2. Since the matrix $\mathbf{R}^{\mathrm{T}} \cdot \mathbf{R}$ is always symmetric, the matrix equation $\mathbf{R}^{\mathrm{T}} \cdot \mathbf{R} = \mathbf{I}$ can be decomposed into $\frac{n(n+1)}{2}$ independent scalar equations. For instance, for $n = 2$, we have $\frac{2(2+1)}{2} = 3$ scalar equations:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \Leftrightarrow \begin{cases} a^2 + b^2 = 1 \\ ac + bd = 0 \\ c^2 + d^2 = 1 \end{cases}$$

As a consequence, the set $SO(n)$ forms a manifold of dimension $d = n^2 - \frac{n(n+1)}{2}$.

- For $n = 1$, we get $d = 0$. The set $SO(1)$ is a singleton which contains a single rotation matrix: $R = 1$.
- For $n = 2$, we get $d = 1$. We need a unique parameter (or angle) to represent the rotations of $SO(2)$.
- For $n = 3$, we get $d = 3$. We need 3 parameters (or angles) to represent $SO(3)$.

1.3 Lie algebra

An *algebra* is an algebraic structure $(\mathcal{A}, +, \times, \cdot)$ over the field $\mathbb{R}$, if

- (i) $(\mathcal{A}, +, \cdot)$ is a vector space over $\mathbb{R}$;
- (ii) the multiplication rule $\times$ of $\mathcal{A} \times \mathcal{A} \to \mathcal{A}$ is left- and right-distributive with respect to $+$
- (iii) for all $\alpha, \beta \in \mathbb{R}$, and for all $x, y \in \mathcal{A}$, $\alpha \cdot x \times \beta \cdot y = (\alpha\beta) \cdot (x \times y)$.

Note that in general, an algebra is non-commutative ($x \times y \neq y \times x$) and non-associative ($(x \times y) \times z \neq x \times (y \times z)$). A *Lie algebra* $(\mathcal{G}, +, [\,], \cdot)$ is a non-commutative and non-associative algebra in which multiplication, denoted by a so-called *Lie bracket*, verifies (i) $[\cdot, \cdot]$ that is bilinear, i.e., linear with respect to each variable ; (ii) $[x, y] = -[y, x]$ (antisymmetry) and (iii) $[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0$ (Jacobi relation).

For Lie groups we can define the associated *Lie algebras [6].* Lie algebras allow us to consider infinitesimal motions around a given element (*i.e.*, a rotation matrix) in order to use derivatives or differential methods.

Consider the rotation matrix $\mathbf{I}$ of $SO(n)$ corresponding to the identity. If we move $\mathbf{I}$ by adding a small matrix say $\mathbf{A} \cdot dt$ of $\mathbb{R}^{n \times n}$, we generally do not obtain a rotation matrix. We are interested by matrices $\mathbf{A}$ such that $\mathbf{I} + \mathbf{A} \cdot dt \in SO(n)$. We have

$$(\mathbf{I} + \mathbf{A} \cdot dt)^{\mathrm{T}} \cdot (\mathbf{I} + \mathbf{A} \cdot dt) = \mathbf{I}$$

i.e., $\mathbf{A} \cdot dt + \mathbf{A}^{\mathrm{T}} \cdot dt = \mathbf{0} + o(dt)$. Therefore, $\mathbf{A}$ should be skew-symmetric. It means that we are able to move in $SO(n)$ around $\mathbf{I}$ by adding infinitesimal skew-symmetric matrices $\mathbf{A} \cdot dt$ that are not elements of $SO(n)$. This corresponds to a new operation in $SO(n)$ which is not the multiplication we already had. Formally, we define the *Lie algebra* associated to $SO(n)$ as follows

$$Lie\,(SO(n)) = \left\{ \mathbf{A} \in \mathbb{R}^{n \times n} \,|\, \mathbf{I} + \mathbf{A} \cdot dt \in SO(n) \right\}$$

and it corresponds skew-symmetric matrices of $\mathbb{R}^{n \times n}$.

If now we want to move around any matrix $\mathbf{R}$ of $SO(n)$, we generate a rotation matrix around $\mathbf{I}$ and we transport it to $\mathbf{R}$. We get $\mathbf{R}\,(\mathbf{I} + \mathbf{A} \cdot dt)$ where $\mathbf{A}$ is skew-symmetric. It means that we add to $\mathbf{R}$ the matrix $\mathbf{R} \cdot \mathbf{A} \cdot dt$.

In robotics, Lie groups are often used to describe transformations (such as translation or rotations). The Lie algebra corresponds to velocities or equivalently to infinitesimal transformations. Lie group theory is useful, but requires some non trivial mathematical backgrounds that are beyond the scope of this book. We will try to focus on $SO(3)$ or use more classical tools such as Euler angles, rotation vectors, which are probably less general but sufficient for control purposes.

1.4 Rotation vector

If $\mathbf{R}$ is a rotation matrix depending on time t, by differentiating the relation $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}$, we get

$$\dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}} + \mathbf{R} \cdot \dot{\mathbf{R}}^{\mathrm{T}} = \mathbf{0}.$$

Thus, the matrix $\dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}}$ is a skew-symmetric matrix (*i.e.*, it satisfies $\mathbf{A}^{\mathrm{T}} = -\mathbf{A}$ and therefore its diagonal contains only zeros, and for each element of $\mathbf{A}$, we have $a_{ij} = -a_{ji}$). We may therefore write, in the case where $\mathbf{R}$ is of dimension 3×3:

$$\dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}. \tag{1.1}$$

The vector $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$ is called the *rotation vector* or *Euler vector* associated with the pair $\left(\mathbf{R}, \dot{\mathbf{R}}\right)$. It must be noted that $\dot{\mathbf{R}}$ is not a matrix with good properties (such as for instance the fact of being skew-symmetric). On the other hand, the matrix $\dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}}$ has the structure of Equation (1.1) since it allows to be positioned within the coordinate system in which the rotation is performed and this, due to the change of basis performed by $\mathbf{R}^{\mathrm{T}}$. We will define the *cross product* between two vectors $\boldsymbol{\omega}$ and $\mathbf{x} \in \mathbb{R}^3$ as follows:

$$\boldsymbol{\omega} \wedge \mathbf{x} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \wedge \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_3\omega_y - x_2\omega_z \\ x_1\omega_z - x_3\omega_x \\ x_2\omega_x - x_1\omega_y \end{pmatrix} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

For each vector $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$, we may associate the skew-symmetric matrix:

$$\wedge(\boldsymbol{\omega}) = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$$

which can be interpreted as the matrix associated with a cross product by the vector $\boldsymbol{\omega}$. The matrix $\wedge(\boldsymbol{\omega})$ is also written $\boldsymbol{\omega}\wedge$. It is also called the *small adjoint* of $\boldsymbol{\omega}$ and denoted by $\mathrm{ad}(\boldsymbol{\omega})$. It should not be confused with the *large adjoint* $\mathrm{Ad}(\mathbf{R})$ also used in this context, but not here in this book.

Proposition 2. *If $\mathbf{R}(t)$ is a rotation matrix that depends on time, its rotation vector is given by:*

$$\boldsymbol{\omega} = \wedge^{-1}\left(\dot{\mathbf{R}} \cdot \mathbf{R}^{T}\right). \tag{1.2}$$

Proof. This relation is a direct consequence of Equation (1.1). □

Proposition 3. *If* $\mathbf{R}$ *is a rotation matrix in* $\mathbb{R}^3$ *and if* $\mathbf{a}$ *is a vector of* $\mathbb{R}^3$*, we have:*

$$\wedge(\mathbf{R}\cdot\mathbf{a}) = \mathbf{R}\cdot\wedge(\mathbf{a})\cdot\mathbf{R}^T. \tag{1.3}$$

Proof. Let $\mathbf{x}$ be a vector of $\mathbb{R}^3$. We have:

$$\begin{aligned}\wedge(\mathbf{R}\cdot\mathbf{a})\cdot\mathbf{x} &= (\mathbf{R}\cdot\mathbf{a})\wedge\mathbf{x} = (\mathbf{R}\cdot\mathbf{a})\wedge\left(\mathbf{R}\cdot\mathbf{R}^{\mathrm{T}}\mathbf{x}\right)\\ &= \mathbf{R}\cdot\left(\mathbf{a}\wedge\mathbf{R}^{\mathrm{T}}\cdot\mathbf{x}\right) = \mathbf{R}\cdot\wedge(\mathbf{a})\cdot\mathbf{R}^{\mathrm{T}}\cdot\mathbf{x}.\end{aligned}$$

□

Proposition 4. *(duality). We have:*

$$\mathbf{R}^T\dot{\mathbf{R}} = \wedge\left(\mathbf{R}^T\boldsymbol{\omega}\right). \tag{1.4}$$

This relation expresses the fact that the matrix $\mathbf{R}^T\dot{\mathbf{R}}$ *is associated with the rotation vector* $\boldsymbol{\omega}$*, associated with* $\mathbf{R}(t)$ *but expressed in the coordinate system associated with* $\mathbf{R}$ *whereas* $\dot{\mathbf{R}}\cdot\mathbf{R}^T$ *is associated to the same vector, but this time expressed in the coordinate system of the standard basis.*

Proof. We have:

$$\mathbf{R}^{\mathrm{T}}\dot{\mathbf{R}} \;= \mathbf{R}^{\mathrm{T}}\left(\dot{\mathbf{R}}\cdot\mathbf{R}^{\mathrm{T}}\right)\mathbf{R} \overset{(1.2)}{=} \mathbf{R}^{\mathrm{T}}\cdot\wedge(\boldsymbol{\omega})\cdot\mathbf{R} \overset{(1.3)}{=} \wedge\left(\mathbf{R}^{\mathrm{T}}\boldsymbol{\omega}\right).$$

□

1.5 Rodrigues rotation formulas

Matrix exponential. Given a square matrix $\mathbf{M}$ of dimension n, its exponential can be defined as:

$$e^{\mathbf{M}} = \mathbf{I}_n + \mathbf{M} + \frac{1}{2!}\mathbf{M}^2 + \frac{1}{3!}\mathbf{M}^3 + \cdots = \sum_{i=0}^{\infty}\frac{1}{i!}\mathbf{M}^i$$

where $\mathbf{I}_n$ is the identity matrix of dimension n. It is clear that $e^{\mathbf{M}}$ is of the same dimension as $\mathbf{M}$. Here are some of the important properties concerning the exponentials of matrices. If $\mathbf{0}_n$ is the zero matrix of $n\times n$ and if $\mathbf{M}$ and $\mathbf{N}$ are two matrices $n\times n$, then:

$e^{\mathbf{0}_n} = \mathbf{I}_n$
$e^{\mathbf{M}}\cdot e^{\mathbf{N}} = e^{\mathbf{M}+\mathbf{N}}$ (if the matrices commute)
$\frac{d}{dt}\left(e^{\mathbf{M}t}\right) = \mathbf{M}\cdot e^{\mathbf{M}t}$

Matrix logarithm. Given a matrix $\mathbf{M}$ of dimension n, the matrix $\mathbf{L}$ is said to be a matrix logarithm of $\mathbf{M}$ if $e^{\mathbf{L}} = \mathbf{M}$. As for complex numbers, the exponential function is not a one-to-one function and matrices may have more than one logarithm. Using power series, we define the logarithm of a square matrix as

$$\log\mathbf{M} = \sum_{i=1}^{\infty}\frac{(-1)^{i+1}}{i}(\mathbf{M}-\mathbf{I}_n)^i.$$

The sum is convergent if $\mathbf{M}$ is close to identity.

Rodrigues formulas. A rotation matrix $\mathbf{R}$ has an axis represented by a unit vector $\mathbf{n}$ and an angle α with respect to this axis. From $\mathbf{n}$ and α we can also generate the matrix $\mathbf{R}$. The link $\mathbf{R} \leftrightarrow (\mathbf{n}, \alpha)$ is made by the following *Rodrigues formulas*

$$\begin{array}{lrcl} (i) & \mathbf{R} & = & e^{\alpha \mathbf{n} \wedge} \\ (ii) & \alpha \mathbf{n} & = & \mathrm{Log}(\mathbf{R}) \end{array} \tag{1.5}$$

where

$$\mathrm{Log}(\mathbf{R}) = \wedge^{-1}(\log \mathbf{R})$$

Equation *(ii)* is the reciprocal of *(i)*. In these formulas, $\alpha\mathbf{n}\wedge$ is a notation to represent the matrix $\wedge(\alpha\mathbf{n})$. As it will be shown in the exercise,

$$\mathbf{R} = e^{\alpha \mathbf{n} \wedge} = \exp(\alpha \mathbf{n} \wedge) = \mathrm{Exp}(\alpha \mathbf{n})$$

is a rotation matrix and $\mathbf{n}$ is an eigenvector associated with the eigenvalue 1 of $\mathbf{R}$. The capitalized version *Exp, Log* for *exp, log* can be seen as shortcuts to avoid using the $\wedge$ operator [7][8]. This is illustrated by Figure 1.3.

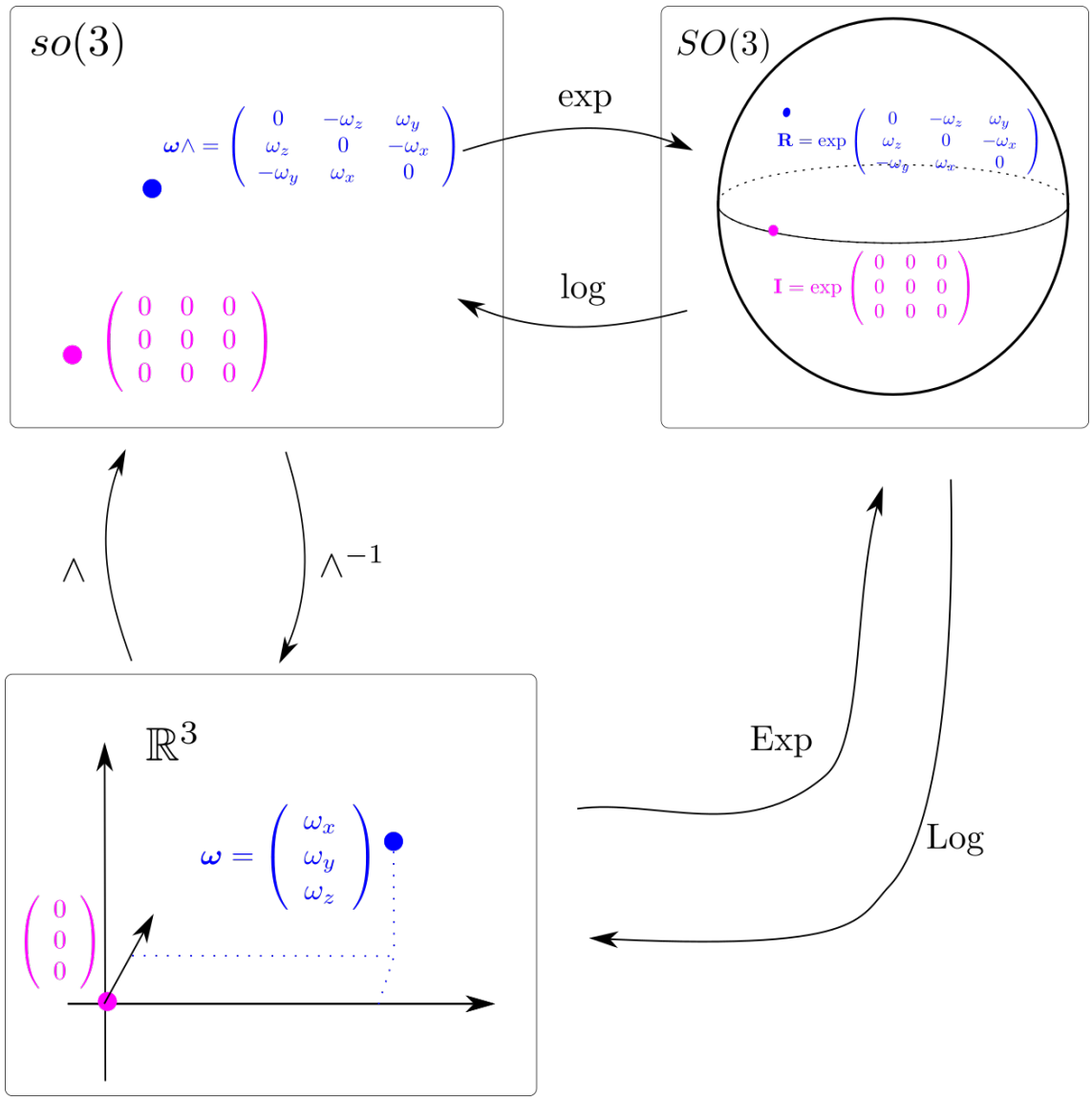

Figure 1.3: Correspondences between the Lie group SO(3) and the Lie algebra so(3)

The following proposition provides an analytical expression for Log and Exp in the case of $SO(3)$.

Proposition 5. *Given a rotation matrix* $\mathbf{R}$ *of* $SO(3)$*, its logarithm (which is a vector of* $so(3)$*) is given by*

$$\begin{array}{rcl} Log\mathbf{R} & = & \frac{\alpha}{2\sin\alpha}\cdot\wedge^{-1}\left(\mathbf{R}-\mathbf{R}^T\right) \\ \alpha & = & acos\left(\frac{tr(\mathbf{R})-1}{2}\right) \end{array} \tag{1.6}$$

Given a vector of $\boldsymbol{\omega}\in so(3)$*, its exponential (which is a rotation matrix) is given by*

$$\begin{array}{rcl} Exp\boldsymbol{\omega} & = & \mathbf{I}+\frac{\sin\alpha}{\alpha}\cdot\omega\wedge+\frac{1-\cos\alpha}{\alpha^2}\cdot(\omega\wedge)^2 \\ \alpha & = & \|\boldsymbol{\omega}\| \end{array}$$

Proof. We only prove (1.6). Take a rotation matrix $\mathbf{R} = e^{\alpha \mathbf{n}\wedge}$ with eigen values $1, \lambda_1, \lambda_2$ and eigen vectors $\mathbf{n}, \mathbf{v}_1, \mathbf{v}_2$. Consider the generalized polynomial $f(x) = x - x^{-1}$, where x is the indeterminate. From the correspondence theorem of eigen values/vectors, the eigen values of $f(\mathbf{R}) = \mathbf{R} - \mathbf{R}^{-1} = \mathbf{R} - \mathbf{R}^{\mathrm{T}}$ are $f(1) = 0, f(\lambda_2), f(\lambda_3)$ and the eigen vectors are still $\mathbf{n}, \mathbf{v}_1, \mathbf{v}_2$. Now,

$$\begin{array}{lll} (i) & f(\mathbf{R}) \text{ is skew symmetric} & (f^{\mathrm{T}}(\mathbf{R}) + f(\mathbf{R}) = \mathbf{R}^{\mathrm{T}} - \mathbf{R} + \mathbf{R} - \mathbf{R}^{\mathrm{T}} = \mathbf{0}) \\ (ii) & f(\mathbf{R}) \cdot \mathbf{n} = \mathbf{0} & (\mathbf{n} \text{ is the eigen vector of } f(\mathbf{R}) \text{ associated to } 0) \end{array} \tag{1.7}$$

Thus, the two matrices $f(\mathbf{R})$ and $(\mathbf{n}\wedge)$ are proportional. As a consequence $\wedge^{-1}(f(\mathbf{R}))$ provides us the axis of $\mathbf{R}$.

To find the angle α, we use the property that the trace of a matrix is *similarity-invariant*, which means that for any invertible matrix $\mathbf{P}$, we have $\mathrm{tr}(\mathbf{R}) = \mathrm{tr}(\mathbf{P}^{-1} \cdot \mathbf{R} \cdot \mathbf{P})$. Indeed

$$\mathrm{tr}((\mathbf{P}^{-1} \cdot \mathbf{R}) \cdot \mathbf{P}) = \mathrm{tr}(\mathbf{P} \cdot (\mathbf{P}^{-1} \cdot \mathbf{R})) = \mathrm{tr}(\mathbf{R})$$

Take $\mathbf{P}$ as the rotation matrix with transforms $\mathbf{R}$ into a rotation along the first axis. We get

$$\mathrm{tr}(\mathbf{R}) = \mathrm{tr} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} = 1 + 2\cos\alpha,$$

which provides us the angle of the rotation. □

1.6 Evolution of the rotation matrix

Proposition 6. *The evolution of the rotation matrix $\mathbf{R}(t)$ of a rigid body follows the integration scheme:*

$$\mathbf{R}(t + dt) = \mathbf{R}(t) \cdot Exp(dt \cdot \boldsymbol{\omega}_r(t)). \tag{1.8}$$

where $\boldsymbol{\omega}_r = \mathbf{R}^T \boldsymbol{\omega}$ is the rotation vector of the body expressed in its own frame.

Proof. Since $\mathbf{R}^{\mathrm{T}}(t)\dot{\mathbf{R}}(t) = (\omega_r(t)\wedge)$ (see 1.4), we have

$$\dot{\mathbf{R}}(t) = \mathbf{R}(t) \cdot (\omega_r(t)\wedge).$$

If dt is infinitely small, we get

$$\begin{array}{rcl} \mathbf{R}(t + dt) & = & \mathbf{R}(t) + dt \cdot \dot{\mathbf{R}}(t) + o(dt) \\ & = & \mathbf{R}(t) + dt \cdot \mathbf{R}(t) \cdot (\omega_r(t)\wedge) + o(dt) \\ & = & \mathbf{R}(t) \cdot (\mathbf{I} + dt \cdot (\omega_r(t)\wedge) + o(dt)) \\ & = & \mathbf{R}(t) \cdot \exp(dt \cdot \omega_r(t)\wedge) + o(dt) \\ & = & \mathbf{R}(t) \cdot \mathrm{Exp}(dt \cdot \omega_r(t)) + o(dt) \end{array}$$

□

For numerical reasons, when we integrate for a long time, the matrix $\mathbf{R}(t)$ may loose the property $\mathbf{R}\cdot\mathbf{R}^{\mathrm{T}} = \mathbf{I}$. To avoid this, at each iteration, a normalization step is needed, *i.e.*, the current matrix $\mathbf{R}$ should be projected on to $SO(3)$. For this, we can perform a QR factorization. A QR-decomposition of a square matrix $\mathbf{M}$ provides two matrices $\mathbf{Q}$,$\mathbf{R}$ such that $\mathbf{M} = \mathbf{Q}\cdot\mathbf{R}$ where $\mathbf{Q}$ is a rotation matrix and $\mathbf{R}$ is triangular. When $\mathbf{M}$ is almost a rotation matrix, then $\mathbf{R}$ is almost diagonal and also almost a rotation, *i.e.*, on the diagonal of $\mathbf{R}$ the entries are approximately ± 1 and outside the entries are almost zeros. The following PYTHON code performs the projection of $\mathbf{M}$ on $SO(3)$ and generates the rotation matrix $\mathbf{M}_2$.

```
Q,R = numpy.linalg.qr(M)
v=diag(sign(R))
M2=Q@diag(v)
```

1.7 Interpolation

Consider a Lie group G (for instance, the rotations), with two elements elements $a, b \in G$. We would like to interpolate between these elements, according to a parameter $t \in [0,1]$. An interpolation from a to b is a function such that:

$$\begin{aligned} f &: G \times G \times \mathbb{R} \to G \\ & f(a,b,0) = a \\ & f(a,b,1) = b \end{aligned}$$

A possible interpolation is:

$$f(a,b,t) = \exp\left(t \cdot \log\left(b \cdot a^{-1}\right)\right) \cdot a$$

or equivalently

$$f(a,b,t) = b^t \cdot a^{1-t}.$$

Note $f(a,b,t)$ is always on G, due to the properties of the exponential map. The resulting path is along a geodesic of the manifold G.

1.8 Coordinate system change

Let $\mathcal{R}_0 : (\mathbf{o}_0, \mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0)$ and $\mathcal{R}_1 : (\mathbf{o}_1, \mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1)$ be two coordinate systems and let $\mathbf{u}$ be a vector of $\mathbb{R}^3$ (refer to Figure 1.4). We have the following relation:

$$\begin{aligned} \mathbf{u} &= x_0\mathbf{i}_0 + y_0\mathbf{j}_0 + z_0\mathbf{k}_0 \\ &= x_1\mathbf{i}_1 + y_1\mathbf{j}_1 + z_1\mathbf{k}_1 \end{aligned}$$

where (x_0, y_0, z_0) and (x_1, y_1, z_1) are the coordinates of $\mathbf{u}$ in $\mathcal{R}_0$ and $\mathcal{R}_1$, respectively.

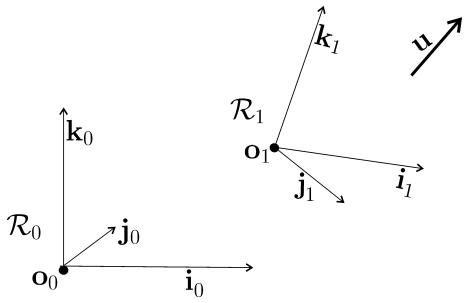

Figure 1.4: Changing the coordinate system $\mathcal{R}_0$ to the system $\mathcal{R}_1$

Thus, for any vector $\mathbf{v}$ we have:

$$\langle x_0\mathbf{i}_0 + y_0\mathbf{j}_0 + z_0\mathbf{k}_0, \mathbf{v}\rangle = \langle x_1\mathbf{i}_1 + y_1\mathbf{j}_1 + z_1\mathbf{k}_1, \mathbf{v}\rangle .$$

By taking respectively $\mathbf{v} = \mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0$, we obtain the following three relations:

$$\left\{\begin{array}{lcl} \langle x_0\mathbf{i}_0 + y_0\mathbf{j}_0 + z_0\mathbf{k}_0, \mathbf{i}_0\rangle &=& \langle x_1\mathbf{i}_1 + y_1\mathbf{j}_1 + z_1\mathbf{k}_1, \mathbf{i}_0\rangle \\ \langle x_0\mathbf{i}_0 + y_0\mathbf{j}_0 + z_0\mathbf{k}_0, \mathbf{j}_0\rangle &=& \langle x_1\mathbf{i}_1 + y_1\mathbf{j}_1 + z_1\mathbf{k}_1, \mathbf{j}_0\rangle \\ \langle x_0\mathbf{i}_0 + y_0\mathbf{j}_0 + z_0\mathbf{k}_0, \mathbf{k}_0\rangle &=& \langle x_1\mathbf{i}_1 + y_1\mathbf{j}_1 + z_1\mathbf{k}_1, \mathbf{k}_0\rangle \end{array}\right.$$

However, since the basis $(\mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0)$ of $\mathcal{R}_0$ is orthonormal, $\langle \mathbf{i}_0, \mathbf{i}_0\rangle = \langle \mathbf{j}_0, \mathbf{j}_0\rangle = \langle \mathbf{k}_0, \mathbf{k}_0\rangle = 1$ and $\langle \mathbf{i}_0, \mathbf{j}_0\rangle = \langle \mathbf{j}_0, \mathbf{k}_0\rangle = \langle \mathbf{i}_0, \mathbf{k}_0\rangle = 0$. Thus, these three relations become:

$$\left\{\begin{array}{lcl} x_0 &=& x_1 \cdot \langle \mathbf{i}_1, \mathbf{i}_0\rangle + y_1 \cdot \langle \mathbf{j}_1, \mathbf{i}_0\rangle + z_1 \cdot \langle \mathbf{k}_1, \mathbf{i}_0\rangle \\ y_0 &=& x_1 \cdot \langle \mathbf{i}_1, \mathbf{j}_0\rangle + y_1 \cdot \langle \mathbf{j}_1, \mathbf{j}_0\rangle + z_1 \cdot \langle \mathbf{k}_1, \mathbf{j}_0\rangle \\ z_0 &=& x_1 \cdot \langle \mathbf{i}_1, \mathbf{k}_0\rangle + y_1 \cdot \langle \mathbf{j}_1, \mathbf{k}_0\rangle + z_1 \cdot \langle \mathbf{k}_1, \mathbf{k}_0\rangle \end{array}\right.$$

Or in matrix form:

$$\underbrace{\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}}_{=\mathbf{u}|_{\mathcal{R}_0}} = \underbrace{\begin{pmatrix} \langle \mathbf{i}_1, \mathbf{i}_0\rangle & \langle \mathbf{j}_1, \mathbf{i}_0\rangle & \langle \mathbf{k}_1, \mathbf{i}_0\rangle \\ \langle \mathbf{i}_1, \mathbf{j}_0\rangle & \langle \mathbf{j}_1, \mathbf{j}_0\rangle & \langle \mathbf{k}_1, \mathbf{j}_0\rangle \\ \langle \mathbf{i}_1, \mathbf{k}_0\rangle & \langle \mathbf{j}_1, \mathbf{k}_0\rangle & \langle \mathbf{k}_1, \mathbf{k}_0\rangle \end{pmatrix}}_{=\mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1}} \cdot \underbrace{\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix}}_{=\mathbf{u}|_{\mathcal{R}_1}} \tag{1.9}$$

We can see a rotation matrix $\mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1}$ whose columns are the coordinates of $\mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1$ expressed in the absolute system $\mathcal{R}_0$. Thus

$$\mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1} = \left(\begin{array}{c|c|c} & & \\ \mathbf{i}_1|_{\mathcal{R}_0} & \mathbf{j}_1|_{\mathcal{R}_0} & \mathbf{k}_1|_{\mathcal{R}_0} \\ & & \end{array}\right)$$

This matrix depends on time and links the frame $\mathcal{R}_1$ to $\mathcal{R}_0$. The matrix $\mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1}$ is often referred to as a *direction cosine matrix* since its components involve the direction cosines of the basis vectors of

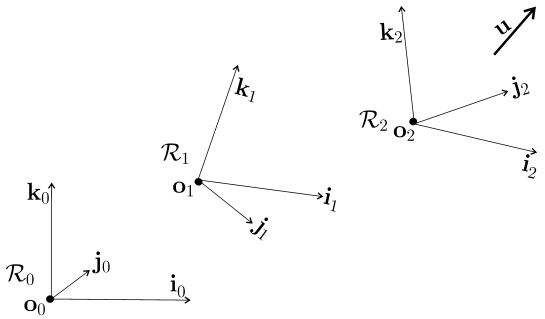

Figure 1.5: Composition of rotations

the two coordinate systems. Likewise, if we would have several systems $\mathcal{R}_0, \ldots, \mathcal{R}_n$ (see Figure 1.5), we would have:

$$\mathbf{u}|_{\mathcal{R}_0} = \mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1} \cdot \mathbf{R}_{\mathcal{R}_1}^{\mathcal{R}_2} \cdot \ \ldots \ \cdot \mathbf{R}_{\mathcal{R}_{n-1}}^{\mathcal{R}_n} \cdot \mathbf{u}|_{\mathcal{R}_n}.$$

Dead Reckoning. Let us consider the situation of a robot moving in a three-dimensional environment. Let us call $\mathcal{R}_0 : (\mathbf{o}_0, \mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0)$ its reference frame (for example, the frame of the robot at an initial time). The position of the robot is represented by the vector $\mathbf{p}(t)$ expressed in $\mathcal{R}_0$ and its attitude (*i.e.*, its orientation) by the rotation matrix $\mathbf{R}(t)$ which represents the coordinates of the vectors $\mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1$ of the coordinate system $\mathcal{R}_1$ of the robot expressed in the coordinate system $\mathcal{R}_0$, at time t. It follows that:

$$\mathbf{R}(t) = \left(\left| \ \mathbf{i}_1|_{\mathcal{R}_0} \ \right| \left| \ \mathbf{j}_1|_{\mathcal{R}_0} \ \right| \left| \ \mathbf{k}_1|_{\mathcal{R}_0} \ \right| \right) = \mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1}(t).$$

This matrix can be returned by a precise attitude unit positioned on the robot. If the robot is also equipped with a *Doppler Velocity Log* (or DVL) which provides it with its speed vector $\mathbf{v}_r$ relative to the ground or the seabed, expressed in the coordinate system $\mathcal{R}_1$ of the robot, then the speed vector $\mathbf{v}$ of the robot satisfies:

$$\underbrace{\mathbf{v}|_{\mathcal{R}_0}}_{\dot{\mathbf{p}}(t)} \overset{(1.9)}{=} \underbrace{\mathbf{R}_{\mathcal{R}_0}^{\mathcal{R}_1}}_{\mathbf{R}(t)} \cdot \underbrace{\mathbf{v}|_{\mathcal{R}_1}}_{\mathbf{v}_r(t)}$$

Or equivalently

$$\dot{\mathbf{p}}\,(t) = \mathbf{R}(t) \cdot \mathbf{v}_r(t). \tag{1.10}$$

Dead reckoning consists of integrating this state equation from the knowledge of $\mathbf{R}(t)$ and $\mathbf{v}_r(t)$.

Exercises

EXERCISE 1.– *Properties of the matrix* $\boldsymbol{\omega}\wedge$

See the correction video at https://youtu.be/D3DbfurFWXo

Let us consider the vector $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$ and its associated skew symmetric matrix

$$\wedge(\boldsymbol{\omega}) = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$$

.

1) Show that the eigenvalues of $\wedge(\boldsymbol{\omega})$ are $\{0, \|\boldsymbol{\omega}\| \cdot i, -\|\boldsymbol{\omega}\| \cdot i\}$. Give an eigenvector associated with 0. Discuss.

2) Show that the vector $\wedge(\boldsymbol{\omega})\ \mathbf{x} = \boldsymbol{\omega} \wedge \mathbf{x}$ is a vector perpendicular to $\boldsymbol{\omega}$ and $\mathbf{x}$, such that the trihedron $(\boldsymbol{\omega}, \mathbf{x}, \boldsymbol{\omega} \wedge \mathbf{x})$ is direct.

3) Show that the norm of $\boldsymbol{\omega} \wedge \mathbf{x}$ is the surface of the parallelogram $\mathcal{A}$ mediated by $\boldsymbol{\omega}$ and $\mathbf{x}$.

EXERCISE 2.– *Jacobi identity*

See the correction video at https://youtu.be/gD04pW7iYBw

The Jacobi identity for the vector product in $\mathbb{R}^3$ is written as:

$$\mathbf{a} \wedge (\mathbf{b} \wedge \mathbf{c}) + \mathbf{c} \wedge (\mathbf{a} \wedge \mathbf{b}) + \mathbf{b} \wedge (\mathbf{c} \wedge \mathbf{a}) = \mathbf{0}.$$

1) Show that this identity is equivalent to:

$$\wedge(\mathbf{a} \wedge \mathbf{b}) = \wedge(\mathbf{a}) \cdot \wedge(\mathbf{b}) - \wedge(\mathbf{b}) \cdot \wedge(\mathbf{a})$$

where $\wedge(\boldsymbol{\omega})$ is the skew symmetric matrix associated to the vector $\boldsymbol{\omega} \in \mathbb{R}^3$.

2) In the space of skew-symmetric matrices, the Lie bracket is defined as follows:

$$[\mathbf{A}, \mathbf{B}] = \mathbf{A} \cdot \mathbf{B} - \mathbf{B} \cdot \mathbf{A}.$$

Show that :

$$\wedge(\mathbf{a} \wedge \mathbf{b}) = [\wedge(\mathbf{a}), \wedge(\mathbf{b})].$$

3) Recall that the infinitesimal rotation following the skew symmetric matrix $\mathbf{A}$ is $\exp(\mathbf{A}\cdot dt) \simeq \mathbf{I} + \mathbf{A}\cdot dt + \frac{\mathbf{A}^2}{2}\cdot dt^2 + \circ(dt^2)$. Consider two skew-symmetric matrices $\mathbf{A}, \mathbf{B}$ and define the two infinitesimal rotation matrices

$$\mathbf{R}_a = e^{\mathbf{A}\cdot dt} \text{ and } \mathbf{R}_b = e^{\mathbf{B}\cdot dt}.$$

Compute a Taylor expansion of second order associated to the rotation given by

$$\mathbf{R}_a^{-1}\mathbf{R}_b^{-1}\mathbf{R}_a\mathbf{R}_b.$$

From this result, give on interpretation of the Lie bracket $[\mathbf{A}, \mathbf{B}] = \mathbf{AB} - \mathbf{BA}$.

4) Verify that the set $(\mathbb{R}^3, +, \wedge, \cdot)$ forms a *Lie algebra*.

EXERCISE 3.– *Varignon's formula*

See the correction video at https://youtu.be/dMhwZLTA_cA

Let us consider a solid body whose center of gravity remains at the origin of a Galilean coordinate system and is rotating around an axis Δ with a rotation vector of $\boldsymbol{\omega}$. Give the equation of the trajectory of a point $\mathbf{x}$ of the body.

EXERCISE 4.– *Quaternions*

See the correction video at https://youtu.be/d5dRd_SjDTU

Quaternion were discovered by W. R. Hamilton are also used to represent a rotation in a 3-dimensional space (see, *e.g.*, [9]). A quaternion $\mathring{q}$ is an extension of the complex number. It corresponds to a scalar s plus a vector $\mathbf{v}$ of $\mathbb{R}^3$. We use the equivalent following notations:

$$\mathring{q} = s + v_1 i + v_2 j + v_3 k = s + < v_1, v_2, v_3 > = s + < \mathbf{v} >,$$

where

$$i^2 = j^2 = k^2 = ijk = -1.$$

1) From these relations, fill the following multiplication table:

$\cdot$	1	i	j	k
1	1	i	j	k
i	i	?	?	?
j	j	?	?	?
k	k	?	?	?

2) A unit-quaternion $\mathring{q}$ is a quaternion with a unit magnitude:

$$|\mathring{q}|^2 = s^2 + v_1^2 + v_2^2 + v_3^2 = 1.$$

Consider the rotation obtained by a rotation around the unit vector $\mathbf{v}$ with an angle θ. From the Rodrigues formula, we know that the corresponding rotation matrix is $\exp(\theta \cdot \mathbf{v}\wedge)$. To this rotation, we associate the quaternion:

$$\mathring{q} = \cos(\frac{\theta}{2}) + \sin(\frac{\theta}{2}) \cdot < \mathbf{v} > .$$

Check that this quaternion has a unit magnitude.

3) Show that the quaternion $\mathring{q}$ and its opposite $-\mathring{q}$, both correspond to the same rotation in $\mathbb{R}^3$.

4) Assuming that the composition of rotations corresponds to a multiplication of quaternions, show that

$$(s + < \mathbf{v} >)^{-1} = s + < -\mathbf{v} > .$$

5) We consider the Euler rotation matrix with $\psi = \theta = \varphi = \frac{\pi}{2}$. It corresponds to a rotation around the unit vector $\mathbf{v}$ with an angle α. Give the values of $\mathbf{v}$ and α using 3 methods: the geometrical method (using hands only), the matrices and the quaternions.

Exercise 5.– *(Lie group $SE(2)$)*

See the correction video at https://youtu.be/vQWkOBpBdhk

The group of 2D rigid transformations of the plane is named *SE(2)*. The classical matrix representation is the following:

$$\mathbf{P} = \begin{pmatrix} \cos p_3 & -\sin p_3 & p_1 \\ \sin p_3 & \cos p_3 & p_2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}$$

where $\mathbf{t} = (p_1, p_2)$ is the translation and p_3 is the angle for the rotation $\mathbf{R}$.

1) Show that $SE(2)$ as defined above is a Lie group.

2) Using the derivatives around (at $\mathbf{p} = \mathbf{0}$), find the associated Lie algebra.

3) We consider the Dubins car

$$\begin{cases} \dot{x}_1 &= u_1 \cdot \cos x_3 \\ \dot{x}_2 &= u_1 \cdot \sin x_3 \\ \dot{x}_3 &= u_2 \end{cases}$$

The state vector $\mathbf{x} = (x_1, x_2, x_3)$ is a pose and can be represented by an element of $\mathbf{P} \in SE(2)$. For the integration, we should apply a formula similar to $\mathbf{R}(t + dt) = \mathbf{R}(t) \cdot \mathrm{Exp}(dt \cdot \boldsymbol{\omega}_r(t))$ (see equation 1.8) we had in $SO(3)$. Give the corresponding formula for $SE(2)$.

4) Assume that

$$\begin{array}{rcl} u_1(t) &=& 1 \\ u_2(t) &=& 1 \\ \mathbf{x}(0) &=& (1, 0, \frac{\pi}{2})^{\mathrm{T}} \end{array}$$

Write a program which integrate the trajectory with three different approches: exact, Euler and the exponential integration scheme and compare. We will take $t \in [0, 6]$ and a sampling time equal to $dt = 0.5$.

5) Using the interpolation formula, find a path $\mathbf{P}(t), t \in [0, 1]$ in SE(2) such that $\mathbf{P}(0) = \mathbf{P}_0, \mathbf{P}(1) = \mathbf{P}_1$, where $\mathbf{P}_0$ and $\mathbf{P}_1$ are the pose matrices associated to $\mathbf{x}(0) = (-1, -1, -1)^{\mathrm{T}}$ and $\mathbf{x}(1) = (1, 1, 1)^{\mathrm{T}}$, respectively.

EXERCISE 6.– *Car on the sphere*

See the correction video at https://youtu.be/ipgs4DBQosk

Consider a car moving in the plane described by the state equations:

$$\begin{cases} \dot{x}_1 &= x_4 \cos x_3 \\ \dot{x}_2 &= x_4 \sin x_3 \\ \dot{x}_3 &= u_1 \\ \dot{x}_4 &= u_2 \end{cases}$$

where (x_1, x_2, x_3) is the pose of the car, x_4 is the speed, u_1 is the rotation rate and u_2 is the acceleration (see 1.6, left).

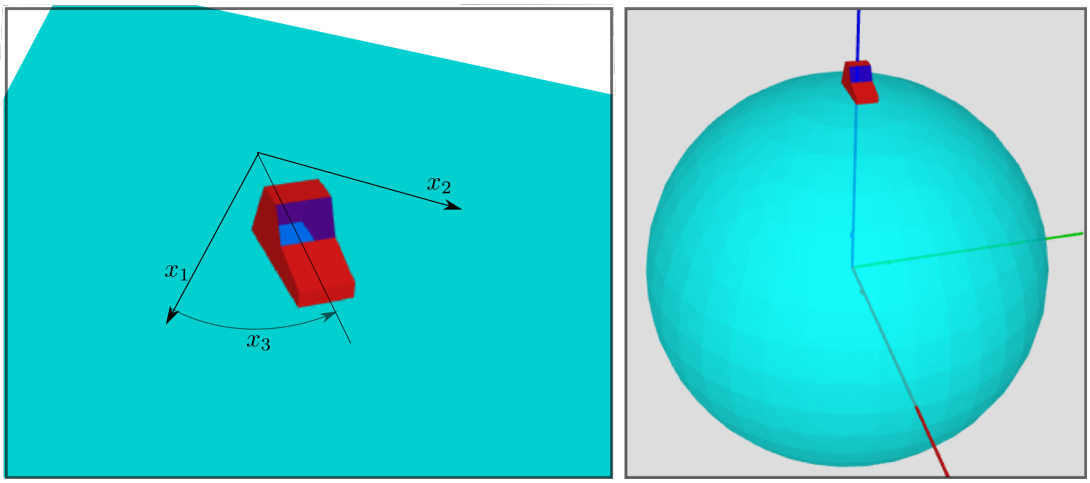

Figure 1.6: Left: car on the plane; Right: car on the sphere

For the input, we choose

$$\mathbf{u} = \begin{pmatrix} 0.1 \cdot \cos(0.1 \cdot t) + 0.02 \\ 1 - x_4 \end{pmatrix}$$

and for the initial state, we take $\mathbf{x} = (0, 0, 0, 0)$.

1) What is the shape of the manifold corresponding to the state space. Simulate the system for $t \in [0, 200]$.

2) We take the same car, with the same input $\mathbf{u}$. But now, the car moves on a sphere with radius $r = 20$ as illustrated by Figure 1.6, right. What is the shape of the state-space?

3) Provide a simulation of the car moving on the sphere. The simulation should avoid the singularities.

Chapter 2

Euler angles

2.1 Definition

In related literature, the angles proposed by Euler in 1770 to represent the orientation of solid bodies in space are not uniquely defined. We mainly distinguish between the roll-yaw-roll, roll-pitch-roll and roll-pitch-yaw formulations. It is the latter that we will choose since it is imposed in the mobile robotics. Within this roll-pitch-yaw formulation, the Euler angles are sometimes referred to as *Cardan angles*. Any rotation matrix of $\mathbb{R}^3$ can be expressed in the form of the product of three matrices as follows:

$$\mathbf{R}(\varphi, \theta, \psi) = \underbrace{e^{\psi \mathbf{k}\wedge}}_{\mathbf{R}_\psi} \cdot \underbrace{e^{\theta \mathbf{j}\wedge}}_{\mathbf{R}_\theta} \cdot \underbrace{e^{\varphi \mathbf{i}\wedge}}_{\mathbf{R}_\varphi}$$

where $\mathbf{i} = (1,0,0)^{\mathrm{T}}$, $\mathbf{j} = (0,1,0)^{\mathrm{T}}$, $\mathbf{k} = (0,0,1)^{\mathrm{T}}$. In developed form, we get:

$$\begin{pmatrix} \cos\theta\cos\psi & -\cos\varphi\sin\psi + \sin\theta\cos\psi\sin\varphi & \sin\psi\sin\varphi + \sin\theta\cos\psi\cos\varphi \\ \cos\theta\sin\psi & \cos\psi\cos\varphi + \sin\theta\sin\psi\sin\varphi & -\cos\psi\sin\varphi + \sin\theta\cos\varphi\sin\psi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{pmatrix} \tag{2.1}$$

$$\underbrace{\phantom{\begin{matrix}\cos\theta\cos\psi\\ \cos\theta\sin\psi\\ -\sin\theta\end{matrix}}}_{\mathbf{i}_1|_{\mathcal{R}_0}} \qquad \underbrace{\phantom{\begin{matrix}-\cos\varphi\sin\psi + \sin\theta\cos\psi\sin\varphi\\ \cos\psi\cos\varphi + \sin\theta\sin\psi\sin\varphi\\ \cos\theta\sin\varphi\end{matrix}}}_{\mathbf{j}_1|_{\mathcal{R}_0}} \qquad \underbrace{\phantom{\begin{matrix}\sin\psi\sin\varphi + \sin\theta\cos\psi\cos\varphi\\ -\cos\psi\sin\varphi + \sin\theta\cos\varphi\sin\psi\\ \cos\theta\cos\varphi\end{matrix}}}_{\mathbf{k}_1|_{\mathcal{R}_0}}$$

The angles φ, θ, ψ are the *Euler angles* and are respectively called the *bank*, the *elevation* and the *heading*. The terms *roll*, *pitch*, *yaw* are often employed, although they correspond, respectively, to variations of bank, elevation and heading.

Gimbal lock. When $\theta = \frac{\pi}{2}$ (the same effect happens as soon as $\cos\theta = 0$), we have

$$\begin{aligned}
\mathbf{R}(\varphi,\theta,\psi) &= \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \underbrace{\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{pmatrix}}_{= \begin{pmatrix} 0 & \sin\varphi & \cos\varphi \\ 0 & \cos\varphi & -\sin\varphi \\ -1 & 0 & 0 \end{pmatrix}} \\
&= \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \overbrace{\begin{pmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}} \\
&= \begin{pmatrix} \cos(\psi-\varphi) & -\sin(\psi-\varphi) & 0 \\ \sin(\psi-\varphi) & \cos(\psi-\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}
\end{aligned}$$

and thus,

$$\frac{d\mathbf{R}}{d\psi} = -\frac{d\mathbf{R}}{d\varphi}.$$

This corresponds to a singularity which tells us that when $\theta = \frac{\pi}{2}$, we cannot move on the manifold of rotation matrices $SO(3)$, in all directions using the Euler angles. Equivalently, this means that some trajectories $\mathbf{R}(t)$ cannot be followed by Euler angles.

Rotation matrix to Euler angles. Given a rotation matrix $\mathbf{R}$, we can easily find the three Euler angles by solving, following Equation (2.1), the equations:

$$\begin{cases} -\sin\theta = r_{31} & \\ \cos\theta\sin\varphi = r_{32} & \cos\theta\cos\varphi = r_{33} \\ \cos\theta\cos\psi = r_{11} & \cos\theta\sin\psi = r_{21} \end{cases}$$

By imposing $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, $\varphi \in [-\pi, \pi]$, $\psi \in [-\pi, \pi]$, we find:

$$\begin{aligned}
\varphi &= \text{atan2}(r_{32}, r_{33}) \\
\theta &= -\arcsin r_{31} \\
\psi &= \text{atan2}(r_{21}, r_{11})
\end{aligned}$$

Here atan2 is the two-argument arctangent function defined by

$$\alpha = \text{atan2}(y,x) \Leftrightarrow \alpha \in]-\pi,\pi] \text{ and } \exists r > 0 \mid \begin{cases} x = r\cos\alpha \\ y = r\sin\alpha \end{cases} \tag{2.2}$$

2.2 Rotation vector of a moving Euler matrix

Let us consider a solid body moving in a coordinate system $\mathcal{R}_0$ and a coordinate system $\mathcal{R}_1$ attached to this body (refer to Figure 2.1). The conventions chosen here are those of the

SNAME (*Society of Naval and Marine Engineers*). The two coordinate systems are assumed to be orthonormal. Let $\mathbf{R}(t) = \mathbf{R}(\psi(t), \theta(t), \varphi(t))$ be the rotation matrix that links the two systems. We need to find the instantaneous rotation vector $\boldsymbol{\omega}$ of the solid body relative to $\mathcal{R}_0$ in function of $\varphi, \theta, \psi, \dot{\varphi}, \dot{\theta}, \dot{\psi}$. We have:

$$\begin{array}{lll} & \dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}} & \\ \overset{(2.1)}{=} & \frac{d}{dt}\left(\mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{R}_\varphi\right) \cdot \mathbf{R}_\varphi^{\mathrm{T}} \cdot \mathbf{R}_\theta^{\mathrm{T}} \cdot \mathbf{R}_\psi^{\mathrm{T}} & \\ = & \left(\dot{\mathbf{R}}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{R}_\varphi + \mathbf{R}_\psi \cdot \dot{\mathbf{R}}_\theta \cdot \mathbf{R}_\varphi + \mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \dot{\mathbf{R}}_\varphi\right) \cdot \mathbf{R}_\varphi^{\mathrm{T}} \cdot \mathbf{R}_\theta^{\mathrm{T}} \cdot \mathbf{R}_\psi^{\mathrm{T}} & \\ \overset{(1.2)}{=} & \underbrace{\dot{\mathbf{R}}_\psi \cdot \mathbf{R}_\psi^{\mathrm{T}}}_{=\wedge(\dot{\psi}\mathbf{k})} + \mathbf{R}_\psi \cdot \underbrace{\dot{\mathbf{R}}_\theta \cdot \mathbf{R}_\theta^{\mathrm{T}}}_{=\wedge(\dot{\theta}\mathbf{j})} \cdot \mathbf{R}_\psi^{\mathrm{T}} + \mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \underbrace{\dot{\mathbf{R}}_\varphi \cdot \mathbf{R}_\varphi^{\mathrm{T}}}_{=\wedge(\dot{\varphi}\mathbf{i})} \cdot \mathbf{R}_\theta^{\mathrm{T}} \cdot \mathbf{R}_\psi^{\mathrm{T}} & \\ \overset{(1.3)}{=} & \wedge(\dot{\psi}\mathbf{k}) + \wedge(\dot{\theta} \cdot \mathbf{R}_\psi \cdot \mathbf{j}) + \wedge(\dot{\varphi} \cdot \mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{i}) & \\ = & \wedge(\dot{\psi}\mathbf{k} + \dot{\theta} \cdot \mathbf{R}_\psi \cdot \mathbf{j} + \dot{\varphi} \cdot \mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{i}). & \end{array}$$

Thus

$$\begin{array}{lll} \boldsymbol{\omega}|_{\mathcal{R}_0} & \overset{(1.2)}{=} & \wedge^{-1}\left(\dot{\mathbf{R}} \cdot \mathbf{R}^{\mathrm{T}}\right) \\ & = & \dot{\psi} \cdot \mathbf{k} + \dot{\theta} \cdot \mathbf{R}_\psi \cdot \mathbf{j} + \dot{\varphi} \cdot \mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{i}. \end{array}$$

If we compute the quantities $\mathbf{k}$, $\mathbf{R}_\psi \mathbf{j}$ and $\mathbf{R}_\psi \cdot \mathbf{R}_\theta \cdot \mathbf{i}$, we get the result:

$$\boldsymbol{\omega}|_{\mathcal{R}_0} = \begin{pmatrix} \cos\theta\cos\psi & -\sin\psi & 0 \\ \cos\theta\sin\psi & \cos\psi & 0 \\ -\sin\theta & 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}.$$

Combining with (2.1) and the fact that $\boldsymbol{\omega}_{|\mathcal{R}_0} = \mathbf{R}\left(\varphi, \theta, \psi\right) \cdot \boldsymbol{\omega}_{|\mathcal{R}_1}$, we get

$$\begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \tan\theta\sin\varphi & \tan\theta\cos\varphi \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \frac{\sin\varphi}{\cos\theta} & \frac{\cos\varphi}{\cos\theta} \end{pmatrix} \cdot \boldsymbol{\omega}_{|\mathcal{R}_1}. \tag{2.3}$$

We observe a singularity if $\cos\theta = 0$. We will therefore take care to never have an elevation θ equal to $\pm\frac{\pi}{2}$, to avoid the gimbal lock.

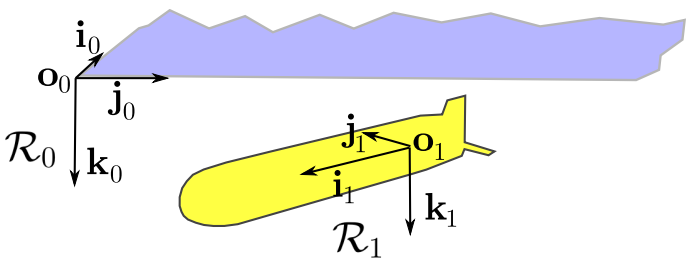

Figure 2.1: The coordinate system $\mathcal{R}_1 : (\mathbf{o}_1, \mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1)$ attached to the robot

Exercises

EXERCISE 7.– *Immersion*

See the correction video at https://youtu.be/E-XX52wbBQ8

Consider the pendulum described by the state equations:

$$\left\{ \begin{pmatrix} \dot{\theta} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} \omega \\ -\sin\theta - \frac{1}{2}\omega \end{pmatrix} \right.$$

1) Considering that the two states (θ, ω) and $(\theta + 2k\pi, \omega), k \in \mathbb{Z}$, are identical, show that the state space corresponds to a cylinder.

2) We want to represent the cylinder which is a two-dimensional manifold in the larger dimensional Cartesian space $\mathbb{R}^3$. We call this representation an *immersion*. Set $x_1 = \cos\theta$, $x_2 = \sin\theta$, $x_3 = \omega$. Give the state equations in the $\mathbf{x}$-space. Which constraint should satisfy $\mathbf{x}$?

3) Provide a Runge-Kutta simulation on the $\mathbf{x}$-space of the system. The initial vector is chosen as $(-1, 0, 6)$. Draw the corresponding trajectory and interpret.

EXERCISE 8.– *Car on the torus*

See the correction video at https://youtu.be/TcZ0f1iVB-A

Consider a car moving on the torus, parameterized by the two radius r_1, r_2, as illustrated by Figure 2.2. The pose of the car is represented by 3 angles x_1, x_2, x_3. The speed of the car is always equal to 1. The car has a single input u corresponding to its rotation rate with respect to the ground.

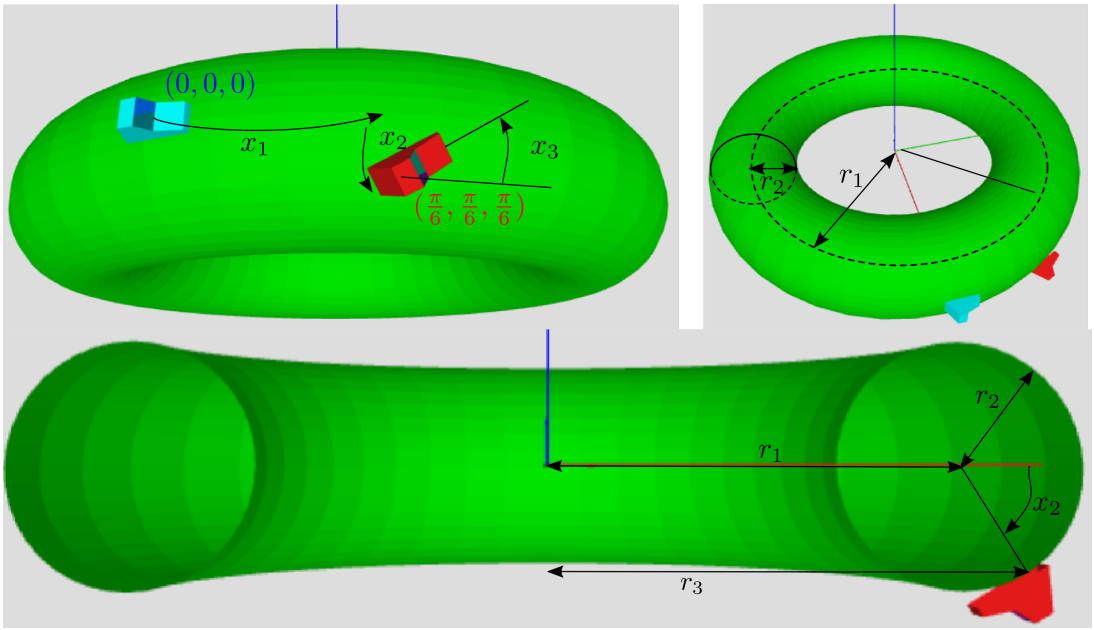

Figure 2.2: Representation of the pose of the car on the torus

1) What is the shape of the manifold corresponding to the state space. Give the state equations of the system.

2) Simulate the system for $r_1 = 10, r_2 = 6$ and $t \in [0, 100]$. The initial state is taken as $x_1 = x_2 = x_3 = 0$. For the control, we choose $u = 0.1$.

EXERCISE 9.– *Manipulator robot*

See the correction video at http://youtu.be/TPlHx0PE6P4

A manipulator robot, such as *Staubli* represented on Figure 2.3, is composed of several rigid arms. We retrieve the coordinates of the end effector, at the extremity of the robot, using a series of geometric transformations. We can show that a parametrization with four degrees of freedom allows to represent these transformations. There are several possible parametrizations, each with its own advantages and disadvantages. The most widely used one is probably the *Denavit-Hartenberg* parametrization. In the case where the articulations are rotational joints (as is the case of the Staubli robot where the joins can turn), the parametrization represented by the figure might prove to be practical since it makes drawing the robot easier. This transformation is the composition of four elementary transformations: (i) a translation of length r following z ; (ii) a translation of length d following x ; (iii) a rotation of α around y and (iv) a rotation of θ (the variable activated around z. Using the figure for drawing the arms and the photo for the robot, perform a realistic simulation of the robot's movement.

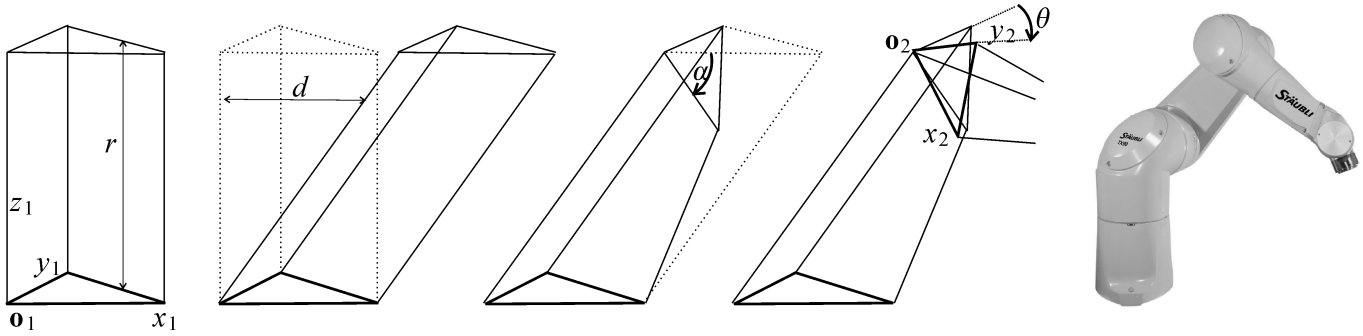

Figure 2.3: Parametrization for the direct geometric model of a manipulator robot

Chapter 3

Inertial unit

Assume that we are enclosed inside a box near to the Earth without any possibility to see the environment outside. Inside the box we may perform some inertial experiments such as observing bodies translating or rotating. These experiments allow us to measure indirectly the accelerations and the angular speed of the box in its own frame. The principle of an inertial unit is to estimate the pose (position and orientation) of the box from these inertial measurements only (called *proprioceptive measurements*), assuming that the initial pose is known.

3.1 Mechanization equations

For this purpose, we will describe the motion of our box by a kinematic model the inputs of which are the accelerations and the angular speeds these are directly measurable in the coordinate system of the box. The state vector is composed of

- the vector $\mathbf{p} = (p_x, p_y, p_z)$ that gives the coordinates of the center of the box expressed in the absolute inertial coordinate system $\mathcal{R}_0$,
- the orientation (which can either be given by the three Euler angles (φ, θ, ψ) or a rotation matrix $\mathbf{R}$) and
- the speed vector $\mathbf{v}_r$ of the box expressed in its own coordinate system $\mathcal{R}_1$.

The inputs of the system are

- the acceleration $\mathbf{a}_r = \mathbf{a}_{\mathcal{R}_1}$ of the center of the box also given in $\mathcal{R}_1$ and
- the vector $\boldsymbol{\omega}_r = \boldsymbol{\omega}_{\mathcal{R}_1/\mathcal{R}_0|\mathcal{R}_1} = (\omega_x, \omega_y, \omega_z)$ corresponding to the rotation vector of the box relative to $\mathcal{R}_0$ expressed in $\mathcal{R}_1$.

We have indeed to express $\mathbf{a}$, $\boldsymbol{\omega}$ in the coordinate system of the box since these quantities are generally measured *via* the sensors attached on it. The first state equation is:

$$\dot{\mathbf{p}} \overset{(1.10)}{=} \mathbf{R}\,(\varphi, \theta, \psi) \ \cdot \ \mathbf{v}_r.$$

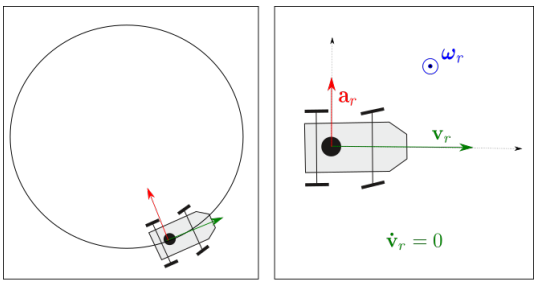

Figure 3.1: Illustration of the formula $\dot{\mathbf{v}}_r = \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r$ when $\dot{\mathbf{v}}_r = \mathbf{0}$. Left: the robot follows a circle; right: representation of the vectors in the robot frame.

Let us differentiate this equation. We obtain:

$$\ddot{\mathbf{p}} = \dot{\mathbf{R}} \cdot \mathbf{v}_r + \mathbf{R} \cdot \dot{\mathbf{v}}_r$$

with $\mathbf{R} = \mathbf{R}(\varphi, \theta, \psi)$. From this equation, we isolate $\dot{\mathbf{v}}_r$ to get

$$\dot{\mathbf{v}}_r = \underbrace{\mathbf{R}^{\mathrm{T}} \cdot \ddot{\mathbf{p}}}_{\mathbf{a}_r} - \mathbf{R}^{\mathrm{T}}\dot{\mathbf{R}} \cdot \mathbf{v}_r \overset{(1.4)}{=} \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r.$$

which constitutes the second state equation. Figure 3.1 shows a situation where a robot has a constant speed and follows a circle. The speed vector $\mathbf{v}_r$ is a constant whereas $\mathbf{a}_r$ is different from zero.

We obtain the *navigation mechanization equations* [10]:

$$\left\{ \begin{array}{rcl} \dot{\mathbf{p}} & = & \mathbf{R}(\varphi, \theta, \psi) \cdot \mathbf{v}_r \\ \begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} & \overset{(2.3)}{=} & \begin{pmatrix} 1 & \tan\theta \sin\varphi & \tan\theta \cos\varphi \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \frac{\sin\varphi}{\cos\theta} & \frac{\cos\varphi}{\cos\theta} \end{pmatrix} \cdot \boldsymbol{\omega}_r \\ \dot{\mathbf{v}}_r & = & \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{array} \right. \tag{3.1}$$

Using rotation matrix instead of Euler angles. From Equation (1.4), we have $\mathbf{R}^{\mathrm{T}}\dot{\mathbf{R}} = \wedge\left(\mathbf{R}^{\mathrm{T}}\boldsymbol{\omega}\right)$. Thus, the navigation mechanization equations can be written without any Euler angles as

$$\left\{ \begin{array}{rcl} \dot{\mathbf{p}} & = & \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} & = & \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) \\ \dot{\mathbf{v}}_r & = & \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{array} \right. \tag{3.2}$$

The main advantage of this representation is that we do not have the singularity that exists in (2.3) for $\cos\theta = 0$. But instead, we have redundancies, since $\mathbf{R}$ of dimension 9 replaces the 3 Euler angles.

3.2 With gravity

In case where there exists a gravity $\mathbf{g}(\mathbf{p})$ which depends on $\mathbf{p}$, the actual acceleration $\mathbf{a}_r$ corresponds to the gravity plus the measured acceleration $\mathbf{a}_r^{mes}$. In this case, the inertial unit can be

described by the following state equations

$$\left\{ \begin{array}{lcl} \dot{\mathbf{p}} & = & \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} & = & \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) \\ \dot{\mathbf{v}}_r & = & \mathbf{R}^{\mathrm{T}} \cdot \mathbf{g}(\mathbf{p}) + \mathbf{a}_r^{mes} - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{array} \right. \tag{3.3}$$

In this case, the input of the system are $\mathbf{a}_r^{mes}$ and $\boldsymbol{\omega}_r$ which are obtained by the accelerometers and the gyroscope [11]. To integrate this state equation for localization purpose, we need to know the initial state and a *gravity map* $\mathbf{g}(\mathbf{p})$.

3.3 Integration scheme

A pure inertial unit (without hybridization and without taking into account Earth's gravity) represents the robot by the kinematic model of Figure 3.2, which itself uses mechanization equations. The input vector $\mathbf{u} = (\mathbf{a}_r, \boldsymbol{\omega}_r)$ corresponds to the measured inertial inputs (accelerations and rotation speeds viewed by an observer on the ground, but expressed in the frame of the robot).

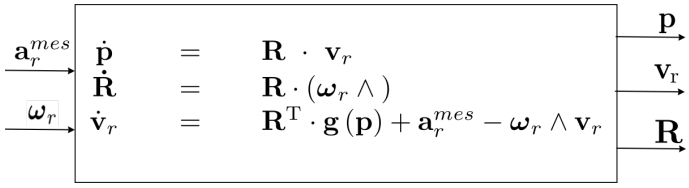

Figure 3.2: Navigation mechanization equations

We may use a numerical integration method such the Euler method:

$$\begin{array}{lcl} \mathbf{x}(t+dt) & := & \mathbf{x}(t) + dt \cdot \mathbf{f}(\mathbf{x}(t), \mathbf{R}(t), \mathbf{u}(t)) \\ \mathbf{R}(t+dt) & := & \mathbf{R}(t) \cdot e^{dt \cdot \boldsymbol{\omega}_r(t)} \end{array}$$

with

$$\begin{array}{rcl} \mathbf{x} & = & (\mathbf{p}, \mathbf{v}_r) \\ \mathbf{f}(\mathbf{x}, \mathbf{R}, \mathbf{u}) & = & \left(\begin{array}{c} \mathbf{R} \cdot \mathbf{v}_r \\ \mathbf{R}^{\mathrm{T}} \cdot \mathbf{g}(\mathbf{p}) + \mathbf{a}_r^{mes} - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{array} \right) \end{array}$$

If we use a Runge-Kutta integration scheme instead, we get

$$\begin{array}{lcl} \mathbf{x}(t+dt) & := & \mathbf{x}(t) + \frac{dt}{4} \cdot \mathbf{f}(\mathbf{x}(t), \mathbf{R}(t), \mathbf{u}(t)) + \frac{3dt}{4} \cdot \mathbf{f}\left(\mathbf{x}(t) + \frac{2dt}{3} \mathbf{f}(\mathbf{x}(t), \mathbf{R}(t), \mathbf{u}(t)), \mathbf{R}(t) \cdot e^{\frac{2}{3} dt \cdot \boldsymbol{\omega}_r(t)}, \mathbf{u}(t)\right) \\ \mathbf{R}(t+dt) & := & \mathbf{R}(t) \cdot e^{dt \cdot \boldsymbol{\omega}_r(t)} \end{array}$$

as illustrated by Figure 3.3.

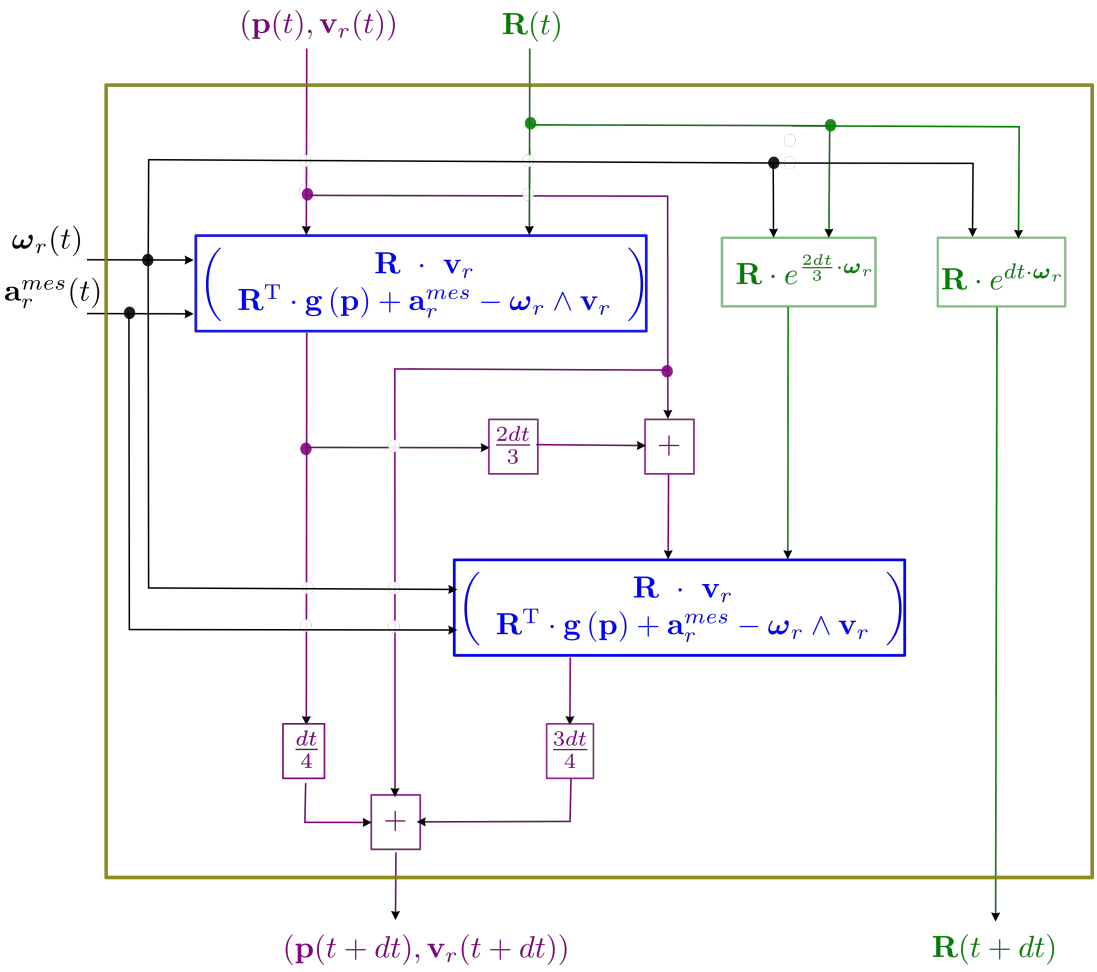

Figure 3.3: Runge Kutta for integrating the mechanization equations

3.4 Dead reckoning

For dead reckoning (*i.e.*, localization without external sensors) there are generally extremely precise laser gyrometers (around 0.001 deg/s.). These make use of the Sagnac effect (in a circular optical fiber turning around itself, the time taken by light to travel an entire round-trip depends on the path direction). Using three fibers, these gyrometers generate the vector $\boldsymbol{\omega}_r = (\omega_x, \omega_y, \omega_z)$. There are also accelerometers capable of measuring the acceleration $\mathbf{a}_r$ with a very high degree of precision. In pure inertial mode, we determine our position by integration of Equations (3.3) only using the acceleration $\mathbf{a}_r$ and the rotation speed $\boldsymbol{\omega}_r$, both expressed in the coordinate system of the box. In the case where we are measuring the quantity $\mathbf{v}_r$ (also expressed in the frame of the inertial unit attached to the robot) with a *Doppler velocity log*, we only need to integrate the first and the last of these three equations. Finally, when the robot is a correctly ballasted submarine or a terrestrial robot moving on a relatively plane ground, we know *a priori* that on average the bank and the elevation are equal to zero. We may therefore incorporate this information through a Kalman filter in order to limit the drift in positioning. An efficient inertial unit integrates an amalgamation of all the available information.

Exercises

Exercise 10.– *Schuler pendulum*

See the correction video at https://youtu.be/15pV-tfFLV8

One of the fundamental components of an inertial unit is the inclinometer. This sensor gives the vertical direction. Traditionally, we use a pendulum (or a plumb line) for this. However, when we are moving, due to the accelerations the pendulum starts to oscillate and it can no longer be used to measure the vertical direction. Here, we are interested in designing a pendulum for which any horizontal acceleration does not lead to oscillations. Let us consider a pendulum with two masses m at each end, situated at a distance ℓ_1 and ℓ_2 from the axis of rotation of the rod (see Figure 3.4). The axis moves over the surface of the Earth. We assume that ℓ_1 and ℓ_2 are small in comparison to Earth's radius r.

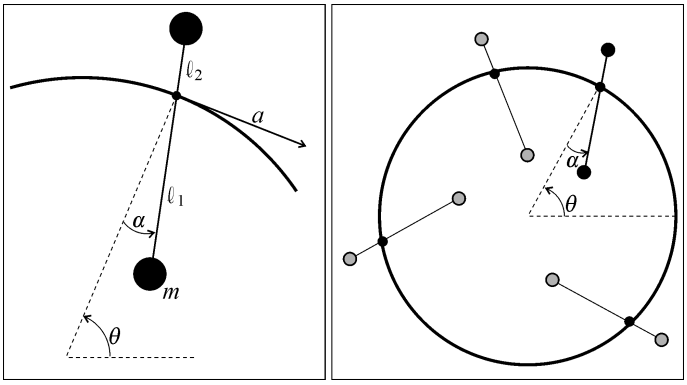

Figure 3.4: Inclinometer pendulum moving over the surface of the Earth

1) Find the state equations of the system.

2) Let us assume that $\alpha = \dot{\alpha} = 0$. For which values of ℓ_1 and ℓ_2 does the pendulum remain vertical, for any horizontal movement of the pendulum? What values does ℓ_2 have to take if we let $\ell_1 = 1$ m and we take $r = 6\ 400$ km for the Earth's radius?

3) Let us assume that, as a result of disturbances, the pendulum starts to oscillate. The period of these oscillations is called the *Schuler period*. Calculate this period.

4) Simulate the system graphically by taking a Gaussian white noise as the acceleration input. Since the system is conservative, an Euler integration method will not perform well (the pendulum

would gain energy). A higher-order integration scheme, such as that of *Runge Kutta*, should be used. This is given by:

$$\mathbf{x}(t+dt) \simeq \mathbf{x}(t) + dt \cdot \left(\frac{\mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right)}{4} + \tfrac{3}{4}\mathbf{f}(\mathbf{x}(t) + \frac{2dt}{3}\, \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \mathbf{u}(t + \tfrac{2}{3}dt)) \right).$$

For an easier graphical representation, we will take $r = 10$ m, $\ell_1 = 1$ m and $g = 10$ ms^{-2}. Discuss the results.

Exercise 11.– *Schuler oscillations in an inertial unit*

See the correction video at https://youtu.be/9vsZCnPXCPo

Consider the Earth which is considered as static, without rotation, with a gravity field directed toward the center and with a magnitude $g = 9.81ms^{-2}$. An inertial unit is fixed in a robot $\mathcal{R}_1$ at the surface or the Earth of radius r. Its position is with $\mathbf{p} = (r, 0, 0)$ and its Euler angles rate is $(\varphi, \theta, \psi) = (0, 0, 0)$ as in Figure 3.5. The robot $\mathcal{R}_1$ will always be static in the exercise.

1) Give the values for $\mathbf{a}_r^{mes}$ and $\boldsymbol{\omega}_r$ collected by the inertial unit of $\mathcal{R}_1$.

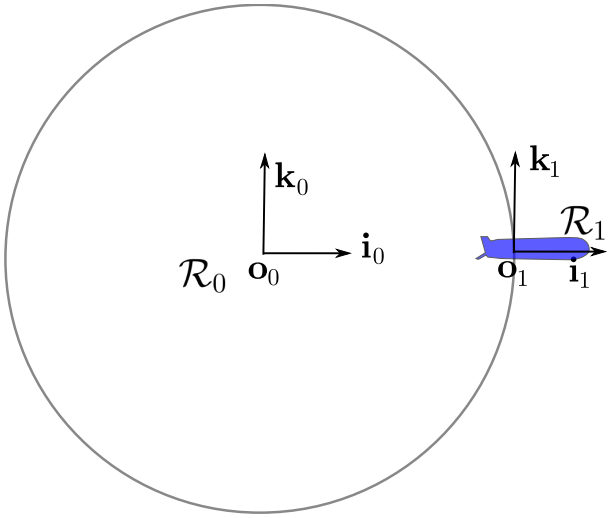

Figure 3.5: The inertial unit inside the robot is static

2) We now want to lure the inertial unit by initializing the robot at some other place, say $\mathcal{R}_2$, and finding a trajectory so that the unit believes that it is fixed in $\mathcal{R}_1$. More precisely, we want to find a motion $\mathbf{p}(t) = (x(t), y(t), z(t))$ for $\mathcal{R}_2$ so that the inertial unit senses exactly the inputs $\mathbf{a}_r^{mes}, \boldsymbol{\omega}_r$ of $\mathcal{R}_1$. For this, we assume that at time $t = 0$,$(\varphi, \theta, \psi) = (0, 0, 0)$ for $\mathcal{R}_2$. For simplicity, we also assume that $\mathcal{R}_2$ remains in the plane $y = 0$. Simplify the state equations for $\mathcal{R}_2$ in our specific case.

3) Provide a simulation of the trajectory of $\mathcal{R}_2$, initialized at $\mathbf{p}(0) = (r, 0, 1)^{\mathrm{T}}$. Interpret.

4) Linearize the state equation for $\mathcal{R}_2$ at the state vector corresponding to $\mathcal{R}_1$. Compute the eigen values and give an interpretation.

5) Assuming that the inertial unit is not perfectly initialized, what can you conclude for the corresponding error propagation.

EXERCISE 12.– *3D robot graphics*

See the correction video at https://youtu.be/t4ay2KENg-k

Drawing two- or three-dimensional robots or objects on the screen is widely used for simulation in robotics. The classic method (used by OPENGL) relies on modeling the posture of objects using a series of affine transformations (rotations, translations, homotheties) of the form:

$$\mathbf{f}_i : \begin{array}{ccc} \mathbb{R}^n & \to & \mathbb{R}^n \\ \mathbf{x} & \mapsto & \mathbf{A}_i\mathbf{x}+\mathbf{b}_i \end{array}$$

with $n = 2$ or 3. However, the manipulation of compositions of affine functions is less simple than that of linear applications. The idea of the transformation in *homogeneous coordinates* is to transform a system of affine equations into a system of linear equations. Notice first of all that an affine equation of the type $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ can be written as:

$$\begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}.$$

We will thus define the *homogeneous transformation* of a vector as follows:

$$\mathbf{x} \mapsto \mathbf{x}_{\rm h} = \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}.$$

Thus, an equation of the type:

$$\mathbf{y} = \mathbf{A}_3\left(\mathbf{A}_2\left(\mathbf{A}_1\mathbf{x}+\mathbf{b}_1\right)+\mathbf{b}_2\right)+\mathbf{b}_3$$

involving the composition of three affine transformations, can be written as:

$$\mathbf{y}_{\rm h} = \begin{pmatrix} \mathbf{A}_3 & \mathbf{b}_3 \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{A}_2 & \mathbf{b}_2 \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{A}_1 & \mathbf{b}_1 \\ \mathbf{0} & 1 \end{pmatrix} \mathbf{x}_{\rm h}.$$

A *sketch* is a matrix with two or three rows (depending on the object being in the plane or in space) and n columns representing the n vertices of a rigid polygon embodying the object. It is important that the union of all the segments formed by two consecutive points of the sketch forms all the vertices of the polygon that we wish to represent.

1) Let us consider the underwater robot (or AUV for *Autonomous Underwater Vehicle*) whose sketch matrix in homogeneous coordinates is :

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & 10 & 0 & 0 & 10 & 0 & 0 \\ -1 & 1 & 0 & -1 & -0.2 & 0 & 0.2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Draw this sketch in perspective view on a piece of paper.

2) The state equations of the robot are the following:

$$\left\{\begin{array}{rcl} \dot{\mathbf{p}} & = & \begin{pmatrix} v\cos\theta\cos\psi \\ v\cos\theta\sin\psi \\ -v\sin\theta \end{pmatrix} \\ \dot{v} & = & u_1 \\ \begin{pmatrix} \dot{\varphi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} & = & \begin{pmatrix} 1 & \tan\theta\sin\varphi & \tan\theta\cos\varphi \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \frac{\sin\varphi}{\cos\theta} & \frac{\cos\varphi}{\cos\theta} \end{pmatrix} \cdot \begin{pmatrix} -0.1\ \sin\varphi\cdot\cos\theta \\ v\cdot u_2 \\ v\cdot u_3 \end{pmatrix} \end{array}\right.$$

where (φ, θ, ψ) are the three Euler angles. The inputs of the system are the tangential acceleration u_1, the pitch rudder u_2 and the yaw rudder u_3. The state corresponds to $(\mathbf{p}, v, \varphi, \theta, \psi)$. Program a function able to draw the robot in 3D together with its shadow, in the plane x-y. Verify that the drawing is correct by moving the six degrees of freedom of the robot one by one. Obtain a three-dimensional representation such as the one illustrated in Figure 3.6.

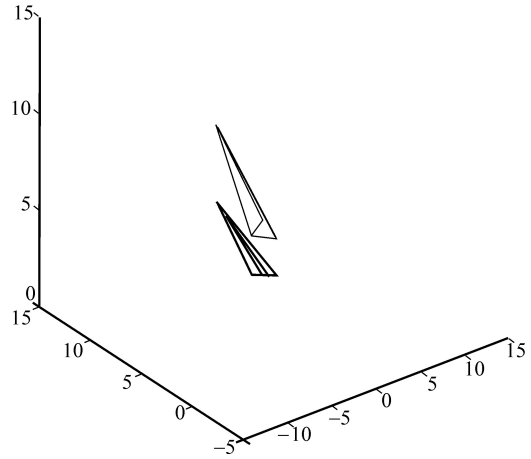

Figure 3.6: 3D representation of the robot together with its shadow in the horizontal plane

3) Simulate the robot in various conditions. In the simulation, draw the instantaneous rotation vector of the robot.

4) Provide a model and a simulation which uses a rotation matrix for the orientation instead of the Euler angles. Compare with the simulation made previously.

Chapter 4

Dynamic modeling

4.1 Principle

A robot (airplane, submarine, boat) can often be considered as a solid body whose inputs are the (tangential and angular) accelerations. These quantities are analytic functions of the forces that are at the origin of the robot's movement. For the dynamic modeling of a submarine the reference work is the book of Fossen [12], but the related notions can also be used for other types of solid robots such as planes, boats or quadrotors. In order to obtain a dynamic model, it is sufficient to take the kinematic equations and to consider that the angular and tangential accelerations are caused by forces and torques which become the new inputs of our system. The link between the accelerations and the forces is done by Newton's second law (or the *fundamental principle of dynamics*). Thus, for instance if $\mathbf{f}$ is the external force expressed in the inertial frame and m is the mass of the robot, we have

$$m\ddot{\mathbf{p}} = \mathbf{f}.$$

Since the speed and accelerations are generally measured by sensors embedded in the system, we generally prefer to express the speed and accelerations in the frame of the robot:

$$m\mathbf{a}_r = \mathbf{f}_r,$$

where $\mathbf{f}_r$ is the applied forces vector expressed in the robot frame. The same type of relation, known as the *Euler's rotation equation*, exists for rotations. It is given by

$$\mathbf{I}\dot{\boldsymbol{\omega}}_r + \boldsymbol{\omega}_r \wedge (\mathbf{I}\boldsymbol{\omega}_r) = \boldsymbol{\tau}_r \tag{4.1}$$

where $\boldsymbol{\tau}_r$ is the applied torque and $\boldsymbol{\omega}_r$ is the rotation vector both expressed in the robot frame. The inertia matrix $\mathbf{I}$ is attached to the robot (*i.e.*, computed in the robot frame) and we generally choose the robot frame to make $\mathbf{I}$ diagonal. Recall that the inertia matrix of a solid body occupying the volume V is

$$\mathbf{I} = \int_V \rho(x,y,z) \begin{pmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{pmatrix} dx\, dy\, dz$$

where $\rho(x,y,z)$ is the density.

Relation 4.1 is a consequence of the *Euler's second law* which states that in an inertial frame, the time derivative of the angular momentum $\mathcal{L} = \mathbf{R}\cdot\mathbf{I}\cdot\mathbf{R}^{\mathrm{T}}\cdot\boldsymbol{\omega}$ equals the applied torque $\boldsymbol{\tau}$. In the inertial frame, this can be expressed as

$$\begin{array}{rl} & \frac{d}{dt}\mathcal{L} = \boldsymbol{\tau} \\ \Leftrightarrow & \frac{d}{dt}\left(\mathbf{R}\cdot\mathbf{I}\cdot\boldsymbol{\omega}_r\right) = \mathbf{R}\cdot\boldsymbol{\tau}_r \\ \Leftrightarrow & \dot{\mathbf{R}}\mathbf{I}\cdot\boldsymbol{\omega}_r + \mathbf{R}\cdot\mathbf{I}\cdot\dot{\boldsymbol{\omega}}_r = \mathbf{R}\cdot\boldsymbol{\tau}_r \\ \Leftrightarrow & \mathbf{R}^{\mathrm{T}}\dot{\mathbf{R}}\mathbf{I}\cdot\boldsymbol{\omega}_r + \mathbf{I}\cdot\dot{\boldsymbol{\omega}}_r = \boldsymbol{\tau}_r \\ \Leftrightarrow & \boldsymbol{\omega}_r \wedge (\mathbf{I}\boldsymbol{\omega}_r) + \mathbf{I}\dot{\boldsymbol{\omega}}_r = \boldsymbol{\tau}_r. \end{array}$$

4.2 Equation of a 3D robot

The state equations for the robot are given by

$$\left\{\begin{array}{lcll} \dot{\mathbf{p}} & = & \mathbf{R}\ \cdot\ \mathbf{v}_r & (i) \\ \dot{\mathbf{R}} & = & \mathbf{R}\cdot(\boldsymbol{\omega}_r\wedge\ ) & (ii) \\ \dot{\mathbf{v}}_r & = & \mathbf{R}^{\mathrm{T}}\cdot\mathbf{g} + \frac{1}{m}\cdot\mathbf{f}_r - \boldsymbol{\omega}_r\wedge\mathbf{v}_r & (iii) \\ \dot{\boldsymbol{\omega}}_r & = & \mathbf{I}^{-1}\cdot(\boldsymbol{\tau}_r - \boldsymbol{\omega}_r\wedge\mathbf{I}\cdot\boldsymbol{\omega}_r) & (iv) \end{array}\right. \tag{4.2}$$

where $\mathbf{p}$ is the position, $\mathbf{R}$ is the orientation matrix of the robot, $\mathbf{v}_r$ the speed in the robot frame, $\boldsymbol{\omega}_r$ the rotation vector in the robot frame and $\mathbf{g} = (0,0,g)$ is the gravity. The three first equations (i), (ii), (iii) correspond to the kinematic equations and have already been derived (see Equation 3.3). Note that in Equation (iii), the acceleration $\mathbf{a}_r$ (expressed in the robot frame) comes from the Newton's second law where $\mathbf{f}_r$ is the *resultant*. Equation (iv) comes from the Euler's rotation equation (4.1) where $\boldsymbol{\tau}_r$ is the torque vector.

4.3 Modeling a quadrotor

As an illustration, we consider a quadrotor (see Figure 4.2) for which we want a dynamic model. The robot has four propellers that can be tuned independently. This will allow us to control the attitude and position of the robot by changing the speeds of the motors. The drone matrice 600 represented on Figure 4.1 has 6 propelers but can also be considered has a quadrotor since all its propellers have a vertical orientation.

Figure 4.1: Matrice 600 of ENSTA-Bretagne

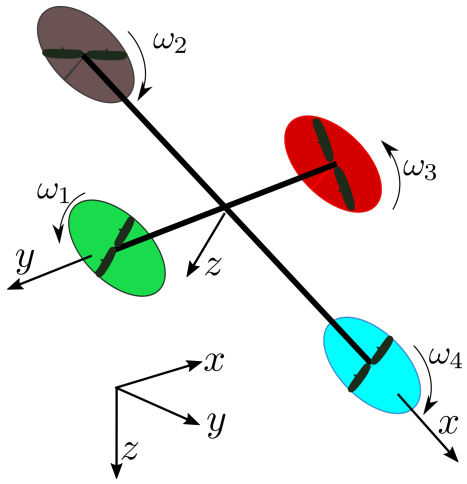

Figure 4.2: Quadrotor

We distinguish the front/rear propellers (blue and black), rotating clockwise and right/left propellers (red and green), rotating counterclockwise. The value of the force generated by the ith propeller is proportional to the squared speeds of the rotors, *i.e.*, is equal to $\beta \cdot \omega_i \cdot |\omega_i|$, where β is the thrust factor. Denote by δ the drag factor and ℓ the distance between any rotor and the center of the robot. The forces and torques generated on the robot are

$$\begin{pmatrix} \tau_0 \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} = \begin{pmatrix} \beta & \beta & \beta & \beta \\ -\beta\ell & 0 & \beta\ell & 0 \\ 0 & -\beta\ell & 0 & \beta\ell \\ -\delta & \delta & -\delta & \delta \end{pmatrix} \cdot \begin{pmatrix} \omega_1 \cdot |\omega_1| \\ \omega_2 \cdot |\omega_2| \\ \omega_3 \cdot |\omega_3| \\ \omega_4 \cdot |\omega_4| \end{pmatrix}$$

where τ_0 is the total thrust generated by the rotors and τ_1, τ_2, τ_3 are the torques generated by differences in the rotor speeds.

Since the resultant and the torque are given by

$$\mathbf{f}_r = \begin{pmatrix} 0 \\ 0 \\ -\tau_0 \end{pmatrix} \text{ and } \boldsymbol{\tau}_r = \begin{pmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix},$$

from (4.2), we get that the state equations for the quadrotor are given by

$$\begin{cases} \dot{\mathbf{p}} &= \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} &= \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) \\ \dot{\mathbf{v}}_r &= \mathbf{R}^{\mathrm{T}} \cdot \mathbf{g} + \begin{pmatrix} 0 \\ 0 \\ -\frac{\tau_0}{m} \end{pmatrix} - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \\ \dot{\boldsymbol{\omega}}_r &= \mathbf{I}^{-1} \cdot (\boldsymbol{\tau}_r - \boldsymbol{\omega}_r \wedge (\mathbf{I} \cdot \boldsymbol{\omega}_r)) \end{cases}$$

where $\mathbf{p}$ is the position and $\mathbf{R}$ is the orientation matrix of the robot.

Exercises

EXERCISE 13.– *Modeling an underwater robot*

See the correction video at https://youtu.be/Ox8zavW9WYw

The robot we will be modeling is the Redermor (*greyhound of the sea* in the Breton language). It is represented on Figure 4.3. It is an entirely autonomous underwater robot. This robot, developed by GESMA (Groupe d'Etude Sous-Marine de l'Atlantique - *Atlantic underwater research group*), has a length of 6 m, a diameter of 1 m and a weight of 3 800 kg. It has an efficient propulsion and control system with the aim of finding mines on the seabed.

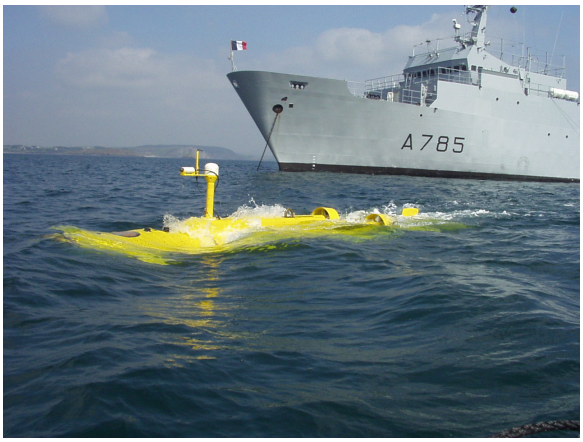

Figure 4.3: *Redermor* built by GESMA (Groupe d'Etude Sous-Marine de l'Atlantique - *Atlantic underwater research group*), on the water surface, still close to the boat it was launched from

Let us build a local coordinate system $\mathcal{R}_0 : (\mathbf{o}_0, \mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0)$ over the area traveled by the robot. The point $\mathbf{o}_0$ is placed on the surface of the ocean. The vector $\mathbf{i}_0$ indicates north, $\mathbf{j}_0$ indicates east and $\mathbf{k}_0$ is oriented towards the center of the Earth. Let $\mathbf{p} = (p_x, p_y, p_z)$ be the coordinates of the center of the robot expressed in the coordinate system $\mathcal{R}_0$. The state variables of the underwater robot are its position $\mathbf{p}$ in the coordinate system $\mathcal{R}_0$, its tangential v and its three Euler angles φ, θ, ψ. Its inputs are the tangential acceleration $\dot{v}$ as well as three control rudders which act respectively on

$\omega_x, \omega_y, \omega_z$. More formally, we have:

$$\begin{cases} u_1 &= \dot{v} \\ vu_2 &= \omega_y \\ vu_3 &= \omega_z \\ u_4 &= \omega_x \end{cases}$$

where the factor v preceding u_2, u_3 indicates that the robot is only able to turn left/right (through u_3) or up/down (through u_2) when it is advancing. Give the kinematic state model for this system.

EXERCISE 14.– *Dzhanibekov effect*

See the correction video at https://youtu.be/EA1Rh5MgGKI

The Dzhanibekov effect corresponds to the instability of the rotation of a rigid body object around its second principal axes. We consider a parallelepiped of mass m given in Figure 4.4. We assume that the density is uniform.

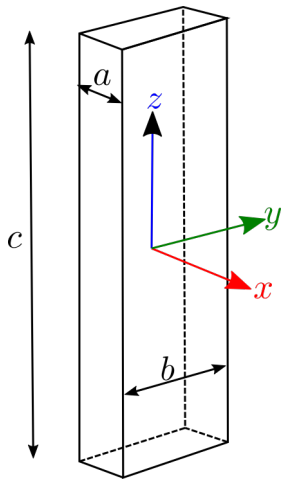

Figure 4.4: Parallelepiped body used to illustrate the Dzhanibekov effect

1) Recall that the inertia matrix of a solid body occupying the volume V is

$$\mathbf{I} = \int_V \rho(x,y,z) \begin{pmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{pmatrix} dx\, dy\, dz$$

where $\rho(x,y,z)$ is the density. Compute the inertia matrix of the parallelepiped.

2) The Euler equation of a free rigid body is given by

$$\dot{\boldsymbol{\omega}}_r = -\mathbf{I}^{-1} \cdot (\boldsymbol{\omega}_r \wedge (\mathbf{I} \cdot \boldsymbol{\omega}_r))$$

where $\boldsymbol{\omega}_r = (\omega_1, \omega_2, \omega_3)$ is the rotation vector expressed in the body frame. Give the state equation governing the rotation of our parallelepiped. The state vector is taken as $(\omega_1, \omega_2, \omega_3)$.

3) Give the equilibrium points of the system is the state-space. Conclude about about the rotation of any solid asteroid in the rotating space.

4) Study the stability of the rotation along the steady states.

5) Take $a = 0.4, b = 1, c = 3, m = 1$. Illustrate by a simulation, an unstable situation with a rotation of the parallelepiped rotating along one of its principle axis.

EXERCISE 15.– *Euler field*

See the correction video at https://youtu.be/in5wrQkVv4A

The goal of this exercise is to provide a geometrical interpretation of the behavior of a solid body in rotation. We consider the Euler equation given by

$$\dot{\boldsymbol{\omega}}_r = -\mathbf{I}^{-1} \cdot (\boldsymbol{\omega}_r \wedge (\mathbf{I} \cdot \boldsymbol{\omega}_r))$$

where $\boldsymbol{\omega}_r$ is the rotation vector expressed in the body frame and $\mathbf{I}$ is the inertia matrix given by

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

1) Show that the kinetic energy given by

$$E_K = \frac{1}{2} \boldsymbol{\omega}_r^{\mathrm{T}} \mathbf{I} \boldsymbol{\omega}_r$$

is constant.

2) Draw in three dimensional figure, the ellipsoid $\mathcal{E}$ corresponding to the set of all $\boldsymbol{\omega}_r$ such that energy equal to $E_0 = 10$J. Draw the vector field describing the evolution of $\boldsymbol{\omega}_r$ on this ellipsoid.

3) Simulate with a Runge-Kutta method the evolution of the rotation vector for an initial condition $\boldsymbol{\omega}_r(0) = (4, \sqrt{2}, 0)$. Draw the corresponding trajectory on the ellipsoid $\mathcal{E}$. Interpret the result.

4) Simulate an evolution for $\boldsymbol{\omega}_r(0) = (0.01, \sqrt{10}, 0)$. Conclude.

EXERCISE 16.– *Floating wheel*

See the correction video at https://youtu.be/mTh7n6mg03g

Consider a wheel floating and spinning in the space as shown on Figure 4.5.

We assume that the wheel is solid, similar to a homogeneous disk of mass $m = 1$ and radius $\rho = 1$. Its inertia matrix is given by:

$$\mathbf{I} = \begin{pmatrix} \frac{m\rho^2}{2} & 0 & 0 \\ 0 & \frac{m\rho^2}{4} & 0 \\ 0 & 0 & \frac{m\rho^2}{4} \end{pmatrix}$$

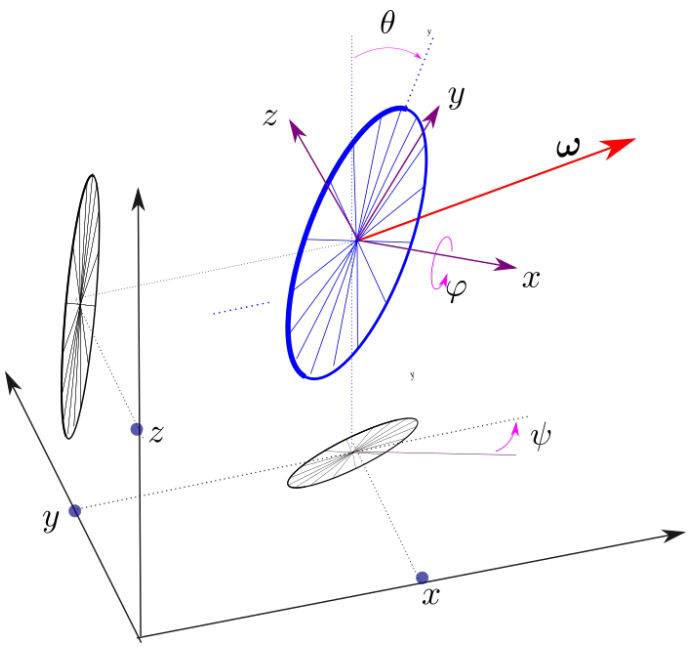

Figure 4.5: Floating wheel

1) Give the state equations of the wheel. The state variables we choose are (i) the coordinates $\mathbf{p} = (x, y, z)$ of the center of the wheel, (ii) the orientation (φ, θ, ψ) of the wheel, (iii) the speed $\mathbf{v}_r$ of the center of the wheel center in the wheel frame, and (iv) the rotation vector $\boldsymbol{\omega}_r$ expressed in the wheel frame. We thus have a system of order 12.

2) The torque-free precession occurs when the angular velocity vector changes its orientation with time. Provide a simulation illustrating the precession.

3) Show that the angular momentum $\mathcal{L} = \mathbf{R} \cdot \mathbf{I} \cdot \mathbf{R}^{\mathrm{T}} \cdot \boldsymbol{\omega}$ and the kinetic energy $E_K = \frac{1}{2}\mathcal{L}^{\mathrm{T}} \cdot \boldsymbol{\omega}$ are both constant. Verify with a simulation.

4) Using symbolic computation, compute the Jacobian matrix of the evolution function $\mathbf{f}$ at point

$$\mathbf{x}_0 = (0, 0, 0, 0, 0, 0, 0, 0, 0, \omega_x, \omega_y, \omega_z)^{\mathrm{T}} .$$

Give an interpretation.

Chapter 5

Inertial control

In this chapter, we consider the equations (4.2) given by

$$\left\{\begin{array}{lcll} \dot{\mathbf{p}} & = & \mathbf{R} \cdot \mathbf{v}_r & (i) \\ \dot{\mathbf{R}} & = & \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) & (ii) \\ \dot{\mathbf{v}}_r & = & \mathbf{R}^{\mathrm{T}} \cdot \mathbf{g} + \frac{1}{m} \cdot \mathbf{f}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r & (iii) \\ \dot{\boldsymbol{\omega}}_r & = & \mathbf{I}^{-1} \cdot (\boldsymbol{\tau}_r - \boldsymbol{\omega}_r \wedge \mathbf{I} \cdot \boldsymbol{\omega}_r) & (iv) \end{array}\right. \tag{5.1}$$

This model can be qualified as *inertial* since two inertial parameters are involved : the mass m and the inertia matrix $\mathbf{I}$. A control loop on such a system is also qualified as *inertial* since it requires the knowledge of these two inertial parameters only.

We show how to choose the forces $\mathbf{f}_r$ and the torque $\boldsymbol{\tau}_r$ to track a virtual pose represented by a desired position $\mathbf{p}_d(t)$ and a desired orientation $\mathbf{R}_d(t)$. We assume these quantities have a closed form and thus their two first derivatives $\dot{\mathbf{p}}_d, \ddot{\mathbf{p}}_d, \dot{\mathbf{R}}_d, \ddot{\mathbf{R}}_d$, are available.

5.1 Control the accelerations

We first note that if we choose

$$\left\{\begin{array}{lcl} \mathbf{f}_r & = & m\dot{\mathbf{v}}_r^d - m\mathbf{R}^{\mathrm{T}} \cdot \mathbf{g} + m\boldsymbol{\omega}_r \wedge \mathbf{v}_r \\ \boldsymbol{\tau}_r & = & \mathbf{I} \cdot \dot{\boldsymbol{\omega}}_r^d + \boldsymbol{\omega}_r \wedge \mathbf{I} \cdot \boldsymbol{\omega}_r \end{array}\right.$$

The system simplifies into

$$\left\{\begin{array}{lcl} \dot{\mathbf{p}} & = & \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} & = & \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) \\ \dot{\mathbf{v}}_r & = & \dot{\mathbf{v}}_r^d \\ \dot{\boldsymbol{\omega}}_r & = & \dot{\boldsymbol{\omega}}_r^d \end{array}\right.$$

where the new inputs are $\dot{\mathbf{v}}_r^d$ and $\dot{\boldsymbol{\omega}}_r^d$ the values we want for $\dot{\mathbf{v}}_r$ and $\dot{\boldsymbol{\omega}}_r$. We now have a model which is purely kinematic. The parameters $m, \mathbf{I}$ do not occur anymore.

5.2 Positioner

The *positioner* is a controller which computes the acceleration $\dot{\mathbf{v}}_r$ to apply to the robot to reach a desired target $\mathbf{p}_d(t)$. We define the error

$$\mathbf{e} = \mathbf{p}_d - \mathbf{p}.$$

To cancel this error, by choosing the right acceleration, we take here a proportional and derivative controller. For this, we want the error to satisfy the error equation

$$\ddot{\mathbf{e}} + \alpha_1 \dot{\mathbf{e}} + \alpha_0 \mathbf{e} = \mathbf{0}.$$

where α_0, α_1 are chosen to generate a stable characteristic polynomial.

In the robot frame, the error equation becomes $\ddot{\mathbf{e}}_r + \alpha_1 \dot{\mathbf{e}}_r + \alpha_0 \mathbf{e}_r = \mathbf{0}$, where

$$\begin{array}{rcccl} \mathbf{e}_r & = & \mathbf{R}^{\mathrm{T}}\mathbf{e} & = & \mathbf{R}^{\mathrm{T}}\left(\mathbf{p}_d - \mathbf{p}\right) \\ \dot{\mathbf{e}}_r & = & \mathbf{R}^{\mathrm{T}}\dot{\mathbf{e}} & = & \mathbf{R}^{\mathrm{T}}\left(\dot{\mathbf{p}}_d - \dot{\mathbf{p}}\right) = \mathbf{R}^{\mathrm{T}}\dot{\mathbf{p}}_d - \mathbf{v}_r \\ \ddot{\mathbf{e}}_r & = & \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{e}} & = & \mathbf{R}^{\mathrm{T}}\left(\ddot{\mathbf{p}}_d - \ddot{\mathbf{p}}\right) \\ & & & = & \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{p}}_d - \mathbf{R}^{\mathrm{T}}(\dot{\mathbf{R}}\mathbf{v}_r + \mathbf{R}\dot{\mathbf{v}}_r) \\ & & & = & \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{p}}_d - \boldsymbol{\omega}_r \wedge \mathbf{v}_r - \dot{\mathbf{v}}_r \end{array}$$

Thus

$$\begin{array}{ll} & \ddot{\mathbf{e}}_r + \alpha_1 \dot{\mathbf{e}}_r + \alpha_0 \mathbf{e}_r = \mathbf{0} \\ \Leftrightarrow & \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{p}}_d - \boldsymbol{\omega}_r \wedge \mathbf{v}_r - \dot{\mathbf{v}}_r + \alpha_1 \dot{\mathbf{e}}_r + \alpha_0 \mathbf{e}_r = \mathbf{0} \\ \Leftrightarrow & \dot{\mathbf{v}}_r = \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{p}}_d - \boldsymbol{\omega}_r \wedge \mathbf{v}_r + \alpha_1 \dot{\mathbf{e}}_r + \alpha_0 \mathbf{e}_r = \mathbf{0} \end{array}$$

The expression of the positioner is thus

$$\dot{\mathbf{v}}_r = \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{p}}_d - \boldsymbol{\omega}_r \wedge \mathbf{v}_r + \alpha_1(\mathbf{R}^{\mathrm{T}}\dot{\mathbf{p}}_d - \mathbf{v}_r) + \alpha_0 \mathbf{R}^{\mathrm{T}}(\mathbf{p}_d - \mathbf{p}).$$

This means that if we are able to produce the forces and torques that generate the acceleration $\dot{\mathbf{v}}_r$ then, the error $\mathbf{e} = \mathbf{p}_d - \mathbf{p}$ will converge to zero.

5.3 Orientator

The *orientator* is a controller which tells us how to change the rotation vector in order to follow a desired orientation $\mathbf{R}_d(t)$.

We need to recall that

$$\frac{d}{dt}\log(\mathbf{R}(t)) = \mathbf{R}^{\mathrm{T}}(t) \cdot \dot{\mathbf{R}}(t). \tag{5.2}$$

Indeed,

$$
\begin{array}{ll}
& \frac{d}{dt}\exp(\log(\mathbf{R}(t)) = \frac{d}{dt}\mathbf{R}(t) \\
\Rightarrow & \exp(\log(\mathbf{R}(t)) \cdot \frac{d}{dt}\log(\mathbf{R}(t)) = \dot{\mathbf{R}}(t) \\
\Rightarrow & \frac{d}{dt}\log(\mathbf{R}(t)) = \mathbf{R}^{\mathrm{T}}(t) \cdot \dot{\mathbf{R}}(t)
\end{array}
$$

In $SO(3)$, the orientation error can be represented by $\mathbf{R}_d \cdot \mathbf{R}^{\mathrm{T}}$. Now, this quantity cannot be qualified as an error since $SO(3)$ has no addition operator. In the corresponding Lie algebra, we define the error as

$$\mathbf{e} = \mathrm{Log}(\mathbf{R}_d\mathbf{R}^{\mathrm{T}})$$

where $\mathbf{e}$ corresponds to the rotation vector we have to follow for 1sec to go from $\mathbf{R}$ to $\mathbf{R}_d$. To cancel thus error, by choosing the right angular acceleration $\dot{\boldsymbol{\omega}}_r^d$, we can consider a proportional and derivative controller. For this, we want the error to satisfy

$$\ddot{\mathbf{e}} + \alpha_1\dot{\mathbf{e}} + \alpha_0\mathbf{e} = \mathbf{0},$$

where α_0, α_1 are chosen to generate a stable characteristic polynomial. Now,

$$
\begin{array}{rcl}
\mathbf{e}\wedge & = & \log(\mathbf{R}_d\mathbf{R}^{\mathrm{T}}) \\
\dot{\mathbf{e}}\wedge & = & \frac{d}{dt}\log(\mathbf{R}_d\mathbf{R}^{\mathrm{T}}) \\
& \overset{(5.2)}{=} & \mathbf{R}\mathbf{R}_d^{\mathrm{T}} \cdot \frac{d}{dt}(\mathbf{R}_d\mathbf{R}^{\mathrm{T}}) \\
& = & \mathbf{R}\mathbf{R}_d^{\mathrm{T}} \cdot (\mathbf{R}_d\dot{\mathbf{R}}^{\mathrm{T}} + \dot{\mathbf{R}}_d\mathbf{R}^{\mathrm{T}}) \\
& = & \underbrace{\mathbf{R}\dot{\mathbf{R}}^{\mathrm{T}}}_{=-(\boldsymbol{\omega}\wedge)} + \mathbf{R}\ \underbrace{\mathbf{R}_d^{\mathrm{T}}\dot{\mathbf{R}}_d}_{=(\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d)\wedge}\ \mathbf{R}^{\mathrm{T}} \\
& \overset{(1.3)}{=} & -(\boldsymbol{\omega}\wedge) + ((\mathbf{R}\cdot\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d)\wedge)
\end{array}
$$

Thus

$$
\begin{array}{rcl}
\mathbf{e} & = & \mathrm{Log}(\mathbf{R}_d\mathbf{R}^{\mathrm{T}}) \\
\dot{\mathbf{e}} & = & -\boldsymbol{\omega} + \mathbf{R}\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d \\
\ddot{\mathbf{e}} & = & -\dot{\boldsymbol{\omega}} + \dot{\mathbf{R}}\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \mathbf{R}\dot{\mathbf{R}}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \mathbf{R}\mathbf{R}_d^{\mathrm{T}}\dot{\boldsymbol{\omega}}_d
\end{array}
$$

In the robot frame, the error equation $\ddot{\mathbf{e}} + \alpha_1\dot{\mathbf{e}} + \alpha_0\mathbf{e} = \mathbf{0}$, becomes $\ddot{\mathbf{e}}_r + \alpha_1\dot{\mathbf{e}}_r + \alpha_0\mathbf{e}_r = \mathbf{0}$, where

$$
\begin{array}{rcl}
\mathbf{e}_r = \mathbf{R}^{\mathrm{T}}\mathbf{e} & = & \mathbf{R}^{\mathrm{T}}\mathrm{Log}(\mathbf{R}_d\mathbf{R}^{\mathrm{T}}) \\
\dot{\mathbf{e}}_r = \mathbf{R}^{\mathrm{T}}\dot{\mathbf{e}} & = & -\mathbf{R}^{\mathrm{T}}\boldsymbol{\omega} + \mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d \\
\ddot{\mathbf{e}}_r = \mathbf{R}^{\mathrm{T}}\ddot{\mathbf{e}} & = & -\underbrace{\mathbf{R}^{\mathrm{T}}\dot{\boldsymbol{\omega}}}_{=\dot{\boldsymbol{\omega}}_r} + \underbrace{\mathbf{R}^{\mathrm{T}}\dot{\mathbf{R}}}_{=\boldsymbol{\omega}_r\wedge}\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \dot{\mathbf{R}}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \mathbf{R}_d^{\mathrm{T}}\dot{\boldsymbol{\omega}}_d
\end{array}
\qquad (5.3)
$$

The differential equation for the error is

$$
\begin{array}{ll}
& \mathbf{0} = \ddot{\mathbf{e}}_r + \alpha_1\dot{\mathbf{e}}_r + \alpha_0\mathbf{e}_r \\
\Leftrightarrow & \mathbf{0} \overset{(5.3)}{=} -\dot{\boldsymbol{\omega}}_r + (\boldsymbol{\omega}_r\wedge)\cdot\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \dot{\mathbf{R}}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \mathbf{R}_d^{\mathrm{T}}\dot{\boldsymbol{\omega}}_d + \alpha_1\dot{\mathbf{e}}_r + \alpha_0\mathbf{e}_r \\
\Leftrightarrow & \dot{\boldsymbol{\omega}}_r = (\boldsymbol{\omega}_r\wedge)\mathbf{R}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \dot{\mathbf{R}}_d^{\mathrm{T}}\boldsymbol{\omega}_d + \mathbf{R}_d^{\mathrm{T}}\dot{\boldsymbol{\omega}}_d + \alpha_1\dot{\mathbf{e}}_r + \alpha_0\mathbf{e}_r
\end{array}
$$

As a consequence, to provide the robot an orientation $\mathbf{R}_d(t)$ we can take the following controller

	Orientator (in : $\mathbf{R}_d, \dot{\mathbf{R}}_d, \ddot{\mathbf{R}}_d, \mathbf{R}, \boldsymbol{\omega}_r$; out: $\dot{\boldsymbol{\omega}}_r^d$)
1	$\boldsymbol{\omega}_d = \wedge^{-1}(\dot{\mathbf{R}}_d \cdot \mathbf{R}_d^{\mathrm{T}})$
2	$\dot{\boldsymbol{\omega}}_d = \wedge^{-1}(\dot{\mathbf{R}}_d \cdot \dot{\mathbf{R}}_d^{\mathrm{T}} + \ddot{\mathbf{R}}_d \cdot \mathbf{R}_d^{\mathrm{T}})$
3	$\mathbf{e}_r = \mathbf{R}^{\mathrm{T}} \cdot \mathrm{Log}(\mathbf{R}_d \mathbf{R}^{\mathrm{T}})$
4	$\dot{\mathbf{e}}_r = -\boldsymbol{\omega}_r + \mathbf{R}_d^{\mathrm{T}} \boldsymbol{\omega}_d$
5	$\dot{\boldsymbol{\omega}}_r^d = \left((\boldsymbol{\omega}_r \wedge) \mathbf{R}_d^{\mathrm{T}} + \dot{\mathbf{R}}_d^{\mathrm{T}} \right) \cdot \boldsymbol{\omega}_d + \mathbf{R}_d^{\mathrm{T}} \dot{\boldsymbol{\omega}}_d + \alpha_1 \dot{\mathbf{e}}_r + \alpha_0 \mathbf{e}_r$

We can take a characteristic polynomial for the error which is stable as for instance: $(s+1) = s^2 + 2s + 1$. In this case, $\alpha_0 = 1$ and $\alpha_1 = 2$.

5.4 Controller

As illustrated by Figure 5.1, we are now able to generate the right resultant $\mathbf{f}_r$ and the right torque $\boldsymbol{\tau}_r$ in order to track a time dependent position $\mathbf{p}_d(t)$ with the right time dependent orientation $\mathbf{R}_d(t)$.

To get generate the torque $\boldsymbol{\tau}_r$, we can either use the external forces such as propellers or use inertial disks inside the robot.

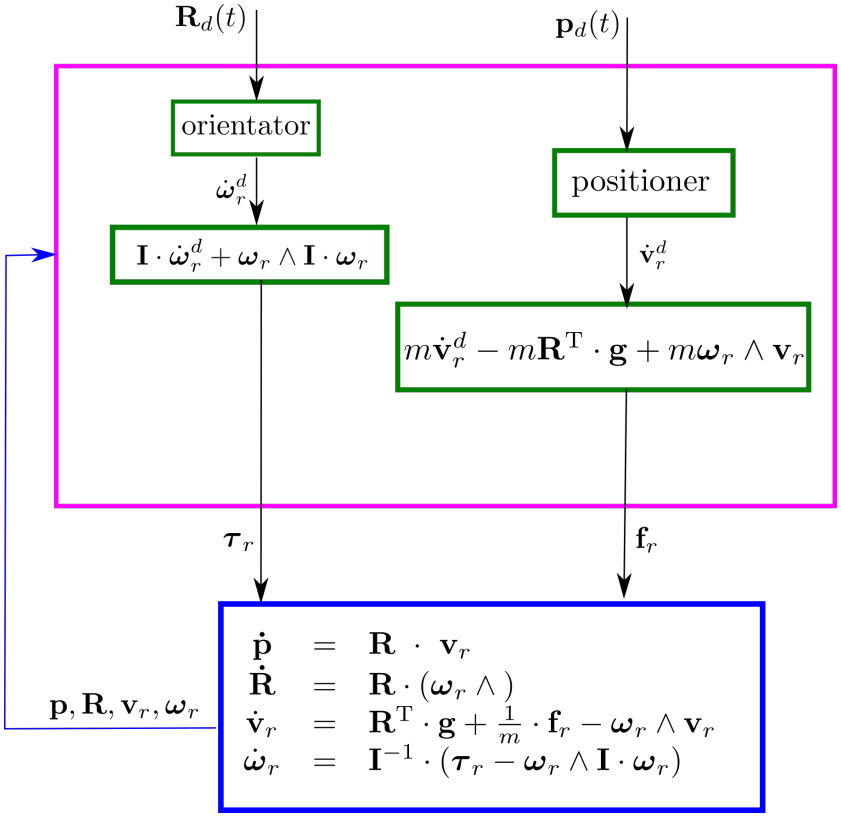

Figure 5.1: Orientator and positioner

5.5 Two-dimensional robots

If the robot moves in the plane, then, we can still apply Equation 5.1 which now becomes simpler. This is illustrated by Figure 5.2. For a two dimensional robot, we get

$$\mathbf{R} = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{p} = \begin{pmatrix} p_1 \\ p_2 \\ 0 \end{pmatrix}, \boldsymbol{\omega}_r = \begin{pmatrix} 0 \\ 0 \\ \omega \end{pmatrix},$$

$$\boldsymbol{\omega}_r\wedge = \begin{pmatrix} 0 & -\omega & 0 \\ \omega & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \dot{\boldsymbol{\omega}}_r = \begin{pmatrix} 0 \\ 0 \\ \dot{\omega} \end{pmatrix}, \mathbf{f}_r = \begin{pmatrix} f_{r1} \\ f_{r2} \\ 0 \end{pmatrix}$$

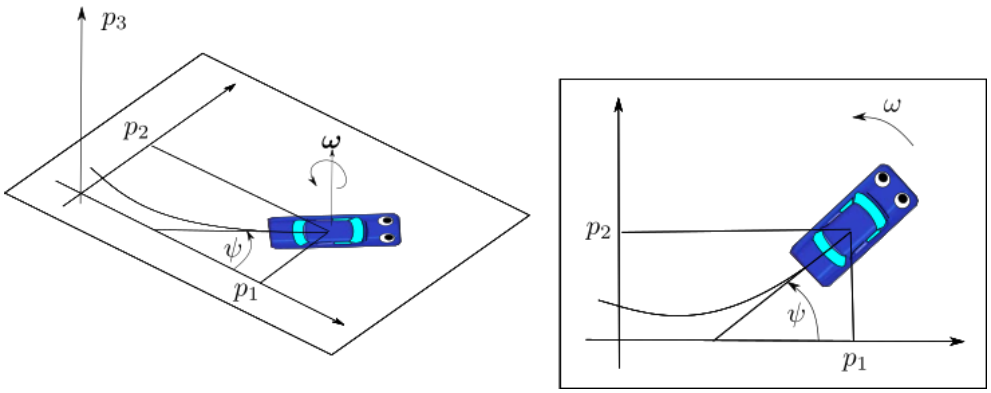

Figure 5.2: Robot in the plane

If we select only the x, y components as represented in Figure 5.2, right, we get

$$\begin{cases} \dot{\mathbf{p}} &= \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} &= \mathbf{R} \cdot \omega\wedge \\ \dot{\mathbf{v}}_r &= \frac{1}{m} \cdot \mathbf{f}_r - (\omega\wedge)\mathbf{v}_r \\ \dot{\omega} &= \frac{\tau}{I_{33}} \end{cases} \tag{5.4}$$

In this equation, ω is a scalar and we have

$$\omega\wedge = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}.$$

There is no need to specify if it is in the robot frame or in the world frame: they are the same. Moreover,

$$\mathbf{R} = \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix}$$

and the vectors $\mathbf{v}_r, \mathbf{f}_r$ are both two dimensional. The main difference between Equations (4.2) and (5.4) is the Euler equation which collapses into $\dot{\omega} = \frac{\tau}{I_{33}}$. This is due to the fact that ω,τ are now both scalar instead of three dimensional vectors.

The equations for the orientator and the positioner remain valid.

Exercises

EXERCISE 17.– *Lie bracket for control*

See the correction video at https://youtu.be/PfitQ7ZuNZs

The motivation of Lie bracket for control is illustrated by Figure 5.3 on the car parking problem. We would like to move the blue car sideway between the two red cars already parked. Moving sideway is not possible, but we can approach this direction using many small forward-backward maneuvers. Building this new sideway direction corresponds to the Lie bracket operator between two dynamics. The goal of this exercise is two illustrate Lie brackets on a simple Dubins car.

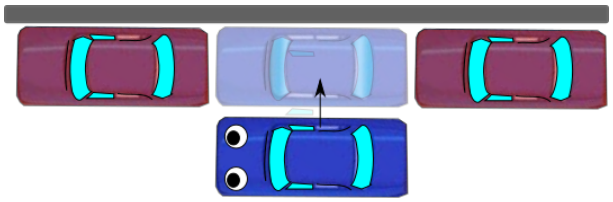

Figure 5.3: The inertial unit inside the robot is static

Consider two vector fields $\mathbf{f}$ and $\mathbf{g}$ of $\mathbb{R}^n$ corresponding to two state equations $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ and $\dot{\mathbf{x}} = \mathbf{g}(\mathbf{x})$. We define the Lie bracket between these two vector fields as

$$[\mathbf{f}, \mathbf{g}] = \frac{d\mathbf{g}}{d\mathbf{x}} \cdot \mathbf{f} - \frac{d\mathbf{f}}{d\mathbf{x}} \cdot \mathbf{g}.$$

We can check that the set of vector fields equipped with the Lie bracket is a Lie algebra. The proof is not difficult but tedious and will not be asked in this exercise.

1) Consider the linear vector fields $\mathbf{f}(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$, $\mathbf{g}(\mathbf{x}) = \mathbf{B} \cdot \mathbf{x}$. Compute $[\mathbf{f}, \mathbf{g}](\mathbf{x})$.

2) Consider the system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \cdot u_1 + \mathbf{g}(\mathbf{x}) \cdot u_2.$$

We propose to apply the following cyclic sequence:

$t \in [0, \delta]$	$t \in [\delta, 2\delta]$	$t \in [2\delta, 3\delta]$	$t \in [3\delta, 4\delta]$	$t \in [4\delta, 5\delta]$	$\dots$
$\mathbf{u} = (1, 0)$	$\mathbf{u} = (0, 1)$	$\mathbf{u} = (-1, 0)$	$\mathbf{u} = (0, -1)$	$\mathbf{u} = (1, 0)$	$\dots$

where δ is an infinitesimal time period δ. This periodic sequence will be denoted by

$$\{(1,0),(0,1),(-1,0),(0,-1)\}$$

Show that

$$\mathbf{x}(t+2\delta) = \mathbf{x}(t-2\delta) + [\mathbf{f},\mathbf{g}](\mathbf{x}(t))\,\delta^2 + o(\delta^2).$$

3) Which periodic sequence should we take in order to follow the field $\nu \cdot [\mathbf{f},\mathbf{g}]$. We will first consider the case $\nu \geq 0$ and then the case $\nu \leq 0$.

4) Propose a controller such that the system becomes

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \cdot a_1 + \mathbf{g}(\mathbf{x}) \cdot a_2 + [\mathbf{f},\mathbf{g}](\mathbf{x}) \cdot a_3$$

where $\mathbf{a} = (a_1, a_2, a_3)$ is the new input vector.

5) Consider a Dubins car described by the following state equations:

$$\begin{cases} \dot{x} &= u_1 \cos\theta \\ \dot{y} &= u_1 \sin\theta \\ \dot{\theta} &= u_2 \end{cases}$$

where u_1 is the speed of the cart, θ its orientation and (x,y) the coordinates of its center. Using Lie Brackets, add a new input to the system, *i.e.*, a new direction of control.

6) Propose a simulation to check the good behavior of your controller. To do this take δ small enough to be consistent with the second order Taylor approximation but large with respect to the sampling time dt. For instance, we may take $\delta = \sqrt{dt}$. The initial state vector is taken as $\mathbf{x}(0) = (0,0,1)$.

7) Propose a controller so that the car goes up. Consider the same questions, where the car goes down, right and left, respectively.

Exercise 18.– *Follow the equator*

See the correction video at https://youtu.be/8brHzqFHhSg

We consider a mobile body moving around a planet, the Earth, with a radius ρ_E rotating along the vertical axis with a rotation rate $\dot{\psi}_E$.

1) Draw the Earth rotating with a static body at position $\mathbf{p}$ with the following frames (see Figure 5.4):

1. $\mathcal{R}_0$ is the Earth-centered inertial frame, fixed with respect to the stars
2. $\mathcal{R}_1$ is the Earth frame and turning with the Earth with a rotational speed $\dot{\psi}_E$
3. $\mathcal{R}_2$ is the navigation frame, ego-centered and pointing toward the sky
4. $\mathcal{R}_3$ is the ego-centered frame of the robot.

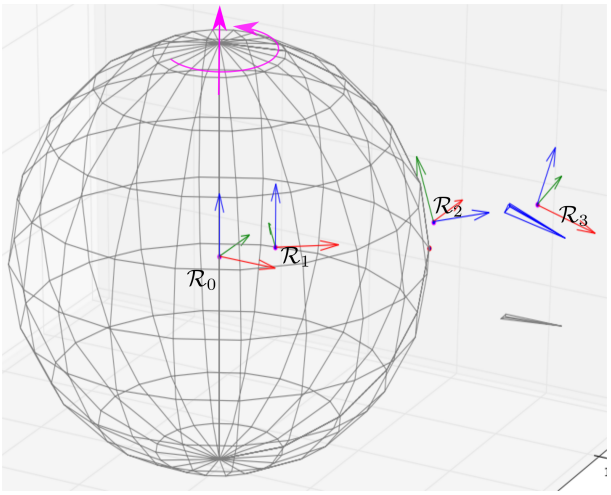

Figure 5.4: Earth with the different frames. The origin of the frames have been shifted for more visibility

For the simulation, take $\dot{\psi}_E = 0.2$ rad/s, $\rho_E = 10$m.

2) Provide the kinematic state equations for the body. The state variables are taken as $(\mathbf{p}, \mathbf{R}_3, \mathbf{v}_3)$, where $\mathbf{p}$ is the position of the body in $\mathcal{R}_0$, $\mathbf{R}_3$ is the orientation of the body, and $\mathbf{v}_3$ is the speed of the body seen from $\mathcal{R}_0$ and expressed in $\mathcal{R}_3$. The inputs are chosen as $\boldsymbol{\omega}_3$, the rotation vector of the body and $\mathbf{a}_3$, the measured acceleration, both expressed in $\mathcal{R}_3$. Propose a simulation with the Earth rotating and the body moving in the space with a gravity given by $\mathbf{g}(\mathbf{p}) = -9.81 \cdot \rho_T^2 \frac{\mathbf{p}}{\|\mathbf{p}\|^3}$.

3) The planet is now assumed to be made of water and the body is an underwater robot equipped with actuators (such as propellers) to move inside the water. The gravity is taken as $\mathbf{g}(\mathbf{p}) = -\frac{9.81}{\rho_T}\mathbf{p}$. The water of the rotating planet drags the robot through friction forces (tangential and rotational). Moreover, the robot which has the same density as the surrounding fluid receives an Archimedes upward buoyant force. Propose a dynamical model describing the motion of the robot. The inputs will be the rotational acceleration $\mathbf{u}_{\omega 3}$ et the tangential acceleration $\mathbf{u}_{a3}$ expressed in $\mathcal{R}_3$. These acceleration result from the action of the actuators creating forces and torques on the robot. Propose a simulation illustrating all these effects.

4) The robot behaves like a 3D Dubins-like car model. More precisely, it is equipped with a single propeller which push the robot forward without any possibility to control the speed. We take $\mathbf{u}_{a3} = (1, 0, 0)^{\mathrm{T}}$. We assume that the robot can chose its rotation vector using rudders, jet pumps or inertial wheels. Find a controller such that the robot goes to East along the equator. Validate by a simulation.

EXERCISE 19.– *Modeling and control of a torpedo*

See the correction video at https://youtu.be/4kLr7KnyNi8

We consider the autonomous underwater vehicle (AUV), a *Riptide*, represented by Figure 5.5. For its locomotion, the robot uses three fins and one propeller. The inputs are u_0, the rotation speed

of the propeller and the angles u_1, u_2, u_3 of the fins. We assume that we have no *side slip* effect. This means that the velocity vector $\mathbf{v}_r$ has always the direction of the robot, or equivalently, the robot has no lateral speed.

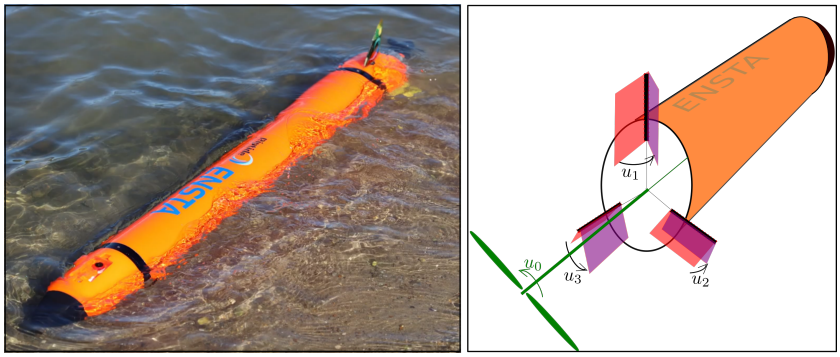

Figure 5.5: The Riptide robot has one propeller and three fins

1) Recall that the mechanization kinematic equations of a solid body are given by

$$\begin{cases} \dot{\mathbf{p}} &= \mathbf{R} \cdot \mathbf{v}_r \\ \dot{\mathbf{R}} &= \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) \\ \dot{\mathbf{v}}_r &= \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{cases}$$

where $\mathbf{R}$ is the orientation matrix, $\boldsymbol{\omega}_r$ is the rotation vector expressed in the robot frame, $\mathbf{a}_r$ is the acceleration vector in the robot frame, and $\mathbf{p}$ is the position of the robot in the world frame. Find a kinematic state space model for the robot, where the inputs are $\boldsymbol{\omega}_r$ and the longitudinal acceleration a_{rx}.

2) Considering that the robot has no side slip, propose a dynamic model for the robot. We will assume that the robot has a cylinder shape, that its buoyancy is neutral, and the added mass is equal to the mass of the robot. The inputs are now, u_0, u_1, u_2, u_3.

3) Propose a controller for the dynamical model. More precisely, we want that the closed loop system has the desired rotation vector $\bar{\boldsymbol{\omega}}_r$ and the desired acceleration $\bar{a}_{rx}$.

4) Illustrate the behavior of the system using a simulation. The parameters will be chosen as $p_1 = 1, p_2 = 2, p_3 = 1, p_4 = 4$. Try to get a trajectory similar to that illustrated by Figure 5.5.

EXERCISE 20.– *Geodesic*

See the correction video at https://youtu.be/S1-j554Szac

Consider a n-dimensional manifold embedded into $\mathbb{R}^m$, $m > n$ with the parametric equations

$$\mathbf{x} = \mathbf{r}(\mathbf{q}),$$

where $\mathbf{q} \in \mathbb{R}^n$ represents the configuration vector (or the vector of the degrees of freedom) of a robot and $\mathbf{x} \in \mathbb{R}^m$ is coordinate vector in the workspace. We assume that we can control the acceleration

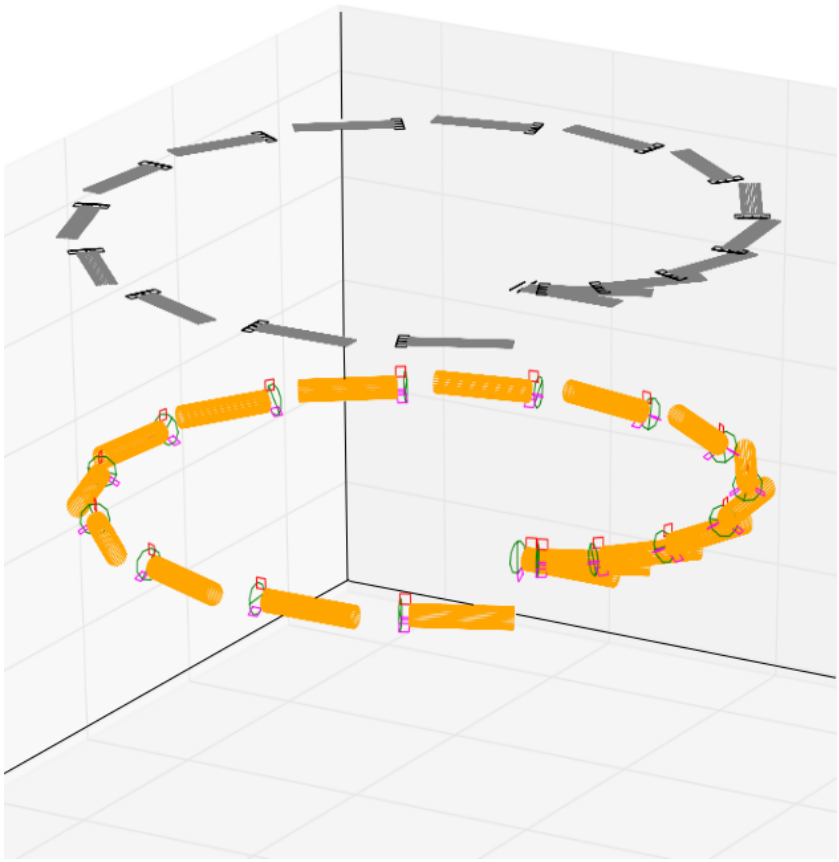

Figure 5.6: The Riptide (orange) is controlled to turn left with a desired speed of $1ms^{-1}$. The surface shadow at the surface is painted gray.

$\ddot{\mathbf{q}}$ of $\mathbf{q}$ via a control $\mathbf{u}$. In this exercise, we want to find a controller (see Figure 5.7) which computes $\mathbf{u}$ such that $\ddot{\mathbf{x}} - \mathbf{w}$ is as small as possible. We will then make the correspondance with the notion of geodesic used to characterize the shortest path between two points of a manifold [13].

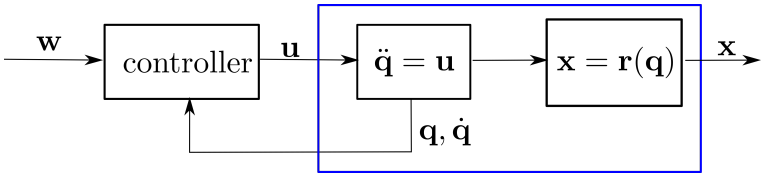

Figure 5.7: The controller generates $\mathbf{u}$ in order to minimize $\|\ddot{\mathbf{x}} - \mathbf{w}\|$

1) Consider a curve $\mathbf{x}(t) = \mathbf{r}(\mathbf{q}(t))$. Give an expression of $\ddot{\mathbf{x}}$ as a function of $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$.

2) We want $\ddot{\mathbf{x}} = \mathbf{w}$ where $\mathbf{w}$ is the wanted acceleration. Now, due to the fact that $\mathbf{x}(t)$ should stay on the manifold $\mathbf{r}(\mathbb{R}^n)$, all accelerations $\ddot{\mathbf{x}}$ are not allowed. Only the projection of $\ddot{\mathbf{x}}$ on the tangent plane can be chosen as desired. This is illustrated by Figure 5.8 where the car can turn left and right but should stay on the surface. This means that we cannot have any impact on the component of $\ddot{\mathbf{x}}$ which is orthogonal to the manifold (red arrow in the figure).

Find the controller (see Figure 5.8) which computes $\mathbf{u}$ such that $\|\ddot{\mathbf{x}} - \mathbf{w}\|$ is as small as possible.

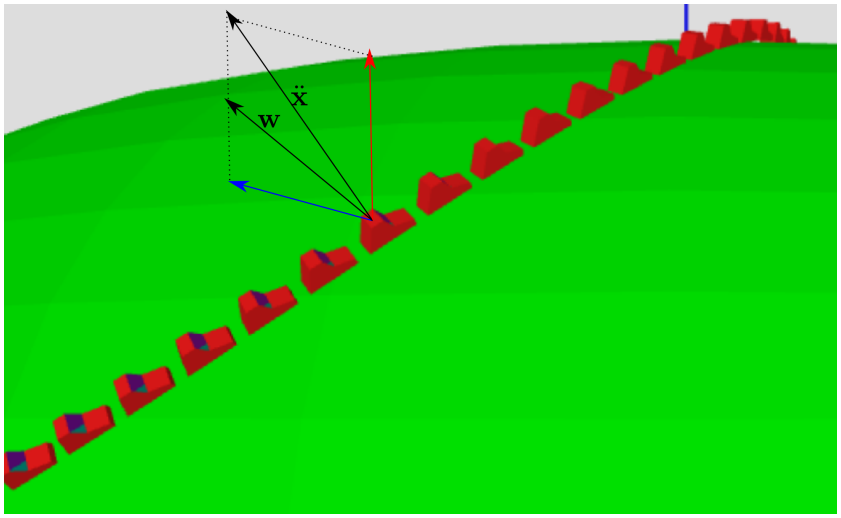

Figure 5.8: When moving on the manifold, we can only have an action the acceleration along the manifold (blue) but not on the transverse component (red)

3) When we do not want any acceleration along the manifold, we chose $\mathbf{w} = \mathbf{0}$ and the corresponding trajectory is called a *geodesic*. What is the equation of a geodesic starting $\mathbf{q}(0) = \mathbf{q}_0$ and $\dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_0$.

4) We consider the torus, given by

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathbf{r}(q_1, q_2) = \begin{pmatrix} (a_1 + a_2 \cos q_2) \cos q_1 \\ (a_1 + a_2 \cos q_2) \sin q_1 \\ a_2 \sin q_2 \end{pmatrix}$$

with $a_1 = 10, a_2 = 6$. Compute and draw a geodesic on the torus starting from $\mathbf{q}(0) = (0, 0)$ and $\dot{\mathbf{q}}(0) = (0.1, 0.2)$, $t \in [0, 50]$. An illustration of what you should obtain is given by Figure 5.9.

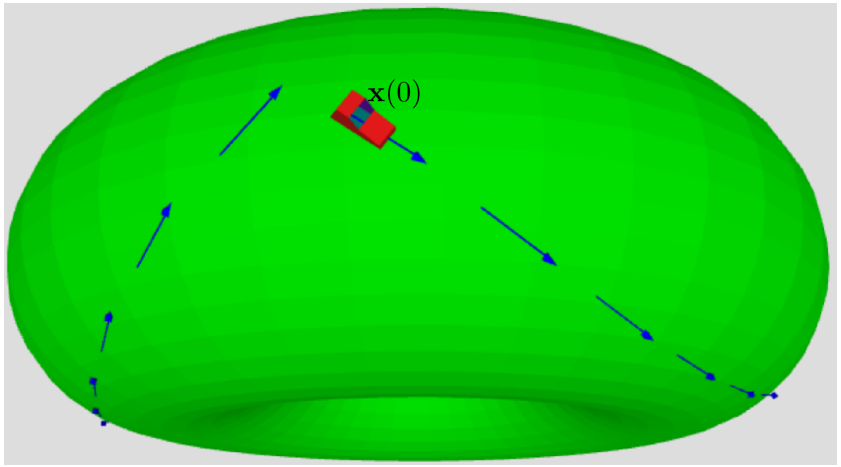

Figure 5.9: The geodesic oscillates around the external equator

5) We consider the ellipsoid, described by

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \mathbf{r}(q_1, q_2) = \begin{pmatrix} a_1 \cos q_2 \cos q_1 \\ a_2 \cos q_2 \sin q_1 \\ a_3 \sin q_2 \end{pmatrix}$$

where $a_1 = 8, a_2 = 4, a_3 = 2$. Write a program which computes a geodesic starting from $\mathbf{q}(0) = (0, 0)$ and such that $\mathbf{q}(10) = (2, 1)$.

EXERCISE 21.– *Helicopter looping*

See the correction video at https://youtu.be/aCI-Tgydct8

Consider the helicopter robot described by Figure 5.5. It has a main rotor on top and a small tail rotor. In order to control the helicopter, one can tilt the angle of each blade on both rotors. The main rotor is used to move the helicopter up and down, and to make the helicopter tilt forward, backward, left, or right. By tilting a blade to increase the blade's angle of attack, the pilot can increase the force of lift that is pushing up on that blade [14].

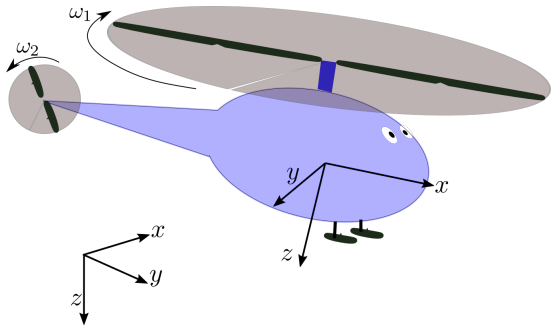

Figure 5.10: Helicopter

A state space model is given by

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{R} \cdot \mathbf{v}_r & (i) \\ \dot{\mathbf{R}} = \mathbf{R} \cdot (\boldsymbol{\omega}_r \wedge) & (ii) \\ \dot{\mathbf{v}}_r = \mathbf{R}^{\mathrm{T}} \cdot \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + \frac{1}{m} \cdot \begin{pmatrix} 0 \\ 0 \\ -\tau_0 \end{pmatrix} - \boldsymbol{\omega}_r \wedge \mathbf{v}_r & (iii) \\ \dot{\boldsymbol{\omega}}_r = \mathbf{I}^{-1} \cdot \left(\begin{pmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} - \boldsymbol{\omega}_r \wedge (\mathbf{I} \cdot \boldsymbol{\omega}_r) \right) & (iv) \end{cases}$$

where

$$\begin{pmatrix} \tau_0 \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix} = \begin{pmatrix} \beta_1\omega_1^2 & 0 & 0 & 0 \\ 0 & \beta_2\omega_1^2 & 0 & 0 \\ 0 & 0 & \beta_3\omega_1^2 & 0 \\ -\delta_1\omega_1^2 & 0 & 0 & -\beta_4\ell\omega_2^2 \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}.$$

In this model, $\mathbf{R}$ corresponds to the orientation, $\mathbf{p}$ the position, $\mathbf{v}_r$ the speed vector expressed in the robot frame, $\boldsymbol{\omega}_r$ the rotation vector also expressed in the robot frame, τ_0 is the total thrust generated by the main rotor, τ_1 is the roll torque, τ_2 is the pitch torque, τ_3 is head torque generated by the tail rotor and $\mathbf{u}$ the input vector. We assume that $\omega_1 = \omega_2 = 100$, $\beta_1 = 0.02$, $\beta_2 = \beta_3 = \frac{\beta_1}{10}$, $\beta_4 = 0.002$, $\delta_1 = \frac{\beta_1}{5}$ all expressed in the international unit system.

1) Provide a controller to that the helicopter has the desired orientation $\mathbf{R}^d$ and the a desired total thrust τ_0^d.

2) Give a controller for the system, so that the helicopter performs a looping, as illustrated by Figure 5.5.

3) Validate using a simulation.

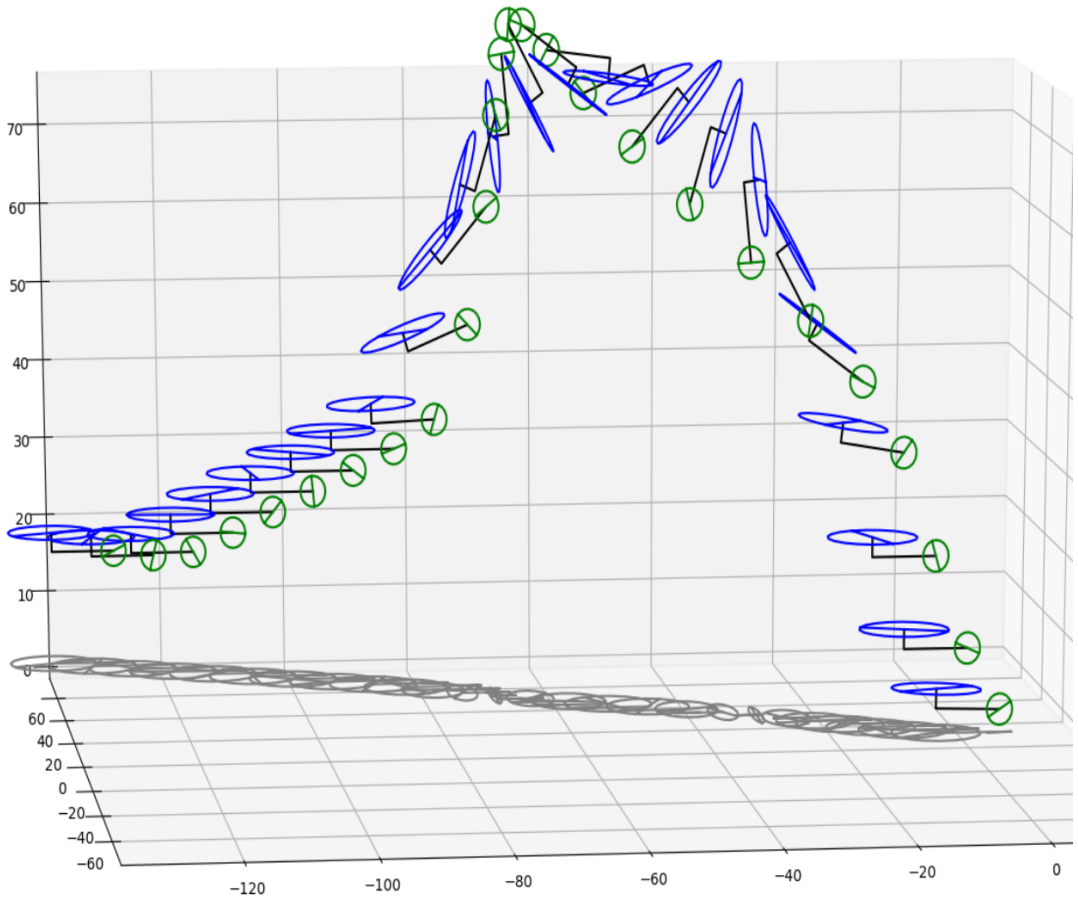

Figure 5.11: Helicopter performing a looping

EXERCISE 22.– *Hexarotor*

See the correction video at https://youtu.be/3HVC9IMDB4o

A hexarotor is composed of a body and with 6 rotors. Each rotor is at position $\mathbf{q}(i)$ with a direction $\mathbf{d}(i)$ and yields a force f_i in the direction of $\mathbf{d}(i)$. In the body frame, the coordinates of $\mathbf{q}(i), \mathbf{d}(i)$ are given by the following table:

i	1	2	3	4	5	6
$\mathbf{q}(i)$	$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} \frac{1}{2} \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ \frac{1}{2} \\ 0 \end{pmatrix}$
$\mathbf{d}(i)$	$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$

This is illustrated by Figure 5.12. Each force f_i contributes to the resultant on the robot as $\mathbf{d}(i) \cdot f_i$ and to the torque as $\mathbf{q}(i) \wedge \mathbf{d}(i) \cdot f_i$.

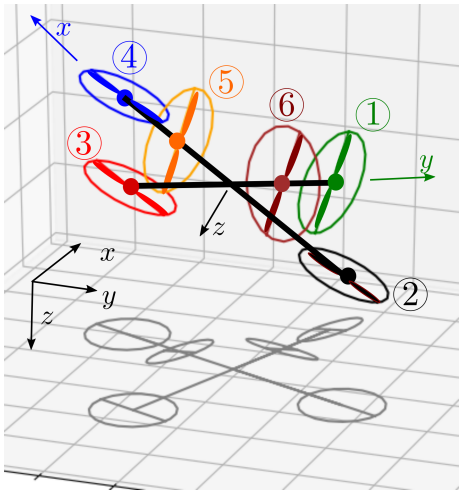

Figure 5.12: Hexarotor with its six propellers

1) Find the state equations of the hexarotor. The state is be composed of the position $\mathbf{p}$, the orientation $\mathbf{R}$, the speed $\mathbf{v}_r$ in the robot frame and the rotation vector $\boldsymbol{\omega}_r$ in the robot frame. The inputs are $f_1, \ldots, f_6$.

2) Find a controller that makes the hexarotor able to dock on a platform at position $\mathbf{p}_d$ with an orientation $\mathbf{R}_d$.

3) Illustrate on a simulation in the case where the position and the orientation for the platform

(which corresponds to the target to be reached) are given by

$$\mathbf{p}_d = \begin{pmatrix} \sin\frac{3t}{10} \\ \cos\frac{4t}{10} \\ \frac{1}{10}\cdot\sin\frac{3t}{10} \end{pmatrix}, \quad \mathbf{R}_d = \mathrm{Exp}\begin{pmatrix} \sin t \\ \cos 2t \\ t \end{pmatrix}$$

Bibliography

[1] R. Beard and T. McLain. *Small Unmanned Aircraft, Theory and Practice*. Princeton University Press, 2012.

[2] V. Creuze. Robots marins et sous-marins ; perception, modélisation, commande. *Techniques de l'ingénieur*, 2014.

[3] J.P. Laumond. *La robotique mobile*. Hermès, Paris, France, 2001.

[4] L. Jaulin. *Représentation d'état pour la modélisation et la commande des systèmes (Coll. Automatique de base)*. Hermès, London, 2005.

[5] L. Jaulin. *Automation for Robotics*. ISTE editions, 2015.

[6] P.J. Olver. *Applications of Lie Groups to Differential Equations*. Graduate Texts in Mathematics. Springer New York, 2012.

[7] J. Sola, J. Deray, and D. Atchuthan. A micro lie theory for state estimation in robotics. *arXiv:1812.01537*, 2018.

[8] Richard M. Murray, Zexiang Li, and Shankar Sastry. *A mathematical introduction to robotics manipulation*. CRC Press, 1994.

[9] P. Corke. *Robotics, Vision and Control*. Springer, Berlin Heidelberg, 2011.

[10] L. Farrell. *Aided navigation: GPS with high rate sensors*. McGraw-Hill, 2008.

[11] James Blaine Scarborough. *The Gyroscope*. Interscience Publ., 1958.

[12] T. Fossen. *Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, 2002.

[13] P. Pokorny. Geodesics revisited. *Chaotic Modeling and Simulation*, pages 281–298, 2012.

[14] R.W. Prouty. *Helicopter Performence, Stability and Control*. Krieger Publishing Company, USA, 1986.