Math for Computer Vision and Navigation

Ferdi van der Heijden University of Twente - RAM 12/3/2019

This document summarizes the mathematical background needed to study the principles for 3D computer vision and navigation technology. It has been setup as follows:

Basics:
Part A and part B are excerpts of math at Bachelor level. They address the following topics:
- Part A: Linear vector spaces
  - o Definition of vectors.
  - o Properties of vector spaces: inner product, norms, projection, orthogonality, and distance measures.
- Part B: Matrix algebra.

Intermediate level:
Part C and part D are at Master level. The text is self-contained with the following topics:
- Part C: how to represent position and orientation of objects in 3D space?
- Part D: alternative representations of orientation; besides rotation matrices: Euler angles, Axis/angle, and quaternions.

Advanced level:
- Part E: kinematics of a moving object: linear and angular velocities and accelerations

Equation Section 1

A. Vector spaces

Vectors are mathematical objects that have some prescribed properties. Vectors are often denoted by bold faced, lower case characters, e.g. $\mathbf{p}$, $\mathbf{q}$ .and $\mathbf{r}$. By definition, a *vector space* is a set of vectors in which two operations are specified:

- The vector addition: $\mathbf{r} = \mathbf{p} + \mathbf{q}$
- The scalar product: $\mathbf{r} = \alpha\mathbf{p}$

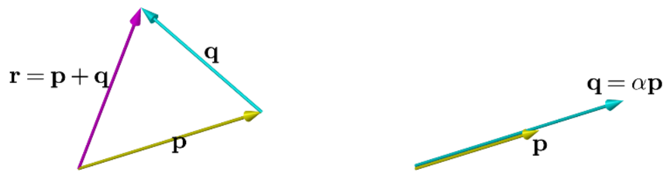
The vector addition is an operation on two vectors that produces a third vector. The scalar product is an operation on one vector and a scalar. It produces another vector.

A vector space is only a vector space if it satisfies eight conditions, called axioms:

$$\begin{array}{cc} \mathbf{p}+\mathbf{q}=\mathbf{q}+\mathbf{p} & \mathbf{p}+\mathbf{0}=\mathbf{p} \\ (\mathbf{p}+\mathbf{q})+\mathbf{r}=\mathbf{p}+(\mathbf{q}+\mathbf{r}) & \mathbf{p}+(-\mathbf{p})=\mathbf{0} \\ \alpha(\mathbf{p}+\mathbf{q})=\alpha\mathbf{p}+\alpha\mathbf{q} & (\alpha\beta)\mathbf{p}=\alpha(\beta\mathbf{p}) \\ (\alpha+\beta)\mathbf{p}=\alpha\mathbf{p}+\beta\mathbf{p} & 1\mathbf{p}=\mathbf{p} \end{array} \tag{A1}$$

A vector $\mathbf{r}$ is said to be *linearly independent* from $\mathbf{p}$ and $\mathbf{q}$ if in any way no α and β can be found such that $\mathbf{r} = \alpha\mathbf{p} + \beta\mathbf{q}$.

We can think of vectors as arrows: entities that have a length and a direction. This enables easy visualization of vectors:



$\mathbf{r}=\mathbf{p}+\mathbf{q}$
$\mathbf{q}$
$\mathbf{p}$
$\mathbf{q}=\alpha\mathbf{p}$
$\mathbf{p}$

These graphs might suggest that vectors are entities in a 2D or 3D space, but vector spaces are not confined to that.

Example: polynomial functions

The set of all polynomial functions of degree N :

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_N x^N$$

is a vector space. This follows readily by considering two vectors: $f(x) = a_0 + \cdots + a_N x^N$ and $g(x) = b_0 + \cdots + b_N x^N$, and showing that vector addition, $f(x) + g(x)$, and scalar product $\alpha f(x)$, both end up in polynomials of the same degree, and thus in vectors that are lying in the same vector space. Also, it should be verified that these two operations comply with the eight axioms.

Many vector spaces can be represented by an ordered set of elements. For the polynomial function example, these elements are the coefficient $a_0, \cdots, a_N$ .Usually, these elements are vertically aligned in a column, e.g.:

$$\mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix}$$

in which the set a_n with $n = 0, \cdots, N$ are the elements. For an inline expression this takes too much space, and the transposed notation, $\mathbf{a} = \begin{bmatrix} a_0 & a_1 & \cdots & a_N \end{bmatrix}^{\mathrm{T}}$ is often used. Note that, without the transpose, $\mathbf{a} = \begin{bmatrix} a_0 & a_1 & \cdots & a_N \end{bmatrix}$, the result is a row vector.

The number of elements is the *dimension* of the vector. In the polynomial function example, the dimension was $N+1$. Thus, a polynomial function of degree N has a dimension of $N+1$. Of course, this is confusing. Therefore, unlike polynomial functions, the enumeration of the elements almost always starts at 1, thus: $\mathbf{a} = \begin{bmatrix} a_1 & a_2 & \cdots & a_N \end{bmatrix}^{\mathrm{T}}$, so that the dimension of this vector is just N .

The space of N -dimensional vectors, of which the elements are real numbers, is denoted by $\mathbb{R}^N$. If the elements are complex numbers, the space is denoted by $\mathbb{C}^N$.

A.1 Properties of vector spaces

A.1.1 The norm of a vector space

The *norm* of a vector space is an axiomatic defined assignment of a real number to a vector. If $\mathbf{f}$ is a vector with elements f_n , then its norm is denoted by $\|\mathbf{f}\|$. A norm can be defined freely as long as it complies with the following three axioms:

$$\begin{array}{ll} 1) & \|\mathbf{f}\| \geq 0, \text{ where } \|\mathbf{f}\| = 0 \text{ if and only if } \mathbf{f} = \mathbf{0} \\ 2) & \|\alpha\mathbf{f}\| = |\alpha| \|\mathbf{f}\| \\ 3) & \|\mathbf{f}+\mathbf{g}\| \leq \|\mathbf{f}\| + \|\mathbf{g}\| \end{array} \tag{A2}$$

A well-known norm in $\mathbb{R}^N$ is the so-called L$_p$ norm:

$$\|\mathbf{f}\|_p = \sqrt[p]{\sum_{n=1}^{N} |f_n|^p} \quad \text{with } p \geq 1 \tag{A3}$$

With $p = 2$, we have the Euclidean norm which is often used because it measures the geometrical length of a vector in the 2- and 3-dimensional case. For this reason, $\|\mathbf{f}\|$ always denotes the Euclidean norm, unless it is specified otherwise.

If we are only interested in the direction of a vector, and not in its length, then we may want to normalize the vector:

$$\mathbf{n}_\mathbf{p} = \frac{\mathbf{p}}{\|\mathbf{p}\|} \qquad \text{(A4)}$$

The vector $\mathbf{n}_\mathbf{p}$ has unit length, and holds only information about the direction of $\mathbf{p}$.

A.1.2 Inner product

The *inner product* is an axiomatic defined assignment of a real number to a pair of vectors. Most commonly, this assignment is the so-called *vector dot product*. Let $\mathbf{p}$ and $\mathbf{q}$ be two vectors from $\mathbb{R}^N$, that is $\mathbf{p} = [p_1 \quad \cdots \quad p_N]^\mathrm{T}$ and $\mathbf{q} = [q_1 \quad \cdots \quad q_N]^\mathrm{T}$. In general, the inner product is denoted by $(\mathbf{p},\mathbf{q})$. If the dot product is chosen as inner product, then:

$$(\mathbf{p},\mathbf{q}) = \mathbf{p}^\mathrm{T}\mathbf{q} = \mathbf{q}^\mathrm{T}\mathbf{p} = \sum_{n=1}^{N} p_n q_n \qquad \text{(A5)}$$

Note that the Euclidean norm of a vector $\mathbf{p}$ follows from:

$$\|\mathbf{p}\|^2 = \mathbf{p}^\mathrm{T}\mathbf{p} \qquad \text{(A6)}$$

Associated with the inner product is an *angle* φ between the two vectors, defined by:

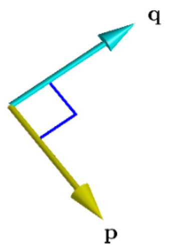
$$(\mathbf{p},\mathbf{q}) = \|\mathbf{p}\|\|\mathbf{q}\|\cos\varphi \qquad \text{(A7)}$$

In the 2 and 3-dimensional case, the angle φ indeed represents the angle between the arrows.

Matlab code for inner product:
If **p** and **q** are column vectors, then `p'*q` or `dot(p,q)` provides the inner product. If **p** and **q** are row vectors, then `p*q'` would be the inner product.

A.1.3 Orthogonality

Two vectors are said to be *orthogonal*, $\mathbf{p} \perp \mathbf{q}$, if $\varphi = 90^0$, which is equivalent to $\mathbf{p}^T\mathbf{q} = 0$. A graphical representation is given below.



The blue rectangle symbolizes the orthogonality of the two vectors.

In the case of orthogonality, Pythagoras' theorem applies:

$$\mathbf{p} \perp \mathbf{q} \quad \Leftrightarrow \quad \|\mathbf{p}+\mathbf{q}\|^2 = \|\mathbf{p}\|^2 + \|\mathbf{q}\|^2 \qquad \text{(A8)}$$

Example: Pythagoras in 2D
Suppose: $\mathbf{a} = [a_1 \quad 0]^\mathrm{T}$ and $\mathbf{b} = [0 \quad b_2]^\mathrm{T}$. Then clearly $\mathbf{a} \perp \mathbf{b}$. We form a third vector: $\mathbf{c} = \mathbf{a} + \mathbf{b}$. Then according to Pythagoras, the length $\|\mathbf{c}\|$ follows from: $\|\mathbf{c}\|^2 = a_1^2 + b_2^2$ which indeed is $\|\mathbf{a}\|^2 + \|\mathbf{b}\|^2$.

A.1.4 Projection

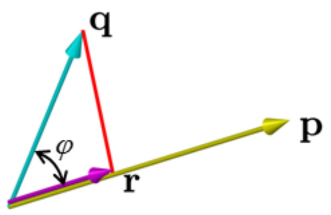
The *projection* $\mathbf{r}$ of $\mathbf{q}$ on $\mathbf{p}$ is defined by

$$\mathbf{r} = \alpha\mathbf{p} \;\text{ such that }\; (\mathbf{p},\mathbf{r}) = (\mathbf{p},\mathbf{q}) \qquad \text{(A9)}$$

Since $(\mathbf{p},\mathbf{r}) = \alpha(\mathbf{p},\mathbf{p}) = \alpha\|\mathbf{p}\|^2$, we find

$$\alpha = \frac{(\mathbf{p},\mathbf{q})}{\|\mathbf{p}\|^2} = \frac{\mathbf{p}^\mathrm{T}\mathbf{q}}{\mathbf{p}^\mathrm{T}\mathbf{p}} = \|\mathbf{q}\|\cos\varphi \qquad \text{(A10)}$$

See figure, below.



A.1.5 Distance measures

A *distance measure* is a non-negative real number that is assigned to a pair of vectors, and that complies with three axioms. $\rho(\mathbf{x},\mathbf{y})$ is a distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ if the following three axioms are met:

$$\begin{aligned} &1) \quad \rho(\mathbf{x},\mathbf{y}) = 0 \;\text{ if and only if }\; \mathbf{x} = \mathbf{y} \\ &2) \quad \rho(\mathbf{x},\mathbf{y}) = \rho(\mathbf{y},\mathbf{x}) \\ &3) \quad \rho(\mathbf{x},\mathbf{y}) \le \rho(\mathbf{x},\mathbf{z}) + \rho(\mathbf{y},\mathbf{z}) \end{aligned} \qquad \text{(A11)}$$

Any norm $\| \;\; \|$ of a vector can be used to define a distance measure:

$$\rho(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\| \qquad \text{(A12)}$$

However, there are also distance measures that are not derived from a norm. For instance, with the definition:

$$\rho(\mathbf{x},\mathbf{y}) = \begin{cases} 0 & \text{if } \; \mathbf{x} = \mathbf{y} \\ 1 & \text{if } \; \mathbf{x} \neq \mathbf{y} \end{cases} \qquad \text{(A13)}$$

$\rho(.,.)$ complies with the three axioms, but $\rho(\mathbf{x},\mathbf{0})$ is not a norm of $\mathbf{x}$. Equation Section (Next)

B. Matrix algebra

A matrix is an arrangement of numbers on a 2D grid:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ \vdots & & & \vdots \\ h_{N1} & h_{N2} & \cdots & h_{NM} \end{bmatrix} \qquad \text{(B1)}$$

The grid consists of N rows and M columns. As such the matrix consists of NM numbers, and the matrix is said to be $N \times M$ dimensional. Usually, a matrix is denoted by a bold-faced capital, such as $\mathbf{H}$. The elements are denoted by the same character, but now italic and in lower case, e.g. h_{ij}. The set of $N \times M$ matrices is denoted by $\mathbb{R}^{N \times M}$.

The matrix addition and scalar multiplication for matrices are defined as follows[1]:

$$\begin{aligned} \mathbf{F} &= \mathbf{G} + \mathbf{H} && \text{with elements } f_{ij} = g_{ij} + h_{ij} \\ \mathbf{F} &= \alpha \mathbf{H} && \text{with elements } f_{ij} = \alpha h_{ij} \end{aligned} \qquad \text{(B2)}$$

A third operation on a matrix is the scalar addition:

$$\mathbf{F} = \mathbf{G} + \alpha \quad \text{with elements } f_{ij} = g_{ij} + \alpha \qquad \text{(B3)}$$

Note, that a matrix with just one column, i.e. $M = 1$, is equivalent to an N-dimensional vector: $\mathbb{R}^{N \times 1} \sim \mathbb{R}^N$.

'Dimension' in Matlab

For matrices and arrays, the term *dimension* can have different connotations:

- *Number of array dimensions*. This is the dimension of the orthogonal grid. In the case of matrices, it is always 2: a matrix is a 2D array. Arrays can also have higher dimensional grids. Full color images are arrays with dimension 3. This is also the case with volumetric medical data. Matlab function to get the dimension: **`ndims`**.
- *Array dimensions*. These are the sizes of each dimension of an array. For an $N \times M$ matrix, they are N and M. Matlab function: **`size`**.
- *Number of array elements*. As matrices fulfill the requirements for vector spaces, a matrix can also be regarded as a vector. In this connotation, the dimension of an $N \times M$ matrix is NM. Matlab function: **`numel`**.

The elements in Matlab arrays and matrices can be rearranged on different orthogonal grids. For instance, a 6×12 matrix can be rearranged to an 8×9 grid. Matlab function: **`reshape`**.
To rearrange an $N \times M$ matrix to a $NM \times 1$ array (i.e. to vectorize the data), use the semicolon operator, e.g. **`H(:)`**.

B.1 Matrix-vector and matrix-matrix products

B.1.1 Matrix-vector product

Matrices offer a very concise way to represent a system of linear equations. Consider, for instance, the following set of equations:

$$\begin{aligned} y_1 &= h_{11}x_1 + h_{12}x_2 + h_{13}x_3 \\ y_2 &= h_{21}x_1 + h_{22}x_2 + h_{23}x_3 \end{aligned}$$

By introducing:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \end{bmatrix}$$

the equation is fully defined by:

$$\mathbf{y} = \mathbf{H}\mathbf{x} \qquad \text{(B4)}$$

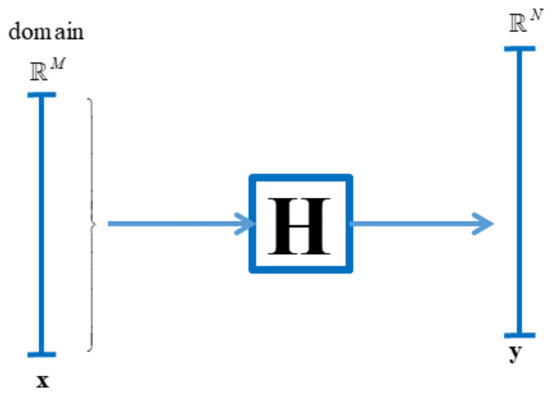
This equation represents the *matrix-vector product*. For the case in which $\mathbf{x}$ is M-dimensional and $\mathbf{y}$ is N-dimensional, the matrix $\mathbf{H}$ must be $N \times M$ dimensional. The multiplication is defined by:

$$y_i = \sum_{j=1}^{M} h_{ij} x_j \quad \text{for } i = 1, \cdots, N \qquad \text{(B5)}$$

B.1.2 Linear mappings

Matrix-vector products can be regarded as linear mappings of vectors. Consider, for instance, a vector $\mathbf{x} \in \mathbb{R}^M$ and an $N \times M$ dimensional matrix $\mathbf{H}$. Then the multiplication $\mathbf{Hx}$ will produce a vector $\mathbf{y} = \mathbf{Hx}$ in an N-dimensional space. The vector space $\mathbb{R}^M$ is the *domain* of the mapping $\mathbf{H}$. Each vector in this set is mapped to another vector which is lying in the vector space $\mathbb{R}^N$. The procedure is illustrated in the figure below.

[1] These operations fulfill the eight axioms for linear vector spaces. Therefore, the set of all matrices of a given size form a vector space by itself.



Mathematically, this mapping is expressed as:

$$\mathbf{H}: \quad \mathbb{R}^M \rightarrow \mathbb{R}^N \tag{B6}$$

The mapping is called *linear* because if we have two vectors in $\mathbb{R}^M$, $\mathbf{p}$ and $\mathbf{q}$,and two scalars, α and β, then the following property holds true:

$$\mathbf{H}(\alpha\mathbf{p}+\beta\mathbf{q}) = \alpha\mathbf{Hp}+\beta\mathbf{Hq} \tag{B7}$$

B.1.3 Matrix-matrix products

The *matrix-matrix* product $\mathbf{F} = \mathbf{HG}$ combines two matrices:

$$f_{ij} = \sum_{k=1}^{K} h_{ik} g_{kj} \tag{B8}$$

Here, $\mathbf{H}$ is $N \times K$ dimensional, $\mathbf{G}$ is $K \times M$ dimensional, and the resulting product is $N \times M$ dimensional. Note, that the number of columns of $\mathbf{H}$ must equal the number of rows of $\mathbf{G}$.

B.2 Special matrices and properties of matrices

The *null-matrix* $\mathbf{0}$. This is a matrix fully filled with zeros. Matlab function: **zeros**. If regarded $\mathbf{0}$ as an operator, then this operator maps each vector to the zero vector: $\mathbf{0x} = \mathbf{0}$.

A *square matrix* is a matrix with equal number of rows and columns: $N = M$. Such a matrix maps its domain $\mathbb{R}^M$ to itself.

$$\mathbf{H}: \quad \mathbb{R}^M \rightarrow \mathbb{R}^M$$

In this case, the mapping is called a *linear operator*.

A *diagonal matrix* is a square matrix fully consisting of zeros except at the diagonal:

$$\mathbf{H} = \begin{bmatrix} h_{11} & & 0 \\ & \ddots & \\ 0 & & h_{NN} \end{bmatrix} = \operatorname{diag}([h_{11} \quad \cdots \quad h_{NN}])$$

Matlab function: **diag**. Regarded as a mapping, the matrix $\mathbf{H}$ *scales* the elements of the vector $\mathbf{x}$. That is, $y_n = h_{nn}x_n$. Thus, a diagonal matrix *stretches* the space.

The *unit matrix* $\mathbf{I}$. This matrix is a square matrix, fully filled with zero, except for the diagonal elements which are unit:

$$\mathbf{I} = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix}$$

Matlab function: **eye**. Regarded as an operator, $\mathbf{I}$ maps each vector $\mathbf{x}$ to itself: $\mathbf{Ix} = \mathbf{x}$.

The *transpose operator* swaps the rows and the columns of the matrix. An $N \times M$ matrix $\mathbf{H}$ with elements h_{ij} becomes an $M \times N$ matrix $\mathbf{H}^{\mathrm{T}}$ with elements h_{ji}. Matlab: **H'** or **transpose(H)**. Properties:

$$(\mathbf{AB})^{\mathrm{T}} = \mathbf{B}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}$$

A *symmetric matrix* is a square matrix for which $h_{ij} = h_{ji}$ or $\mathbf{H} = \mathbf{H}^{\mathrm{T}}$.

The *trace* of a square matrix is the sum of its diagonal elements:

$$trace(\mathbf{H}) = \sum_{n=1}^{N} h_{nn} \tag{B9}$$

Matlab function: **trace**.

The *determinant* $|\mathbf{H}|$ of a square matrix $\mathbf{H}$ is recursively defined with its co-matrices. The co-matrix $\mathbf{H}_{n,m}$ is an $(N-1)\times(N-1)$-matrix that is derived from $\mathbf{H}$ by exclusion of the n-th row and the m-th column. The following equations define the determinant:

$$\begin{aligned} &\text{If} \quad N = 1: \quad |\mathbf{H}| = h_{1,1} \\ &\text{If} \quad N > 1: \quad |\mathbf{H}| = \sum_{m=1}^{N} (-1)^{m-1} h_{1,m} |\mathbf{H}_{1,m}| \end{aligned} \tag{B10}$$

Matlab function: **det**. Properties: $|\alpha\mathbf{H}| = \alpha^N |\mathbf{H}|$ and $|\mathbf{HG}| = |\mathbf{H}||\mathbf{G}|$.

B.3 Matrix inversion

The *inverse of a square matrix* $\mathbf{H}$ is denoted $\mathbf{H}^{-1}$, and is defined as the matrix for which

$$\mathbf{H}\mathbf{H}^{-1} = \mathbf{I} \tag{B11}$$

The inverse exists if, and only if, $\det(\mathbf{H}) \neq 0$. The matrix is said to be *invertible* then. If the inverse does not exist, the matrix is called *singular*.

Consider a set of N linear equations with N unknown variables. For instance:

$$\begin{aligned} y_1 &= h_{11}x_1 + h_{12}x_2 \\ y_2 &= h_{21}x_1 + h_{22}x_2 \end{aligned}$$

in which the variables y_n and coefficients h_{ij} are assumed to be known, and the variables x_n are unknown. The two equations represent two lines in $\mathbb{R}^2$. The intersection point of the two lines is the solution of the equations. If the two lines do not intersect (they are parallel), there is no (unique) solution, and the associated matrix is singular.

The equations are written concisely as $\mathbf{y} = \mathbf{H}\mathbf{x}$. This equation is solved by:

$$\mathbf{x} = \mathbf{H}^{-1}\mathbf{y} \tag{B12}$$

If the inverse of the matrix $\mathbf{H}$ exists, then this matrix maps each vector $\mathbf{x}$ from $\mathbb{R}^M$ to a unique vector $\mathbf{y}$, In addition, the operation $\mathbf{H}\mathbf{x}$ can 'hit' then any vector in $\mathbb{R}^M$. That is, for every $\mathbf{y} \in \mathbb{R}^M$, there is a unique corresponding $\mathbf{x}$. In other words, there is a one-to-one relation between $\mathbf{x}$ and $\mathbf{y}$. This is called a *surjection*.

The Matlab function for matrix inversion: **`inv`**. The Matlab expression for $\mathbf{H}^{-1}\mathbf{y}$ is **`H\y`**.

Some inversion properties:

$$(\mathbf{H}^{-1})^{\mathrm{T}} = (\mathbf{H}^{\mathrm{T}})^{-1}$$

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$$

$$(\mathbf{A}^{-1} + \mathbf{H}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{H})^{-1} = \mathbf{A} - \mathbf{A}\mathbf{H}^{\mathrm{T}}\left(\mathbf{H}\mathbf{A}\mathbf{H}^{\mathrm{T}} + \mathbf{B}\right)^{-1}\mathbf{H}\mathbf{A}$$

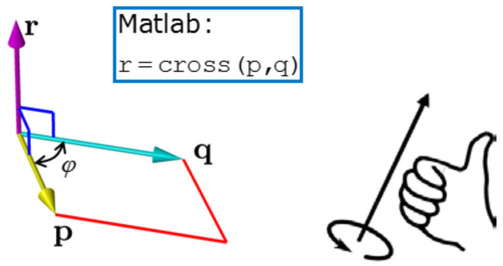
The latter is called the *matrix inversion lemma*.

B.4 The cross product

The cross product is only applicable to 3D spaces. In $\mathbb{R}^3$ the cross product of two vectors $\mathbf{p}$ and $\mathbf{q}$ is defined as a third vector $\mathbf{r} = \mathbf{p} \times \mathbf{q}$ with the following properties (see figure below):

$$\begin{aligned} &\|\mathbf{r}\| = \|\mathbf{p}\|\|\mathbf{q}\|\sin\varphi \\ &\mathbf{r} \perp \mathbf{p} \quad \text{and} \quad \mathbf{r} \perp \mathbf{q} \end{aligned} \tag{B13}$$

The direction of $\mathbf{r}$ follows the right-hand rule. That is, if the fingers of the right hand follows the rotation from $\mathbf{p}$ to $\mathbf{q}$, then $\mathbf{r}$ is in the direction of the thumb.



Matlab:
```
r = cross(p,q)
```

Note that $\|\mathbf{p}\|\|\mathbf{q}\|\sin\varphi$ is the area of the parallelogram. Consequently, we have:

$$\begin{aligned} &\mathbf{r} = \mathbf{0} && \text{if } \sin\varphi = 0; \text{ that is if } \mathbf{p} = \alpha\mathbf{q} \\ &\|\mathbf{r}\| = \|\mathbf{p}\|\|\mathbf{q}\| && \text{if } \mathbf{p} \perp \mathbf{q} \end{aligned} \tag{B14}$$

The cross product is calculated as follows:

$$\mathbf{r} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} = \begin{bmatrix} p_y q_z - p_z q_y \\ -p_x q_z + p_z q_x \\ p_x q_y - p_y q_x \end{bmatrix} \tag{B15}$$

The cross product can also be regarded as a matrix-vector multiplication. The vector $\mathbf{p}$ is then cast into a 3×3 skew symmetric (antisymmetric) matrix, as follows:

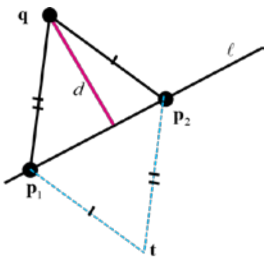
$$[\mathbf{p}]_\times \overset{def}{=} \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix} \tag{B16}$$

so that the cross product can be written as a matrix-vector multiplication:

$$\mathbf{p} \times \mathbf{q} = [\mathbf{p}]_\times \mathbf{q} \tag{B17}$$

Example: distance between a point and a line

The distance between a point $\mathbf{q}$ and a line ℓ defined by two points $\mathbf{p}_1$ and $\mathbf{p}_2$ is easily obtained using the cross product. For that purpose, consider the geometry in the figure below:



We ask for the shortest distance d. This distance follows readily from considering the parallelogram that is spanned by $\mathbf{p}_1 - \mathbf{q}$ and $\mathbf{p}_2 - \mathbf{q}$. The area of this parallelogram follows from the cross product:

$$area = \left\| (\mathbf{p}_1 - \mathbf{q}) \times (\mathbf{p}_2 - \mathbf{q}) \right\|$$

However, the same area also equals the base and the height of the parallelogram:

$$area = d \left\| \mathbf{p}_2 - \mathbf{p}_1 \right\|$$

The distance equals:

$$d = \frac{\left\| (\mathbf{p}_1 - \mathbf{q}) \times (\mathbf{p}_1 - \mathbf{q}) \right\|}{\left\| \mathbf{p}_2 - \mathbf{p}_1 \right\|} \qquad \text{(B18)}$$

B.5 Eigenvalues

Sometimes it is very useful to decompose a matrix into other matrices. For instance, singular value decomposition rewrites a given matrix $\mathbf{H}$ into $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}$. The matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{S}$ have nice properties as will be shown in Section B.6.

A widely used tool to get such a decomposition is eigenvalue analysis of a matrix. This concept applies to square matrices only. Suppose $\mathbf{H}$ is an $N \times N$ matrix, then $\mathbf{v}$ is an *eigenvector* of $\mathbf{H}$ if the operation $\mathbf{H}\mathbf{v}$ does not change the direction of $\mathbf{v}$. It only alters its length:

$$\mathbf{H}\mathbf{v} = \lambda \mathbf{v} \qquad \text{(B19)}$$

The scaling factor λ is a scalar which is called the *eigenvalue* associated with $\mathbf{v}$. Note that if $\mathbf{v}$ is an eigenvector, then so is $\alpha\mathbf{v}$ with $\alpha \neq 0$. Therefore, we require eigenvectors to have unit length: $\|\mathbf{v}\| = 1$.

Suppose that $\mathbf{H}$ has N linearly independent eigenvectors $\{\mathbf{v}_1, \cdots, \mathbf{v}_N\}$ with N eigenvalues $\{\lambda_1, \cdots, \lambda_N\}$. We put the eigenvectors side by side to form an $N \times N$ matrix:

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_N \end{bmatrix} \qquad \text{(B20)}$$

Next, we form a $N \times N$ diagonal matrix $\Lambda = \operatorname{diag}(\lambda_1, \cdots, \lambda_N)$. We then have:

$$\mathbf{V}\Lambda = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_N \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_N \end{bmatrix} = \begin{bmatrix} \lambda_1 \mathbf{v}_1 & \cdots & \lambda_N \mathbf{v}_N \end{bmatrix} \qquad \text{(B21)}$$

Since $\mathbf{H}\mathbf{V} = \begin{bmatrix} \mathbf{H}\mathbf{v}_1 & \cdots & \mathbf{H}\mathbf{v}_N \end{bmatrix}$, it follows that:

$$\mathbf{H}\mathbf{V} = \mathbf{V}\Lambda \qquad \text{or} \qquad \mathbf{H} = \mathbf{V}\Lambda\mathbf{V}^{-1} \qquad \text{(B22)}$$

Hence, we have decomposed $\mathbf{H}$ in $\mathbf{V}$, Λ, and $\mathbf{V}^{-1}$.

An even more interesting decomposition is achieved if the matrix $\mathbf{H}$ is symmetric: $\mathbf{H} = \mathbf{H}^{\mathrm{T}}$. In that case, it can be proven that:

a) All eigenvalues are real, i.e. imaginary parts are zero.
b) Any pair of eigenvectors is orthogonal: $\mathbf{v}_n \perp \mathbf{v}_m$.

The consequence of b), together with $\|\mathbf{v}_n\| = 1$, for the matrix $\mathbf{V}$ is that:

$$\mathbf{V}\mathbf{V}^{\mathrm{T}} = \mathbf{I} \qquad \text{or} \qquad \mathbf{V}^{-1} = \mathbf{V}^{\mathrm{T}} \qquad \text{(B23)}$$

Matrices with this property are said to be *orthonormal.* For these matrices, the following properties hold true:

$$\begin{gathered} (\mathbf{V}\mathbf{y})^{\mathrm{T}} (\mathbf{V}\mathbf{x}) = \mathbf{y}^{\mathrm{T}}\mathbf{x} \\ \|\mathbf{V}\mathbf{x}\| = \mathbf{V}\|\mathbf{x}\| \end{gathered} \qquad \text{(B24)}$$

In other words, the operator $\mathbf{V}$ does not change the length of a vector, nor does it change the angles between pairs of vectors. The operator merely *rotates* the whole vector space.

The decomposition of the symmetric matrices becomes:

$$\mathbf{H} = \mathbf{V}\Lambda\mathbf{V}^{\mathrm{T}} \qquad \text{(B25)}$$

The interpretation is as follows. The operation $\mathbf{H}\mathbf{x}$ can be regarded as a sequence of operations:

a) A rotation $\mathbf{V}^{\mathrm{T}}$ acting on the vector $\mathbf{x}$.
b) An elementwise multiplication (stretching) Λ acting on the elements of $\mathbf{V}^{\mathrm{T}}\mathbf{x}$.
c) A back rotation $\mathbf{V}$ acting on the vector $\Lambda\mathbf{V}^{\mathrm{T}}\mathbf{x}$.

The attentive reader sees an analogy with convolution, which is equivalent to a) Fourier transform, b) elementwise multiplication, and c) inverse Fourier transform.

B.6 Singular value decomposition

Singular value decomposition (SVD) is a mathematical theorem that states that any matrix $\mathbf{H}$ can be decomposed into two orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$, and a third matrix $\mathbf{S}$ which is a diagonal matrix:

$$\mathbf{H} = \mathbf{USV}^T \qquad \text{(B26)}$$

Suppose that $\mathbf{H}$ is a $N \times M$ matrix. Then:

- $\mathbf{U}$ is an orthonormal matrix, i.e. $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ with size $N \times N$.
- $\mathbf{V}$ is an orthonormal matrix, i.e. $\mathbf{V}^T\mathbf{V} = \mathbf{I}$ with size $M \times M$.
- $\mathbf{S}$ is a diagonal matrix with size $N \times M$. The diagonal elements s_{nn} are called the singular values.

Equation (B26) can be written in an alternative form. If we define: $\mathbf{U} = [\mathbf{u}_1 \ \cdots \ \mathbf{u}_N]$ and $\mathbf{V} = [\mathbf{v}_1 \ \cdots \ \mathbf{v}_M]$, where $\mathbf{u}_i$ and $\mathbf{v}_j$ are the columns in the $\mathbf{U}$ and $\mathbf{V}$ matrices, then:

$$\mathbf{H} = \sum_{i=1}^{K} s_{ii}\mathbf{u}_i\mathbf{v}_i^T \quad \text{with} \quad K = \min(N, M) \qquad \text{(B27)}$$

Note that some of the singular values might be zero.

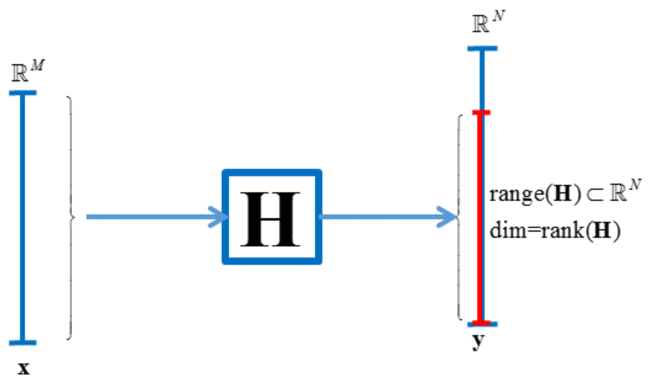
Corollary 1
From $\mathbf{HH}^T = \mathbf{USV}^T(\mathbf{USV}^T)^T = \mathbf{USV}^T\mathbf{VSU}^T = \mathbf{US}^2\mathbf{U}^T$ it follows that $\mathbf{u}_i$ are the eigenvectors of $\mathbf{HH}^T$ with associated eigenvalues s_{ii}^2.

Corollary 2
From $\mathbf{H}^T\mathbf{H} = (\mathbf{USV}^T)^T\mathbf{USV}^T = \mathbf{VSU}^T\mathbf{USV}^T = \mathbf{VS}^2\mathbf{V}^T$ it follows that $\mathbf{v}_i$ are the eigenvectors of $\mathbf{H}^T\mathbf{H}$ with associated eigenvalues s_{ii}^2.

B.7 The range and the rank of a matrix

Consider a $N \times M$ matrix $\mathbf{H}$. The operation $\mathbf{y} = \mathbf{Hx}$ maps the M dimensional space $\mathbb{R}^M$ of $\mathbf{x}$ to the N dimensional space $\mathbb{R}^N$ of $\mathbf{y}$.

The *range* of the matrix is the set of all possible outcomes $\mathbf{y}$ when $\mathbf{x}$ is varied over the whole M dimensional space $\mathbb{R}^M$. The range is a linear subspace of $\mathbb{R}^N$. This is because if $\mathbf{y}_1$ and $\mathbf{y}_2$ are in the range of $\mathbf{H}$, then there exists an $\mathbf{x}_1$ and an $\mathbf{x}_2$ such that $\mathbf{y}_1 = \mathbf{Hx}_1$ and $\mathbf{y}_2 = \mathbf{Hx}_2$. For any linear combination $\alpha\mathbf{x}_1 + \beta\mathbf{x}_2$ we have $\alpha\mathbf{y}_1 + \beta\mathbf{y}_2 = \mathbf{H}(\alpha\mathbf{x}_1 + \beta\mathbf{x}_2)$. Thus, $\alpha\mathbf{y}_1 + \beta\mathbf{y}_2$ is also in the range of $\mathbf{H}$.



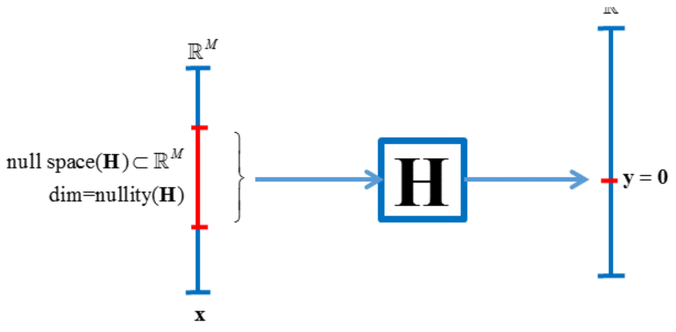
The range can be found by considering the singular value decomposition of the matrix $\mathbf{H}$:

$$\mathbf{H} = \mathbf{USV}^T == \sum_{i-1}^{K} s_{ii}\mathbf{u}_i\mathbf{v}_i^T \quad \text{with} \quad K = \min(N, M) \quad \text{(B28)}$$

Each $\mathbf{u}_i$ for which the associated singular value s_{ii} is not zero is in the range of $\mathbf{H}$. Since the matrix $\mathbf{U}$ is orthonormal, all these vectors $\mathbf{u}_i$ with non-zero s_{ii} form an orthonormal basis for the range. The dimension of the range equals the number of those vectors. This number is called the *rank* of the matrix, and is often denoted by $\mathrm{rank}(\mathbf{H})$.

B.8 The null space of a matrix

Consider a $N \times M$ matrix $\mathbf{H}$. The operation $\mathbf{y} = \mathbf{Hx}$ maps the M dimensional space $\mathbb{R}^M$ of $\mathbf{x}$ to the N dimensional space $\mathbb{R}^N$ of $\mathbf{y}$.



$\mathbb{R}^M$
null space$(\mathbf{H}) \subset \mathbb{R}^M$
dim=nullity$(\mathbf{H})$
$\mathbf{x}$
$\mathbf{H}$
$\mathbf{y} = \mathbf{0}$

The *null space* is the set of vectors $\mathbf{x}$ for which $\mathbf{Hx} = \mathbf{0}$. If $\mathbf{x}_1$ and $\mathbf{x}_2$ are vectors from the null space, then so is any linear combination $\alpha\mathbf{x}_1+\beta\mathbf{x}_2$. Thus, the null space is a linear subspace of $\mathbb{R}^M$.

The null space can be found by considering the singular value decomposition of the matrix $\mathbf{H}$. See above.

$$\mathbf{H}=\mathbf{U}\mathbf{S}\mathbf{V}^T == \sum_{i-1}^{K} s_{ii}\mathbf{u}_i\mathbf{v}_i^T \quad \text{with} \quad K=\min(N,M) \tag{B29}$$

Each $\mathbf{v}_i$ for which the associated singular value s_{ii} is not zero is not lying in the null space. All the other vectors are within the null space. Since the matrix $\mathbf{V}$ is orthonormal, all these vectors $\mathbf{v}_i$ in the null space form an orthonormal basis for the null space, and the dimension of this null space equals the number of those vectors. This dimension is sometimes called the *nullity* of the matrix.

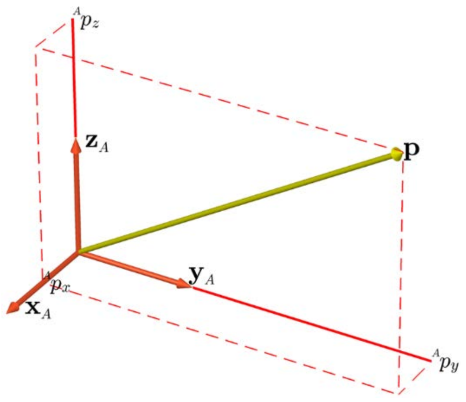
Equation Section (Next)

C. 3D pose

We consider the position and angular position of rigid objects in 3D spaces. The angular position is often called the *orientation* or the *attitude*. Taken together, the position and orientation are called the *pose*.

C.1 A position of a point in 3D

Given a coordinate system A that consists of three orthogonal basis vectors $(\mathbf{x}_A, \mathbf{y}_A, \mathbf{z}_A)$. They are orthogonal, e.g. $\mathbf{x}_A \perp \mathbf{x}_B$, and have unit length, e.g. $\|\mathbf{x}_A\| = 1$. These basis vectors span a 3D space. See figure.



A point $\mathbf{p}$ is represented in A as a vector:

$$\mathbf{p} = {}^A p_x \mathbf{x}_A + {}^A p_y \mathbf{y}_A + {}^A p_z \mathbf{z}_A \tag{C1}$$

Such a representation is conveniently denoted by its coordinates, stacked into a 3D vector in $\mathbb{R}^3$:

$$^A\mathbf{p} = \begin{bmatrix} ^A p_x \\ ^A p_y \\ ^A p_z \end{bmatrix} \tag{C2}$$

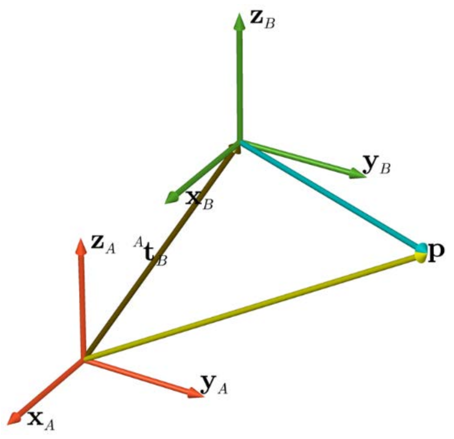
$^A\mathbf{p}$ is just a possible *representation* of the point $\mathbf{p}$. The same point can equally well be represented in some other coordinate system B, such that $^B\mathbf{p}$ refers to the same point $\mathbf{p}$.

C.2 Two coordinate systems

We consider two coordinate systems and describe the relations between the two.

C.2.1 Translated coordinate systems

Given two coordinate systems A and B. We assume that one is a translated version of the other. That is, B is shifted with respect to A, but it is not rotated. See figure below.



The origin of B is at a position $\mathbf{t}_B$ that is represented in A by $^A\mathbf{t}_B = \begin{bmatrix} ^A t_x & ^A t_y & ^A t_z \end{bmatrix}^{\mathrm{T}}$. Likewise, $^B\mathbf{t}_A$ is the position of the origin of A expressed in B. With the logic of this notation, $^B\mathbf{t}_B = \mathbf{0}$ and $^A\mathbf{t}_A = \mathbf{0}$.

The translation of B with respect to A is opposite to the translation of A with respect to B, so that we have:

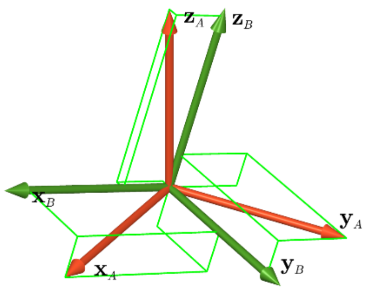
$$^B\mathbf{t}_A = -{}^A\mathbf{t}_B \tag{C3}$$

A point $\mathbf{p}$ can be represented both in A and in B. These representations are related to each other according to:

$$\begin{aligned} ^A\mathbf{p} &= {}^B\mathbf{p} - {}^B\mathbf{t}_A \\ ^B\mathbf{p} &= {}^A\mathbf{p} - {}^A\mathbf{t}_B \end{aligned} \tag{C4}$$

C.2.2 Rotated coordinate systems

We consider two coordinate systems that are rotated versions of each other. See figure.



The system A is described in terms of system B by defining the basis vectors of A expressed in those of B:

$$\begin{aligned} \mathbf{x}_A &= r_{xx}\mathbf{x}_B + r_{xy}\mathbf{y}_B + r_{xz}\mathbf{z}_B \\ \mathbf{y}_A &= r_{yx}\mathbf{x}_B + r_{yy}\mathbf{y}_B + r_{yz}\mathbf{z}_B \\ \mathbf{z}_A &= r_{zx}\mathbf{x}_B + r_{zy}\mathbf{y}_B + r_{zz}\mathbf{z}_B \end{aligned} \tag{C5}$$

This is conveniently expressed in matrix-vector notation:

$$\begin{bmatrix}\mathbf{x}_A\\ \mathbf{y}_A\\ \mathbf{z}_A\end{bmatrix}=\begin{bmatrix}r_{xx}&r_{xy}&r_{xz}\\ r_{yx}&r_{yy}&r_{yz}\\ r_{zx}&r_{zy}&r_{zz}\end{bmatrix}\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}\quad\text{or:}\quad\begin{bmatrix}\mathbf{x}_A\\ \mathbf{y}_A\\ \mathbf{z}_A\end{bmatrix}={}^A\mathbf{R}_B\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}\qquad\text{(C6)}$$

The matrix:

$${}^A\mathbf{R}_B=\begin{bmatrix}r_{xx}&r_{xy}&r_{xz}\\ r_{yx}&r_{yy}&r_{yz}\\ r_{zx}&r_{zy}&r_{zz}\end{bmatrix}\qquad\text{(C7)}$$

is called the *rotation matrix*.

For $\mathbf{x}_A$ we find:

$$\mathbf{x}_A=r_{xx}\mathbf{x}_B+r_{xy}\mathbf{y}_B+r_{xz}\mathbf{z}_B\quad\text{thus}\quad{}^B\mathbf{x}_A=\begin{bmatrix}r_{xx}\\ r_{xy}\\ r_{xz}\end{bmatrix}\qquad\text{(C8)}$$

Applying this to $\mathbf{y}_A$ $\mathbf{z}_A$ as well, we conclude:

$${}^A\mathbf{R}_B=\begin{bmatrix}{}^B\mathbf{x}_A & {}^B\mathbf{y}_A & {}^B\mathbf{z}_A\end{bmatrix}^{\mathrm{T}}\qquad\text{(C9)}$$

Note that, for instance, $r_{xx}\mathbf{x}_B$ is the projection of $\mathbf{x}_A$ on $\mathbf{x}_B$. Therefore,

$$r_{xx}=\frac{(\mathbf{x}_A,\mathbf{x}_B)}{\|\mathbf{x}_B\|^2}=\cos\varphi_{xx}$$

where φ_{xx} is the angle between $\mathbf{x}_A$ and $\mathbf{x}_B$. Likewise, $r_{xy}=\cos\varphi_{xy}$ with φ_{xy} the angle between $\mathbf{x}_A$ and $\mathbf{y}_B$, and so on. Apparently:

$${}^A\mathbf{R}_B=\begin{bmatrix}\cos(\varphi_{xx})&\cos(\varphi_{xy})&\cos(\varphi_{xz})\\ \cos(\varphi_{yx})&\cos(\varphi_{yy})&\cos(\varphi_{yz})\\ \cos(\varphi_{zx})&\cos(\varphi_{zy})&\cos(\varphi_{zz})\end{bmatrix}\qquad\text{(C10)}$$

For this reason, the rotation matrix is also called the *direct cosine matrix*.

To express the basis vectors of B in A we define the matrix ${}^B\mathbf{R}_A$. Inspection of eq (C10) reveals that ${}^B\mathbf{R}_A={}^A\mathbf{R}_B^{\mathrm{T}}$. Furthermore, we have:

$$\begin{bmatrix}\mathbf{x}_A\\ \mathbf{y}_A\\ \mathbf{z}_A\end{bmatrix}={}^A\mathbf{R}_B\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}={}^A\mathbf{R}_B{}^B\mathbf{R}_A\begin{bmatrix}\mathbf{x}_A\\ \mathbf{y}_A\\ \mathbf{z}_A\end{bmatrix}\qquad\text{(C11)}$$

The conclusion is:

$${}^A\mathbf{R}_B^{-1}={}^A\mathbf{R}_B^{\mathrm{T}}={}^B\mathbf{R}_A\qquad\text{(C12)}$$

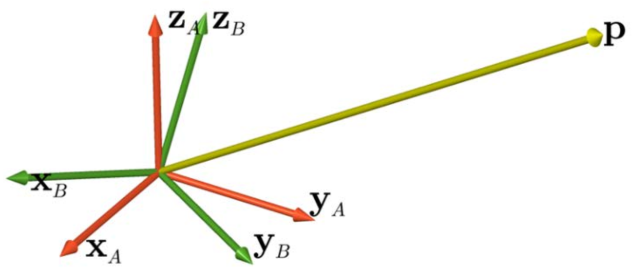
Corollary 3

Rotation matrices have 3 degrees of freedom. This latter follows from the following construction:

1. Consider the first column $\begin{bmatrix}r_{xx} & r_{xy} & r_{xz}\end{bmatrix}^{\mathrm{T}}$ of a rotation matrix. We can choose two elements freely, but within the constraint of unit length $r_{xx}^2+r_{xy}^2+r_{xz}^2=1$. The third element follows from this constraint.
2. Consider the second column $\begin{bmatrix}r_{yx} & r_{yy} & r_{yz}\end{bmatrix}^{\mathrm{T}}$. We can choose one element freely, but within the constraint of unit length $r_{yx}^2+r_{yy}^2+r_{yz}^2=1$. The other two elements follow from this constraint, but also from the orthogonality constraint $r_{xx}r_{yx}+r_{xy}r_{yy}+r_{xz}r_{yz}=0$.
3. The third column follows directly from the unit length constraint $r_{zx}^2+r_{zy}^2+r_{zz}^2=1$, and the two orthogonality constraints: $r_{xx}r_{zx}+r_{xy}r_{zy}+r_{xz}r_{zz}=0$ and $r_{yx}r_{zx}+r_{yy}r_{zy}+r_{yz}r_{zz}=0$.

C.2.3 Point representation in two rotated systems

Consider a point $\mathbf{p}$ that is represented in two rotated coordinate systems A and B



The point $\mathbf{p}$ has two representation:

$$\begin{aligned}\mathbf{p}&={}^Ap_x\mathbf{x}_A+{}^Ap_y\mathbf{y}_A+{}^Ap_z\mathbf{z}_A\\ &={}^Bp_x\mathbf{x}_B+{}^Bp_y\mathbf{y}_B+{}^Bp_z\mathbf{z}_B\end{aligned}$$

which can be written as:

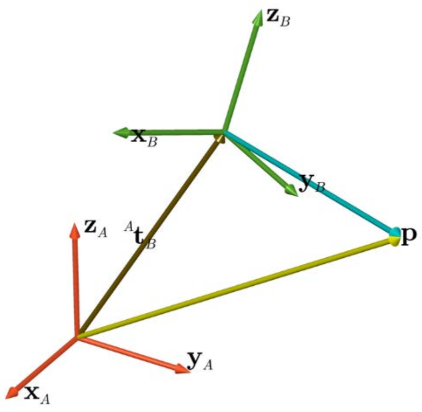
$$\mathbf{p}={}^A\mathbf{p}^{\mathrm{T}}\begin{bmatrix}\mathbf{x}_A\\ \mathbf{y}_A\\ \mathbf{z}_A\end{bmatrix}={}^A\mathbf{p}^{\mathrm{T}}\,{}^A\mathbf{R}_B\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}=\left({}^B\mathbf{R}_A{}^A\mathbf{p}\right)^{\mathrm{T}}\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}={}^B\mathbf{p}^{\mathrm{T}}\begin{bmatrix}\mathbf{x}_B\\ \mathbf{y}_B\\ \mathbf{z}_B\end{bmatrix}$$

From which follows:

$$\begin{aligned}{}^A\mathbf{p}&={}^A\mathbf{R}_B{}^B\mathbf{p}\\ {}^B\mathbf{p}&={}^B\mathbf{R}_A{}^A\mathbf{p}\end{aligned}\qquad\text{(C13)}$$

C.2.4 Rotated and translated frames

Lastly, we consider two coordinate systems that are rotated and translated versions of each other. Such coordinate systems are called *frames*. We have frame A which we consider as the *reference frame*. We have frame B that is defined with respect to A by means of $\left\{{}^A\mathbf{t}_B, {}^A\mathbf{R}_B\right\}$:



A point $\mathbf{p}$ is represented in B by ${}^B\mathbf{p}$. The same point is represented in A by:

$$ {}^A\mathbf{p} = {}^A\mathbf{R}_B {}^B\mathbf{p} + {}^A\mathbf{t}_B \tag{C14} $$

Note, that the notation works out intuitively. The subscript ${}_B$ and the superscript ^{B} are situated next to each other ${}_B^B$ indicating that the variables ${}^A\mathbf{R}_B$ and ${}^B\mathbf{p}$ "fit". The two scripts neutralize each other, The final result on the right side is an expression in A, which is consistent with the left side.

Going back, from a representation in A to one in B is likewise: ${}^B\mathbf{p} = {}^B\mathbf{R}_A {}^A\mathbf{p} + {}^B\mathbf{t}_A$. Substitution of (C14) yields ${}^B\mathbf{p} = {}^B\mathbf{R}_A\left({}^A\mathbf{R}_B {}^B\mathbf{p} + {}^A\mathbf{t}_B\right) + {}^B\mathbf{t}_A$, from which:

$$ \begin{aligned} {}^B\mathbf{t}_A &= -{}^B\mathbf{R}_A {}^A\mathbf{t}_B \\ {}^A\mathbf{t}_B &= -{}^A\mathbf{R}_B {}^B\mathbf{t}_A \end{aligned} \tag{C15} $$

This brings an alternative expression for (C14):

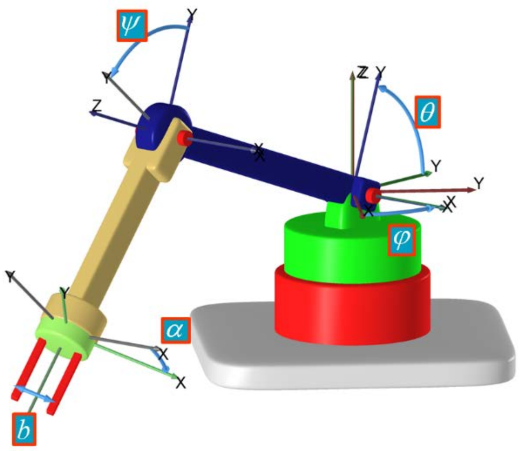
$$ {}^A\mathbf{p} = {}^A\mathbf{R}_B\left({}^B\mathbf{p} - {}^B\mathbf{t}_A\right) \tag{C16} $$

Here too, the notation is consistent: subtraction of vectors, i.e. translation, is only meaningful if the vectors are represented in the same frame.

C.3 Homogeneous transformations

In stereo vision, two frames may suffice to describe the geometry, but in other computer vision applications, more frames are needed. This occurs, for example, in *multi-view geometry* and also the processing of an image sequence from a moving camera. The situation also occurs in surgical navigation, and in robotics.

To describe, for instance, the kinematics of the robotic device, shown below, each rigid part of the construction has its own frame.



To describe the frame $\left\{{}^5\mathbf{R}_1, {}^5\mathbf{t}_1\right\}$ attached to the red fingers relative to frame attached to the red robotic base, we have to go through a sequence of five frames $1,\cdots,5$ attached to the different parts. Each transformation is described by a parameter that represents the state of the corresponding actuator, i.e. the angles φ, θ, ψ, α and the shift b.

It is cumbersome to apply eq (C14) for calculating the representation ${}^5\mathbf{p}$ of a point, given its representation ${}^1\mathbf{p}$ in the robotic reference frame:

$$ \begin{aligned} {}^5\mathbf{p} &= {}^5\mathbf{R}_4 {}^4\mathbf{p} + {}^5\mathbf{t}_4 \\ &= {}^5\mathbf{R}_4\left({}^4\mathbf{R}_3 {}^3\mathbf{p} + {}^4\mathbf{t}_3\right) + {}^5\mathbf{t}_4 \\ &= {}^5\mathbf{R}_4\left({}^4\mathbf{R}_3(\cdots) + {}^4\mathbf{t}_3\right) + {}^5\mathbf{t}_4 \qquad \text{and so on} \end{aligned} $$

Things become even more complicated if we want to inverse the relation, i.e. finding ${}^1\mathbf{p}$ given ${}^5\mathbf{p}$.

Homogeneous coordinates are invented to by-pass this complexity. The homogeneous representation of a 3D point ${}^1\mathbf{p}$ is a 4D vector ${}^1\underline{\mathbf{p}}$:

$$ {}^1\underline{\mathbf{p}} = \begin{bmatrix} {}^1p_x \\ {}^1p_y \\ {}^1p_z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^1\mathbf{p} \\ 1 \end{bmatrix} \tag{C17} $$

Definition of the homogeneous transformation matrix:

$$ {}^{2}\mathbf{T}_{1} = \left[\begin{array}{ccc|c} \multicolumn{3}{c|}{{}^{2}\mathbf{R}_{1}} & {}^{2}\mathbf{t}_{1} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] \qquad \text{(C18)} $$

converts the *inhomogeneous* equation (C14) into a *homogeneous* equation:

$$ {}^{2}\underline{\mathbf{p}} = {}^{2}\mathbf{T}_{1}{}^{1}\underline{\mathbf{p}} \qquad \text{(C19)} $$

Indeed, substitution of (C17) and (C19) in (C19) shows that:

$$ \begin{bmatrix} {}^{2}\mathbf{p} \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} \multicolumn{3}{c|}{{}^{2}\mathbf{R}_{1}} & {}^{2}\mathbf{t}_{1} \\ \hline 0 & 0 & 0 & 1 \end{array} \right] \begin{bmatrix} {}^{1}\mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{2}\mathbf{R}_{1}{}^{1}\mathbf{p} + {}^{2}\mathbf{t}_{1} \\ 1 \end{bmatrix} $$

The following two properties make the homogenous representations very useful:

$$ \begin{array}{rl} \text{cascading:} & {}^{3}\mathbf{T}_{1} = {}^{3}\mathbf{T}_{2}{}^{2}\mathbf{T}_{1} \\ \text{inversion:} & {}^{2}\mathbf{T}_{1} = {}^{1}\mathbf{T}_{2}^{-1} \end{array} \qquad \text{(C20)} $$

In the robotic example, the forward transformation becomes simply: ${}^{5}\underline{\mathbf{p}} = {}^{5}\mathbf{T}_{4}{}^{4}\mathbf{T}_{3}{}^{3}\mathbf{T}_{2}{}^{2}\mathbf{T}_{1}\,{}^{1}\underline{\mathbf{p}}$. The backward transformation is: ${}^{1}\underline{\mathbf{p}} = \left({}^{5}\mathbf{T}_{4}{}^{4}\mathbf{T}_{3}{}^{3}\mathbf{T}_{2}{}^{2}\mathbf{T}_{1}\right)^{-1}\,{}^{5}\underline{\mathbf{p}}$.

C.4 Representation of pose

3D objects not only have a position, but also an orientation. Position and orientation, taken together, is the *pose* of an object. To define the pose, first a frame must be defined that is attached to some point of the object. The pose of that object is given then by the position and orientation of that frame with respect to some reference frame. The representation of pose is either explicitly by the rotation matrix and origin position: $\left\{ {}^{ref}\mathbf{R}_{obj},\ {}^{ref}\mathbf{t}_{obj} \right\}$, or more conveniently by a transformation matrix: ${}^{ref}\mathbf{T}_{obj}$

C.5 Representations and functions in Matlab

Representation of points in Matlab

Matrix-vector products are found in two forms:

pre-mulitiplication: $\mathbf{y} = \mathbf{A}\mathbf{x}$ This applies to column vectors

post-multiplication: $\mathbf{y} = \mathbf{x}\mathbf{B}$ This applies to row vectors

The two forms are easily converted into each other:

$$ \text{if } \mathbf{y} = \mathbf{A}\mathbf{x} \text{ then: } \mathbf{y}^{\mathrm{T}} = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}} $$

Expressions can be given in one or the other form. In most textbooks, column vectors are used. Consequently, all expressions appear in the pre-multiplication form. A problem with Matlab is that it is not consistent with this. In many toolboxes, row vectors with post-multiplication are assumed. This is sometimes indicated in the documentation by mentioning the 'transposed form' as opposed to the 'non-transposed form'.

If a series of points are to be represented, Matlab often assumes row vectors. A set of N points are represented by a $N \times 3$ or $N \times 4$ (homogeneous) array. For instance, two points $\mathbf{p}_1 = \begin{bmatrix} p_{11} & p_{12} & p_{13} & 1 \end{bmatrix}^{\mathrm{T}}$ and $\mathbf{p}_2 = \begin{bmatrix} p_{21} & p_{22} & p_{23} & 1 \end{bmatrix}^{\mathrm{T}}$ are stored in a 2×4 array:

```
P = [p11 p12 p13 1;
     p21 p22 p23 1];
```

If these points are to be transformed by a 4×4 transformation matrix $\mathbf{T}$, e.g. $\mathbf{T}\mathbf{p}_1$, Matlab applies post-multiplication **`P*T`**. The consequence is that the matlab matrix is the transposed version of $\mathbf{T}$. That is: **`T=`**$\mathbf{T}^{\mathrm{T}}$ **.**

Useful Functions in Matlab

- **`cart2hom`** and **`hom2cart`** - Conversion between Cartesian coordinates and homogeneous coordinates. These functions are from the Robotics System Toolbox. The functions assume row vector representation of points.
- **`makehgtform`** - Create 4-by-4 transform matrix. This function is part of the set standard graphics functions of Matlab. The resulting matrix is in column vector representation, i.e non-transposed form.

D. Four representations of orientation and rotation

D.1 Rotation matrices

In section C.2.2, the rotation matrix ${}^{A}\mathbf{R}_B$, also called DCM, was introduced to specify the orientation of frame B relative to frame A. The interpretation is simple. Consider how the basis vectors

$$\mathbf{x}_B = \begin{bmatrix}1\\0\\0\end{bmatrix} \quad \mathbf{y}_B = \begin{bmatrix}0\\1\\0\end{bmatrix} \quad \mathbf{z}_B = \begin{bmatrix}0\\0\\1\end{bmatrix} \tag{D1}$$

of frame B are represented in frame A:

$${}^{A}\mathbf{R}_B\begin{bmatrix}\mathbf{x}_B & \mathbf{y}_B & \mathbf{z}_B\end{bmatrix} = {}^{A}\mathbf{R}_B = \begin{bmatrix}{}^{A}\mathbf{r}_1 & {}^{A}\mathbf{r}_2 & {}^{A}\mathbf{r}_3\end{bmatrix} \tag{D2}$$

Thus, the columns of ${}^{A}\mathbf{R}_B$ are the basis vectors of frame B, expressed in coordinates of frame A. Likewise, the columns of ${}^{B}\mathbf{R}_A$ are the basis vectors of frame A, expressed in coordinates of frame B.

A rotation matrix can also be regarded as a rotation of an object. Suppose that the points of an object are represented by a set ${}^{B}\mathbf{p}(k),\ k = 1, 2, \cdots$, then ${}^{A}\mathbf{R}_B{}^{B}\mathbf{p}(k)$ is the rotated version of the object, expressed in frame A.

Adding a second rotation, now applied to frame A with reference to a new frame C is defined by a rotation matrix ${}^{C}\mathbf{R}_A$. The addition of this new rotation to frame A is reflected in the rotation matrices by means of a matrix multiplication:

$${}^{C}\mathbf{R}_B = {}^{C}\mathbf{R}_A{}^{A}\mathbf{R}_B \tag{D3}$$

See eq (C20). It is clear that adding a zero rotation is the same as multiplication by a unit rotation matrix ${}^{C}\mathbf{R}_A = \mathbf{I}$.

D.1.1 Pros and cons

Rotation matrices have a nice property: the representation is unique. For a given rotation, there is only one associated rotation matrix. For a given rotation matrix, there is only one associated rotation.

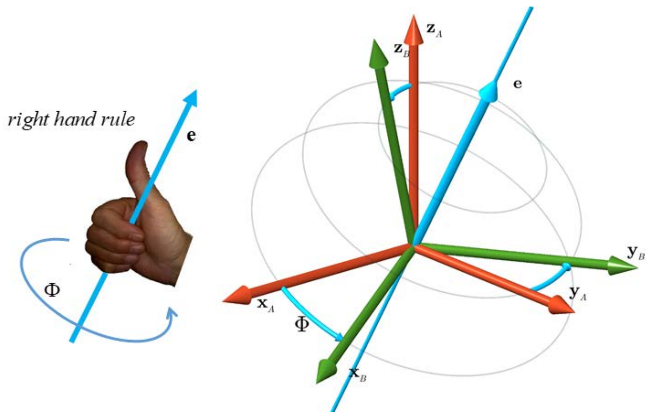
For navigation, rotation matrices are not always most appropriate. The matrices are redundant: 9 parameters to represent a quantity with 3 degrees of freedom. In other words, after each manipulation of a rotation matrix, we have to assure that in the resulting rotation matrix, 6 constraints are satisfied. This can be cumbersome and unstable. For instance, an application could require an interpolation from ${}^{A}\mathbf{R}_B$ to ${}^{A}\mathbf{R}_C$ consisting of a sequence of small rotations of equal size to obtain a smooth rotational movement. This is difficult to achieve with rotation matrices.

Another disadvantage of the rotation matrix occurs if we want to know whether the orientation of frame B is almost the same as the one of frame A. Is frame B almost aligned with frame A? To answer this, the angle between frame A and B must be quantified. This angle is not directly available, but can be derived from the trace of the rotation matrix as will be shown in eq (D10) in section D.2.

D.2 Axis-angle representation: the rotation vector

Euler's principal rotation theorem:

Each frame can be brought from an arbitrary initial orientation to another final orientation by means of a single rotation about an axis. The representation of this axis is the same in both the initial and final frame.



The figure above shows frame A and frame B. There is a rotation axis $\mathbf{e}$ which has equal representations in the two frames: $\mathbf{e} = {}^{A}\mathbf{e} = {}^{B}\mathbf{e}$. Since only the direction of the axis matters, we require unit length: $\|\mathbf{e}\| = 1$. The angle of rotation is denoted by Φ. Its direction is defined with the right-hand rule. Its absolute value $|\Phi|$ is the magnitude of rotation. A small value means that A and B are almost aligned.

As the axis is represented by the unit vector $\mathbf{e}$, the axis and the angle can be represented by a single 3D vector $\Phi\mathbf{e}$, which is called *the rotation vector*. The elements of this vector are sometimes called the *Rodriques parameters*.

D.2.1 Non-uniqueness

Each additional multiple of 360^0 to the angle Φ does not change the orientation of frame B. It only changes the amount of rotation to get there, starting from frame A. Thus, Φ can be replaced by $\Phi + 2\pi N$ with N any integer. The same orientation can also be reached by rotating the other way around. That is, Φ can also be replaced by $\Phi - 2\pi$. However, the two rotations Φ and $\Phi - 2\pi$ differ in the way that the orientation is reached. In the current example, Φ is the short way to rotate, whereas $\Phi - 2\pi$ is the long way.

The rotation axis is not unique either. If the opposite direction $-\mathbf{e}$ was chosen, the same rotations can be described by opposite angles: $-\Phi$ and $2\pi - \Phi$. Hence, there are four different principal rotations that all describe two different rotations, but that end up in just one orientation. Note also that if the rotation is zero, the axis $\mathbf{e}$ is undefined.

D.2.2 Conversion to and from rotation matrix

The rotation does not affect the rotation axis:

$$ {}^{A}\mathbf{e} = {}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{e} = {}^{B}\mathbf{e} \qquad \text{(D4)}$$

From this, it follows that ${}^{B}\mathbf{e}$ is the eigenvector of the matrix ${}^{A}\mathbf{R}_{B}$ with eigenvalue 1. This could be used to convert a rotation matrix into an axis-angle representation. An easier way to find the conversion between rotation matrix and axis-angle representation is *Rodrigues' rotation formula*. Suppose that Φ is the angle, and $\mathbf{e}$ is the rotation axis, that will rotate frame A to frame B, according the right-hand rule. Rodriques showed that if $\mathbf{e} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^{\mathrm{T}}$ is the rotation axis, then:

$$ {}^{B}\mathbf{R}_{A} = \mathbf{I} - \sin(\Phi)[\mathbf{e}]_{\times} + (1-\cos(\Phi))[\mathbf{e}]_{\times}^{2} \qquad \text{(D5)}$$

Here $[\mathbf{e}]_{\times}$ is the skew-symmetric matrix formed by $\mathbf{e}$:

$$ [\mathbf{e}]_{\times} \overset{def}{=} \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix} \qquad \text{(D6)}$$

Note that matrix multiplication with $[\mathbf{e}]_{\times}$ represents a cross product. That is, for an arbitrary vector $\mathbf{x}$, we have $\mathbf{e} \times \mathbf{x} = [\mathbf{e}]_{\times}\mathbf{x}$. See eq. (B16) and (B17).

The exponential function of a square matrix is defined by the series:

$$ \exp(\mathbf{A}) \overset{def}{=} \mathbf{I} + \mathbf{A} + \tfrac{1}{2}\mathbf{A}^2 + \cdots \tfrac{1}{n!}\mathbf{A}^n + \cdots \qquad \text{(D7)}$$

It can be proven that, with this definition, Rodrigues' rotation formula can also be expressed as:

$$ {}^{B}\mathbf{R}_{A} = \exp\left(-\Phi[\mathbf{e}]_{\times}\right) \qquad \text{(D8)}$$

The rotation back from B to A follows from ${}^{B}\mathbf{R}_{A} = {}^{B}\mathbf{R}_{A}^{\mathrm{T}}$:

$$ \begin{aligned} {}^{A}\mathbf{R}_{B} &= \mathbf{I} + \sin(\Phi)[\mathbf{e}]_{\times} + (1-\cos(\Phi))[\mathbf{e}]_{\times}^{2} \\ &= \exp\left(\Phi[\mathbf{e}]_{\times}\right) \end{aligned} \qquad \text{(D9)}$$

By expanding eq (D5), the conversion from rotation matrix to axis-angle is found. Suppose that the elements of ${}^{B}\mathbf{R}_{A}$ are denoted by r_{nm}, then:

$$ \begin{aligned} \Phi &= \arccos \tfrac{1}{2}(r_{11} + r_{22} + r_{33} - 1) \\ \mathbf{e} &= \frac{-1}{2\sin\left({}^{B}\Phi_{A}\right)} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \end{aligned} \qquad \text{(D10)}$$

D.2.3 Pros and Cons

The advantage of the axis/angle representation is twofold. First, it has a simple interpretation. Second, the magnitude of rotations is easily expressed as just $|\Phi|$. Therefore, a smooth interpolation between two different orientations is also easily accomplished.

A disadvantage is the already mentioned four-fold ambiguities. A second disadvantage is the complexity to concatenate two consecutive rotations with different rotational axes. To do so, first the representations must be converted to rotation matrices, so that eq (D3) applies, and then it has to be back transformed to axis-angle.

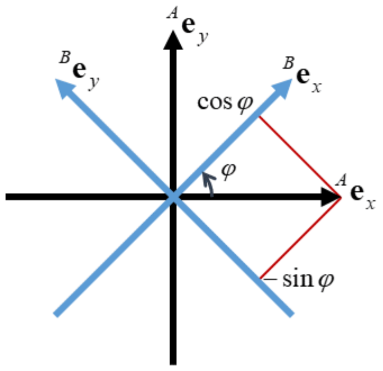
D.3 Euler angles

Euler angles are the most commonly used representation for the orientation of a body. Starting with a reference pose, the orientation is described by three sequential rotations about the axes of the coordinate system. The order in this sequence matters. " i-j-k Euler angles" means that we rotate first about the i-th axis, then about the j-th axis, and finally about the k-th axis. As an example, (3-2-1), or 'zyx' means:

- First, rotating about the z-axis.
- Then, rotating about the y-axis.
- Finally, rotating about the x-axis.

These rotations are taken over the axes of the rotating body itself, the so-called *intrinsic rotations*.

The rotation matrices that are associated with the rotations around these axes are called principal rotations. As usual, a positive angle corresponds to a counterclockwise rotation around an axis according to the right-hand rule. See the figure below. Here, A is regarded as the reference frame, e.g. an earth-fixed frame, and B is the frame of the body. The literature is not consistent in the definition whether the angles are defined from the reference frame to the moving frame, or reversed. In this syllabus, the principal rotations are defined from the reference frame A to the body frame B. That is ${}^{B}\mathbf{R}_{A}$. This definition is consistent with the functions of the Robotic Systems Toolbox of Matlab: It is also applied in, for instance, Farrell in [1]. In contrast, Fossen [2] defines a rotation from frame B to frame A. In applying software, and while reading literature, one has to check which definition is applied.



With our definition, we have:

$$\mathbf{R}_{x,\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix}$$
$$\mathbf{R}_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \qquad \text{(D11)}$$
$$\mathbf{R}_{z,\varphi} = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In the 'zxy' convention, the orientation of B is obtained by rotating A first around the z-axis, then the y-axis, and finally around the x-axis:

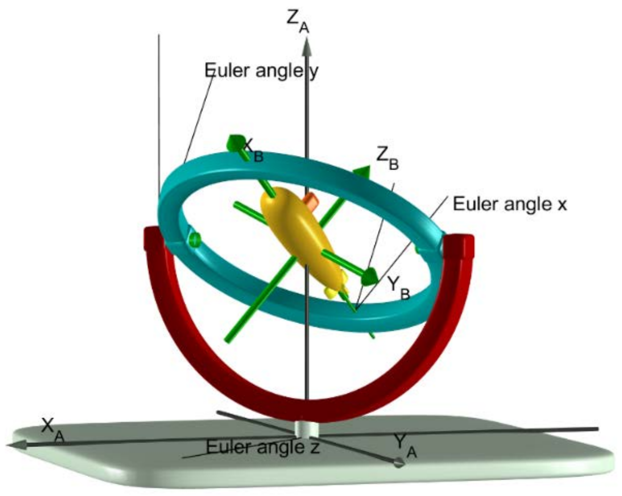
$${}^{B}\mathbf{R}_{A}(\varphi,\theta,\psi) = \mathbf{R}_{x,\psi}\mathbf{R}_{y,\theta}\mathbf{R}_{z,\varphi} \qquad \text{(D12)}$$

This is called an *extrinsic* rotation as they are applied to the axes of the reference frame.

To go back from frame B to frame A, we can use ${}^{A}\mathbf{R}_{B} = {}^{B}\mathbf{R}_{A}^{\mathrm{T}}$, yielding:

$${}^{A}\mathbf{R}_{B}(\varphi,\theta,\psi) = \mathbf{R}_{z,\varphi}^{\mathrm{T}}\mathbf{R}_{y,\theta}^{\mathrm{T}}\mathbf{R}_{x,\psi}^{\mathrm{T}} \qquad \text{(D13)}$$

This is called the *intrinsic* rotation. Note that the order of rotations is now reversed. Note also that, if the definition of the direction of an Euler angle is defined counterclockwise, as in some literature, then eq (D13) changes into ${}^{A}\mathbf{R}_{B}(\varphi,\theta,\psi) = \mathbf{R}_{z,\varphi}\mathbf{R}_{y,\theta}\mathbf{R}_{x,\psi}$, These expressions seem to be counterintuitive with the order of the 'zxy' convention. Yet, the expression is correct as will be shown in the example below. The figure below shows a cardan suspension that illustrates the intrinsic principal rotations.



In navigation and control applications, the 'zyx' convention is often used. The names of the angles are then:

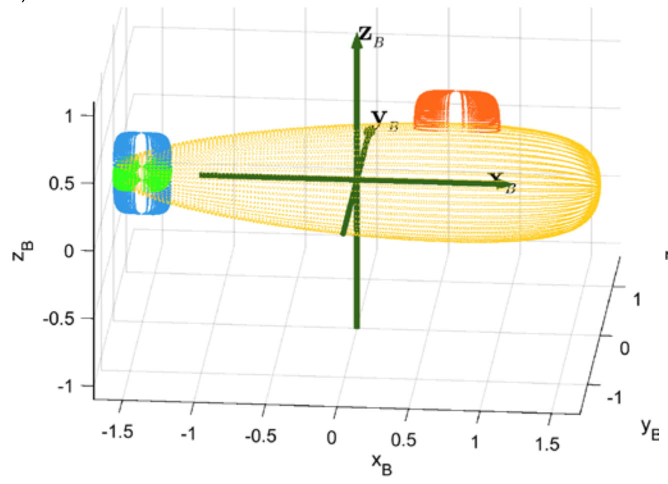
- Euler angle around z axis: *yaw*.
- Euler angle around y axis: *pitch*.
- Euler angle around x axis: *roll*.

11 other conventions exist of which 6 are symmetric, e.g. ‘zxz’, ‘zyz’ and 6 are asymmetric: ‘zyx’, ‘yzx’, and so on. In astronomy, the ‘zxz’ convention is popular to describe orbit planes.

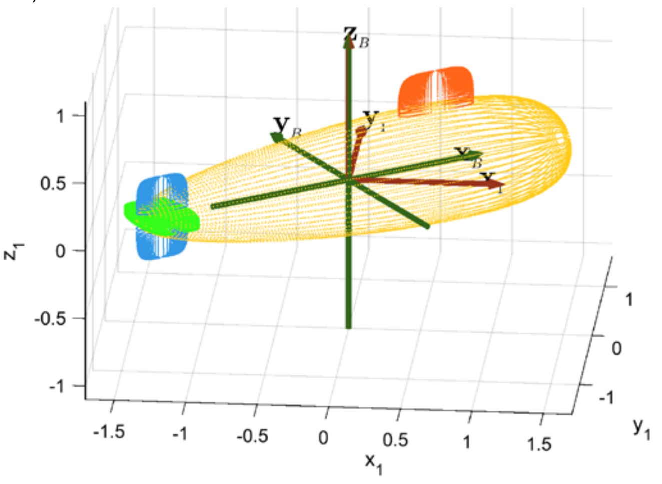
Whatever convention is chosen, there is always a so-called singularity. This always occurs for typical values of the second rotation angle, in which the values of the first and third angles are not unique[2]. For asymmetric conventions, this happens when the angle +90^0 and -90^0. For symmetric conventions the singularities are at 0^0 and 180^0.

[2] Compare this, for instance, with the latitudinal and longitudinal earth coordinates. At a pole, (latitude = 90^0), the longitudes are not defined.

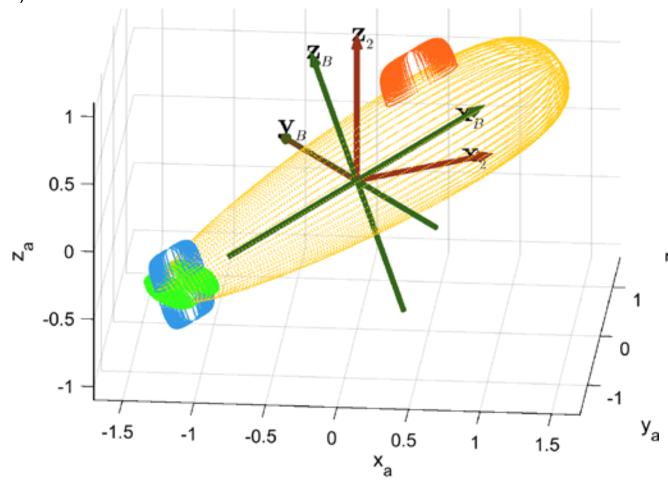
a)



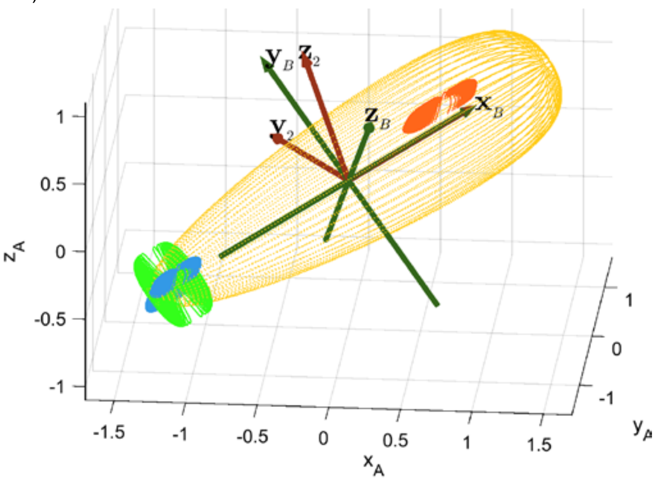
b)



c)



d)



Example: rotating an object over its own axes

Suppose, an object/body/vehicle is represented in a frame B which is rigidly attached to this body. The body itself is represented by a set of points ${}^{B}\mathbf{p}_k$ expressed in its own frame. This set, defined in this frame, lays down the reference orientation of the body. A graphical representation of the object is shown in Figure a) above.

Suppose that this object has an orientation in another frame A that is described with Euler angles in 'zyx' convention, as follows:
- yaw is 35⁰
- pitch is 20⁰
- roll is 45⁰.
How can we represent and visualize the object in frame A?

The use of Euler angles is demonstrated by successively rotating the object around the body-attached axes. In accordance with eq (D13) we first apply ${}^{1}\mathbf{p}_k = \mathbf{R}_{z,\varphi}^{\mathrm{T}}\,{}^{B}\mathbf{p}_k$ to rotate the object around its own z-axis. The result is shown in Figure b).

The next step is to rotate the object around the y-axis. Maybe, intuitively, this is done with ${}^{2}\mathbf{p}_k = \mathbf{R}_{y,\theta}^{\mathrm{T}}\,{}^{1}\mathbf{p}_k$, but this is wrong, as we would then rotate around the y_1-axis, whereas it should be around the body-attached y_B-axis. Therefore, we first have to rotate ${}^{1}\mathbf{p}_k$ back to the original B frame, perform the rotation, and then roll it back over the z-axis. That is: ${}^{2}\mathbf{p}_k = \mathbf{R}_{z,\varphi}^{\mathrm{T}}\mathbf{R}_{y,\theta}^{\mathrm{T}}\mathbf{R}_{z,\varphi}^{-\mathrm{T}}\,{}^{1}\mathbf{p}_k$. The full operation becomes ${}^{2}\mathbf{p}_k = \mathbf{R}_{z,\varphi}^{\mathrm{T}}\mathbf{R}_{y,\theta}^{\mathrm{T}}\,{}^{B}\mathbf{p}_k$. Figure c) shows the result.

The last step is the rotation over the body-attached x_B-axis. Following the same line of reasoning, the expression becomes ${}^{A}\mathbf{p}_k = \mathbf{R}_{z,\varphi}^{\mathrm{T}}\mathbf{R}_{y,\theta}^{\mathrm{T}}\mathbf{R}_{x,\psi}^{\mathrm{T}}\,{}^{B}\mathbf{p}_k$, which is compatible with eq (D13). The result is shown in Figure d).

In each step of this construction, angles are defined according to the right-hand rule when going from an old frame to the new frame. For instance, in the first rotation over the z-axis, the angle between the old $\mathbf{x}_1$-axis to the new $\mathbf{x}_B$-axis is +35⁰.

D.3.1 Conversion to and from rotation matrix

With the chosen principal rotations, each set of Euler angles can be mapped to the corresponding rotation matrix. Substitution of (D11) in (D12) shows how the rotation matrix is calculated from the Euler angles:

$$ {}^{B}\mathbf{R}_A(\psi,\theta,\varphi) = \begin{bmatrix} c_\varphi c_\theta & c_\theta s_\varphi & -s_\theta \\ c_\varphi s_\theta s_\psi - c_\psi s_\varphi & c_\psi c_\varphi + s_\psi s_\theta s_\varphi & c_\theta s_\psi \\ s_\psi s_\varphi + c_\psi c_\varphi s_\theta & c_\psi s_\varphi s_\theta - c_\varphi s_\psi & c_\theta c_\psi \end{bmatrix} \tag{D14}$$

For brevity, the shortcuts $c_\varphi = \cos\varphi$ and $s_\varphi = \sin\varphi$, and so on, have been made. The inverse conversion follows directly from (D14):

$$ \varphi = \operatorname{atan}\left(\frac{r_{12}}{r_{11}}\right) \qquad \theta = \operatorname{asin}(-r_{13}) \qquad \psi = \operatorname{atan}\left(\frac{r_{23}}{r_{33}}\right) \tag{D15}$$

In the atan function, the quadrants must be checked. If, for instance, r_{21} and r_{11} are both negative, 180^0 should be added to ψ.

D.3.2 Pros and Cons

Euler angles are easy to interpret, but there are some serious disadvantages:

- The singularities which occur in the second rotation. This is often called the *gimbal lock*.
- It is difficult to describe a sequence of consecutive rotations by means of Euler angles. One could convert the sets of Euler angles to rotation matrices by using (D12), then multiplying the found matrices, and finally converting the rotation matrix back to Euler angles using (D15).
- The magnitude of the overall rotation cannot easily be deduced directly by Euler angles. If all three angles are small, then the difference in orientations of A and B is also small. But a small difference in orientation does not guarantee that the Euler angles will be small.

D.4 Quaternions

Complex numbers:

A complex number $\mathbf{z} = a + b\mathbf{j}$ consists of a real part a and an imaginary part $b\mathbf{j}$. It has been set up such that a consistent framework for addition, subtraction, multiplication and division is established. This has been obtained from the axiom that $\mathbf{j}^2 = -1$, resulting in the following multiplication table:

	1	$\mathbf{j}$
1	1	$\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	-1

A complex number with unit length, e.g. $\mathbf{z} = \cos\phi + \mathbf{j}\sin\phi$ can be used to represent orientations in 2D. A possible advantage of doing so is when describing two consecutive rotations $\mathbf{z}_1 = \cos\phi_1 + \mathbf{j}\sin\phi_1$ and $\mathbf{z}_2 = \cos\phi_2 + \mathbf{j}\sin\phi_2$. The result is just $\mathbf{z}_1\mathbf{z}_2$.

Quaternions:

Quaternions are a number system that generalizes complex numbers from 2D to a 4D system. The basis of a quaternion consists of a scalar (the real part in a complex number), and 3 so-called quaternion units: $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$. A quaternion is a quadruple of real numbers that are combined as follows:

$$ \mathbf{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} \tag{D16}$$

q_0 is the *scalar* part. The triple $[q_1, q_2, q_3]$ is the vectorial part[3].

In 1843, the mathematician Hamilton invented a consistent framework for quaternions. It is based on the axioms:

$$ \mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1 \tag{D17}$$

This leads to the following multiplication table:

	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
1	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
$\mathbf{i}$	$\mathbf{i}$	-1	$\mathbf{k}$	$-\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	$-\mathbf{k}$	-1	$\mathbf{i}$
$\mathbf{k}$	$\mathbf{k}$	$\mathbf{j}$	$-\mathbf{i}$	-1

This system is not commutative, e.g. $\mathbf{ij} \neq \mathbf{ji}$.

A quaternion as a representation of orientation:

A quaternion represents the orientation and rotation in 3D by coding the axis-angle representation. Suppose that $\mathbf{e} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^{\mathrm{T}}$ is the principal rotation axis, and Φ is the rotation angle, then the associated quaternion is defined as:

[3] Beware: some authors use the notation $\mathbf{q} = q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} + q_4$, so that q_4 is the scalar part.

$$q_0 = \cos\tfrac{1}{2}\Phi \qquad \begin{aligned} q_1 &= e_1 \sin\tfrac{1}{2}\Phi \\ q_2 &= e_2 \sin\tfrac{1}{2}\Phi \\ q_3 &= e_3 \sin\tfrac{1}{2}\Phi \end{aligned} \qquad \text{(D18)}$$

Since a quaternion has 4 parameters, there must be one constraint. This constraint follows from the requirement that the rotation axis has unit length: $\|\mathbf{e}\|^2 = e_1^2 + e_2^2 + e_3^2 = 1$. Only its direction matters. In general, $\cos^2\alpha + \sin^2\alpha = 1$. Thus, the quaternions must also have unit length:

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1 \qquad \text{(D19)}$$

The requirement that $e_1^2 + e_2^2 + e_3^2 = 1$ means that the vector $\mathbf{e}$ must be situated on the surface of sphere with unit radius. Equation (D19) states that the quaternion must be situated on the 3D surface of a 4D hypersphere with unit radius.

Interpretation:
Since $q_0 = \cos\tfrac{1}{2}\Phi$, it is a direct measure of the magnitude of rotation:

$q_0 = 1$: no rotation at all.
$q_0 \approx 1$: small rotation.
$q_0 = 0$: rotation of 180⁰.
$q_0 < 0$: rotation larger than 180⁰.

The direction of the vectorial part $[q_1 \quad q_2 \quad q_3]^{\mathrm{T}}$ is the rotation axis. Its length equals $\sin\tfrac{1}{2}\Phi$. Therefore, a quaternion is well defined if $\Phi = 0$. This is unlike the axis-angle representation, in which the vector $\mathbf{e}$ is undefined in case of zero rotations.

D.4.1 Non-uniqueness

The axis-angle representation $(\mathbf{e}, \Phi)$ is equivalent with the axis-angle representation $(-\mathbf{e}, -\Phi)$. However, $\cos(-\tfrac{1}{2}\Phi) = \cos(\tfrac{1}{2}\Phi)$ and $-\sin(-\tfrac{1}{2}\Phi) = \sin(\tfrac{1}{2}\Phi)$. Therefore, the two equivalent axis-angle representations have just one quaternion representation.

The pair $(\mathbf{e}, \Phi)$ represents the same orientation as $(\mathbf{e}, \Phi - 2\pi)$. Yet, they differ in the sense that $(\mathbf{e}, \Phi)$ could be the short way to rotate frame B to its orientation relative to frame A. If this is true, and $\Phi \geq 0$, then $(\mathbf{e}, \Phi - 2\pi)$ is the long way around. If the short angle is negative, $\Phi < 0$, then $(\mathbf{e}, \Phi + 2\pi)$ is the long way. This property is reflected in the quaternions by realizing that: $\sin(\tfrac{1}{2}\Phi) = -\sin\tfrac{1}{2}(\Phi - 2\pi) =$ $= -\sin\tfrac{1}{2}(\Phi + 2\pi)$ and $\cos(\tfrac{1}{2}\Phi) = -\cos\tfrac{1}{2}(\Phi - 2\pi) = = -\cos\tfrac{1}{2}(\Phi + 2\pi)$. The consequence of this is that if the quaternion $\mathbf{q}$ represents the short angle rotation, then the long way rotation is represented by $-\mathbf{q}$. This is consistent with the earlier observation that $q_0 < 0$ represents a rotation larger than 180⁰, and that $q_0 > 0$ is associated with a rotation smaller than 180⁰.

D.4.2 Conversions

The rotation matrix that is associated with quaternions follows from their connection between the axis-angle representation, and from Rodriques formula, eq (D5):

$${}^{A}\mathbf{R}_B = \begin{bmatrix} 1-2(q_2{}^2+q_3{}^2) & 2(q_0q_3+q_1q_2) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & 1-2(q_1{}^2+q_3{}^2) & 2(q_0q_1+q_2q_3) \\ 2(q_0q_2+q_1q_3) & 2(q_2q_3-q_0q_1) & 1-2(q_1{}^2+q_2{}^2) \end{bmatrix} \qquad \text{(D20)}$$

The inverse relation follows from inspecting the trace of this matrix. This yields:

$$\begin{aligned} q_0 &= \pm\tfrac{1}{2}\sqrt{r_{11}+r_{22}+r_{33}+1} & q_2 &= \frac{r_{31}-r_{13}}{4q_0} \\ q_1 &= \frac{r_{23}-r_{32}}{4q_0} & q_3 &= \frac{r_{12}-r_{21}}{4q_0} \end{aligned} \qquad \text{(D21)}$$

This is singular if $q_0 = 0$. A variant of eq (D21), called *Sheppard's method*, avoids this singularity.

D.4.3 Pros and Cons

The major benefits of quaternions are:

- There is no singularity. This is in contrast with both Euler angles (gimbal lock) and axis-angle representation (axis undefined if the rotation angle is zero).
- There is only one constraint as opposed to rotation matrices that have six constraints.
- A sequence of two rotations, i.e. $\mathbf{q}^A$ and $\mathbf{q}^B$ is accomplished by quaternion multiplication:

$$\mathbf{q}^C = \mathbf{q}^B \otimes \mathbf{q}^A = \begin{bmatrix} q_0^B & -q_1^B & -q_2^B & -q_3^B \\ q_1^B & q_0^B & q_3^B & -q_2^B \\ q_2^B & -q_3^B & q_0^B & q_1^B \\ q_3^B & q_2^B & -q_1^B & q_0^B \end{bmatrix} \begin{bmatrix} q_0^A \\ q_0^A \\ q_0^A \\ q_0^A \end{bmatrix} \qquad \text{(D22)}$$

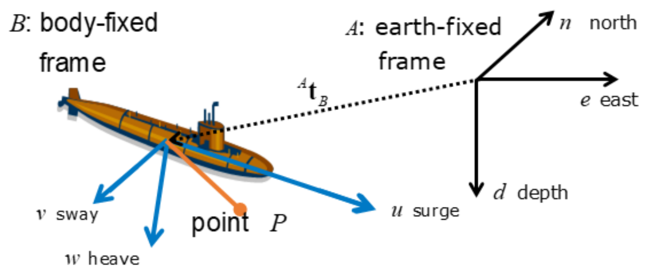
  Note that the matrix, which accomplishes the quaternion multiplication is orthonormal, and thus can easily be inverted to implemented subtraction of rotations.
- A disadvantage is that the quaternions are difficult to visualize.

Useful Functions in Matlab (Robotic system toolbox)

`axang2quat`	axis-angle rotation to quaternion
`axang2rotm`	axis-angle rotation to rotation matrix
`axang2tform`	axis-angle rotation to homogeneous transformation
`eul2quat`	Euler angles to quaternion
`eul2rotm`	Euler angles to rotation matrix
`eul2tform`	Euler angles to homogeneous transformation
`quat2axang`	quaternion to axis-angle rotation
`quat2eul`	quaternion to Euler angles
`quat2rotm`	quaternion to rotation matrix
`quat2tform`	quaternion to homogeneous transformation
`rotm2axang`	rotation matrix to axis-angle rotation
`rotm2eul`	rotation matrix to Euler angles
`rotm2quat`	rotation matrix to quaternion
`rotm2tform`	rotation matrix to homogeneous transformation
`tform2axang`	homogeneous transformation to axis-angle rotation
`tform2eul`	Extract Euler angles from homogeneous transformation
`tform2quat`	Extract quaternion from homogeneous transformation
`tform2rotm`	Extract rotation matrix from homogeneous transformation
`tform2trvec`	Extract translation vector from homogeneous transformation
`angdiff`	Difference between two angles
`cart2hom`	Cartesian coordinates to homogeneous coordinates
`hom2cart`	homogeneous coordinates to Cartesian coordinates
`trvec2tform`	translation vector to homogeneous transformation

E. Velocities and accelerations in 3D

In the mathematical characterization of motion, three coordinate systems (frames), are at stake: two frames that move relative to each another, and a third frame which is used to express this motion in. As an example, and to make the situation more comprehensible, we will consider a marine example with an earth-fixed frame, a body-fixed frame (the submarine), and a point P. See figure:



The point may move within the body. At the same time, the body can move wrt (= with respect to) the earth.

We will consider the velocity and acceleration of the point P in the case that it is moving with respect to the body B. We will do so for the situation that the body moves, but still keeps its orientation with respect to the earth A. Next, we'll see what happens if the body is also rotating with respect to the earth.

There are two coordinate systems at stake: B and A. The position of the origin of B relative to A is denoted by ${}^{A}\mathbf{t}_B$. Likewise, the orientation of B relative to A is denoted by a rotation matrix ${}^{A}\mathbf{R}_B$. The position of P, expressed in B coordinates is ${}^{B}\mathbf{p}$. When expressed in A, it is given by ${}^{A}\mathbf{p} = {}^{A}\mathbf{R}_B {}^{B}\mathbf{p} + {}^{A}\mathbf{t}_B$.

E.1 Linear velocities and accelerations

If the body does not rotate with respect to the earth, its rotation matrix ${}^{A}\mathbf{R}_B$ is fixed. Its position ${}^{A}\mathbf{t}_B$ might still change in time. The velocity of the body wrt the earth is found as:

$$ {}^{A}\mathbf{v}_{AB} = \frac{d}{dt}\,{}^{A}\mathbf{t}_B \qquad \text{(E1)} $$

The post-fixed subscript AB means that ${}^{A}\mathbf{v}_{AB}$ is the velocity of B wrt A. The pre-fixed superscript A indicates that this velocity is expressed in coordinates of frame A.

The point P might be moving wrt the body. Its velocity relative to B, and expressed in B is:

$$ {}^{B}\mathbf{v}_{BP} = \frac{d}{dt}\,{}^{B}\mathbf{p} \qquad \text{(E2)} $$

We could also express the same motion in earth coordinates, that is A coordinates, which then would be:

$$ {}^{A}\mathbf{v}_{BP} = {}^{A}\mathbf{R}_B {}^{B}\mathbf{v}_{BP} \qquad \text{(E3)} $$

Finally, we could consider the velocity of the point wrt the earth, and express this in earth coordinates:

$$ {}^{A}\mathbf{v}_{AP} = {}^{A}\mathbf{v}_{BP} + {}^{A}\mathbf{v}_{AB} \qquad \text{(E4)} $$

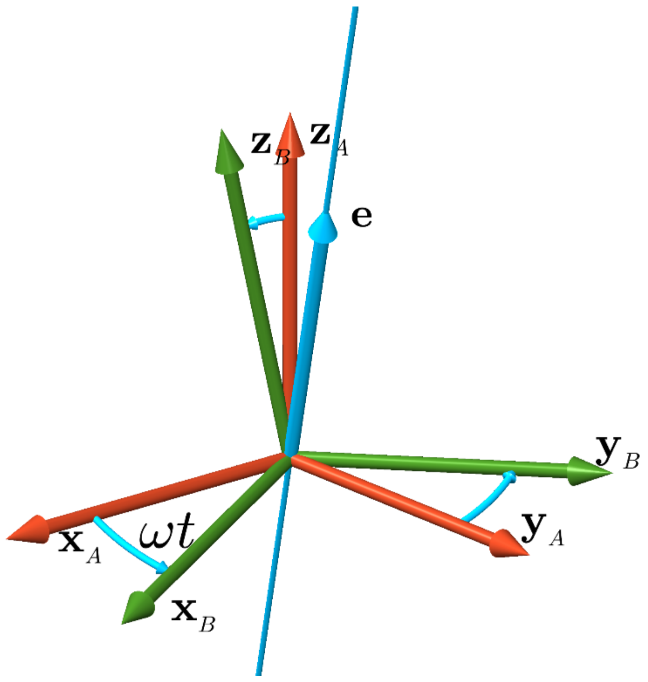
For the linear acceleration of the point, the same line of reasoning holds. If ${}^{A}\mathbf{a}_{AB}$ and ${}^{B}\mathbf{a}_{BP}$ are the accelerations of the body wrt the earth, and the point wrt the body, respectively, then:

$$ {}^{A}\mathbf{a}_{AP} = {}^{A}\mathbf{R}_B {}^{B}\mathbf{a}_{BP} + {}^{A}\mathbf{a}_{AB} = {}^{A}\mathbf{a}_{BP} + {}^{A}\mathbf{a}_{AB} \qquad \text{(E5)} $$

E.2 Rotations and angular velocities

E.2.1 Rotating frames

We consider again the two frames A and B, and assume for the moment that their origins coincide. Frame B is spinning around a rotation axis $\mathbf{e}$ in frame A with a turning rate, or angular velocity of ω (rad/s). See figure.



Such a spinning rotation is usually denoted by a vector $\boldsymbol{\omega} = \omega\,\mathbf{e}$, where $\mathbf{e}$ is a unit vector. Thus, $|\omega| = \|\boldsymbol{\omega}\|$. Note, that if ω is constant, the angle between the two frames would be linearly increasing with time: $\Phi(t) = \omega t + \omega_0$. Since different coordinate systems are at state, here too, it makes sense to add a prefix and postfix: ${}^{A}\boldsymbol{\omega}_{AB}$ indicating that this vector represents a rotation of frame B wrt A, and that it is expressed in coordinates of frame A. We'll only use this notation, when needed. Note that the rotation axis is invariant under the rotation, so that ${}^{A}\boldsymbol{\omega}_{AB} = {}^{A}\mathbf{R}_{B}\,{}^{B}\boldsymbol{\omega}_{AB}$. Furthermore, we have ${}^{A}\boldsymbol{\omega}_{BA} = -{}^{A}\boldsymbol{\omega}_{AB}$.

Suppose now, that at time t, the frames A and B coincide. At time $t+dt$, frame B will be rotated by an amount $\omega\,dt$. What will be the corresponding rotation matrix $\mathbf{R}$ if dt approaches zero?

The answer follows from Rodrigues' formula in eq (D9):

$$\begin{aligned} {}^{A}\mathbf{R}_{B} &= \mathbf{I} + \sin(\omega\,dt)[\mathbf{e}]_{\times} + (1-\cos(\omega\,dt))[\mathbf{e}]_{\times}^{2} \\ &= \mathbf{I} + \omega\,dt[\mathbf{e}]_{\times} \end{aligned} \tag{E6}$$

This follows simply from $\sin\omega\,dt \simeq \omega\,dt$ and $\cos\omega\,dt \simeq 1$ provided that dt approaches zero.

Suppose that we define the matrix

$$\boldsymbol{\Omega} \overset{def}{=} \omega[\mathbf{e}]_{\times} = \begin{bmatrix} 1 & -\omega_z & \omega_y \\ \omega_z & 1 & -\omega_x \\ -\omega_y & \omega_x & 1 \end{bmatrix} \tag{E7}$$

then eq (E6) further simplifies to:

$$\mathbf{R} = \mathbf{I} + \boldsymbol{\Omega}dt \tag{E8}$$

Properties of the skew-matrix cross product operator:

$\boldsymbol{\Omega}$ is defined, such that for any $\mathbf{p}$, we have $\boldsymbol{\Omega}\mathbf{p} = \boldsymbol{\omega}\times\mathbf{p}$, Since $\boldsymbol{\Omega} = -\boldsymbol{\Omega}^{\mathrm{T}}$, and $\boldsymbol{\omega}\times\mathbf{p} = -\mathbf{p}\times\boldsymbol{\omega}$, we have the following properties:

$$\begin{aligned} \boldsymbol{\Omega}\mathbf{p} &= \boldsymbol{\omega}\times\mathbf{p} \\ -\boldsymbol{\Omega}\mathbf{p} &= \mathbf{p}\times\boldsymbol{\omega} \end{aligned} \tag{E9}$$

The cross product is rotation invariant in the sense that $\mathbf{R}(\boldsymbol{\omega}\times\mathbf{p}) = \mathbf{R}\boldsymbol{\omega}\times\mathbf{R}\mathbf{p}$, This implies $\mathbf{R}\boldsymbol{\omega}\times\mathbf{p} = \mathbf{R}(\boldsymbol{\omega}\times\mathbf{R}^{\mathrm{T}}\mathbf{p})$. Therefore, the following property holds:

$$\text{if} \quad \boldsymbol{\Omega}\mathbf{p} = \boldsymbol{\omega}\times\mathbf{p} \quad \text{then} \quad \mathbf{R}\boldsymbol{\Omega}\mathbf{R}^{\mathrm{T}}\mathbf{p} = \mathbf{R}\boldsymbol{\omega}\times\mathbf{p} \tag{E10}$$

Corollary:

If at time t, the orientation of frame B wrt frame A is given by the rotation matrix ${}^{A}\mathbf{R}_{B}(t)$, then the rate at which the rotation matrix changes is:

$$\begin{aligned} {}^{A}\dot{\mathbf{R}}_{B}(t) &= {}^{A}\mathbf{R}_{B}(t)\,{}^{B}\boldsymbol{\Omega}_{AB} \\ {}^{B}\dot{\mathbf{R}}_{A}(t) &= -\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{R}_{A}(t) \end{aligned} \tag{E11}$$

The dot represents differentiation, $\dot{\mathbf{R}} = d\mathbf{R}/dt$ .The corollary is because ${}^{A}\mathbf{R}_{B}(t+dt) = {}^{A}\mathbf{R}_{B}(t)(\mathbf{I} + \boldsymbol{\Omega}dt)$ and $\boldsymbol{\Omega}^{\mathrm{T}} = -\boldsymbol{\Omega}$.

E.2.2 Velocity of a point in two moving frames

We consider the following case:

- Frame B moves wrt frame A with a linear velocity ${}^{A}\mathbf{v}_{AB}$ and an angular velocity of ${}^{B}\boldsymbol{\omega}_{AB}$.
- The point P is moving wrt frame B with a velocity of ${}^{B}\mathbf{v}_{BP}$.

The question is what is the velocity of P wrt frame A? The answer is given by expressing the position of the point in frame A:

$${}^{A}\mathbf{p} = {}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{t}_{B} \tag{E12}$$

The velocity of P wrt frame A is obtained by differentiating this expression wrt to the time:

$$\begin{aligned} {}^{A}\mathbf{v}_{AP} &= \frac{d}{dt}\,{}^{A}\mathbf{p} = \frac{d}{dt}\left({}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{t}_{B}\right) \\ &= {}^{A}\mathbf{R}_{B}\left({}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p} + {}^{B}\mathbf{v}_{BP}\right) + {}^{A}\mathbf{v}_{AB} \\ &= {}^{A}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{v}_{BP} + {}^{A}\mathbf{v}_{AB} \\ \text{where } {}^{A}\boldsymbol{\Omega}_{AB} &= {}^{A}\mathbf{R}_{B}\,{}^{B}\boldsymbol{\Omega}_{AB} \end{aligned} \tag{E13}$$

The first term on the right hand side describes the motion of the point due to the rotation of frame B wrt frame A. The second term is due to the velocity of the point within frame B. The third term is the linear velocity of frame B wrt frame A.

E.2.3 Addition of angular velocities

Consider two frames B and C, and a fixed one A. Assume that the origins of the three frames coincide. We then have: ${}^{A}\mathbf{p} = {}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{R}_{C}\,{}^{C}\mathbf{p}$, and ${}^{A}\mathbf{R}_{C} = {}^{A}\mathbf{R}_{B}\,{}^{B}\mathbf{R}_{C}$. We assume that frame B rotates wrt A, This is described by the angular velocity vector ${}^{B}\boldsymbol{\omega}_{AB}$ and the associated matrix ${}^{B}\boldsymbol{\Omega}_{AB}$. Likewise, the rotation of frame C wrt frame B is described by ${}^{C}\boldsymbol{\omega}_{BC}$ and ${}^{C}\boldsymbol{\Omega}_{BC}$.

The question is: what will be the angular velocity ${}^{A}\boldsymbol{\omega}_{AC}$, or equivalently ${}^{A}\boldsymbol{\Omega}_{AC}$, i.e. the rotation of frame C wrt frame A?

According to (E11), ${}^{A}\dot{\mathbf{R}}_C = {}^{A}\mathbf{R}_C\,{}^{C}\boldsymbol{\Omega}_{AC}$. Therefore, if we would be able to find an expression for ${}^{A}\dot{\mathbf{R}}_C$, the question is answered:

$$\begin{aligned}{}^{A}\dot{\mathbf{R}}_C &= \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\mathbf{R}_C\right)\\ &= {}^{A}\dot{\mathbf{R}}_B\,{}^{B}\mathbf{R}_C + {}^{A}\mathbf{R}_B\,{}^{B}\dot{\mathbf{R}}_C \\ &= {}^{A}\mathbf{R}_B\,{}^{B}\Omega_{AB}\,{}^{B}\mathbf{R}_C + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{R}_C\,{}^{C}\Omega_{BC}\end{aligned} \qquad \text{(E14)}$$

This expression can be further simplified by substitution of ${}^{A}\mathbf{R}_C = {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{R}_C$, and by introducing ${}^{A}\mathbf{R}_B^{\mathrm{T}}\,{}^{A}\mathbf{R}_B = \mathbf{I}$:

$$\begin{aligned}{}^{A}\dot{\mathbf{R}}_C &= {}^{A}\mathbf{R}_B\,\underbrace{{}^{B}\mathbf{R}_C\,{}^{B}\mathbf{R}_C^{\mathrm{T}}}_{=\mathbf{I}}\,{}^{B}\Omega_{AB}\,{}^{B}\mathbf{R}_C + {}^{A}\mathbf{R}_C\,{}^{C}\Omega_{BC}\\ &= {}^{A}\mathbf{R}_C\,{}^{B}\mathbf{R}_C^{\mathrm{T}}\,{}^{B}\Omega_{AB}\,{}^{B}\mathbf{R}_C + {}^{A}\mathbf{R}_C\,{}^{C}\Omega_{BC}\\ &= {}^{A}\mathbf{R}_C\left({}^{C}\mathbf{R}_B\,{}^{B}\Omega_{AB}\,{}^{B}\mathbf{R}_C + {}^{B}\Omega_{BC}\right)\end{aligned} \qquad \text{(E15)}$$

Thus, the combined rotation is given by:

$${}^{A}\Omega_{AC} = {}^{C}\Omega_{AC} = {}^{C}\mathbf{R}_B\,{}^{B}\Omega_{AB}\,{}^{C}\mathbf{R}_B^{\mathrm{T}} + {}^{C}\Omega_{BC} \qquad \text{(E16)}$$

Translated back to angular velocities vectors, and by using (E10), we have:

$${}^{C}\boldsymbol{\omega}_{AC} = {}^{C}\mathbf{R}_B\,{}^{B}\boldsymbol{\omega}_{AB} + {}^{C}\boldsymbol{\omega}_{BC} \qquad \text{(E17)}$$

This emphasizes the property that angular velocities vectors are free linear vectors in 3D space which have meaningful addition and multiplication operators.

E.3 Acceleration of a point in rotating frames

We consider again two frames A and B and a point P:

- Frame B moves wrt frame A with a linear velocity ${}^{A}\mathbf{v}_{AB}$ and an angular velocity of ${}^{A}\boldsymbol{\omega}_{AB} = {}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\omega}_{AB}$.
- Point P moves wrt frame B with a velocity of ${}^{B}\mathbf{v}_{BP}$.

The question is what is the acceleration of P wrt frame A?

The answer is given by differentiating the expression for the velocity shown in eq (E13):

$${}^{A}\mathbf{a}_{AP} = \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{v}_{BP} + {}^{A}\mathbf{v}_{AB}\right) \qquad \text{(E18)}$$

Application of the chain rule yields:

$$\begin{aligned}{}^{A}\mathbf{a}_{AP} &= \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{v}_{BP} + {}^{A}\mathbf{v}_{AB}\right)\\ &= \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p}\right) + \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\mathbf{v}_{BP}\right) + {}^{A}\mathbf{a}_{AB}\end{aligned} \qquad \text{(E19)}$$

The first and second term in this expression is expanded by using the chain rule as follows:

$$\begin{aligned}\frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p}\right) &= {}^{A}\dot{\mathbf{R}}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\dot{\boldsymbol{\Omega}}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{v}_{BP}\\ &= {}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}^2_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\dot{\boldsymbol{\Omega}}_{AB}\,{}^{B}\mathbf{p} + {}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{v}_{BP}\\ \frac{d}{dt}\left({}^{A}\mathbf{R}_B\,{}^{B}\mathbf{v}_{BP}\right) &= {}^{A}\dot{\mathbf{R}}_B\,{}^{B}\mathbf{v}_{BP} + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{a}_{BP}\\ &= {}^{A}\mathbf{R}_B\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{v}_{BP} + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{a}_{BP}\end{aligned}$$
(E20)

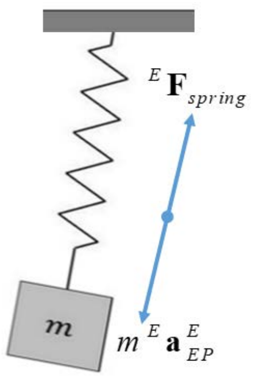
Rearrangement and grouping of the terms yield:

$${}^{A}\mathbf{a}_{AP} = {}^{A}\mathbf{R}_B\left({}^{B}\boldsymbol{\Omega}^2_{AB}\,{}^{B}\mathbf{p} + {}^{B}\dot{\boldsymbol{\Omega}}_{AB}\,{}^{B}\mathbf{p} + 2\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{v}_{BP} + {}^{B}\mathbf{a}_{BP}\right) + {}^{A}\mathbf{a}_{AB}$$
(E21)

This expression has the following interpretation:

${}^{B}\boldsymbol{\Omega}^2_{AB}\,{}^{B}\mathbf{p}$:	centripetal acceleration
${}^{B}\dot{\boldsymbol{\Omega}}_{AB}\,{}^{B}\mathbf{p}$:	acceleration due to non-constant spinning
$2\,{}^{B}\boldsymbol{\Omega}_{AB}\,{}^{B}\mathbf{v}_{BP}$:	Coriolis acceleration due to velocity of the point within frame B
${}^{B}\mathbf{a}_{BP}$:	acceleration of the point within frame B
${}^{A}\mathbf{a}_{AB}$:	acceleration of frame B wrt frame A

Example: Centripetal and Coriolis acceleration on to earth



Consider a setup consisting of a free-hanging mass-spring-damper system that is suspended from a pivot point. The mass is a point mass. The spring is a linear spring. Assume that the whole system, i.e. mass and spring together, is linearly moving wrt to the earth E with a constant velocity ${}^{E}\mathbf{v}_{EP}$. We only consider static forces: the system has reached its steady state and does not oscillate. Under these conditions, the force in the spring counterbalances all other forces, ${}^{E}\mathbf{F}_{spring} = m\,{}^{E}\mathbf{a}^{E}_{EP}$, and the mass is at complete rest relative to the pivot point. We want an expression for ${}^{E}\mathbf{a}^{E}_{EP}$.

The earth spins around its axis with a turning rate of $\omega = 2\pi/(24\cdot 3600) = 7.27\cdot 10^{-5}\,\mathrm{rad/s}$. Without loss of generality, assume that the earth fixed frame E is earth centered. Its z-axis, the pole-to-pole axis, is the rotation axis. The rotation is then with reference to a so-called inertial frame I. This frame and the earth frame share the same origin, and the same z-axis. The angular velocity is ${}^{I}\boldsymbol{\omega}_{IE} = {}^{E}\boldsymbol{\omega}_{IE} = \begin{bmatrix}0 & 0 & \omega\end{bmatrix}^{\mathrm{T}}$. Effects of the earth motion around the sun is assumed to be negligible.

If the earth would not rotate, the only force that needs to be counteracted by the spring is the gravitational force $m\,{}^{I}\mathbf{a}_{IP}$. This force is pointing to the center of the earth. ${}^{I}\mathbf{a}_{IP}$ is the gravitational acceleration, usually denoted by $\mathbf{g}$.

However, the earth does rotate, and eq (E13) applies with $A = I$ and $E = B$. Thus, ${}^{I}\mathbf{v}_{IP} = {}^{I}\Omega_{IE}{}^{E}\mathbf{p} + {}^{I}\mathbf{v}_{EP}$, where P is the point mass. Differentiation wrt time yields ${}^{I}\mathbf{a}_{IP} = {}^{I}\mathbf{a}_{EP} + {}^{I}\Omega_{IE}\left({}^{I}\Omega_{IE}{}^{E}\mathbf{p} + {}^{I}\mathbf{v}_{EP}\right)$. It follows:

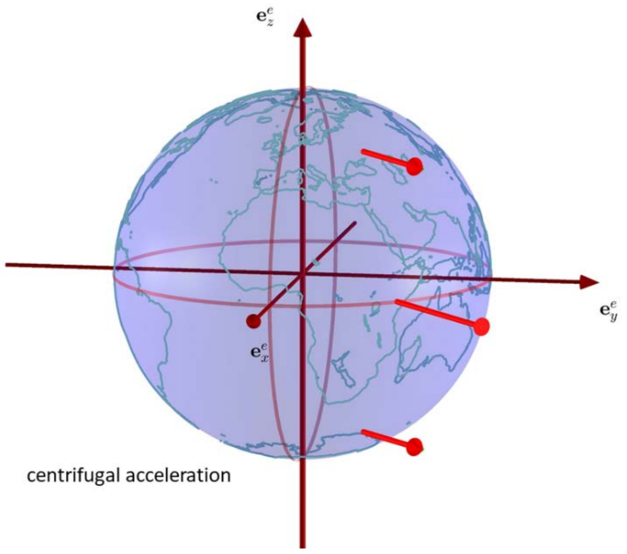
$$ {}^{I}\mathbf{a}_{EP} = {}^{I}\mathbf{a}_{IP} - {}^{I}\Omega_{IE}{}^{I}\Omega_{IE}{}^{E}\mathbf{p} - {}^{I}\Omega_{IE}{}^{I}\mathbf{v}_{EP} \tag{E22} $$

Thus, expressed in the inertial frame, there are two additional forces on the spring: $-m\,{}^{I}\Omega_{IE}{}^{I}\Omega_{IE}{}^{E}\mathbf{p}$, which is the centrifugal force, and $-m\,{}^{I}\Omega_{IE}{}^{I}\mathbf{v}_{EP}$, which is the Coriolis force.

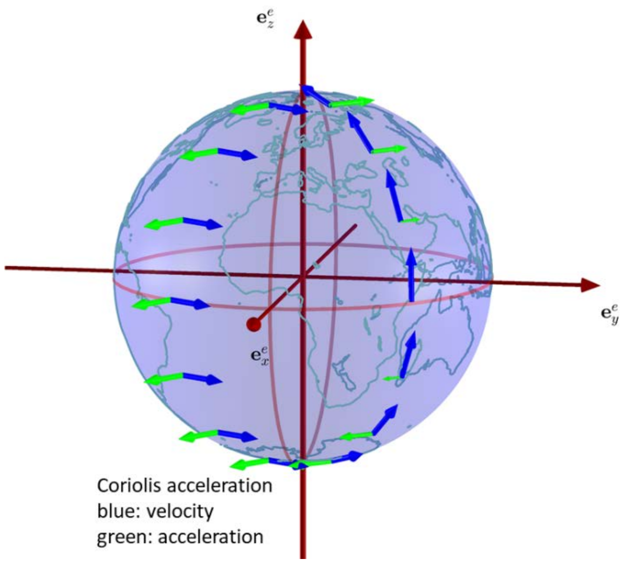
Since we are living on the earth's surface, it is more convenient to express these forces in the E frame. Using the relations ${}^{E}\mathbf{p} = {}^{E}\mathbf{R}_{I}{}^{I}\mathbf{p}$ and ${}^{E}\mathbf{v}_{EP} = {}^{E}\mathbf{R}_{I}{}^{I}\mathbf{v}_{EP}$, we have:

$$ \begin{aligned} {}^{E}\mathbf{a}_{EP} = {}^{E}\dot{\mathbf{v}}_{EP} &= {}^{E}\mathbf{R}_{I}\left({}^{I}\mathbf{a}_{EP} + {}^{I}\Omega_{EI}{}^{I}\mathbf{v}_{EP}\right) \\ &= {}^{E}\mathbf{a}_{IP} - {}^{E}\Omega_{IE}{}^{E}\Omega_{IE}{}^{E}\mathbf{p} - 2\,{}^{E}\Omega_{IE}{}^{E}\mathbf{v}_{EP} \end{aligned} \tag{E23} $$

This is the earth's relative acceleration of the point expressed in earth-fixed coordinates. The first term is the gravitational acceleration ${}^{E}\mathbf{a}_{IP} = {}^{E}\mathbf{R}_{I}{}^{I}\mathbf{a}_{EP}$. The second term $-{}^{E}\Omega_{IE}{}^{E}\Omega_{IE}{}^{E}\mathbf{p}$ is the centrifugal acceleration, which is pointing outward, orthogonal to the earth's rotation axis. This vector is experienced as a force, and is the cause of the gravity force not exactly pointing towards the center of the earth. See figure. The effect is that, on the equator, a person's weight is about 0.3% less than it would be on a pole.



The term $-2\,{}^{E}\Omega_{IE}{}^{E}\mathbf{v}_{EP}$ is the so-called Coriolis acceleration. It is proportional to the velocity of the point with respect to the earth. It is orthogonal to both the rotation axis and to the direction of the velocity. See Figure.



Whether the effect is negligible depends on the speed and the duration of the motion. Suppose, for instance, that a point at 53⁰ latitude, e.g. in the Netherlands, has a speed of $v = 1\,\mathrm{m/s}$, and the duration is $T = 10\,\mathrm{s}$, then the travelled distance would be $vT = 10\,\mathrm{m}$, and the deviation due to Coriolis acceleration would be around $5.9\,\mathrm{mm}$. This might be negligible. At a larger scale, the effect cannot be neglected. For instance, the atmosphere might move with a speed of $v = 40\,\mathrm{km/hr}$ during a period of $T = 5\,\mathrm{hr}$. The travelled distance is then $200\,\mathrm{km}$, while the Coriolis distance would be $211\,\mathrm{km}$.

E.4 Direction derivatives and integration

Consider the following problem: a body B is provided with a measurement device, that measures its angular velocity. This could be accomplished by means of a gyroscope, or maybe by means of cameras and optical flow calculations. The found angular velocity, ${}^B\boldsymbol{\omega}_{AB}(t) = \begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^{\mathrm{T}}$ performs a rotation of frame B wrt a reference frame A. It is expressed in its own coordinate system of frame B. The time dependency is emphasized by the notation ${}^B\boldsymbol{\omega}_{AB}(t)$. For brevity's sake, the shorter notation ${}^B\boldsymbol{\omega}_{AB}$ is used if this would not lead to confusion.

The problem that is addressed in this section is how can we integrate the angular velocity of the body so as to keep track of the orientation of the body B with reference to frame A. In other words, how does ${}^A\mathbf{R}_B(t)$ evolve in time, starting from a known reference orientation ${}^A\mathbf{R}_B(0)$ at $t=0$?

Suppose that a point $\mathbf{p}$ is fixed in frame B. Then its coordinates, at a given time t_k, are given by ${}^A\mathbf{p}(t_k) = {}^A\mathbf{R}_B(t_k)\,{}^B\mathbf{p}$. The rotation performed from t_k to time t_k+dt is described by a rotation matrix ${}^{t_k}\mathbf{R}_{t_k+dt}$. We then have ${}^A\mathbf{p}(t_k+dt) = {}^A\mathbf{R}_B(t_k+dt)\,{}^B\mathbf{p} = {}^A\mathbf{R}_B(t_k)\,{}^{t_k}\mathbf{R}_{t_k+dt}\,{}^B\mathbf{p}$, According to eq (E6), ${}^{t_k}\mathbf{R}_{t_k+dt} = \mathbf{I} + {}^B\boldsymbol{\Omega}_{AB}dt$, which holds true if dt approaches zero. Therefore:

$$ {}^A\mathbf{R}_B(t_k+dt) = {}^A\mathbf{R}_B(t_k)\left(\mathbf{I} + {}^B\boldsymbol{\Omega}_{AB}(t_k)dt\right) \qquad \text{(E24)}$$

In an algorithmic design, implemented in discrete-time with a time-index k and a sampling interval of Δ, i.e. $t_k = k\Delta$, the numerical, recursive approximation would be:

$$ {}^A\mathbf{R}_B(k) = {}^A\mathbf{R}_B(k-1)\left(\mathbf{I} + {}^B\boldsymbol{\Omega}_{AB}(k-1)\Delta\right) \qquad k=1,2,\cdots \qquad \text{(E25)}$$

The big disadvantage of eq (E25) is that it does not enforce the orthonormality constraint of a rotation matrix. This implementation is very sensitive to accumulation errors. Its use is not recommended, and alternatives, as presented below, are the preferred ones.

E.4.1 Using Rodriques' rotation formula

Rodriques' rotation formula converts an axis/angle representation of an orientation to a rotation matrix:

$$ {}^B\mathbf{R}_A = \mathbf{I} - \sin(\Phi)[\mathbf{e}]_\times + (1-\cos(\Phi))[\mathbf{e}]_\times^2 \qquad \text{see (D5)}$$

This formula can be used to get a closed-form expression for ${}^A\mathbf{R}_B(t)$, or its discrete-time equivalence ${}^A\mathbf{R}_B(k)$. For that purpose, define the vector and matrix:

$$ \boldsymbol{\upsilon}(t) = \int_{\tau=0}^{t} {}^B\boldsymbol{\omega}_{AB}(\tau)d\tau \qquad \boldsymbol{\Upsilon}(t) = \int_{\tau=0}^{t} {}^B\boldsymbol{\Omega}_{AB}(\tau)d\tau \qquad \text{(E26)}$$

Note that $\boldsymbol{\Upsilon} = [\boldsymbol{\upsilon}]_\times$. Application in Rodriques' formula yields:

$$ {}^A\mathbf{R}_B(t) = {}^A\mathbf{R}_B(0)\left(\mathbf{I} + \frac{\sin\|\boldsymbol{\upsilon}(t)\|}{\|\boldsymbol{\upsilon}(t)\|}\boldsymbol{\Upsilon}(t) + \frac{1-\cos\|\boldsymbol{\upsilon}(t)\|}{\|\boldsymbol{\upsilon}(t)\|^2}\boldsymbol{\Upsilon}^2(t)\right)$$

(E27)

A recursive implementation in discrete-time is as follows:

$$ \begin{aligned} &{}^A\mathbf{R}_B(k) = {}^A\mathbf{R}_B(k-1)\left(\mathbf{I} + \frac{\sin\Phi}{\Phi}\boldsymbol{\Upsilon} + \frac{1-\cos\Phi}{\Phi^2}\boldsymbol{\Upsilon}^2\right) \\ &\text{with:} \quad \Phi = \left\|{}^B\boldsymbol{\omega}_{AB}(k-1)\right\|\Delta \\ &\qquad\qquad \boldsymbol{\Upsilon} = {}^B\boldsymbol{\Omega}_{AB}(k-1)\Delta \end{aligned} \qquad \text{(E28)}$$

Here, it is assumed that the sampling period is small enough to apply the following approximation:

$$ \int_{\tau=(k-1)\Delta}^{k\Delta} {}^B\boldsymbol{\omega}_{AB}(\tau)d\tau \cong {}^B\boldsymbol{\omega}_{AB}(k-1)\Delta$$

Note that in eq (E28), if Φ is very, small, then $\sin(\Phi)/\Phi \cong 1$, and $1-\cos\Phi \cong -\tfrac{1}{2}\Phi^2$. Thus, in fact, the solution given in eq (E25) is a truncated Taylor series expansion of eq (E28).

E.4.2 Euler angle derivatives

Instead of a rotation matrix, the orientation of an object can also be captured by Euler angles. Suppose that these Euler angles in 'zyx' convention are bundled in a three tuple $\boldsymbol{\eta}(t) = \begin{bmatrix}\varphi & \theta & \psi\end{bmatrix}^{\mathrm{T}}$. At time $t_k+\Delta$, the body has rotated around the rotation axis ${}^B\boldsymbol{\omega}_{AB}(t_k)$ with an angle $\left\|{}^B\boldsymbol{\omega}_{AB}(t_k)\right\|\Delta$. This gives rise to changed Euler angles $\boldsymbol{\eta}(t_k+\Delta)$. If Δ approaches zero, then the truncated Taylor series expansion:

$$ \boldsymbol{\eta}(t_k+\Delta) = \boldsymbol{\eta}(t_k) + \Delta \tfrac{d}{dt}\boldsymbol{\eta}(t)\Big|_{t=t_k} \qquad \text{(E29)}$$

asymptotically holds true. The term $d\boldsymbol{\eta}/dt$ contains the time derivatives of the Euler angles: $\begin{bmatrix}d\varphi/dt & d\theta/dt & d\psi/dt\end{bmatrix}^{\mathrm{T}}$.

The dependency of $d\boldsymbol{\eta}/dt$ on ${}^B\boldsymbol{\omega}_{AB}$ is found in eq (D15) and (E8). Define a matrix $\mathbf{F}(\Delta)$

$$\mathbf{F}(\Delta) \overset{def}{=} {}^A\mathbf{R}_B(t_k+\Delta) = {}^A\mathbf{R}_B(t_k)\left(\mathbf{I} + {}^B\boldsymbol{\Omega}_{AB}\Delta\right) \qquad \text{(E30)}$$

According to (D14), with ${}^A\mathbf{R}_B = {}^B\mathbf{R}_A^{\mathrm{T}}$, we have, for instance:

$$\begin{aligned} f_{11}(\Delta) &= c_\varphi c_\theta - q\Delta\left(s_\varphi s_\psi + c_\varphi c_\psi s_\theta\right) - r\Delta\left(c_\psi s_\varphi - c_\varphi s_\psi s_\theta\right) \\ f_{21}(\Delta) &= s_\varphi c_\theta + q\Delta\left(c_\varphi s_\psi - c_\psi s_\varphi s_\theta\right) + r\Delta\left(c_\varphi c_\psi + s_\varphi s_\psi s_\theta\right) \end{aligned} \qquad \text{(E31)}$$

According to eq (D15):

$$\varphi(t+\Delta) = \mathrm{atan}\left(\frac{f_{21}}{f_{11}}\right) \qquad \text{(E32)}$$

Differentiation of $\varphi(t+\Delta)$ with respect to Δ, and evaluating the result at $\Delta = 0$, provides the answer. That is:

$$\frac{d}{dt}\varphi(t) = \frac{d}{d\Delta}\varphi(t+\Delta)\bigg|_{\Delta=0}$$

Repeating this procedure for all Euler angles, and after some tedious mathematical manipulations, we arrive at:

$$\begin{bmatrix} \frac{d\varphi}{dt} \\ \frac{d\theta}{dt} \\ \frac{d\psi}{dt} \end{bmatrix} = \begin{bmatrix} 0 & \frac{\sin\psi}{\cos\theta} & \frac{\cos\psi}{\cos\theta} \\ 0 & \cos\psi & -\sin\psi \\ 1 & \tan(\theta)\sin(\psi) & \tan(\theta)\cos(\psi) \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \qquad \text{(E33)}$$

In a compact notation, this is written as:

$$\dot{\boldsymbol{\eta}}(t) = \mathbf{J}\left(\boldsymbol{\eta}(t)\right) {}^B\boldsymbol{\omega}_{AB}(t) \qquad \text{(E34)}$$

The discrete-time implementation to keep track of the Euler angles, given measured body-fixed angular velocity, is:

$$\boldsymbol{\eta}(k) = \boldsymbol{\eta}(k-1) + \mathbf{J}\left(\boldsymbol{\eta}(k-1)\right) {}^B\boldsymbol{\omega}_{AB}(k-1)\Delta \qquad \text{(E35)}$$

This works well in applications in which the pitch angle θ is far away from its singular points, i.e. $\theta = \pm\frac{1}{2}\pi$. At these points, the matrix $\mathbf{J}$ becomes instable.

E.4.3 Quaternion derivatives

Another possibility to represent orientation is a quaternion $\mathbf{q}$. See Section D.4. If the orientation is time dependent, because of a rotation, described by the angular velocity ${}^B\boldsymbol{\omega}_{AB}(t) = \begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^{\mathrm{T}}$, we have to consider the quaternion at to different points in time, i.e. $\mathbf{q}(t)$ and $\mathbf{q}(t+dt)$ to say how the rotation affects the quaternion. Using eq (D20), we can transform $\mathbf{q}(t)$ into a rotation matrix ${}^A\mathbf{R}_B(t)$. Next, with eq (E8), we can calculate how the rotation effects the rotation matrix going from ${}^A\mathbf{R}_B(t)$ to ${}^A\mathbf{R}_B(t+dt)$. Then, application of eq (D21) provides the quaternion $\mathbf{q}(t+dt)$.

The time-derivative $\dot{\mathbf{q}}$ follows from:

$$\dot{\mathbf{q}} = \lim_{dt\to 0}\frac{\mathbf{q}(t+dt) - \mathbf{q}(t)}{dt}$$

After some mathematical manipulations, and making use of the property that $\|\mathbf{q}\| = 1$, the following solution is achieved:

$$\dot{\mathbf{q}} = \mathbf{W}\mathbf{q} \quad \text{with} \quad \mathbf{W} = \frac{1}{2}\begin{bmatrix} 0 & \omega_x & \omega_y & \omega_z \\ -\omega_x & 0 & \omega_z & -\omega_y \\ -\omega_y & -\omega_z & 0 & \omega_x \\ -\omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \qquad \text{(E36)}$$

where the matrix $\mathbf{W}$ has been introduced to allow the shorthand notation of $\dot{\mathbf{q}} = \mathbf{W}\mathbf{q}$.

To get a recursive discrete-time implementation, a direct usage of eq (E36) would be $\mathbf{q}(k) = \mathbf{q}(k-1) + \mathbf{W}\mathbf{q}(k-1)\Delta$. However, this violates the condition that $\|\mathbf{q}(k)\| = 1$. Therefore, care must be taken when using eq (E36). A possibility would be to renormalize $\mathbf{q}(k)$ at each time step. A more elegant solution is to solve the differential equation $\dot{\mathbf{q}} = \mathbf{W}\mathbf{q}$ by making use of the following theorem, which can be proven simply by differentiating the expression for $x(t)$:

$$\begin{aligned} \text{if:} \quad & \dot{x}(t) = a(t)x(t) \\ \text{then:} \quad & x(t) = x(0)\exp\left(\int_{\tau=0}^{t} a(\tau)d\tau\right) \end{aligned}$$

Application of this theorem to the vector-matrix equation $\dot{\mathbf{q}} = \mathbf{W}\mathbf{q}$ provides the following solution:

$$\mathbf{q}(t_k) = \exp\left(\int_{\tau=t_{k-1}}^{t_k} \mathbf{W}(\tau)d\tau\right)\mathbf{q}(t_{k-1}) \qquad \text{(E37)}$$

where it is understood that the exponent function is applied as a matrix function[4].

[4] In matlab the function **expm** should be used; not the function **exp**.

Assuming that Δ is small, this equation simplifies to

$$\mathbf{q}(t_k) = \exp\left(\mathbf{W}\Delta\right)\mathbf{q}(t_{k-1}) \quad \text{with} \quad \mathbf{W} \text{ evaluated at } t_{k-1} \qquad \text{(E38)}$$

A further simplification is obtained by making use of the Taylor expansion of $\exp\left(\mathbf{W}\Delta\right)$ and by the property that $\mathbf{W}^2 = -\frac{1}{2}\|\boldsymbol{\omega}\|^2\,\mathbf{I}$. This leads to the following recursive implementation in discrete-time:

$$\mathbf{q}(k) = \left[\cos\left(\tfrac{1}{2}\|\boldsymbol{\omega}\|\Delta\right)\mathbf{I} + \frac{1}{\frac{1}{2}\|\boldsymbol{\omega}\|\Delta}\sin\left(\tfrac{1}{2}\|\boldsymbol{\omega}\|\Delta\right)\mathbf{W}\Delta\right]\mathbf{q}(k-1) \quad \text{(E39)}$$

where $\boldsymbol{\omega} = {}^{B}\boldsymbol{\omega}_{AB}(k-1)$, and $\mathbf{W}$ is evaluated from ${}^{B}\boldsymbol{\omega}_{AB}(k-1)$ according to eq (E36).

E.4.4 Application: state space model for Kalman filtering

In this section, an example will be given that describes the kinematics of a motion of a 3D ridged device, such as a handheld camera or endoscope, etc. This model is needed in state estimators, like Kalman filtering with applications like navigation and visual SLAM.

We consider a free moving object such as a handheld camera. It can move in all directions, and it can be rotated in all directions. This is in contrast with motions with holonomic constraints. These are motions with limited degrees of freedom (DoF). A typical example would be laparoscopy in which the motion is limited by the trocar, i.e. a portal for the placement of laparoscope.

The pose of an object has six DoF. Three of them are needed to capture the position. The other three are for its orientation. For orientation, we have different representations: rotation matrices, Euler angles, and quaternions. The latter embeds the axis/angle representation. Rotation matrices are highly redundant. Euler angles are not redundant, but suffer from instability and singularity at the gimbal points. Therefore, we choose quaternions, which are made up by four variables.

If the model is used in Kalman filtering, or other types of state estimation, the model should be predictive. That is, knowing the state at t_k, indexed by the discrete-time k, it should be able to predict the state at time t_{k+1}, indexed by $k+1$. This is a one-step-ahead prediction. The model should not only be able to predict the state, it must also be able to quantify the uncertainty of this prediction.

To be able to predict, advantage should be taken of the smoothness of the motion. This implies that the source of the prediction errors should be modelled in the derivatives of the pose, e.g. the linear and angular velocities, and not in the pose itself. The integration of the velocities, which is needed to get position and orientation, smooths the error source, and enables the prediction.

So, the simplest model that complies with these considerations, is a model that includes, in addition to pose, also the velocities. The pose is expressed in a reference frame A, but the velocities are device-fixed, and thus expressed in the frame B attached to the device. We thus arrive at the following state vector that is valid at discrete-time k:

$$\mathbf{x}(k) = \begin{bmatrix} {}^{A}\mathbf{p}(k) \\ {}^{A}\mathbf{q}_B(k) \\ {}^{B}\mathbf{v}_{AB}(k) \\ {}^{B}\boldsymbol{\omega}_{AB}(k) \end{bmatrix} \qquad \text{(E40)}$$

This is a 13D vector with 12 DoF.

The objective is to predict the state $\mathbf{x}(k+1)$. For that purpose, a system function is needed that takes the current state $\mathbf{x}(k)$ as input, and produces an output $\mathbf{x}(k+1)$. However, the system should also take into account the uncertainty that enters the system during time t_k and t_{k+1}. The system equation is represented by:

$$\mathbf{x}(k+1) = \mathbf{f}\left(\mathbf{x}(k)\right) + \mathbf{w}(k) \qquad \text{(E41)}$$

Here, $\mathbf{f}(.)$ is the system function. The term $\mathbf{w}(k)$ models the uncertainty that is injected in the system. It accounts for unknown and unpredictable movements of the device.

State space model for the position:

Since the state vector consists of 4 parts, the system function also consists of 4 parts. When the sampling interval Δ is small, the new position is quite simple to model:

$${}^{A}\mathbf{p}(k+1) = {}^{A}\mathbf{p}(k) + {}^{A}\mathbf{R}_B\,{}^{B}\mathbf{v}_{AB}(k)\Delta \qquad \text{(E42)}$$

where ${}^{A}\mathbf{R}_B$ is the rotation matrix associated with the quaternion ${}^{A}\mathbf{q}(k)$. See eq (D20).

State space model for the linear velocity:

For the linear velocity ${}^{B}\mathbf{v}_{AB}(k)$, different models can be applied. A possibility is to assume that the acceleration that takes place during time t_{k-1} and t_k is statistically independent from accelerations between t_k and t_{k+1}, and that these accelerations are not predictable. This is so called white-noise acceleration. The associated model is ${}^B\mathbf{v}_{AB}(k+1) = {}^B\mathbf{v}_{AB}(k) + \mathbf{w}_1(k)$, where $\mathbf{w}_1(k)$ represents white noise which is due to the unpredictable accelerations. In the final model of eq (E41), this term will be embedded in $\mathbf{w}(k)$.

If the sampling interval is very large so that the velocity at time k doesn't carry much information about the velocity at time $k+1$, the model becomes ${}^B\mathbf{v}_{AB}(k+1) = \mathbf{w}_1(k)$. This is white-noise velocity model.

A balance between the two models is found by:

$$\begin{aligned} &{}^B\mathbf{v}_{AB}(k+1) = \mathbf{\Gamma}\,{}^B\mathbf{v}_{AB}(k) + \mathbf{w}_1(k) \\ &\text{with} \quad \mathbf{\Gamma} = \text{diag}([\gamma_x \quad \gamma_y \quad \gamma_z]) \end{aligned} \qquad \text{(E43)}$$

$\mathbf{\Gamma}$ is a diagonal matrix with diagonal elements in the range from 0 to 1. With these parameters, one can tweak the model.

State space model for the orientation:

The state space model for the orientation is provided by eq (E38):

$${}^A\mathbf{q}(k+1) = \exp\left(\mathbf{W}(k)\Delta\right)\,{}^A\mathbf{q}(k) \qquad \text{(E44)}$$

in which the matrix $\mathbf{W}$, defined in eq (E36), is evaluated with the angular velocities ${}^B\boldsymbol{\omega}_{AB}(k)$. Note that eq (E39) can be used as an alternative, as mathematically it is equivalent with eq (E38).

State space model for the angular velocity:

For the angular velocity, the same considerations as for the linear velocity holds true. Therefore, a simple state space model is:

$$\begin{aligned} &{}^B\boldsymbol{\omega}_{AB}(k+1) = \mathbf{\Lambda}\,{}^B\boldsymbol{\omega}_{AB}(k) + \mathbf{w}_2(k) \\ &\text{with} \quad \mathbf{\Lambda} = \text{diag}([\lambda_x \quad \lambda_y \quad \lambda_z]) \end{aligned} \qquad \text{(E45)}$$

The system equation:

Pooling the four parts together, we arrive at the following system function:

$$\mathbf{f}\left(\mathbf{x}(k)\right) = \begin{bmatrix} \mathbf{I} & \mathbf{0} & {}^A\mathbf{R}_B\Delta & \mathbf{0} \\ \mathbf{0} & \exp\left(\mathbf{W}\Delta\right) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Gamma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda} \end{bmatrix} \begin{bmatrix} {}^A\mathbf{p}(k) \\ {}^A\mathbf{q}(k) \\ {}^B\mathbf{v}_{AB}(k) \\ {}^B\boldsymbol{\omega}_{AB}(k) \end{bmatrix} \qquad \text{(E46)}$$

The rotation matrix ${}^A\mathbf{R}_B$ is time variant and depends on the quaternion ${}^A\mathbf{q}(k)$. The matrix $\mathbf{W}$ is also time variant and depends on ${}^B\boldsymbol{\omega}_{AB}(k)$. The diagonal matrices $\mathbf{\Gamma}$ and $\mathbf{\Lambda}$ are constant, and can be used to choose between smooth, inertial motions and more shaky motions.

In eq (E41), the term $\mathbf{w}(k)$ is the process noise. This term adds randomness to the system, so that the state space model becomes stochastic rather than deterministic. It takes the following form:

$$\mathbf{w}(k) = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{w}_1(k) \\ \mathbf{w}_2(k) \end{bmatrix} \qquad \text{(E47)}$$

For Kalman filtering, the assumption is made that the process noise is a Gaussian, zero mean, white noise sequence with a 13×13 covariance matrix $\mathbf{C}_\mathrm{w}$, which has the following structure:

$$\mathbf{C}_\mathrm{w} = \left[\begin{array}{c|cc} \mathbf{0}_{7\times7} & \multicolumn{2}{c}{\mathbf{0}_{7\times6}} \\ \hline \mathbf{0}_{6\times7} & \mathbf{C}_1 & \mathbf{0}_{3\times3} \\ & \mathbf{0}_{3\times3} & \mathbf{C}_2 \end{array}\right] \qquad \text{(E48)}$$

The 3×3 diagonal matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ quantify the randomness of the linear and angular velocity vectors. Their diagonal elements, $\sigma^2_{1,x}, \sigma^2_{1,y}, \sigma^2_{1,z}$ and $\sigma^2_{2,x}, \sigma^2_{2,y}, \sigma^2_{2,z}$ are the variances of the process noise of the x, y and z components of the velocities.

The covariance matrix offers the possibility to adjust the motion somewhat to certain directions. For instance, if the device can only move in forward or backward direction, i.e. surge movement, then $\sigma^2_{1,y}$ and $\sigma^2_{1,z}$ should be very small relative to $\sigma^2_{1,x}$. Another example, if the rotation of the tip of a flexible endoscope is only 'roll', and 'yaw', then $\sigma^2_{2,y}$ (pitch) should be very small compared with $\sigma^2_{2,x}$ (roll), and $\sigma^2_{2,z}$ (yaw).

F. References and further reading:

[1] H. Guo: *Modern Mathematics and Applications in Computer Graphics and Vision*, World Scientific Publishing Company, 2014.
[2] A.N. Kolmogorov, S.V. Fomin: *Introductory Real Analysis*, Dover Publications, New York, 1970.
[3] R.E. Bellman: *Introduction to Matrix Analysis*, McGraw-Hill, New York, 1970.
[4] G. Strang: *Linear Algebra and its Applications*, Harcourt Brace Jovanovich, San Diego, 3rd edition, 1989.
[5] J.A. Farrell, M. Barth: *The Global Positioning System & Inertial Navigation*, McGraw-Hill, New York, 1998.
[6] T.I. Fossen: *Guidance and Control of Ocean Vehicles*, John Wiley and Sons, Chichester, 1994.
[7] J. Civera, A.J. Davison, J.M. Martínez Montiel: *Structure from Motion Using the Extended Kalman Filter*, Springer-Verlag, Berlin, 2012.