Technical Concepts

Orientation, Rotation, Velocity and Acceleration, and the SRM

Version 2.0

20 June 2008

Author:

Paul Berner, PhD

Contributors:

Ralph Toms, PhD, Kevin Trott, Farid Mamaghani, David Shen, Craig Rollins, Edward Powell, PhD

Table of contents

Acknowledgements

This document was developed by the SEDRIS Organization as part of the effort to include a more comprehensive treatment of orientation in the SRM implementation, and in support of the requirements of the Test & Training Enabling Architecture (TENA) project.

The helpful suggestions and feedback from TENA developers, under the auspices of Dr. Ed Powell, were invaluable in developing the scope and in refining the use cases described in this document. Furthermore, the software implementations of the concepts in this document and the refinement of the corresponding application program interface (API) were greatly benefited by the feedback from the TENA development team. In particular, the many contributions of Mr. Terry Burks (Trideum Corp) during the development of the interface and the testing of the implementations of the API were critical in the completion of this effort.

The participation of Mr. Craig Rollins (National Geospatial-Intelligence Agency) in this effort has been essential and invaluable. In addition to his insightful feedback and critical reviews of this document, he expended significant effort in the development of independent test data used to verify the correctness of the implementations.

The software design and implementations of the algorithms, along with the development of a complete suite of testing and verification of the implementations, were developed and produced by Mr. David Shen (SAIC). In the course of this effort, he identified various errors in the formulations and made numerous critical suggestions and contributions.

The practical use of the API is further described in a separate and comprehensive document, produced by Mr. Kevin Trott (Northrop Grumann), entitled "User's Manual for SRM Orientation, Velocity, & Acceleration Transformations".

Paul Berner
Ralph Toms
Kevin Trott
Farid Mamaghani

1 Introduction

One of the characteristics of the SRM[1] (ISO/IEC 18026:2006(E)) that distinguishes it from many other treatments of spatial referencing is the definition of the concept of direction in linear and curvilinear 3D spatial reference frames and the explicit methodology to convert direction representations from one spatial reference frame to another spatial reference frame. Intrinsic to that methodology is the use of orientation operations. Orientation and rotation operators are also important in operating on the vector representation of physical phenomena. These types of operations are important for a significant sector of the intended user domain of the SRM. With the intent to leverage the SRM treatment of the direction concept, this document explores the orientation/rotation operator subject matter domain. In presenting these concepts in a consistent and well defined manner, a framework is laid out to allow the future expansion of the SRM API to explicitly deal with the orientation concept. To this end, this document reviews the rotation/orientation concept in relation to the SRM. In particular, various representations of orientation and rotation and the methods of converting between them are presented.

Many concepts discussed here have been in wide use from the time of Euler's work on the subject. As a result, there are many similar but different treatments in the literature. In particular, there are similar terms with different meanings and, in some cases, the differences are subtle. There are also many differences in notational conventions. For this reason an attempt has been made to provide self contained derivations (assuming the prerequisites) of most of the formulations and algorithms presented here. By following the derivations there should be no mistake as to the intended meanings of the results. To improve the flow of the text, parts of lengthier derivations have been relegated to appendices. The formulation of these concepts as presented here may be incorporated in a future version of the SRM.

1.1 Prerequisites

This document assumes that reader is familiar with the following prerequisite subject matter:

- Linear algebra
    - Vector spaces concepts including:
        - linear operators,
        - vector dot and cross products
    - Matrix algebra
- Calculus, and
- Elementary Physics
    - Rigid body kinematics and dynamics.

[1] See references [1].

See also reference [1] Annex A – Mathematical foundations.

1.2 Notation

The coordinate representation of a three dimensional (3D) vector $\boldsymbol{u}$ with respect to a basis is a column vector $\boldsymbol{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$. To compactly denote a coordinate in a line of text, the transpose is used $\boldsymbol{u} = (u_1, u_2, u_3)^{\mathsf{T}}$.

In this section, let $\boldsymbol{u} = (u_1, u_2, u_3)^{\mathsf{T}}$ and $\boldsymbol{v} = (v_1, v_2, v_3)^{\mathsf{T}}$ be two 3D vectors.

The *Inner product* or *dot product* or *scalar product* of 3D vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is denoted and defined as:

$$\boldsymbol{u} \bullet \boldsymbol{v} = \boldsymbol{u}^{\mathsf{T}} \boldsymbol{v} = u_1 v_1 + u_2 v_2 + u_3 v_3 \tag{0.1}$$

The *norm* or *length of a vector* $\boldsymbol{u}$ is defined as:

$$\|\boldsymbol{u}\| = \sqrt{\boldsymbol{u} \bullet \boldsymbol{u}} \tag{0.2}$$

If θ is the angle between two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ then:

$$\boldsymbol{u} \bullet \boldsymbol{u} = \cos(\theta) \|\boldsymbol{u}\| \|\boldsymbol{v}\| \tag{0.3}$$

The *outer product* of 3D vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is denoted $\boldsymbol{u} \otimes \boldsymbol{v}$ and defined as:

$$\boldsymbol{u} \otimes \boldsymbol{v} = \boldsymbol{u}\boldsymbol{v}^{\mathsf{T}} = \begin{pmatrix} u_1 v_1 & u_1 v_2 & u_1 v_3 \\ u_2 v_1 & u_2 v_2 & u_2 v_3 \\ u_3 v_1 & u_3 v_2 & u_3 v_3 \end{pmatrix} \tag{0.4}$$

Note that:

$$\mathrm{Trace}(\boldsymbol{u} \otimes \boldsymbol{v}) = u_1 v_1 + u_2 v_2 + u_3 v_3 = \boldsymbol{u} \bullet \boldsymbol{v} \tag{0.5}$$

The *vector product* or *cross product* of 3D vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined as:

$$\boldsymbol{u} \times \boldsymbol{v} = (u_2 v_3 - u_3 v_2, u_3 v_1 - u_1 v_3, u_1 v_2 - u_2 v_1)^{\mathsf{T}} = \boldsymbol{S_u} \boldsymbol{v} \tag{0.6}$$

where:

$$\boldsymbol{S_u} = \begin{pmatrix} 0 & -u_3 & +u_2 \\ +u_3 & 0 & -u_1 \\ -u_2 & +u_1 & 0 \end{pmatrix}$$

is the *skew-symmetric matrix associated with a vector* $\boldsymbol{u}$.

See Appendix A for some useful properties of the cross product.

The 3D *identity matrix* is denoted as:

$$\boldsymbol{I}_{3\times3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{0.7}$$

The 3D *zero vector* is noted as:

$$\mathbf{0} = (0,0,0)^{\mathrm{T}} \tag{0.8}$$

The two argument form of arctangent, $\operatorname{arctan2}(y,x)$, returns a value adjusted for the quadrant of the point (x,y). Given real numbers x and y,

$\operatorname{arctan2}(y,x) = \theta$

where: θ is the unique value satisying $-\pi < \theta \le \pi$, and

if $r = 0$,

$\theta = 0$, else

if $r > 0$,

$x = r\cos\theta$ and $y = r\sin\theta$.

where:

$r = \sqrt{x^2 + y^2}$.

2 Vectors, directions, axes and their uses

2.1 Vector space directions

A direction in a Euclidean vector space may be represented as a *unit vector*. That is, a vector $\boldsymbol{n}$ of length 1. Any non-zero $\boldsymbol{u}$ vector may be *normalized* to a unit vector $\boldsymbol{n}$ by dividing by the norm of the vector:

$$\boldsymbol{n} = \frac{1}{\|\boldsymbol{u}\|}\boldsymbol{u}, \quad \text{if } \boldsymbol{u} \ne (0,0,0)^{\mathrm{T}}.$$

Any positive multiple of a unit vector points in the same direction. By requiring unit vectors, each direction has a unique vector representation.

Directions have many application specific uses. For example, a velocity is a direction multiplied by a speed. Force and momentum acting on the center of mass of a body may be similarly represented.

A direction can be used to specify the axis of a rotating body. The axis of a rotating body lies on a line. By specifying the line as a direction, the right hand rule can be used to unambiguously identify which of the two axial rotational directions is acting on the body. Torque and angular momentum acting on a body may be similarly represented.

2.2 Vector directions in the SRM

In the Spatial Reference Model (SRM), the underlying vector space that is associated with a 3D Spatial Reference Frame (SRF) is determined by the Object Reference Model (ORM) of the SRF. For example, the underlying 3D vector space of any 3D SRF based on ORM WGS84 corresponds to the WGS84 geocentric SRF. This associated 3D Euclidean space is called the *object-space* of the ORM.

An SRF associates unique coordinates in a domain of the *coordinate-space* (of coordinate-component-tuples) to corresponding points in object-space. In the special case of a geocentric SRF, the object-space and coordinate-space are indistinguishable[2]. In general, an SRF is either linear or curvilinear. In the linear cases, the vector-space structure of coordinate-space carries over to object-space. In particular, lines through points in a given direction $\boldsymbol{n}$ are all parallel in both coordinate- and object-space. This shows that a direction is translation invariant in a linear SRF. A linear SRF will not preserve angular relationships between directions unless the associated abstract coordinate system (CS) is also orthonormal. In the orthonormal case, angles and distances are preserved.

In the case of a curvilinear SRF, the vector-space structure of the coordinate-space does not carry over[3]. The coordinate-space of an augmented map projection SRF (a map projection augmented with ellipsoidal height as a third dimension) appears to inherit the vector-space structure of $\mathbf{R}^3$, however, the vector properties of the (easting, northing, height)-coordinates do not carry over to object-space. This is illustrated in part by the “up pointing” vector $\boldsymbol{n} = (0, 0, 1)$ that points in different spatial directions (in object-space) depending on the map coordinate location from which $\boldsymbol{n}$ is viewed.

In Figure 1, distinct position points $\boldsymbol{p}$ and $\boldsymbol{q}$ on the ellipsoid surface are projected to augmented map coordinates $(s, t, 0)$ and $(u, v, 0)$. Starting at these map coordinates, the coordinates one unit away in direction $\boldsymbol{n}$ are $(s, t, 1)$ and $(u, v, 1)$ respectively. In an augmented map projection, these coordinates correspond to the position-space points $\boldsymbol{p}'$ and $\boldsymbol{q}'$. The direction from $\boldsymbol{p}$ to $\boldsymbol{p}'$ is not the same as the direction from $\boldsymbol{q}$ to $\boldsymbol{q}'$. This shows that the "up direction" is relative to an observation or reference point.

For each reference point, the SRM defines a uniform method for associating a unique orthonormal linear SRF to each reference point coordinate. This associated linear SRF will be used to specify a direction as "seen" from the reference point. This SRF is called the *local tangent frame* at the reference point. This SRF is defined by as having its origin at the reference point and axis directions given by the normalized tangent vectors to the coordinate curves passing through the reference point as illustrated in Figure 2.

[2] This assumes a common unit of length. The SRM requires the metre as the common unit of length.

[3] In the curvilinear case, even the coordinate domain is not the entire space of *n*-tuples.

All curvilinear SRFs in the SRM are orthogonal so that the local tangent frame will be an orthonormal linear SRF.

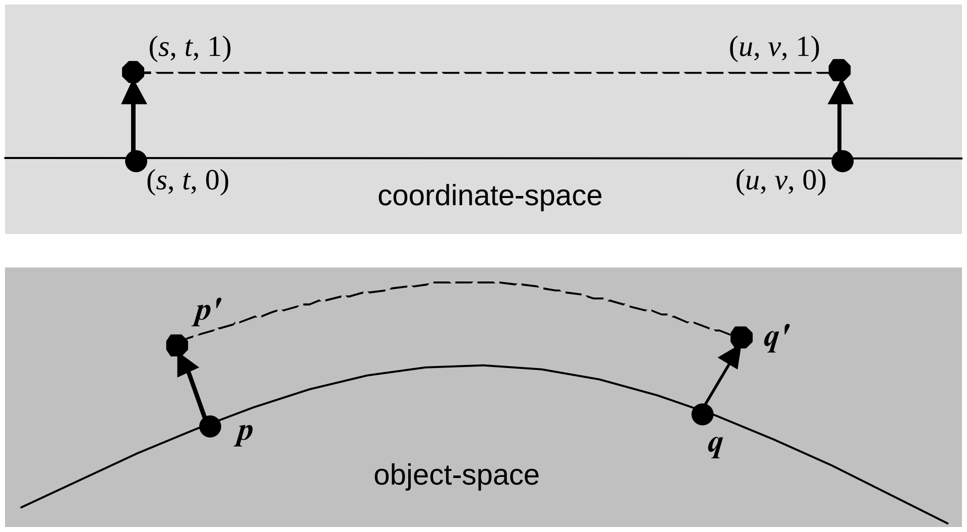


Figure 1 – Directions in an augmented map projection SRF

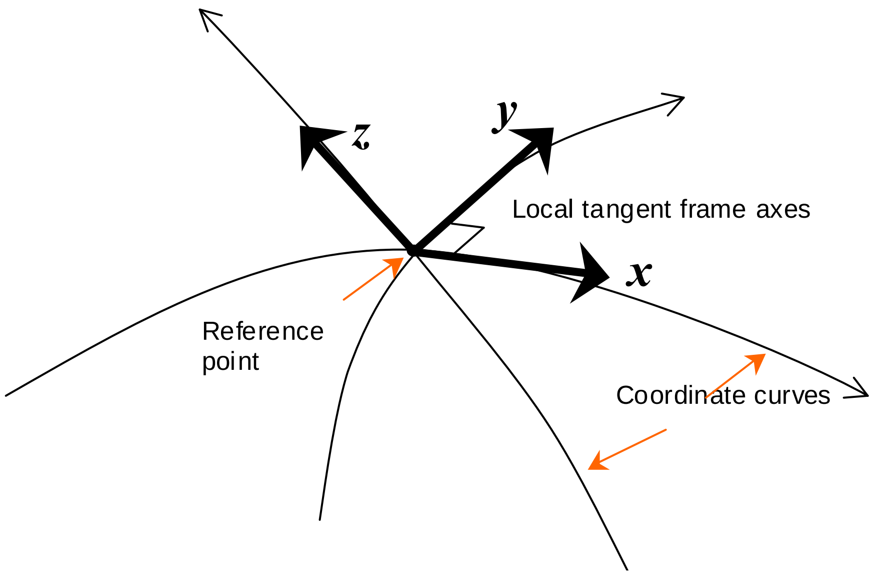


Figure 2 – Local tangent frame axes

Continuing the augmented map projection example, Figure 3 shows the local tangent frames axes (***x*** and ***z***-axes) at points ***p*** and ***q***. The local "up" directions may be specified in either local tangent frame. Since directions are translation invariant in linear SRFs, we may conceptually translate the two local tangent frames to a common origin as in Figure 4.

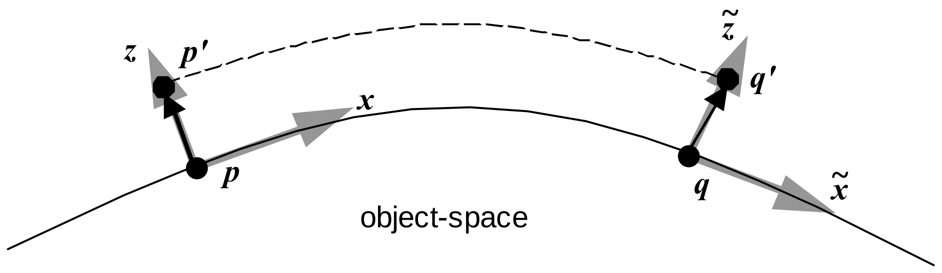


Figure 3 – Local tangent frame axes at p and q

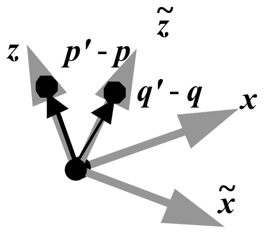


Figure 4 – Direction vectors two local tangent frames

In the SRM, a Direction data type consists of the coordinate of a reference point in a given SRF and a 3-tuple unit vector in the local tangent frame at the reference point. Since there is neither an intrinsic SRF nor an intrinsic reference point in object-space, it is necessary to specify the reference point in order to be able to inter-convert between SRF representations of a given direction. The SRM approach of associating reference points and local tangent frames thus reduces the general problem of inter-converting the representation of a direction between two SRFs to that of inter-converting between two orthonormal linear spaces. This methodology generalizes to the problem of inter-converting any vector quantity[4] between a pair of linear spaces. The treatment given here of this general problem begins with the notion of orientation.

3 Orientation

Consider two orthonormal bases for 3 dimensional Euclidean space $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ and $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$. An *orientation* is an expression of the axis directions of one basis with respect to the other. To illustrate this notion, consider an aircraft at time t_0 aligned with one basis: the center of mass of the airplane is at the vector space origin, the fuselage points in direction $\boldsymbol{x}$, the starboard wing points in direction $\boldsymbol{y}$, and (to complete a right handed system) $\boldsymbol{z}$ points down with respect to the aircraft (see Figure 7 below). At some later time t_1 the airplane is subjected to a roll, pitch, and/or yaw. We subtract the vector that

[4] Not necessarily a direction or a unit vector, but any vector of interest.

represents the displacement of the center of mass from time t_0 to time t_1 and the new directions for the fuselage, starboard wing, and relative down define the $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ directions. The two vector spaces spanned by bases $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ and $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ share the same origin and are thus two bases for the same vector space. The only difference is that $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ has a different orientation with respect to $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$. Orientation is also called *attitude* in some contexts.

3.1 Orientation and rotation

Let $\boldsymbol{r}$ be a point in 3 dimensional Euclidean space. Let $\boldsymbol{E}$ denote that vector space with orthonormal basis $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, and let $\tilde{\boldsymbol{E}}$ denote that vector space with orthonormal basis $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$. The coordinate representation of $\boldsymbol{r}$ with respect to each basis[5] is:

$$\boldsymbol{r} = \left(r_1, r_2, r_3\right)^{\mathrm{T}}, \text{ where } \boldsymbol{r} = r_1\boldsymbol{x} + r_2\boldsymbol{y} + r_3\boldsymbol{z}, \text{ and}$$

$$\boldsymbol{r} = \left(\tilde{r}_1, \tilde{r}_2, \tilde{r}_3\right)^{\mathrm{T}}, \text{ where } \boldsymbol{r} = \tilde{r}_1\tilde{\boldsymbol{x}} + \tilde{r}_2\tilde{\boldsymbol{y}} + \tilde{r}_3\tilde{\boldsymbol{z}}.$$

This coordinate transformation from $\boldsymbol{E}$ to $\tilde{\boldsymbol{E}}$ is denoted $\boldsymbol{\Omega}_{E\to\tilde{E}} : \left(r_1, r_2, r_3\right) \mapsto \left(\tilde{r}_1, \tilde{r}_2, \tilde{r}_3\right)$. This is a linear transformation and can thus be realized as a matrix multiplication:

$$\begin{pmatrix} \tilde{r}_1 \\ \tilde{r}_2 \\ \tilde{r}_3 \end{pmatrix} = \boldsymbol{\Omega}_{E\to\tilde{E}} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}$$

where:

$$\begin{aligned} a_{11} &= \boldsymbol{x} \bullet \tilde{\boldsymbol{x}}, \quad a_{12} = \boldsymbol{y} \bullet \tilde{\boldsymbol{x}}, \quad a_{13} = \boldsymbol{z} \bullet \tilde{\boldsymbol{x}} \\ a_{21} &= \boldsymbol{x} \bullet \tilde{\boldsymbol{y}}, \quad a_{22} = \boldsymbol{y} \bullet \tilde{\boldsymbol{y}}, \quad a_{23} = \boldsymbol{z} \bullet \tilde{\boldsymbol{y}} \\ a_{31} &= \boldsymbol{x} \bullet \tilde{\boldsymbol{z}}, \quad a_{32} = \boldsymbol{y} \bullet \tilde{\boldsymbol{z}}, \quad a_{33} = \boldsymbol{z} \bullet \tilde{\boldsymbol{z}} \end{aligned} \tag{1.1}$$

Since the basis vectors are unit vectors, each dot product in equation (1.1) is the cosine of the angle between the two vectors (see Equation (0.3)). For this reason this matrix $\left[a_{ij}\right]$ is the called *direction cosine matrix*. Note that the columns of the matrix are the $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ basis vectors in $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ coordinate representation while the rows (or columns of the transpose matrix) are the $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{z}}$ basis vectors in $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ coordinate representation.

---

[5] For any orthonormal basis, $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, the basis coefficients may be computed as:
$r_1 = \boldsymbol{r} \bullet \boldsymbol{x}, \;\; r_2 = \boldsymbol{r} \bullet \boldsymbol{y}, \;\; r_3 = \boldsymbol{r} \bullet \boldsymbol{z}$.

Euler's rotation theorem states that this linear transformation is a rotation operation. In particular, the matrix has a unit eigenvector $\boldsymbol{n}$ and three eigenvalues: $1, e^{+i\theta}, e^{-i\theta}$. The line spanned by the vector $\boldsymbol{n}$ is fixed under the transformation and represents the axis of rotation. The angle of rotation is given by θ. Let $\boldsymbol{R_n}(\theta)$ denote the rotation about vector $\boldsymbol{n}$ through angle θ.

Euler's rotation theorem thus shows that orientation and rotation are just two ways of viewing the same transformation. These two ways are closely related, but are not equivalent. Consider Figure 5. On the left side, the point $\boldsymbol{r}$ is rotated by angle θ about the z-axis (which points directly toward the reader) to a new position $\boldsymbol{r}'$.

The coordinates of these two points, $\boldsymbol{r} = (r_1, r_2, r_3)^{\mathrm{T}}$, and $\boldsymbol{r}' = (r_1', r_2', r_3')^{\mathrm{T}}$ are related by the following matrix.

$$\begin{pmatrix} r_1' \\ r_2' \\ r_3' \end{pmatrix} = \boldsymbol{R_z}(\theta) \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}.$$

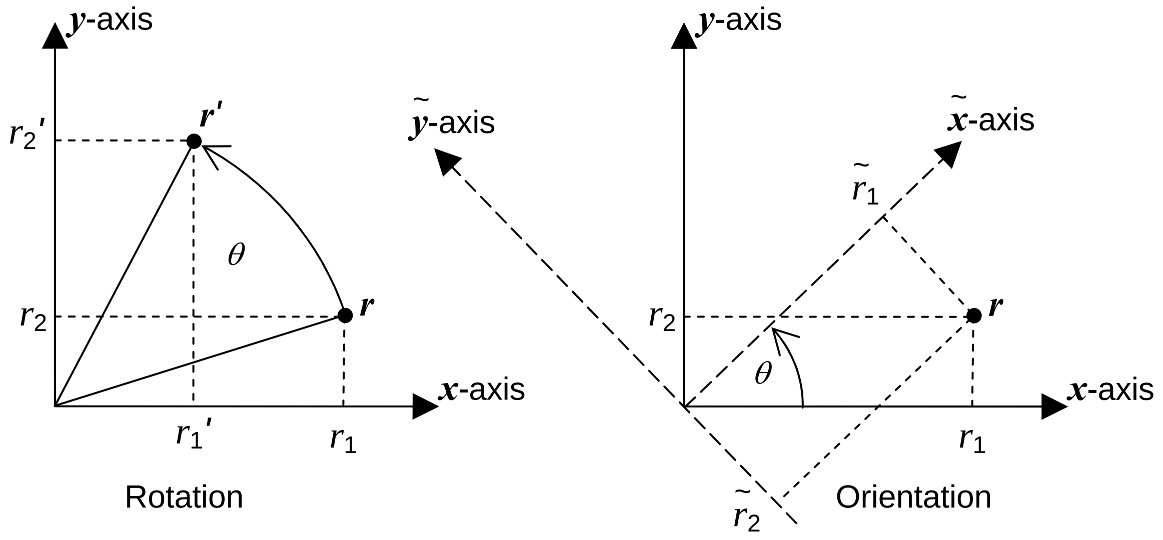


Figure 5 – Rotation and orientation

The right side of the figure shows a second basis whose orientation with respect to the first basis is a rotation by angle θ about the z-axis. In this case (Figure 5), let $\boldsymbol{\Omega_z}(\theta)$ denote the orientation $\boldsymbol{\Omega}_{E \to \tilde{E}}$. The coordinates of the single point $\boldsymbol{r}$ are related by the direction cosine matrix for this case.

$$\begin{pmatrix} \tilde{r}_1 \\ \tilde{r}_2 \\ \tilde{r}_3 \end{pmatrix} = \boldsymbol{\Omega_z}(\theta) \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}.$$

Notice that matrices corresponding to the left and right figures are not the same: $\boldsymbol{R}_z(\theta) = \boldsymbol{\Omega}_z^{\mathrm{T}}(\theta)$, and $\boldsymbol{\Omega}_z(\theta)\boldsymbol{R}_z(\theta) = \boldsymbol{I}_{3\times 3}$. So while both cases, the rotation of a point, and the orientation of one coordinate system with respect to another, involve the same axis of rotation and the same angle of rotation, the corresponding linear operations are, in fact, the inverses of each other. We shall call an operator that performs a rotation, such as the operator on the left side of Figure 5, a *rotation operator* and an operator that changes coordinate system directions, such as on the right side of the Figure, an *orientation operator*.

Note that to transform a coordinate from the $\tilde{\boldsymbol{E}}$ coordinate system back to the $\boldsymbol{E}$ coordinate system, the rotation matrix may be used as the inverse operator:

$$\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \boldsymbol{\Omega}_z^{-1}(\theta) \begin{pmatrix} \tilde{r}_1 \\ \tilde{r}_2 \\ \tilde{r}_3 \end{pmatrix} = \boldsymbol{R}_z(\theta) \begin{pmatrix} \tilde{r}_1 \\ \tilde{r}_2 \\ \tilde{r}_3 \end{pmatrix}$$

Note as well that the inverse of a rotation is a reverse rotation so that $\boldsymbol{\Omega}_z(\theta) = \boldsymbol{R}_z(-\theta)$.

Alternatively, both operations may be treated as rotations from different coordinate frame view points. With respect to the original coordinate system, a rotation $\boldsymbol{R}$ is, in some contexts, called a *coordinate frame rotation.* With respect to the rotated coordinate system, an orientation operation $\boldsymbol{\Omega}$ is, in some contexts, called a *position vector rotation*.

It follows that a representation of a rotation will depend on its intended use or interpretation. This document will address three primary use cases:

Primary use case 1. This primary use case concerns rigid body dynamics. Rigid body dynamics characterize the motion of a rigid body by translation and rotation. Of particular concern in this document are the characterizations of instantaneous rotational kinematics – rotational velocity and rotational acceleration, and rotational dynamics – torque and inertia.

Primary use case 2. This primary use case concerns the descriptions of point positions in one coordinate system with respect to another coordinate system with a different orientation. A sub-case concerns position descriptions between a “space-fixed” or inertial coordinate system and “body-fixed” coordinate system attached to a rigid body that is either static or moving in time.

Primary use case 3. This primary use case combines the first two. Of particular concern is representing rigid body dynamics characterizations computed in one coordinate system in terms of the second coordinate system. The coordinate systems may both be space-fixed, or one may be moving with respect to the other.

3.2 Representing rotations

Rigid body motion exhibits six degrees of freedom - three degrees of freedom for translation and three degrees of freedom for rotation. This means that, in principle, a rotation operation on 3D Euclidean space can be specified by three scalar numbers.

That is indeed the case with Euler angle conventions (see below). However, other less compact specifications are commonly used because they are more amenable to some computations such as performing a rotation operation on a vector, composing rotations, interpolating rotations, and other operations, and/or because they can be measured or modeled directly. Of the various representation methods in prevalent use, each presents various tradeoffs with respect to storage size, and computational complexity, speed, and error control (see 3.5, 3.6, and 3.7). Thus the best representation is dependent on the requirements and computational environment of a user application. For this reason, different representations are in use and interoperability becomes an issue. This issue is compounded by the non-standard meaning of terms in prevalent use. To support interoperability, this document defines these terms and presents various methods and algorithms for key operations and inter-conversions between the representation methods.

3.2.1 Axis-angle vector rotation

The *axis-angle* representation of a rotation, $(\boldsymbol{n}, \theta)$, consists of a unit vector $\boldsymbol{n}$ $(\|\boldsymbol{n}\| = 1)$ and a rotation angle θ. This represents the rotation $\boldsymbol{R}_{\boldsymbol{n}}(\theta)$ through angle θ about the axis spanned by $\boldsymbol{n}$. The rotation direction is determined by the *right hand rule*: conceptually, if the right hand holds the vector $\boldsymbol{n}$ with thumb pointing in the direction of the vector, the fingers point in the direction of increasing θ. Large rotations (greater than one full revolution) are important in some applications, however, in this document angles shall be considered equivalent modulo 2π. As a consequence of Euler's theorem, every rotation operation may be represented as an axis-angle rotation.

This representation uses four scalar parameters $\boldsymbol{n} = (n_1, n_2, n_3)$ and θ. The constraint $\|\boldsymbol{n}\| = 1$ reduces the degrees of freedom down to three degrees of freedom. The axis-angle representation is not unique. In particular, the axis-angle pairs $(\boldsymbol{n}, \theta)$ and $(-\boldsymbol{n}, -\theta)$ represent the same rotation, and when $\theta = 0$, $\boldsymbol{n}$ may be any unit vector.

3.2.1.1 Rodrigues' rotation formula

The rotation of a vector $\boldsymbol{r}$ to a rotated vector $\boldsymbol{r}'$ in terms of $(\boldsymbol{n}, \theta)$ is given by Rodrigues' rotation formula (see Appendix B for its derivation):

$$\boldsymbol{r}' = \cos(\theta)\boldsymbol{r} + (1 - \cos(\theta))(\boldsymbol{r} \bullet \boldsymbol{n})\boldsymbol{n} + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r} \tag{1.2}$$

The terms may be rearranged to the alternate form:

$$\boldsymbol{r}' = \boldsymbol{r} + (1 - \cos(\theta))\boldsymbol{n} \times (\boldsymbol{n} \times \boldsymbol{r}) + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r} \tag{1.3}$$

The matrix form of this formula is:

$$\boldsymbol{r}' = \boldsymbol{R}\,\boldsymbol{r}$$

where:

$$\boldsymbol{R} = \left[\boldsymbol{I}_{3\times3} + \sin(\theta)\boldsymbol{S_n} + \left(1-\cos(\theta)\right)\boldsymbol{S_n^2}\right] \quad (1.4)$$

or, alternatively (see Appendix B):

$$\boldsymbol{R} = \left[\cos(\theta)\boldsymbol{I}_{3\times3} + \left(1-\cos(\theta)\right)\boldsymbol{n}\otimes\boldsymbol{n} + \sin(\theta)\boldsymbol{S_n}\right] \quad (1.5)$$

and

$$\boldsymbol{S_n} = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$$

is the skew-symmetric matrix associated with $\boldsymbol{n}$ (see 1.2). Note that here $\boldsymbol{R}$ is the matrix form of the rotation operator $\boldsymbol{R_n}(\theta)$.

3.2.2 Principal rotations

For a given 3 dimensional Euclidean space, an orthonormal basis may be represented by the coordinate 3-tuples: $\boldsymbol{x} = (1,0,0)^{\mathrm{T}}$, $\boldsymbol{y} = (0,1,0)^{\mathrm{T}}$, and $\boldsymbol{z} = (0,0,1)^{\mathrm{T}}$ with respect to that basis. As an axis of rotation, each of these unit vectors is called a *principal axis*[6] of rotation. A rotation about a principal axis is called a *principal rotation*. Some authors refer to these rotations as *elementary rotations*. The vector space operators: $\boldsymbol{R_x}(\alpha)$, $\boldsymbol{R_y}(\beta)$, and $\boldsymbol{R_z}(\gamma)$ will denote the three principal rotations through the respective angles α, β, and γ modulo 2π. In axis-angle representation, these are the rotations: $(\boldsymbol{x},\alpha)$, $(\boldsymbol{y},\beta)$, and $(\boldsymbol{z},\gamma)$.

When the $\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$ basis is rotated by a principal rotation $\boldsymbol{R_x}(\alpha)$, the resulting basis will have orientation $\boldsymbol{\Omega_x}(\alpha)$ with respect to the $\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$ basis, similarly for rotation $\boldsymbol{R_y}(\beta)$ and orientation $\boldsymbol{\Omega_y}(\beta)$, and rotation $\boldsymbol{R_z}(\gamma)$ and orientation $\boldsymbol{\Omega_z}(\gamma)$ (see 3.1).

3.2.3 Euler angles

Euler angles are a specification of a rotation (or an orientation) obtained by applying three consecutive principal rotations. There are twelve distinct ways to select a sequence of three principal axes and apply the principal rotations (24 if left-handed axes are considered)[7]. Each such ordered selection is an *Euler angle convention*. There is little agreement among authors in names or notations for these conventions.

[6] This term should not be confused with the moment of inertia principal axes (see 5).

[7] There cannot be two consecutive rotations on the same axis as they would combine to a single rotation. Thus, among right-handed axis systems, there are 3 choices for the first rotation axis, 2 choices each for the second and third rotation axes to avoid repeating a preceding axis choice (3x2x2=12).

There are numerous conventions for Euler angles in use and many are named inconsistently. (Note that some authors use a left-handed coordinate system. All coordinate systems in this document are right-handed). The convention defined in the next section (3.2.3.1) uses axes z–x–z (also known as the 3-1-3 convention) and is often called the x-convention. Replacing x with y gives the so-called y-convention (z–y–z or 3-2-3). Quantum physics treatments prefer the y-convention, but x–y–x (or 1-2-1) is also called the y-convention by some authors. The convention using x–y–z (or 1-2-3) is defined in section 3.2.3.2 below.

The Euler angle representation of a rotation or orientation is important, in part, because most inertial systems produce Euler angles as output. In addition, Euler angles are often used to determine orientation in control mechanisms such as robotic arms and motion platforms.

The three principal rotations may either be rotations about the original axes, or about the successively rotated axes. Given a rotation, let $\tilde{x}, \tilde{y}, \tilde{z}$ be the principal axes after the successive rotations are applied to the original x, y, z axes. To distinguish between these two coordinate bases, coordinates with respect to the original basis x, y, z will be called *space-fixed* or static coordinates and those with respect to the sequentially moving $\tilde{x}, \tilde{y}, \tilde{z}$ axes will be called *body-fixed* or rotating coordinates. It is useful to think of the $\tilde{x}, \tilde{y}, \tilde{z}$ as attached to a rigid body that will be rotated.

3.2.3.1 Euler angles in the z-x-z convention

We shall assume that the xy-plane and $\tilde{x}\tilde{y}$-plane intersect in a line. This line is called the *line of nodes* for this convention. The *Euler angles* in the z-x-z convention are the three angles defined as follows:

α is the angle between the x-axis and the line of nodes,
β is the angle between z-axis and the $\tilde{z}$-axis, and
γ is the angle between the line of nodes and the $\tilde{x}$-axis.

In some contexts α is called the *spin* angle, β is called the *nutation* angle, and γ is called the *precession* angle. Many authors use the symbols ϕ, θ, and ψ for these angles, but disagree on the order and angle identification. All angles are considered equivalent modulo 2π.

These three angles specify a rotation as consecutive principal rotations using the z–axis, the x–axis and z–axis again. There are two equivalent specifications, space-fixed and body-fixed.

In the space-fixed specification, all the principal rotations are about the space-fixed principal axes z and x. The first principal rotation is about the z-axis through angle α, followed by the x-axis through angle β, followed by the z-axis again through angle γ. The combined rotation is:

$$\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha).$$

This principal rotation sequence is the *Euler angle z–y–z rotation convention*.

In the body-fixed specification, all the principal rotations are about the body-fixed principal axes. Before any rotation is applied, the space-fixed and body-fixed bases coincide. The first principal rotation is about the $\boldsymbol{z}$-axis through angle γ. This rotates axes $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ to the intermediate orientations $\boldsymbol{x}'$ and $\boldsymbol{y}'$ (in this intermediate orientation, the $\boldsymbol{x}'$-axis lies on the line of nodes). The second rotation is about the intermediate $\boldsymbol{x}'$-axis through angle β. This second rotation moves the $\boldsymbol{y}'$ and $\boldsymbol{z}$ axes to intermediate orientations $\boldsymbol{z}''$ and $\boldsymbol{y}''$. The third rotation is about the $\boldsymbol{z}''$-axis through α which moves axes $\boldsymbol{x}'$ and $\boldsymbol{y}''$ to their final orientations $\boldsymbol{x}'''$ and $\boldsymbol{y}'''$ The combined rotation is $\boldsymbol{R}_{\boldsymbol{z}''}(\alpha)\boldsymbol{R}_{\boldsymbol{x}'}(\beta)\boldsymbol{R}_{\boldsymbol{z}}(\gamma)$. The final body-fixed axis orientations are $\tilde{\boldsymbol{x}} = \boldsymbol{x}'''$, $\tilde{\boldsymbol{y}} = \boldsymbol{y}'''$, $\tilde{\boldsymbol{z}} = \boldsymbol{z}''$. The sequence of body-fixed rotations is illustrated in Figure 6.

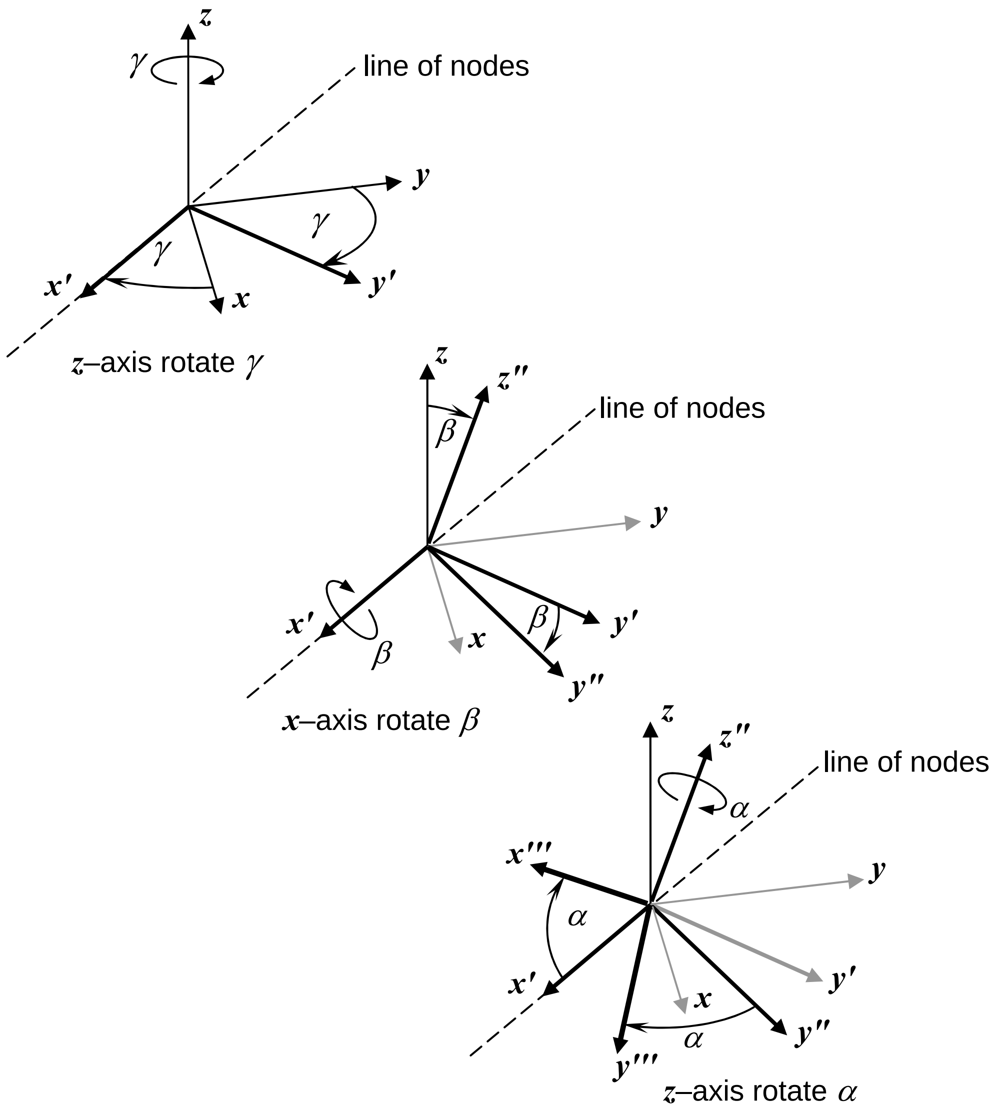


Figure 6 — Euler *z*-*x*-*z* rotation sequence

Observe that order of the three rotation angles is reversed between the space-fixed and body-fixed cases:

$$\begin{aligned}&\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha) \quad \text{space-fixed}\\ &\boldsymbol{R_{z''}}(\alpha)\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\gamma) \quad \text{body-fixed}\end{aligned} \tag{1.6}$$

To show that both expressions produce the same rotation, note that when $\boldsymbol{x'}$ is at its intermediate position on the line of nodes, the second rotation $\boldsymbol{R_{x'}}(\beta)$ is equivalent to first rotating the line of nodes to the $\boldsymbol{x}$-axis using principal rotation $\boldsymbol{R_z}(-\gamma)$, rotating about the $\boldsymbol{x}$-axis (which is the line of nodes at this point) with $\boldsymbol{R_x}(\beta)$ and finally rotating the line of nodes back to its original position with $\boldsymbol{R_z}(\gamma)$. In effect, $\boldsymbol{R_{x'}}(\beta)=\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(-\gamma)$. Similarly, $\boldsymbol{R_{z''}}(\alpha)=\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\alpha)\boldsymbol{R_{x'}}(-\beta)$. Noting that two rotations about the same axis commute and substituting these expressions in the body-fixed formulation gives:

$$\begin{aligned}\boldsymbol{R_{z''}}(\alpha)\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\gamma) &= \left[\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\alpha)\boldsymbol{R_{x'}}(-\beta)\right]\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\gamma)\\ &= \left[\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\alpha)\right]\boldsymbol{R_z}(\gamma)\\ &= \left[\left\{\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(-\gamma)\right\}\boldsymbol{R_z}(\alpha)\right]\boldsymbol{R_z}(\gamma)\\ &= \boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)\boldsymbol{R_z}(-\gamma)\boldsymbol{R_z}(\gamma)\\ &= \boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)\end{aligned}$$

This result:

$$\boldsymbol{R_{z''}}(\alpha)\boldsymbol{R_{x'}}(\beta)\boldsymbol{R_z}(\gamma)=\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha) \tag{1.7}$$

shows that the space-fixed and body-fixed formulations produce the same rotation. Both formulations are important. The matrix formulations of the principal rotations (Equation (1.15)) are expressed with respect to the static space-fixed frame. However, an inertial system attached to the body would read out the angles with respect to the rotating body-fixed frame.

The orientation of the $\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}},\tilde{\boldsymbol{z}}$ axes with respect to the $\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$ axes is the inverse (or transpose) of the rotation so that the angle sequence reverses:

$$\boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma) \tag{1.8}$$

This is *Euler angle z–y–z orientation convention*.

3.2.3.2 Euler angles in the x-y-z convention (Tait-Bryan angles)

In this convention the line of nodes is the intersection of the $\boldsymbol{xy}$-plane and $\tilde{\boldsymbol{y}}\tilde{\boldsymbol{z}}$-plane. The *Euler angles* in this convention are defined as follows:

ϕ is the angle between the line of nodes and the $\tilde{\boldsymbol{y}}$-axis,
θ is the angle between $\boldsymbol{z}$-axis and the $\tilde{\boldsymbol{y}}\tilde{\boldsymbol{z}}$-plane, and
ψ is the angle between the $\boldsymbol{y}$-axis and the line of nodes.

These three angles specify a rotation as principal rotations about the space-fixed principal axes. The first rotation is by angle ϕ about the $\boldsymbol{x}$-axis. The second is by angle θ about the $\boldsymbol{y}$-axis. The third is by angle ψ about the $\boldsymbol{z}$-axis. The combined rotation

$$\boldsymbol{R}_z(\psi)\,\boldsymbol{R}_y(\theta)\,\boldsymbol{R}_x(\phi) \quad \text{space-fixed.} \tag{1.9}$$

is the *Euler angle $\boldsymbol{z}$–$\boldsymbol{y}$–$\boldsymbol{x}$ rotation convention*. The equivalent body-fixed specification is:

$$\boldsymbol{R}_{\tilde{x}}(\phi)\,\boldsymbol{R}_{\tilde{y}}(\theta)\,\boldsymbol{R}_{\tilde{z}}(\psi) \quad \text{body-fixed.} \tag{1.10}$$

The corresponding *Euler angle $\boldsymbol{x}$–$\boldsymbol{y}$–$\boldsymbol{z}$ orientation convention* is the inverse operation:

$$\begin{aligned} &\boldsymbol{\Omega}_x(\phi)\,\boldsymbol{\Omega}_y(\theta)\,\boldsymbol{\Omega}_z(\psi) \quad \text{space-fixed} \\ &\boldsymbol{\Omega}_{\tilde{z}}(\psi)\,\boldsymbol{\Omega}_{\tilde{y}}(\theta)\,\boldsymbol{\Omega}_{\tilde{x}}(\phi) \quad \text{body-fixed} \end{aligned} \tag{1.11}$$

The Euler angles in this convention are variously called *Tait-Bryan angles*, *Cardano angles*, or *nautical angles*. The various names given to these angle symbols include:

ϕ roll or bank or tilt,
θ pitch or elevation, and
ψ yaw or heading or azimuth.

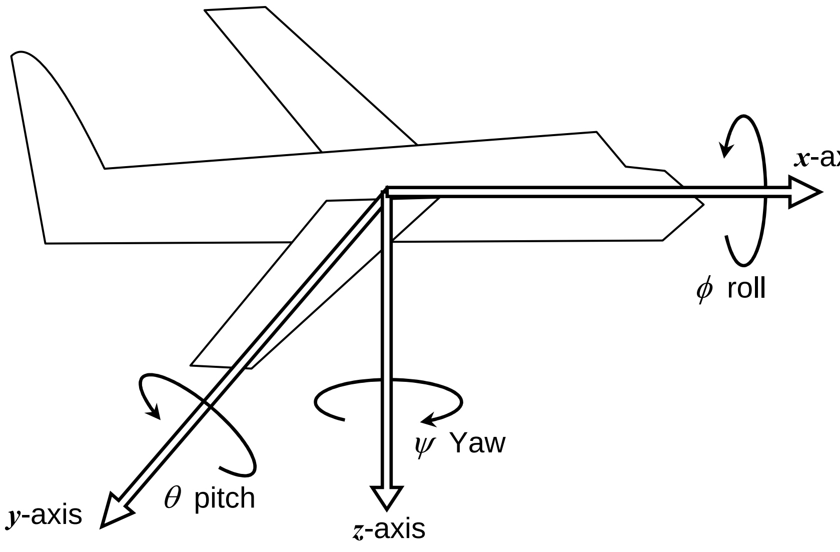


Figure 7 - Tait-Bryan angles

When the fixed body is an aircraft, the common practice is to choose the center of mass as the coordinate system origin with the $\boldsymbol{x}$-axis pointing forward, the $\boldsymbol{y}$-axis pointing starboard, and the $\boldsymbol{z}$-axis pointing down (to complete a right handed system[8]). The "entity coordinate system" defined in the IEEE 1278.1-1995 standard uses the same axis directions, but the coordinate system origin is at the center of the entity bounding volume. In both of these cases the Euler angles in the $\boldsymbol{x}$–$\boldsymbol{y}$–$\boldsymbol{z}$ convention are called the Tait-Bryan angles and the angle names *roll*, *pitch*, and *yaw* are specifically used.

The Euler angle $\boldsymbol{x}$–$\boldsymbol{y}$–$\boldsymbol{z}$ orientation convention is used to specify orientation in DIS packets as specified in the IEEE 1278.1-1995 standard [2]. In that standard Euler angles are defined as the successive rotations needed to transform from the world coordinate system to the entity coordinate system. In that standard, the world coordinate system is the WGS84 Geocentric SRF and the entity coordinate system is as shown in Figure 7. The specified rotation sequence is the Tait-Bryan angles rotation with respect to the body-fixed (rotating) entity frame, shown in Equation (1.10), or the space-fixed equivalent shown in Equation (1.9). To express a world frame coordinate in the entity coordinate frame, the inverse of that rotation is the required orientation operator (Equation (1.11)). The corresponding matrix operator (see 3.2.4.2) is denoted in the IEEE 1278.1-1995 standard as $[\boldsymbol{R}]_{w\to b}$.

3.2.3.3 Gimbal lock

The term gimbal lock refers to a gyroscope mounted in three nested gimbals to provide three degrees of rotational freedom. Each mounting scheme corresponds to an Euler angle convention. In any such mounting scheme, there exist critical angles for the middle gimbal that reduce the rotational degrees of freedom from three to two. In those critical configurations, the gimbals lie in a single plane and rotation within that plane is "locked out" by the gimbal mechanism. This loss of a degree of freedom is termed "gimbal lock".

The case of the Euler angle $\boldsymbol{z}$-$\boldsymbol{x}$-$\boldsymbol{z}$ rotation convention, it is assumed that the $\boldsymbol{xy}$-plane and $\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}$-plane intersect in a line (the line on nodes). That assumption is met when (modulo 2π) $\beta \neq 0$ and $\beta \neq \pi$. If not, $\beta = 0$ or $\beta = \pi$ and the consecutive rotations collapse down to a single principal rotation:

$$\begin{aligned} \beta = 0: &\quad \boldsymbol{R}_z(\gamma)\boldsymbol{R}_{\boldsymbol{x}}(0)\boldsymbol{R}_z(\alpha) = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_z(\ \alpha) = \boldsymbol{R}_z(\gamma+\alpha) \\ \beta = \pi: &\quad \boldsymbol{R}_z(\gamma)\boldsymbol{R}_{\boldsymbol{x}}(\pi)\boldsymbol{R}_z(\alpha) = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_z(-\alpha) = \boldsymbol{R}_z(\gamma-\alpha) \end{aligned}. \qquad (1.12)$$

This situation is illustrated by a spinning table top. The top spins on its spin-axis and precesses about the precession-axis. The angle between the spin- and precession-axes is the nutation angle. When the spin-axis is perfectly vertical (either upright or upside down), the nutation angle is 0 or π and the spin- and precession-axes become indistinguishable from each other as indicated in Equation (1.12).

[8] In this axis assignment, positive pitch tilts the aircraft up (angle of attack), and if the $\boldsymbol{x}$-axis aligns with local North, yaw corresponds to heading and azimuth.

The case of the Euler angle $\boldsymbol{z}$-$\boldsymbol{y}$-$\boldsymbol{x}$ convention (Tait-Bryan angles) it is assumed that the $\boldsymbol{xy}$-plane and $\tilde{\boldsymbol{y}}\tilde{\boldsymbol{z}}$-plane intersect in a line (the line of nodes). That assumption is met when $\theta \neq \pm\pi/2$ modulo 2π. If not, $\theta = \pm\pi/2$ and the $\tilde{\boldsymbol{x}}$-axis becomes parallel to the $\boldsymbol{z}$-axis and the consecutive rotations collapse down to a single principal rotation:

$$\begin{aligned}
\theta = +\pi/2: &\quad \boldsymbol{R_z}(\psi)\boldsymbol{R_y}\left(\frac{\pi}{2}\right)\boldsymbol{R_x}(\phi) = \boldsymbol{R_z}(\psi+\phi) \\
\theta = -\pi/2: &\quad \boldsymbol{R_z}(\psi)\boldsymbol{R_y}\left(\frac{-\pi}{2}\right)\boldsymbol{R_x}(\phi) = \boldsymbol{R_z}(\psi-\phi)
\end{aligned} \tag{1.13}$$

This situation is illustrated by an aircraft as in Figure 7. When the aircraft either climbs vertically, or dives vertically, roll-rotation cannot be distinguished from (plus or minus) yaw-rotation. This occurs at critical pitch angles of $\theta = \pm\pi/2$ as indicated in Equation (1.13).

3.2.4 Rotation and orientation matrices

A rotation (or orientation) operation on vector space is a linear operation, thus for a given basis, it has a matrix representation (see the direction cosine matrix definition in 3.1). If $\boldsymbol{R}$ is a rotation (or orientation) matrix, it satisfies these properties:

$$\begin{aligned}
&\det(\boldsymbol{R}) = 1 \\
&\boldsymbol{R}^{\mathsf{T}} = \boldsymbol{R}^{-1}
\end{aligned} \tag{1.14}$$

Matrices satisfying these properties form an algebraic group with respect to matrix multiplication. This group is known as the *special orthogonal group* of degree 3, SO(3). In particular, the product of any two rotation matrices is itself a rotation matrix and similarly for orientation matrices. (Note: Matrix multiplication is generally not commutative).

A rotation operation may be realized by simple matrix multiplication: $\boldsymbol{r}' = \boldsymbol{Rr}$. The inverse operation is $\boldsymbol{r} = \boldsymbol{R}^{\mathsf{T}}\boldsymbol{r}'$. While this is computationally convenient, the matrix representation does not lend itself well to intuitive visualization of the corresponding rotation. According to Euler's rotation theorem, there exists an axis-angle pair $(\boldsymbol{n}, \theta)$ for which $\boldsymbol{R}$ is the matrix representation of the rotation operator $\boldsymbol{R_n}(\theta)$. Finding this pair $(\boldsymbol{n}, \theta)$ involves the computational problem of finding the eigenvalues of $\boldsymbol{R}$ (see Appendix E).

For this and other reasons, it is useful to able to factor a given rotation matrix into a product of rotation matrices corresponding to a sequence of principal rotations. In general, a rotation matrix can be factored into three or less principal rotations called *principal factors* of the rotation. In particular, a rotation matrix has a factorization for each of the Euler angle convention of sequences of principal rotations.

The matrix forms of the principal rotations (and orientations) are:

$$\boldsymbol{R_x}(\alpha) = \boldsymbol{\Omega}_{\boldsymbol{x}}^{\mathsf{T}}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix},$$
$$\boldsymbol{R_y}(\beta) = \boldsymbol{\Omega}_{\boldsymbol{y}}^{\mathsf{T}}(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}, \text{ and}$$
$$\boldsymbol{R_z}(\gamma) = \boldsymbol{\Omega}_{\boldsymbol{z}}^{\mathsf{T}}(\gamma) = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{1.15}$$

NOTE: We are using $\boldsymbol{R}$ to denote the rotation operation that moves a point by rotation, and $\boldsymbol{\Omega}$ to denote the orientation operation that transforms a coordinate in a coordinate system into a coordinate in a coordinate system that is rotated with respect to the first coordinate system.

The next two sections will deal with matrix factorizations corresponding to Euler angle in z–x–z convention and the Euler angle z–y–x convention (Tait-Bryan angles).

3.2.4.1 Euler angle z-x-z convention matrix factorization

To factor a matrix $\boldsymbol{M} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ that belongs to SO(3) into a sequence of principal factors in the Euler angle z-x-z rotation convention[9] $\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)$, expand the sequence by multiplying the corresponding matrix forms (Equation (1.15)). The resulting matrix is:

$$\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha) =$$
$$\begin{pmatrix} \cos\alpha\cos\gamma - \cos\beta\sin\alpha\sin\gamma & -\sin\alpha\cos\gamma - \cos\beta\cos\alpha\sin\gamma & \sin\beta\sin\gamma \\ \cos\beta\sin\alpha\cos\gamma + \cos\alpha\sin\gamma & \cos\beta\cos\alpha\cos\gamma - \sin\alpha\sin\gamma & -\sin\beta\cos\gamma \\ \sin\beta\sin\alpha & \sin\beta\cos\alpha & \cos\beta \end{pmatrix} \tag{1.16}$$

[9] Note that the order of applying each of principal rotations is right-to-left so that the $\boldsymbol{R_z}(\alpha)$ rotation is conceptually performed first, but matrix multiplication is associative.

Matching the elements of this matrix to those of $\boldsymbol{M}$, we find that $a_{33} = \cos\beta$, $a_{31}/a_{32} = \tan\alpha$, and $a_{13}/(-a_{23}) = \tan\gamma$. These equations lead to the solutions in Table 1 based on the value of a_{33}.

Table 1 – Principal factors for *z-x-z* rotation

Case	Principal factors for rotation $\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)$ (all angles modulo 2π)		
$a_{33} \neq \pm 1$	$\beta = \text{arccos}(a_{33})$ [principal value] $0 < \beta < \pi$	$\alpha =$ $\text{arctan2}(a_{31}, a_{32})$	$\gamma =$ $\text{arctan2}(a_{13}, -a_{23})$
	$\beta = \text{arccos}(a_{33})$ [$2\pi -$ principal value] $\pi < \beta < 2\pi$	$\alpha =$ $\text{arctan2}(-a_{31}, -a_{32})$	$\gamma =$ $\text{arctan2}(-a_{13}, a_{23})$
$a_{33} = -1$	$\beta = \pi$	any value of α	$\gamma =$ $\text{arctan2}(a_{21}, a_{11}) + \alpha$
$a_{33} = +1$	$\beta = 0$	any value of α	$\gamma =$ $\text{arctan2}(a_{21}, a_{11}) - \alpha$

In the case $a_{33} \neq \pm 1$, arccos() is multi-valued so that there are two valid solution sets depending on the quadrants selected for arccosine values[10]. The principal value solution is the commonly used one. The two argument arctangent function arctan2() is defined in section 1.2.

In the case $a_{33} = -1$, using the trigonometric identities for the difference of angles and substituting for $\beta = \pi$, $\sin\beta = 0$ and $\cos\theta = -1$, the matrix expression reduces to :

$$\boldsymbol{R_z}(\gamma)\boldsymbol{R_y}(\pi)\boldsymbol{R_z}(\alpha) = \begin{pmatrix} \cos(\gamma-\alpha) & \sin(\gamma-\alpha) & 0 \\ \sin(\gamma-\alpha) & -\cos(\gamma-\alpha) & 0 \\ 0 & 0 & -1 \end{pmatrix}.$$

[10] Note that computer library functions such as acos() return the principal value only. The second solution for β may obtained by subtracting the principal value from 2π.

This shows that only the difference of the other two angles is determined as $\gamma - \alpha = \text{arctan2}(a_{21}, a_{11})$. Therefore, all values are valid for α if we set $\gamma = \text{arctan2}(a_{21}, a_{11}) + \alpha$. The case $a_{31} = +1$ is similar to the previous case with the sum of the angles determined by $\gamma + \alpha = \text{arctan2}(a_{21}, a_{11})$. These two cases correspond to Equation (1.12) , and are the gimbal lock cases.

To factor the matrix $\boldsymbol{M}$ into a sequence of principal factors in the Euler angle z-x-z orientation convention $\boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma)$, the corresponding orientation matrices in Equation (1.15) are multiplied out to the form:

$$\boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma) = \begin{pmatrix} \cos\alpha\cos\gamma - \cos\beta\sin\alpha\sin\gamma & \cos\beta\sin\alpha\cos\gamma + \cos\alpha\sin\gamma & \sin\beta\sin\alpha \\ -\sin\alpha\cos\gamma - \cos\beta\cos\alpha\sin\gamma & \cos\beta\cos\alpha\cos\gamma - \sin\alpha\sin\gamma & \sin\beta\cos\alpha \\ \sin\beta\sin\gamma & -\sin\beta\cos\gamma & \cos\beta \end{pmatrix} \tag{1.17}$$

This matrix is the transpose of the rotation case (Equation (1.16)). The solutions for the principal factor angles are shown in Table 2.

Table 2 – Principal factors for z-x-z orientation

Case	**Principal factors for orientation $\boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma)$** (all angles modulo 2π)		
$a_{33} \neq \pm 1$	$\beta = \arccos(a_{33})$ [principal value] $0 < \beta < \pi$	$\alpha =$ $\text{arctan2}(a_{13}, a_{23})$	$\gamma =$ $\text{arctan2}(a_{31}, -a_{32})$
	$\beta = \arccos(a_{33})$ [$2\pi -$ principal value] $\pi < \beta < 2\pi$	$\alpha =$ $\text{arctan2}(-a_{13}, -a_{23})$	$\gamma =$ $\text{arctan2}(-a_{31}, a_{32})$
$a_{33} = -1$	$\beta = \pi$	any value of α	$\gamma =$ $\text{arctan2}(a_{12}, a_{11}) + \alpha$
$a_{33} = +1$	$\beta = 0$	any value of α	$\gamma =$ $\text{arctan2}(a_{12}, a_{11}) - \alpha$

As in the rotation factorization, for case $a_{31} \neq \pm 1$, arccos() is multi-valued so that there are two valid solution sets depending on the quadrant selected for arccosine values. The principal value solution is the commonly used one. Here as well, the extreme values $a_{31} = \pm 1$ are the gimbal lock cases (see above).

As can be seen in the preceding tables, the three angle sequence corresponding to a given rotation or orientation operator is not unique modulo 2π. Two sequences, $(\alpha_1, \beta_1, \gamma_1)$ and $(\alpha_2, \beta_2, \gamma_2)$ of z-x-z principal factors specify the same operator if they satisfy one the criteria of the next Table.

Table 3 – Equivalence of z-x-z principal factor sequences

Case equality modulo 2π	**Criteria for the equivalence of angle sequences $(\alpha_1, \beta_1, \gamma_1)$ and $(\alpha_2, \beta_2, \gamma_2)$ for principal factor z-x-z sequences**
$\beta_1 = \beta_2$	$\alpha_1 = \alpha_2,\ \gamma_1 = \gamma_2$ $[\beta_1, \beta_2 \neq 0 \text{ or } \pi]$ (in)equalities modulo 2π
$\beta_1 + \beta_2 = 2\pi$	$\lvert\alpha_2 - \alpha_1\rvert = \pi,\ \lvert\gamma_2 - \gamma_1\rvert = \pi$ $[\beta_1, \beta_2 \neq 0 \text{ or } \pi]$ (in)equalities modulo 2π
$\beta_1 = \beta_2 = \pi$	$\alpha_1 - \gamma_1 = \alpha_2 - \gamma_2$ equality modulo 2π
$\beta_1 = \beta_2 = 0$	$\alpha_1 + \gamma_1 = \alpha_2 + \gamma_2$ equality modulo 2π

3.2.4.2 Tait-Bryan angles matrix factorization

To factor a matrix $\boldsymbol{M} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ that belongs to SO(3) into a sequence of principal factors in the Euler angle z-y-x rotation convention (Tait-Bryan angles) $\boldsymbol{R_z}(\psi)\boldsymbol{R_y}(\theta)\boldsymbol{R_x}(\phi)$, we multiply the corresponding principal rotation matrices (Equation (1.16)) to obtain:

$$\boldsymbol{R_z}(\psi)\,\boldsymbol{R_y}(\theta)\,\boldsymbol{R_x}(\phi) =$$
$$\begin{pmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix} \tag{1.18}$$

Matching the elements of this matrix to those of $\boldsymbol{M}$ we find that $a_{31} = -\sin\theta$, $a_{32}/a_{33} = \tan\phi$, and $a_{21}/a_{11} = \tan\psi$. These equations lead to the solutions in Table 4 for the principal factor angles based on the value of a_{31}.

Table 4 – Principal factors for *z-y-x* rotation

Case	**Principal factors for rotation $\boldsymbol{R_z}(\psi)\,\boldsymbol{R_y}(\theta)\,\boldsymbol{R_x}(\phi)$** (all angles modulo 2π)		
$a_{31} \neq \pm 1$	$\theta = \arcsin(-a_{31})$ [principal value] $-\pi/2 < \theta < \pi/2$	$\phi = \text{arctan2}(a_{32}, a_{33})$	$\psi =$ $\text{arctan2}(a_{21}, a_{11})$
	$\theta = \arcsin(-a_{31})$ [π – principal value] $\pi/2 < \theta < 3\pi/2$	$\phi = \text{arctan2}(-a_{32}, -a_{33})$	$\psi =$ $\text{arctan2}(-a_{21}, -a_{11})$
$a_{31} = -1$	$\theta = \pi/2$	$\phi =$ $\text{arctan2}(a_{12}, a_{13}) + \psi$	any value of ψ
$a_{31} = +1$	$\theta = -\pi/2$	$\phi =$ $\text{arctan2}(-a_{12}, -a_{13}) - \psi$	any value of ψ

In the case $a_{31} \neq \pm 1$, arcsin() is multi-valued so that there are two valid solution sets depending on the quadrant selected for arcsine values[11]. The principal value solution is the commonly used one.

In the case $a_{31} = -1$, using the trigonometric identities for the difference of angles and substituting $\sin\theta = 1$ and $\cos\theta = 0$, the matrix reduces to :

$$\boldsymbol{R_z}(\psi)\boldsymbol{R_y}\left(\frac{\pi}{2}\right)\boldsymbol{R_x}(\phi) = \begin{pmatrix} 0 & \sin(\phi-\psi) & \cos(\phi-\psi) \\ 0 & \cos(\phi-\psi) & -\sin(\phi-\psi) \\ -1 & 0 & 0 \end{pmatrix}.$$

This shows that only the difference of the other two angles is determined as $\phi - \psi = \text{arctan2}(a_{12}, a_{13})$. Therefore, all values are valid for ψ if we set $\phi = \text{arctan2}(a_{12}, a_{13}) + \psi$. The case $a_{31} = +1$ is similar to the previous case with the sum of the angles determined by $\phi + \psi = \text{arctan2}(-a_{12}, -a_{13})$. These two cases correspond to Equation (1.13) and are the gimbal lock cases.

To factor matrix $\boldsymbol{M}$ into a sequence of principal orientation factors of the form $\boldsymbol{\Omega_x}(\phi)\boldsymbol{\Omega_y}(\theta)\boldsymbol{\Omega_z}(\psi)$, multiply the corresponding principal factor matrices to obtain:

$$\boldsymbol{\Omega_x}(\phi)\boldsymbol{\Omega_y}(\theta)\boldsymbol{\Omega_z}(\psi) =$$
$$\begin{pmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \cos\theta\sin\phi \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{pmatrix} \tag{1.19}$$

Note that this matrix is just the transpose of $\boldsymbol{R_z}(\psi)\boldsymbol{R_y}(\theta)\boldsymbol{R_x}(\phi)$ so that solutions use transposed elements as shown in Table 5.

In the Table 5 case $a_{31} \neq \pm 1$, arcsin() is multi-valued so that there are two valid solution sets depending on the quadrant selected for arcsine values. The principal value solution is the commonly used one. The cases $a_{31} = \pm 1$ are the gimbal lock cases (see above).

[11] Note that computer library functions such as asin() return the principal value only. The second solution for θ may obtained by subtracting the principal value from π.

Table 5 – Principal factors for *x*-*y*-*z* orientation

Case	**Principal factors for orientation $\boldsymbol{\Omega_x}(\phi)\boldsymbol{\Omega_y}(\theta)\boldsymbol{\Omega_z}(\psi)$** (all angles modulo 2π)		
$a_{13} \neq \pm 1$	$\theta = \text{arcsin}(-a_{13})$ [principal value] $-\pi/2 < \theta < \pi/2$	$\phi =$ arctan2(a_{23}, a_{33})	$\psi =$ arctan2(a_{12}, a_{11})
	$\theta = \text{arcsin}(-a_{13})$ [$\pi +$ principal value] $\pi/2 < \theta < 3\pi/2$	$\phi =$ arctan2$(-a_{23}, -a_{33})$	$\psi =$ arctan2$(-a_{12}, -a_{11})$
$a_{13} = -1$	$\theta = \pi/2$	$\phi =$ arctan2$(a_{21}, a_{31}) + \psi$	any value of γ
$a_{13} = +1$	$\theta = -\pi/2$	$\phi =$ arctan2$(-a_{21}, -a_{31}) - \psi$	any value of γ

As can be seen in Table 4 and Table 5, the three angle sequence corresponding to a given rotation or orientation operator is not unique modulo 2π. Two such sequences, $(\alpha_1, \beta_1, \gamma_1)$ and $(\alpha_2, \beta_2, \gamma_2)$ specify the same operator if they satisfy one the criteria of Table 6.

Table 6 – Equivalence of *z-y-x* rotation or *x-y-z* orientation principal factor sequences

Case equality modulo 2π	**Criteria for the equivalence of angle sequences (ϕ_1,θ_1,ψ_1) and (ϕ_2,θ_2,ψ_2) for principal factor *z-y-x* rotation or *x-y-z* orientation sequences**
$\theta_1 = \theta_2$	$\phi_1 = \phi_2,\ \psi_1 = \psi_2$ $\left[\theta_1 \neq \pm\frac{\pi}{2} \neq \theta_2\right]$ (in)equalities modulo 2π
$\theta_1 + \theta_2 = \pi$	$\lvert\phi_2 - \phi_1\rvert = \pi,\ \lvert\psi_2 - \psi_1\rvert = \pi$ $\left[\theta_1 \neq \pm\frac{\pi}{2} \neq \theta_2\right]$ (in)equalities modulo 2π
$\theta_1 = \theta_2 = \frac{\pi}{2}$	$\phi_1 - \psi_1 = \phi_2 - \psi_2$ equality modulo 2π
$\theta_1 = \theta_2 = -\frac{\pi}{2}$	$\phi_1 + \psi_1 = \phi_2 + \psi_2$ equality modulo 2π

Factorizations for other Euler angle conventions may be obtained in a similar fashion.

3.2.5 Quaternions

In this section the definition of quaternions is presented. It is then shown that each quaternion induces a rotation operator. The importance of the algebraic structure of the quaternions is that rotations behave well under these operations.

3.2.5.1 Quaternion notations and conventions

The quaternions are a 4-dimensional vector space together with a vector multiplication operation that forms a non-commutative associative algebra. In analogy to complex numbers that are written as $a + \boldsymbol{i}b,\ \boldsymbol{i}^2 = -1$, quaternion axes $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k},$ are defined with the following relationships: $\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$. A quaternion $\boldsymbol{q}$ is denoted as $\boldsymbol{q} = e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k}$. The first term e_0 is called the “real” (or “scalar”) part of $\boldsymbol{q}$ and $e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k}$ is called the “imaginary” (or “vector”) part of $\boldsymbol{q}$.

There are several other conventions used to denote a quaternion. To distinguish conventions in this document, the $e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k}$ convention will be called the *Hamilton form*. The *scalar vector form* uses an ordered pair of a scalar and 3-tuple vector $\boldsymbol{q} = (e_0, \boldsymbol{e})$. The scalar is the real part of $\boldsymbol{q}$ and the vector corresponds to the imaginary part of $\boldsymbol{q}$, $\boldsymbol{e} = (e_1, e_2, e_3)^{\mathrm{T}}$. As can be seen below, the scalar vector form allows for some compact notation. (NOTE: In the literature, the order is sometimes reversed: $\boldsymbol{q} = (\boldsymbol{e}, e_0)$.)

Another convention is the *4-tuple form* which is just the 4-tuple of scalar numbers $\boldsymbol{q} = (e_0, e_1, e_2, e_3)$. Formulations below will be given in each of these three notational conventions. (NOTE: In the literature, the real part is sometimes placed last: $\boldsymbol{q} = (e_1, e_2, e_3, e_4)$ where $e_4 = e_0$.)

3.2.5.2 Quaternion algebra

Let $p = d_0 + d_1\boldsymbol{i} + d_2\boldsymbol{j} + d_3\boldsymbol{k}$ and $\boldsymbol{q}$ be two quaternions and let t be a scalar. Quaternion addition and scalar multiplication (in each notational convention) is defined as usual for 4D vector space:

$$\begin{aligned}
\boldsymbol{p} + t\boldsymbol{q} &= (d_0 + te_0) + (d_1 + te_1)\boldsymbol{i} + (d_2 + te_2)\boldsymbol{j} + (d_3 + te_3)\boldsymbol{k} && \text{[Hamiltion form]} \\
&= (d_0 + te_0,\ \boldsymbol{d} + t\boldsymbol{e}) && \text{[scalar vector form]} \\
&= (d_0 + te_0,\ d_1 + te_1,\ d_2 + te_2,\ d_3 + te_3) && \text{[4-tuple form]}
\end{aligned}$$

Assuming associative multiplication, the quaternion axes relationships gives the quaternion multiplication rule (in each notational convention):

$$\begin{aligned}
\boldsymbol{pq} &= (d_0e_0 - d_1e_1 - d_2e_2 - d_3e_3) \\
&\quad + (d_1e_0 + d_0e_1 + d_2e_3 - d_3e_2)\boldsymbol{i} && \text{[Hamiltion form]} \\
&\quad + (d_2e_0 + d_0e_2 + d_3e_1 - d_1e_3)\boldsymbol{j} \\
&\quad + (d_3e_0 + d_0e_3 + d_1e_2 - d_2e_1)\boldsymbol{k} \\
&= \left((d_0e_0 - \boldsymbol{d} \bullet \boldsymbol{e}), (e_0\boldsymbol{d} + d_0\boldsymbol{e} + \boldsymbol{d} \times \boldsymbol{e})\right) && \text{[Scalar vector form]} \\
&= \begin{pmatrix} (d_0e_0 - d_1e_1 - d_2e_2 - d_3e_3), \\ (d_1e_0 + d_0e_1 + d_2e_3 - d_3e_2), \\ (d_2e_0 + d_0e_2 + d_3e_1 - d_1e_3), \\ (d_3e_0 + d_0e_3 + d_1e_2 - d_2e_1) \end{pmatrix} && \text{[4-tuple form]}
\end{aligned} \tag{1.20}$$

Quaternion multiplication is not commutative (note the cross product term in the scalar vector form is anti-symmetric). However, the quaternion addition and multiplication operations together form an associative algebra.

The conjugate of a quaternion $\boldsymbol{q}$ is defined analogously with complex numbers:

$$\begin{aligned} \boldsymbol{q}^* &= e_0 - e_1\boldsymbol{i} - e_2\boldsymbol{j} - e_3\boldsymbol{k} && \text{[Hamiltion form]} \\ &= \left(e_0, -\boldsymbol{e}\right) && \text{[scalar vector form]} \\ &= \left(e_0, -e_1, -e_2, -e_3\right) && \text{[4-tuple form]} \end{aligned} \tag{1.21}$$

The product of a quaternion with its conjugate is "pure-real" and is called the *norm* of $\boldsymbol{q}$:

$$\begin{aligned} \boldsymbol{q}\boldsymbol{q}^* = \boldsymbol{q}^*\boldsymbol{q} &= e_0^2 + e_1^2 + e_2^2 + e_3^2 && \text{[Hamiltion form]} \\ &= \left(e_0^2 + e_1^2 + e_2^2 + e_3^2,\ \mathbf{0}\right) && \text{[scalar vector form]} \\ &= \left(e_0^2 + e_1^2 + e_2^2 + e_3^2,\ 0,\ 0,\ 0\right) && \text{[4-tuple form]} \end{aligned}$$

The *modulus* of a quaternion is defined as the square root of the norm:

$$\left|\boldsymbol{q}\right| = \sqrt{\boldsymbol{q}\boldsymbol{q}^*} = \sqrt{e_0^2 + e_1^2 + e_2^2 + e_3^2}\ .$$

A quaternion $\boldsymbol{q}$ is a *unit quaternion* if $\left|\boldsymbol{q}\right| = 1$. In that case $\boldsymbol{q}\boldsymbol{q}^* = \boldsymbol{q}^*\boldsymbol{q} = 1$ which implies that, for a unit quaternion, its conjugate is its multiplicative inverse $\boldsymbol{q}^{-1} = \boldsymbol{q}^*$. More generally, the inverse of a (non-unit) quaternion $\boldsymbol{p}$ is $\boldsymbol{p}^{-1} = \dfrac{\boldsymbol{p}^*}{\boldsymbol{p}\boldsymbol{p}^*} = \dfrac{\boldsymbol{p}^*}{\left|\boldsymbol{p}\right|^2}$.

If $\boldsymbol{q} = \left(e_0, \boldsymbol{e}\right)$ is a unit quaternion, then $\boldsymbol{q}$ may be expressed in the form:

$$\boldsymbol{q} = \left(\cos(\alpha), \sin(\alpha)\boldsymbol{n}\right)$$

where:

$$\begin{aligned} &\boldsymbol{n} = \frac{1}{\left\|\boldsymbol{e}\right\|}\boldsymbol{e} \text{ is a unit vector in 3D space,} \\ &\alpha = \operatorname{arctan2}\left(\left\|\boldsymbol{e}\right\|, e_0\right), \text{ and} \\ &\left\|\boldsymbol{e}\right\| = \sqrt{e_1^2 + e_2^2 + e_3^2} \end{aligned} \tag{1.22}$$

Note: The two argument arctangent function arctan2() is defined section 1.2.

3.2.5.3 Quaternion operators on 3D Euclidean space

Each quaternion $\boldsymbol{q}$ corresponds to a linear operator on 3D Euclidean space as follows:

Let $\boldsymbol{r} = (r_1, r_2, r_3)$ be a point in 3D Euclidean space, then the corresponding quaternion is formed by using 0 for the real part and $\boldsymbol{r}$ for the complex part $(0, \boldsymbol{r})$. A unit quaternion $\boldsymbol{q}$ operates on $(0, \boldsymbol{r})$ by left multiplying with $\boldsymbol{q}$ and right multiplying with its conjugate $\boldsymbol{q}^*$. It is shown in Appendix C that the real part of the product $\boldsymbol{q}(0, \boldsymbol{r})\boldsymbol{q}^* = (r_0', \boldsymbol{r}')$, is 0. Thus, $\boldsymbol{q}(0, \boldsymbol{r})\boldsymbol{q}^* = (0, \boldsymbol{r}')$ and the quaternion $\boldsymbol{q}$ associates $\boldsymbol{r}'$ with $\boldsymbol{r}$. Symbolically the operation on $\boldsymbol{r}$ is:.

$$\boldsymbol{r} \mapsto \boldsymbol{r}' = imaginary\ \ part\left\{\boldsymbol{q}(0, \boldsymbol{r})\boldsymbol{q}^*\right\}. \tag{1.23}$$

In terms of $\boldsymbol{q} = (e_0, \boldsymbol{e})$ it is also shown in Appendix C that

$$\boldsymbol{r}' = \left(e_0^2 - \boldsymbol{e} \bullet \boldsymbol{e}\right)\boldsymbol{r} + 2(\boldsymbol{e} \bullet \boldsymbol{r})\boldsymbol{e} + 2e_0\boldsymbol{e} \times \boldsymbol{r}. \tag{1.24}$$

Note that $-\boldsymbol{q} = (-e_0, -\boldsymbol{e})$ produces the same $\boldsymbol{r}'$ so that $\boldsymbol{q}$ and $-\boldsymbol{q}$ produce equivalent rotations.

Since $\boldsymbol{q}$ is a unit quaternion, there exists (Equation (1.22)) an angle α satisfying $\boldsymbol{q} = (\cos(\alpha), \sin(\alpha)\boldsymbol{n})$. By substitution in Equation (1.24):

$$\boldsymbol{r}' = \left(\cos^2(\alpha) - \sin^2(\alpha)\boldsymbol{n} \bullet \boldsymbol{n}\right)\boldsymbol{r} + 2\sin^2(\alpha)(\boldsymbol{n} \bullet \boldsymbol{r})\boldsymbol{n} + 2\cos(\alpha)\sin(\alpha)\boldsymbol{n} \times \boldsymbol{r}$$

if $\theta = 2\alpha$, then

$\cos(\theta) = \cos^2(\alpha) - \sin^2(\alpha) = 1 - 2\sin^2(\alpha)$, and

$\sin(\theta) = 2\cos(\alpha)\sin(\alpha)$, so that

$$\boldsymbol{r}' = \cos(\theta)\boldsymbol{r} + (1 - \cos(\theta))(\boldsymbol{n} \bullet \boldsymbol{r})\boldsymbol{n} + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r}$$

This last expression is Rodrigues' rotation formula (Equation (1.3)) for a counter-clockwise rotation about axis $\boldsymbol{n}$ through angle θ, thus quaternion operation on $\boldsymbol{r}$ is a rotation operation. Unit quaternions in scalar vector form are often written as $\boldsymbol{q} = \left(\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\boldsymbol{n}\right)$ to indicate the corresponding rotation angle θ (see Equation (1.22)).

Let $\boldsymbol{p}$ be any non-zero quaternion and let $\boldsymbol{q} = \frac{\boldsymbol{p}}{|\boldsymbol{p}|^2}$ be its corresponding unit quaternion,

then $\boldsymbol{p}(0,\boldsymbol{r})\boldsymbol{p}^{-1} = \boldsymbol{p}(0,\boldsymbol{r})\frac{\boldsymbol{p}^*}{|\boldsymbol{p}|^2} = \frac{\boldsymbol{p}}{|\boldsymbol{p}|}(0,\boldsymbol{r})\frac{\boldsymbol{p}^*}{|\boldsymbol{p}|} = \boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^*$.

This shows that any non-zero quaternion performs a rotation with the $\boldsymbol{p}(0,\boldsymbol{r})\boldsymbol{p}^{-1}$ operation and that this rotation is identical to the rotation performed by the corresponding unit quaternion $\boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^{-1} = \boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^*$. For this reason, some authors use $\boldsymbol{p}(0,\boldsymbol{r})\boldsymbol{p}^{-1}$ operations for any non-zero quaternion while other restrict the set to unit quaternions only and use the $\boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^*$ operator. Note, however, that Equations (1.22) and (1.24) assume a unit quaternion. A unit quaternion in 4-tuple form is also called the *Euler parameters* (or the *Euler-Rodrigues parameters*) of a rotation.

The quaternion representation of rotation facilitates the computation of the composition of two rotations. If $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ are two unit quaternions, the composite rotation on $\boldsymbol{r}$ is obtained by first rotating with the rotation operation induced by $\boldsymbol{q}_1$ and then rotating the result with the rotation operation induced by $\boldsymbol{q}_2$. This composite rotation is the same as the single rotation induced by the quaternion product $\boldsymbol{q}_2\boldsymbol{q}_1$ since

$$\boldsymbol{q}_2\left\{\boldsymbol{q}_1(0,\boldsymbol{r})\boldsymbol{q}_1^*\right\}\boldsymbol{q}_2^* = \boldsymbol{q}_2\boldsymbol{q}_1(0,\boldsymbol{r})\boldsymbol{q}_1^*\boldsymbol{q}_2^* = \left\{\boldsymbol{q}_2\boldsymbol{q}_1\right\}(0,\boldsymbol{r})\left\{\boldsymbol{q}_2\boldsymbol{q}_1\right\}^*.$$

3.2.5.4 Quaternions in matrix forms

A quaternion $\boldsymbol{q} = (e_0, e_1, e_2, e_3)$ can also be represented as a 2x2 complex matrix or a 4x4 real matrix.

The 2×2 complex matrix form is $\begin{pmatrix} e_0 + ie_1 & e_2 + ie_3 \\ -e_2 + ie_3 & e_0 - ie_1 \end{pmatrix}$.

The 4x4 real matrix form is $\begin{pmatrix} e_0 & -e_1 & e_3 & -e_2 \\ e_1 & e_0 & -e_2 & -e_3 \\ -e_3 & e_2 & e_0 & -e_1 \\ e_2 & e_3 & e_1 & e_0 \end{pmatrix}$.

The advantage of these forms are that quaternion addition and quaternion multiplication operations are just the usual matrix addition and matrix multiplication operations. The conjugate $\boldsymbol{q}^*$ is just the matrix conjugate transpose in the complex 2x2 case and the matrix transpose in the real 4x4 case.

3.2.6 Representation summary

Some important attributes of the representations in this section are summarized in the following Table.

Table 7 – Summary of representation attributes

Represen-tation type	**Data compo-nents**	**Data constraints**	**Ambiguities** (modulo 2π)	**Composition**	**Inverse**
Axis-angle $(\boldsymbol{n}, \theta)$	4	$\Vert\boldsymbol{n}\Vert = 1$	$(\boldsymbol{n}, \theta)$ is equivalent to $(-\boldsymbol{n}, -\theta)$. If $\theta = 0$, $\boldsymbol{n}$ is indeterminate	Convert to/from another representation for the operation.	$(\boldsymbol{n}, -\theta)$ or $(-\boldsymbol{n}, \theta)$
Matrix $\boldsymbol{R}$	9	$\det(\boldsymbol{R}) = 1$ $\boldsymbol{R}^{\mathrm{T}} = \boldsymbol{R}^{-1}$	None	Matrix multiplication	$\boldsymbol{R}^{\mathrm{T}}$
Euler angle conventions	3	None	2 or more *z-x-z* convention: see Table 3 Tait-Bryan angles: see Table 6	Convert to/from another representation for the operation (see Note 2).	See Note 1
Unit quaternion $\boldsymbol{q}$	4	unit constraint: $\boldsymbol{q}\boldsymbol{q}^* = 1$	$\boldsymbol{q}$ is equivalent to $-\boldsymbol{q}$ (see Note 3).	Quaternion multiplication.	$\boldsymbol{q}^*$ or $-\boldsymbol{q}^*$

Note 1: In the Euler angle *z-x-z* rotation convention

$$\left[\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)\right]^{-1} = \boldsymbol{R_z}(-\alpha)\boldsymbol{R_x}(-\beta)\boldsymbol{R_z}(-\gamma) = \boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma)$$

In the Euler angle *z-y-x* rotation convention (Tait-Bryan angles)

$$\left[\boldsymbol{R_z}(\psi)\boldsymbol{R_y}(\theta)\boldsymbol{R_x}(\phi)\right]^{-1} = \boldsymbol{R_x}(-\phi)\boldsymbol{R_y}(-\theta)\boldsymbol{R_z}(-\psi) = \boldsymbol{\Omega_x}(\phi)\boldsymbol{\Omega_y}(\theta)\boldsymbol{\Omega_z}(\psi)$$

Note 2: The composition of Euler angle operations may also be performed in a "direct" method that involves expressions that use combinations of forward and inverse trigonometric functions.

Note 3: Formulae such as Equation (1.24) require the unit quaternion constraint. Other useful relationships such as Equation (1.23) do not have that requirement. For that reason, some applications do not enforce the unit constraint. In the unconstrained case, every non-zero scalar multiple of a given quaternion is rotationally equivalent to it.

3.3 Performing a rotation on an arbitrary point (formulae)

3.3.1 Rotation about the origin

A point represented by vector $\boldsymbol{r}$ is rotated to a new position represented by vector $\boldsymbol{r}'$.

Using the rotation matrix representation

The rotated point is obtained by matrix multiplication.

$$\boldsymbol{r}' = \boldsymbol{R}\,\boldsymbol{r}$$

Using Euler angle sequences

Euler angles are first converted to a matrix (section 3.4.1 below). For the Euler angle z-x-z rotation convention, use the matrix in Equation (1.16). For the Euler angle z–y–x rotation convention (Tait-Bryan angles), use the matrix in Equation (1.18).

In either convention, conversion to quaternion may also be used (see section 3.4.9 below).

Using the axis-angle representation

A counter-clockwise rotation about axis $\boldsymbol{n}$ (a unit vector) through angle θ is given by Rodrigues' rotation formula (Equation (1.2)):

$$\boldsymbol{r}' = \cos(\theta)\boldsymbol{r} + (1-\cos(\theta))(\boldsymbol{r} \bullet \boldsymbol{n})\boldsymbol{n} + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r}\ .$$

Using the quaternion representation

From Equation (1.24), the rotation specified by unit quaternion $\boldsymbol{q} = (e_0, \boldsymbol{e})$ is:

$$\boldsymbol{r}' = (e_0^2 - \boldsymbol{e} \bullet \boldsymbol{e})\boldsymbol{r} + 2(\boldsymbol{e} \bullet \boldsymbol{r})\boldsymbol{e} + 2e_0\boldsymbol{e} \times \boldsymbol{r}$$

See also Equation (1.23) for the direct quaternion multiplication method.

3.3.2 Rotation about another point

To perform a rotation of point $\boldsymbol{q}$ about the point $\boldsymbol{p}$ to obtain a rotated point $\boldsymbol{q}'$, let

$$\boldsymbol{r} = \boldsymbol{q} - \boldsymbol{p}.$$

Rotate $\boldsymbol{r}$ to $\boldsymbol{r}'$, and then let

$$\boldsymbol{q}' = \boldsymbol{r}' + \boldsymbol{p}.$$

3.4 Inter-converting between representations (formulae)

3.4.1 Euler angle convention to matrix

The Euler angle z-x-z rotation convention is converted to a matrix $\boldsymbol{R}$ by forming the matrix product of the corresponding three principal rotation matrices defined in 3.2.2, Equation (1.15).

$$\boldsymbol{R} = \boldsymbol{R}_{\boldsymbol{z}}(\gamma)\boldsymbol{R}_{\boldsymbol{x}}(\beta)\boldsymbol{R}_{\boldsymbol{z}}(\alpha).$$

The resulting matrix is given in Equation (1.16).

The Euler angle z-x-z orientation convention is converted to a matrix $\boldsymbol{\Omega}$ by forming the matrix product of the corresponding three principal orientation matrices defined in 3.2.2, Equation (1.15).

$$\boldsymbol{\Omega} = \boldsymbol{\Omega}_{\boldsymbol{z}}(\alpha)\boldsymbol{\Omega}_{\boldsymbol{x}}(\beta)\boldsymbol{\Omega}_{\boldsymbol{z}}(\gamma).$$

The resulting matrix is given in Equation (1.17).

The Euler angle z-y-x rotation convention (Tait-Bryan angles) is converted to a matrix $\boldsymbol{R}$ by forming the matrix product of the corresponding three principal rotation matrices:

$$\boldsymbol{R} = \boldsymbol{R}_{\boldsymbol{z}}(\psi)\boldsymbol{R}_{\boldsymbol{y}}(\theta)\boldsymbol{R}_{\boldsymbol{x}}(\phi).$$

The resulting matrix is given in Equation (1.18).

The Euler angle x-y-z orientation convention (Tait-Bryan angles, IEEE 1278.1-1995 Convention) is converted to a matrix $\boldsymbol{\Omega}$ by forming the matrix product of the corresponding three principal orientation matrices:

$$\boldsymbol{\Omega} = \boldsymbol{\Omega}_{\boldsymbol{x}}(\phi)\boldsymbol{\Omega}_{\boldsymbol{y}}(\theta)\boldsymbol{\Omega}_{\boldsymbol{z}}(\psi).$$

The resulting matrix is given in Equation (1.19).

3.4.2 Matrix to axis-angle

Given a rotation matrix $\boldsymbol{R} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$, find a corresponding axis-angle representation $(\boldsymbol{n}, \theta)$. The following method to find $(\boldsymbol{n}, \theta)$ is based on Appendix E where it is shown that:

$$\theta = \arccos\left(\left(\frac{\text{Trace}(\boldsymbol{R}) - 1}{2}\right)\right) = \arccos\left(\left(\frac{(a_{11} + a_{22} + a_{33}) - 1}{2}\right)\right), \ 0 \le \theta \le \pi.$$

There are three cases for the computation of $\boldsymbol{n}$ that depend on the value of θ.

Case $\theta = 0$: There is no rotation so $\boldsymbol{n}$ is indeterminant.

Case $0 < \theta < \pi$: Let $\boldsymbol{n} = \frac{1}{\|\boldsymbol{v}\|}\boldsymbol{v}$, where:

$$\boldsymbol{v} = \begin{pmatrix} a_{32} - a_{23} \\ a_{13} - a_{31} \\ a_{21} - a_{12} \end{pmatrix}.$$

In this case, $\| v \| = 2\left|\sin(\theta)\right|$.

Case: $\theta = \pi$: First find the maximum diagonal element a_{11}, a_{22}, or a_{33} of $\boldsymbol{R}$. Then:

Sub-case: a_{11} is the maximum. Let $\boldsymbol{v} = \left(a_{11} + 1, a_{12}, a_{13}\right)^{\mathsf{T}}$.

Sub-case: a_{22} is the maximum. Let $\boldsymbol{v} = \left(a_{21}, a_{22} + 1, a_{23}\right)^{\mathsf{T}}$.

Sub-case: a_{33} is the maximum. Let $\boldsymbol{v} = \left(a_{31}, a_{32}, a_{33} + 1\right)^{\mathsf{T}}$

Finally let $\boldsymbol{n} = \frac{1}{\|\boldsymbol{v}\|}\boldsymbol{v}$.

In all cases, $\left(-\boldsymbol{n}, -\theta\right)$ is a rotationally equivalent solution.
Given an orientation matrix $\boldsymbol{\Omega}$, let $\boldsymbol{R} = \boldsymbol{\Omega}^{\mathsf{T}}$ and compute $\left(\boldsymbol{n}, \theta\right)$ as above.

3.4.3 Axis-angle to rotation matrix

To convert an axis-angle rotation $\left(\boldsymbol{n},\theta\right)$ to the corresponding rotation matrix $\boldsymbol{R}$, use Rodrigues' rotation formula (Equation (1.4) or (1.5)):

$$\begin{aligned}\boldsymbol{R} &= \left[\boldsymbol{I}_{3\times3} + \sin(\theta)\boldsymbol{S}_{\boldsymbol{n}} + \left(1 - \cos(\theta)\right)\boldsymbol{S}_{\boldsymbol{n}}^2\right] \\ &= \left[\cos(\theta)\boldsymbol{I}_{3\times3} + \left(1 - \cos(\theta)\right)\boldsymbol{n}\otimes\boldsymbol{n} + \sin(\theta)\boldsymbol{S}_{\boldsymbol{n}}\right]\end{aligned} \tag{1.25}$$

where:

$$\boldsymbol{S}_{\boldsymbol{n}} = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$$

is the skew-symmetric matrix associated with $\boldsymbol{n}$ (see 1.2).

Substituting $\boldsymbol{n} = (n_1, n_2, n_3)^{\mathsf{T}}$ in the expansion of $\boldsymbol{R}$ to matrix elements yields:

$$\begin{pmatrix} (1-\cos\theta)n_1^2 + \cos\theta & (1-\cos\theta)n_1n_2 - n_3\sin\theta & (1-\cos\theta)n_1n_3 + n_2\sin\theta \\ (1-\cos\theta)n_2n_1 + n_3\sin\theta & (1-\cos\theta)n_2^2 + \cos\theta & (1-\cos\theta)n_2n_3 - n_1\sin\theta \\ (1-\cos\theta)n_3n_1 - n_2\sin\theta & (1-\cos\theta)n_3n_2 + n_1\sin\theta & (1-\cos\theta)n_3^2 + \cos\theta \end{pmatrix} \quad (1.26)$$

The orientation matrix corresponding to $(\boldsymbol{n},\theta)$ is the transpose matrix: $\boldsymbol{\Omega} = \boldsymbol{R}^{\mathsf{T}}$.

3.4.4 Axis-angle to quaternion

Starting with an axis-angle representation $(\boldsymbol{n},\theta)$, where $\boldsymbol{n}$ is a unit vector and angle θ is a counter-clockwise rotation about $\boldsymbol{n}$. Let:

$$e_0 = \cos\left(\frac{\theta}{2}\right)$$
$$\boldsymbol{e} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \sin\left(\frac{\theta}{2}\right)\boldsymbol{n} = \sin\left(\frac{\theta}{2}\right)\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \quad (1.27)$$

Then the corresponding quaternion is:

$$\begin{aligned} \boldsymbol{q} &= e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k} && \text{[Hamiltion form]} \\ &= (e_0, \boldsymbol{e}) && \text{[scalar vector form]} \\ &= (e_0, e_1, e_2, e_3) && \text{[4-tuple form]} \end{aligned} \quad (1.28)$$

A rotationally equivalent quaternion is $-\boldsymbol{q}$.

3.4.5 Matrix to quaternion

Given a rotation matrix $\boldsymbol{R} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$, the corresponding quaternion

$$\begin{aligned} \boldsymbol{q} &= e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k} && \text{[Hamiltion form]} \\ &= (e_0, \boldsymbol{e}) && \text{[scalar vector form]} \\ &= (e_0, e_1, e_2, e_3) && \text{[4-tuple form]} \end{aligned}$$

is computed as follows:

$$e_0^2 = \frac{1}{4}\left(1 + \text{Trace}\left(\boldsymbol{R}\right)\right) = \frac{1}{4}\left(1 + a_{11} + a_{22} + a_{33}\right)$$

if $e_0^2 > 0$,

$$\boldsymbol{e} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \frac{1}{4e_0}\begin{pmatrix} a_{32} - a_{23} \\ a_{13} - a_{31} \\ a_{21} - a_{12} \end{pmatrix},$$

else $e_0 = 0$,

$$e_1^2 = -\frac{1}{2}\left(a_{22} + a_{33}\right),$$

if $e_1^2 > 0$, $\quad e_2 = \dfrac{a_{12}}{2e_1}, \; e_3 = \dfrac{a_{13}}{2e_1},$

else $e_1 = 0$,

$$e_2^2 = \frac{1}{2}\left(1 - a_{33}\right),$$

if $e_2^2 > 0$, $\quad e_3 = \dfrac{a_{23}}{2e_2}$

else $e_2 = 0$, $\quad e_3 = 1$.

A rotationally equivalent quaternion is $-\boldsymbol{q}$.

3.4.6 Quaternion to matrix

Given a unit quaternion:

$$\boldsymbol{q} = e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k} \quad \text{[Hamiltion form]}$$
$$= \left(e_0, \boldsymbol{e}\right) \quad \text{[scalar vector form]}$$
$$= \left(e_0, e_1, e_2, e_3\right) \quad \text{[4-tuple form]}$$

the corresponding rotation matrix is then:

$$\boldsymbol{R} = \begin{pmatrix} 1 - 2\left(e_2^2 + e_3^2\right) & 2\left(e_1e_2 - e_0e_3\right) & 2\left(e_1e_3 + e_0e_2\right) \\ 2\left(e_1e_2 + e_0e_3\right) & 1 - 2\left(e_1^2 + e_3^2\right) & 2\left(e_2e_3 - e_0e_1\right) \\ 2\left(e_1e_3 - e_0e_2\right) & 2\left(e_2e_3 + e_0e_1\right) & 1 - 2\left(e_1^2 + e_2^2\right) \end{pmatrix} \tag{1.29}$$

Since $e_0^2+e_1^2+e_2^2+e_3^2=1$, the diagonal terms in the matrix can be re-written in the following equivalent form:

$$\boldsymbol{R}=\begin{pmatrix} e_0^2+e_1^2-e_2^2-e_3^2 & 2\left(e_1e_2-e_0e_3\right) & 2\left(e_1e_3+e_0e_2\right) \\ 2\left(e_1e_2+e_0e_3\right) & e_0^2-e_1^2+e_2^2-e_3^2 & 2\left(e_2e_3-e_0e_1\right) \\ 2\left(e_1e_3-e_0e_2\right) & 2\left(e_2e_3+e_0e_1\right) & e_0^2-e_1^2-e_2^2+e_3^2 \end{pmatrix} \qquad (1.30)$$

This equation is derived as follows. Equation (1.24) is:

$$\boldsymbol{r}'=\left(e_0^2-\boldsymbol{e}\bullet\boldsymbol{e}\right)\boldsymbol{r}+2\left(\boldsymbol{e}\bullet\boldsymbol{r}\right)\boldsymbol{e}+2e_0\boldsymbol{e}\times\boldsymbol{r}\,.$$

This equation in matrix form (see Appendix A) is:

$$\boldsymbol{r}'=\left[\left(e_0^2-e_1^2-e_2^2-e_3^2\right)\boldsymbol{I}_{3\times3}+2\boldsymbol{e}\otimes\boldsymbol{e}+2e_0\boldsymbol{S_e}\right]\boldsymbol{r}$$

where $\boldsymbol{S_e}=\begin{pmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{pmatrix}$.

is the skew-symmetric matrix associated with $\boldsymbol{e}$ (see 1.2). The expansion of this expression gives Equation (1.30).

3.4.7 Quaternion to axis-angle

Given a unit quaternion

$\boldsymbol{q}=e_0+e_1\boldsymbol{i}+e_2\boldsymbol{j}+e_3\boldsymbol{k}$	[Hamiltion form]
$=\left(e_0,\boldsymbol{e}\right)$	[scalar vector form]
$=\left(e_0,e_1,e_2,e_3\right)$	[4-tuple form]

the corresponding axis-angle representation $\left(\boldsymbol{n},\theta\right)$ can be found as follows. Section 3.4.4 shows that the quaternion corresponding to axis-angle $\left(\boldsymbol{n},\theta\right)$ is

$$\boldsymbol{q}=\left(\cos\left(\frac{\theta}{2}\right),\sin\left(\frac{\theta}{2}\right)\boldsymbol{n}\right).$$

If $e_0\neq1$, this formulation may be reversed to yield:

$$\left(\boldsymbol{n},\theta\right)=\left(\boldsymbol{e}/\left\|\boldsymbol{e}\right\|,\ 2*\text{arctan2}\left(\left\|\boldsymbol{e}\right\|,e_0\right)\right)$$
$$=\left(\frac{\boldsymbol{e}}{\sqrt{1-e_0^2}},\ 2*\text{arctan2}\left(\sqrt{1-e_0^2},e_0\right)\right).$$

If $e_0=1$, $\theta=0$ and $\boldsymbol{n}$ is indeterminate. In either case, $\left(-\boldsymbol{n},-\theta\right)$ is a rotationally equivalent solution.

3.4.8 Matrix to Euler angle convention

Given a matrix $\boldsymbol{M} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ in SO(3), determine the corresponding Euler angles for a given convention.

To factor $\boldsymbol{M}$ in the Euler angle $\boldsymbol{z}$–$\boldsymbol{x}$–$\boldsymbol{z}$ rotation convention $\boldsymbol{M} = \boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)$ use Table 1.

To factor $\boldsymbol{M}$ in the Euler angle $\boldsymbol{z}$–$\boldsymbol{x}$–$\boldsymbol{z}$ orientation convention $\boldsymbol{M} = \boldsymbol{\Omega_z}(\alpha)\boldsymbol{\Omega_x}(\beta)\boldsymbol{\Omega_z}(\gamma)$ use Table 2.

To factor $\boldsymbol{M}$ in the Euler angles $\boldsymbol{z}$-$\boldsymbol{y}$-$\boldsymbol{x}$ rotation convention (Tait-Bryan angles) $\boldsymbol{M} = \boldsymbol{R_z}(\psi)\boldsymbol{R_y}(\theta)\boldsymbol{R_x}(\phi)$, use Table 4.

To factor $\boldsymbol{M}$ in the Euler angles $\boldsymbol{x}$-$\boldsymbol{y}$-$\boldsymbol{z}$ orientation convention (Tait-Bryan angles) $\boldsymbol{M} = \boldsymbol{\Omega_x}(\phi)\boldsymbol{\Omega_y}(\theta)\boldsymbol{\Omega_z}(\psi)$, use Table 5.

3.4.9 Euler angle convention to quaternion

The principal rotations (section 3.2.2) correspond to the following quaternions:

$$\boldsymbol{R_z}(\gamma) \leftrightarrow \left(\cos\left(\frac{\gamma}{2}\right), \sin\left(\frac{\gamma}{2}\right)\boldsymbol{z} \right)$$

$$\boldsymbol{R_y}(\beta) \leftrightarrow \left(\cos\left(\frac{\beta}{2}\right), \sin\left(\frac{\beta}{2}\right)\boldsymbol{y} \right)$$

$$\boldsymbol{R_x}(\alpha) \leftrightarrow \left(\cos\left(\frac{\alpha}{2}\right), \sin\left(\frac{\alpha}{2}\right)\boldsymbol{x} \right)$$

For each Euler convention, multiply the corresponding quaternions. Terms in the resulting product may be simplified using the orthonormal property of the vector set $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$. and various trigonometric identities.

For the Euler angle $\boldsymbol{z}$-$\boldsymbol{x}$-$\boldsymbol{z}$ rotation convention $\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)$, the equivalent quaternion is the product of three corresponding quaternion factors:

$$\boldsymbol{q} = \left(\cos\left(\frac{\gamma}{2}\right), \sin\left(\frac{\gamma}{2}\right)\boldsymbol{z} \right)\left(\cos\left(\frac{\beta}{2}\right), \sin\left(\frac{\beta}{2}\right)\boldsymbol{x} \right)\left(\cos\left(\frac{\alpha}{2}\right), \sin\left(\frac{\alpha}{2}\right)\boldsymbol{z} \right)$$

When multiplied out with the quaternion multiplication rule (Equation (1.20)), the expression reduces to:

$$\boldsymbol{q} = \left(e_0, \boldsymbol{e}\right) = \left(e_0, e_1, e_2, e_3\right)$$

where:

$$e_0 = \left(\cos\left(\frac{\gamma}{2}\right)\cos\left(\frac{\alpha}{2}\right) - \sin\left(\frac{\gamma}{2}\right)\sin\left(\frac{\alpha}{2}\right)\right)\cos\left(\frac{\beta}{2}\right) = \cos\left(\frac{\gamma+\alpha}{2}\right)\cos\left(\frac{\beta}{2}\right)$$

$$e_1 = \left(\cos\left(\frac{\gamma}{2}\right)\cos\left(\frac{\alpha}{2}\right) + \sin\left(\frac{\gamma}{2}\right)\sin\left(\frac{\alpha}{2}\right)\right)\sin\left(\frac{\beta}{2}\right) = \cos\left(\frac{\gamma-\alpha}{2}\right)\sin\left(\frac{\beta}{2}\right)$$

$$e_2 = \left(\sin\left(\frac{\gamma}{2}\right)\cos\left(\frac{\alpha}{2}\right) - \cos\left(\frac{\gamma}{2}\right)\sin\left(\frac{\alpha}{2}\right)\right)\sin\left(\frac{\beta}{2}\right) = \sin\left(\frac{\gamma-\alpha}{2}\right)\sin\left(\frac{\beta}{2}\right)$$

$$e_3 = \left(\sin\left(\frac{\gamma}{2}\right)\cos\left(\frac{\alpha}{2}\right) + \cos\left(\frac{\gamma}{2}\right)\sin\left(\frac{\alpha}{2}\right)\right)\cos\left(\frac{\beta}{2}\right) = \sin\left(\frac{\gamma+\alpha}{2}\right)\cos\left(\frac{\beta}{2}\right)$$

Note that $-\boldsymbol{q}$ is also a solution. The conjugate $\boldsymbol{q}^* = \left(e_0, -\boldsymbol{e}\right) = \left(e_0, -e_1, -e_2, -e_3\right)$ is the quaternion representation of the orientation operator $\boldsymbol{\Omega_z}\left(\alpha\right)\boldsymbol{\Omega_x}\left(\beta\right)\boldsymbol{\Omega_z}\left(\gamma\right)$.

For the Euler angle $\boldsymbol{z}$-$\boldsymbol{y}$-$\boldsymbol{x}$ rotation convention (Tait-Bryan angles) $\boldsymbol{R_z}\left(\psi\right)\boldsymbol{R_y}\left(\theta\right)\boldsymbol{R_x}\left(\phi\right)$, the equivalent quaternion is the product of three corresponding quaternion factors:

$$\boldsymbol{q} = \left(\cos\left(\frac{\psi}{2}\right), \sin\left(\frac{\psi}{2}\right)\boldsymbol{z}\right)\left(\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\boldsymbol{y}\right)\left(\cos\left(\frac{\phi}{2}\right), \sin\left(\frac{\phi}{2}\right)\boldsymbol{x}\right)$$

When multiplied out, the expression reduces to:

$$\boldsymbol{q} = \left(e_0, \boldsymbol{e}\right) = \left(e_0, e_1, e_2, e_3\right)$$

where:

$$e_0 = \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) + \sin\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right)$$

$$e_1 = \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right) - \sin\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right)$$

$$e_2 = \cos\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) + \sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right)$$

$$e_3 = \sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) - \cos\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right)$$

Note that $-\boldsymbol{q}$ is also a solution. The conjugate $\boldsymbol{q}^* = \left(e_0, -\boldsymbol{e}\right) = \left(e_0, -e_1, -e_2, -e_3\right)$ is the quaternion representation of the orientation $\boldsymbol{\Omega_x}\left(\phi\right)\boldsymbol{\Omega_y}\left(\theta\right)\boldsymbol{\Omega_z}\left(\psi\right)$. This quaternion corresponds to the IEEE 1278.1-1995 Convention [2] orientation matrix $\boldsymbol{\Omega_x}\left(\phi\right)\boldsymbol{\Omega_y}\left(\theta\right)\boldsymbol{\Omega_z}\left(\psi\right)$.

3.4.10 Quaternion to Euler angle convention

To convert a unit quaternion

$$\begin{aligned} \boldsymbol{q} &= e_0 + e_1\boldsymbol{i} + e_2\boldsymbol{j} + e_3\boldsymbol{k} && \text{[Hamiltion form]} \\ &= \left(e_0, \boldsymbol{e}\right) && \text{[scalar vector form]} \\ &= \left(e_0, e_1, e_2, e_3\right) && \text{[4-tuple form]} \end{aligned}$$

to the Euler angle ***z–x–z*** rotation convention $\boldsymbol{R_z}(\gamma)\boldsymbol{R_x}(\beta)\boldsymbol{R_z}(\alpha)$, compute:

$$\alpha = \text{arctan2}\left(\left(e_1e_3 + e_0e_2\right), -\left(e_2e_3 - e_0e_1\right)\right)$$

$$\beta = \arccos\left(1 - 2\left(e_1^2 + e_2^2\right)\right) \quad \text{principal value: } 0 < \beta < \pi$$

$$\gamma = \text{arctan2}\left(\left(e_1e_3 - e_0e_2\right), \left(e_2e_3 + e_0e_1\right)\right)$$

(Given this solution, see Table 3 for an equivalent alternate solution.)

This formulation assumes that $0 \neq \left(e_1^2 + e_2^2\right) \neq 1$. Otherwise the conversion is indeterminate with:

case $\left(e_1^2 + e_2^2\right) = 0$:

$$\beta = 0 \text{ and } \alpha + \gamma = \text{arctan2}\left(\left(e_1e_2 - e_0e_3\right), \tfrac{1}{2} - \left(e_2^2 + e_3^2\right)\right)$$

and

case $\left(e_1^2 + e_2^2\right) = 1$:

$$\beta = \pi \text{ and } \alpha - \gamma = \text{arctan2}\left(\left(e_1e_2 - e_0e_3\right), \tfrac{1}{2} - \left(e_2^2 + e_3^2\right)\right)$$

To convert to the Euler angle ***z-y-x*** rotation convention (Tait-Bryan angles) $\boldsymbol{R_z}(\psi)\boldsymbol{R_y}(\theta)\boldsymbol{R_x}(\phi)$, compute:

$$\phi = \text{arctan2}\left(\left(e_2e_3 + e_0e_1\right), \tfrac{1}{2} - \left(e_1^2 + e_2^2\right)\right)$$

$$\theta = \arcsin\left(-2\left(e_1e_3 - e_0e_2\right)\right) \quad \text{principal value: } -\pi/2 < \theta < \pi/2$$

$$\psi = \text{arctan2}\left(\left(e_1e_2 + e_0e_3\right), \frac{1}{2} - \left(e_2^2 + e_3^2\right)\right)$$

(Given this solution, see Table 6 for an equivalent alternate solution.)

This formulation assumes that $2\left(e_1e_3 - e_0e_2\right) \neq \pm 1$. Otherwise the conversion is indeterminate with:

case $2\left(e_1e_3 - e_0e_2\right) = +1$:

$$\theta = -\pi/2 \quad \text{and } \phi + \psi = \text{arctan2}\left(\left(e_1e_2 - e_0e_3\right), \left(e_1e_3 + e_0e_2\right)\right)$$

and

case $2\left(e_1e_3 - e_0e_2\right) = -1$:

$$\theta = \pi/2 \quad \text{and } \phi - \psi = \text{arctan2}\left(\left(e_1e_2 - e_0e_3\right), \left(e_1e_3 + e_0e_2\right)\right)$$

3.5 Considerations for computational and storage efficiency

The selection of a data representation to implement a rotation or orientation operation is highly dependent on the memory, storage, and performance requirements of the application, as well as the hardware and operating system environment. Generally, there is a tradeoff between data storage size and computational efficiency. A data transmission application may need to minimize transfer bit rate in contrast to a real time computer generated animation that may need to constantly compute changing orientations between scene graph nodes and to manipulate large quantities of vertices. As the hardware costs of RAM decrease, the requirements for quantity of data items in memory often tend to increase, and in some cases approach address space limits. The introduction of 64-bit CPUs and operating systems will remove that address space limit.

Consider the axis-angle representation that uses four scalar parameters $\boldsymbol{n} = \left(n_1, n_2, n_3\right)$ and θ. This representation could also be stored using just three scalars:

$$\boldsymbol{s} = \left(s_1, s_2, s_3\right) = \left(\theta n_1, \theta n_2, \theta n_3\right).$$

This is compact, but reconstruction is computationally expensive. It requires 2 adds, 6 multiplies, 1 divide and 1 square root:

$$\theta = \left\|\boldsymbol{s}\right\| = \sqrt{s_1^2 + s_2^2 + s_3^2}$$

$$\boldsymbol{n} = \frac{1}{\theta}\boldsymbol{s} = \left(\frac{s_1}{\theta}, \frac{s_2}{\theta}, \frac{s_3}{\theta}\right)$$

Note also that the recovered angle is non-negative so that if the starting θ is negative, then recovered axis-angle pair is (in terms of the starting values) the reversed sign equivalent pair: $\left(-\boldsymbol{n}, -\theta\right)$.

If many points are to be rotated, it should be noted that Rodrigues' rotation formula requires both the sine and cosine of the angle. Thus, in a computationally intensive application, it may be advantageous to compute the expensive trigonometric values once and store them as part of a five scalar axis-angle rotation data storage type:

$$\left(s_1, s_2, s_3, s_4, s_5\right) = \left(n_1, n_2, n_3, \sin\theta, \cos\theta\right).$$

If the value of θ itself is required often, it can be stored as a sixth scalar. Otherwise, θ

can be reconstructed as $\theta = \text{arctan2}(s_4, s_5)$. The $(n_1, n_2, n_3, \sin\theta, \cos\theta)$ data type is also advantageous in converting to and from quaternion representation (see 3.4.4 and 3.4.7), because half angle / double angle trigonometric identities are less computationally expensive than trigonometric and inverse trigonometric functions.

The composition of two rotations in axis-angle representation is trivial if the two rotations share the same axis. Otherwise, conversion to and from matrix or quaternion representation may be required.

Storage of the Matrix representation requires nine scalars. This representation has the computational advantage that vector rotation is just a matrix-vector multiply operation requiring only scalar multiply and add instructions. The composition of two rotation operators is also just a matrix multiplication. This simplicity makes the matrix representation attractive in many applications in spite of the large storage size.

The quaternion representation is compact requiring only four scalars for storage. Quaternion multiplication requires only scalar multiply and add instructions. A rotation operation requires two quaternion multiplications and a composition operation requires only one quaternion multiply. Compared to Matrix representation, both quaternion point rotation and quaternion composition require fewer add and multiply instructions. Quaternions also have an important computational advantage with respect to interpolation (see below).

The Euler angle conventions are compact, requiring only three scalars for storage, but are computationally inefficient – conversion to and from another type of representation is generally required for rotation and composition operations. An additional disadvantage is the presence of singular points (see 3.2.3.3 - Gimbal lock). The primary importance of the Euler angle conventions are that many physical systems provide measured data or are controlled by data in an Euler angle convention.

See [3] for a discussion of the computational costs of various conversions between representations.

3.6 Interpolation issues

Interpolating between two orientations is important in some types of applications. In distributed simulations, entity orientations are captured at a fixed rate. Some simulated visual systems need orientations at a higher frame rate. These intermediate visual frame rate orientations are interpolated with respect to time between pairs of captured orientations. Projecting state forward in time (dead reckoning) is a related problem. In computer 3D animations, key frames are generated. Intermediate frames at the higher render frame rate are generated by time interpolation between key frames.

Linear interpolation on the parameters of a representation is straightforward, but the resulting operator valued function of the interpolation parameter may have undesirable properties. Interpolating Euler angles may produce very indirect and un-natural rotation sequences and may involve gimbal lock. A linear interpolation of rotation matrices, $(1-t)\boldsymbol{M}_1 + t\boldsymbol{M}_2, 0 < t < 1$ will, in general, not be a rotation matrix.

Ken Shoemake defined an interpolation between a pair of quaternions that produces a uniform rate of rotation change with respect to the interpolation variable. This interpolation scheme is known as Spherical linear interpolation (SLERP), and is widely used in computer animation. In [4], SLERP is extended to allow second order smoothness in a sequence of three or more key frame orientations using splines.

Given two quaternions, $\boldsymbol{q}_0$ and $\boldsymbol{q}_1$, the SLERP interpolated value $\text{Slerp}\left(\boldsymbol{q}_0, \boldsymbol{q}_1, t\right)$ is given by:

$$\begin{aligned}\text{Slerp}\left(\boldsymbol{q}_0, \boldsymbol{q}_1, t\right) &= \boldsymbol{q}_0\left(\boldsymbol{q}_0^{-1}\boldsymbol{q}_1\right)^t \\ &= \frac{\sin\left((1-t)\theta\right)}{\sin(\theta)}\boldsymbol{q}_0 + \frac{\sin(t\theta)}{\sin(\theta)}\boldsymbol{q}_1,\end{aligned}$$

where: $0 \le t \le 1$ and $\cos(\theta) = \boldsymbol{q}_0 \bullet \boldsymbol{q}_1$ and this dot product is the 4-dimensional dot product.

3.7 Error analysis

One approach to error analysis is to note the behavior of the magnitude of first derivative terms with respect to input parameters. That study is beyond the present scope. It will be noted here that in the case of the various Euler angle conventions such magnitudes tend to blow up near gimbal lock singularities. In contrast, in the case of unit quaternions, the magnitudes are uniformly bounded. In [5], parameter rates, Jacobians and linearization are treated for these representations.

4 Rotational kinematics

4.1 Rotational velocity and acceleration

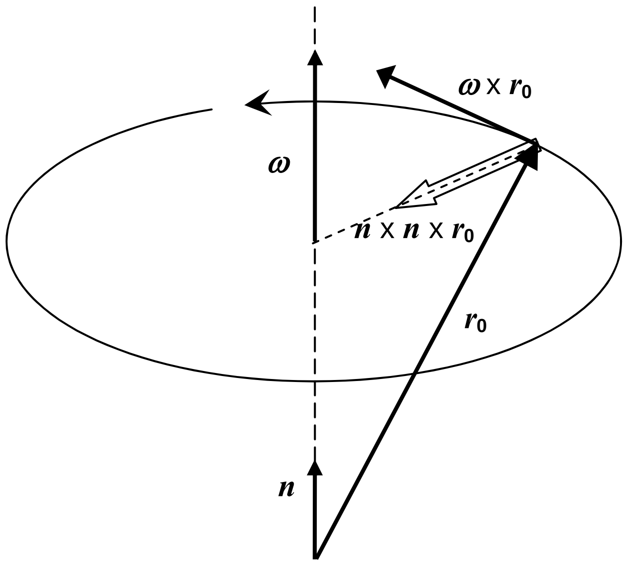
Consider first the special case of rotation of a point about a fixed unit vector axis $\boldsymbol{n}$ as a differentiable function of time, $\boldsymbol{r}(t) = \boldsymbol{R}_{\boldsymbol{n}}(\theta(t))\boldsymbol{r}_0$. Assume that the rotation begins at $t = 0$ so that $\theta(0) = 0$ and $\boldsymbol{r}(0) = \boldsymbol{r}_0$. By the alternate form of Rodrigues' rotation formula (Equation (1.3)):

$$\boldsymbol{r}(t) = \boldsymbol{R}_{\boldsymbol{n}}(\theta(t))\boldsymbol{r}_0 = \boldsymbol{r}_0 + (1 - \cos(\theta))\boldsymbol{n} \times (\boldsymbol{n} \times \boldsymbol{r}_0) + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r}_0 .$$

The rotational velocity at time t is then:

$$\begin{aligned}\frac{d}{dt}\boldsymbol{r}(t) &= \frac{d}{dt}\left(\boldsymbol{r}_0 + (1 - \cos(\theta))\boldsymbol{n} \times (\boldsymbol{n} \times \boldsymbol{r}_0) + \sin(\theta)\boldsymbol{n} \times \boldsymbol{r}_0\right) \\ &= \sin(\theta)\frac{d\theta}{dt}\boldsymbol{n} \times (\boldsymbol{n} \times \boldsymbol{r}_0) + \cos(\theta)\frac{d\theta}{dt}\boldsymbol{n} \times \boldsymbol{r}_0\end{aligned}$$

Thus the velocity is the sum of two vector components. One component $\boldsymbol{n} \times (\boldsymbol{n} \times \boldsymbol{r}_0)$ points toward the rotation axis and the other $\boldsymbol{n} \times \boldsymbol{r}_0$ is tangent to the arc or rotation (see Figure 8).



$\boldsymbol{\omega}$ x $\boldsymbol{r_0}$
$\boldsymbol{\omega}$
$\boldsymbol{n}$ x $\boldsymbol{n}$ x $\boldsymbol{r_0}$
$\boldsymbol{r_0}$
$\boldsymbol{n}$

Figure 8

Evaluating at $t = 0$ gives the instantaneous rotational velocity:

$$\frac{d}{dt}\boldsymbol{r}(0) = \frac{d\theta}{dt}\boldsymbol{n}\times\boldsymbol{r}_0 = \dot{\theta}\,\boldsymbol{n}\times\boldsymbol{r}_0 = \boldsymbol{\omega}\times\boldsymbol{r}_0\text{, where } \boldsymbol{\omega} = \dot{\theta}\,\boldsymbol{n}\,.$$

The rotational acceleration is the second time derivative:

$$\begin{aligned}\frac{d^2}{dt^2}\boldsymbol{r}(t) &= \frac{d}{dt}\left(\sin(\theta)\frac{d\theta}{dt}\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}_0) + \cos(\theta)\frac{d\theta}{dt}\boldsymbol{n}\times\boldsymbol{r}_0\right)\\ &= \left(\cos(\theta)\frac{d\theta}{dt} + \sin(\theta)\frac{d^2\theta}{dt^2}\right)\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}_0) + \left(-\sin(\theta)\frac{d\theta}{dt} + \cos(\theta)\frac{d^2\theta}{dt^2}\right)\boldsymbol{n}\times\boldsymbol{r}_0.\end{aligned}$$

Evaluating at $t = 0$ gives the instantaneous rotational acceleration $\boldsymbol{\alpha}$:

$$\boldsymbol{\alpha} = \dot{\boldsymbol{\omega}} = \dot{\theta}\,\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}_0) + \ddot{\theta}\,\boldsymbol{n}\times\boldsymbol{r}_0.$$

The first term in the right hand expression points towards the rotation axis as is called the centripetal acceleration.

In the general case (in which the rotation axis may vary as function of time), we set $\boldsymbol{r}(t) = \boldsymbol{R}(t)\boldsymbol{r}_0$ with the initial condition $\boldsymbol{R}(0) = \boldsymbol{I}_{3\times3}$ (that is, $\boldsymbol{r}(0) = \boldsymbol{r}_0$). To compute $\dot{\boldsymbol{R}} = \frac{d\boldsymbol{R}(0)}{dt}$, we note that as a rotation matrix we have $\boldsymbol{I}_{3\times3} = \boldsymbol{R}(t)\boldsymbol{R}^{\mathsf{T}}(t)$ so that:

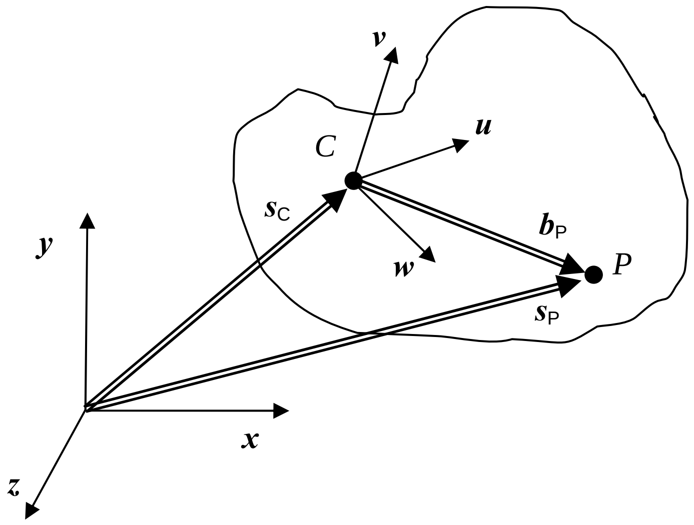
$$\begin{aligned}\frac{d}{dt}\boldsymbol{I}_{3\times3} &= \frac{d}{dt}\left[\boldsymbol{R}(t)\boldsymbol{R}^{\mathsf{T}}(t)\right]\\ \mathbf{0} &= \dot{\boldsymbol{R}}(t)\boldsymbol{R}^{\mathsf{T}}(t) + \boldsymbol{R}(t)\dot{\boldsymbol{R}}^{\mathsf{T}}(t)\end{aligned}$$

At $t = 0$, this expression reduces to $\dot{\boldsymbol{R}}(0) = -\dot{\boldsymbol{R}}^{\mathsf{T}}(0)$, so $\dot{\boldsymbol{R}}(0)$ is skew-symmetric and must be in the form $\dot{\boldsymbol{R}}(0) = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} = \boldsymbol{S}_{\boldsymbol{\omega}}$ where $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)^{\mathsf{T}}$. Thus we have $\frac{d}{dt}\boldsymbol{r}(0) = \boldsymbol{S}_{\boldsymbol{\omega}}\boldsymbol{r} = \boldsymbol{\omega}\times\boldsymbol{r}_0$.

4.2 Orientation ($\boldsymbol{\Omega}$), angular velocity ($\boldsymbol{\omega}$), and angular acceleration ($\boldsymbol{\alpha}$)

We consider three cases, of increasing complexity, of converting the position (and motion) of a point described by a body-fixed coordinate system to its description in a space-fixed coordinate system. The cases are static, rigid motion, and free motion.

Static case:
Consider a rigid body with a body-fixed coordinate system origin at some point C on the body and let P be another point on the body with body-coordinate vector $\boldsymbol{b}_\text{P}$. We are given $\boldsymbol{s}_\text{C}$ the space-fixed coordinate vector of point C and the orientation $\boldsymbol{\Omega}$ of the body coordinate system with respect to the space coordinate system (see Figure 9).



v
u
C
$\boldsymbol{s}_\text{C}$
$\boldsymbol{b}_\text{P}$
y
w
P
$\boldsymbol{s}_\text{P}$
x
z

Figure 9

The space coordinate vector $\boldsymbol{s}_\text{P}$ for the point P can then be computed as:

$$\boldsymbol{s}_\text{P} = \boldsymbol{s}_\text{C} + \boldsymbol{\Omega}\, \boldsymbol{b}_\text{P} \,.$$

In this expression, the term $\boldsymbol{\Omega}\, \boldsymbol{b}_\text{P}$ is to understood as an orientation operator acting on $\boldsymbol{b}_\text{P}$. The implementation of this operation may depend on the specification of the orientation. For example $\boldsymbol{\Omega}\, \boldsymbol{b}_\text{P}$ may be computed:

- by Rodrigues' rotation formula if the orientation was specified as an axis-angle,
- by a quaternion operation if the orientation was specified as a quaternion,
- by matrix multiplication using the directional cosine matrix computed from the body coordinate system basis vectors in space coordinates, or
- with a matrix or a quaternion computed from an Euler angle convention specification of the orientation.

While this comment is applicable to all such operations, we shall use matrix and vector notation in all of the following cases.

<u>Rigid motion case</u>:
In this case the body is in motion so that $\boldsymbol{s}_\mathrm{C}$ and $\boldsymbol{\Omega}$ are functions of time and we have:

$$\begin{aligned}
\boldsymbol{s}_\mathrm{P}(t) &= \boldsymbol{s}_\mathrm{C}(t) + \boldsymbol{\Omega}(t)\boldsymbol{b}_\mathrm{P} \\
\boldsymbol{v}_\mathrm{P}(t) &= \boldsymbol{v}_\mathrm{C}(t) + \boldsymbol{\omega}(t)\times\boldsymbol{b}_\mathrm{P} \\
\boldsymbol{a}_\mathrm{P}(t) &= \boldsymbol{a}_\mathrm{C}(t) + \alpha(t)\times\boldsymbol{b}_\mathrm{P} + \boldsymbol{\omega}(t)\times\left(\boldsymbol{\omega}(t)\times\boldsymbol{b}_\mathrm{P}\right)
\end{aligned}$$

where:

$$\boldsymbol{v}_\mathrm{P}(t) = \dot{\boldsymbol{s}}_\mathrm{P}(t),\ \ \boldsymbol{v}_\mathrm{C}(t) = \dot{\boldsymbol{s}}_\mathrm{C}(t),$$

$$\boldsymbol{a}_\mathrm{P}(t) = \dot{\boldsymbol{v}}_\mathrm{P}(t),\ \ \boldsymbol{a}_\mathrm{C}(t) = \dot{\boldsymbol{v}}_\mathrm{C}(t),$$

$$\boldsymbol{\omega}(t) = \dot{\boldsymbol{\Omega}}(t), \text{ and}$$

$$\alpha(t) = \dot{\boldsymbol{\omega}}(t).$$

Note that $\boldsymbol{b}_\mathrm{P}$ is time independent because it is in fixed position on the rigid body. The orientation operator $\boldsymbol{\Omega}$ converts directions from the body coordinate system to the space coordinate system, so that its derivatives $\boldsymbol{\omega}$ and α have space coordinate system values. The corresponding values in body coordinates are obtained with the inverse operator $\boldsymbol{\Omega}^\mathrm{T}$:

$$\begin{aligned}
\boldsymbol{\omega}_\mathrm{B}(t) &= \boldsymbol{\Omega}^\mathrm{T}(t)\boldsymbol{\omega}(t) \\
\alpha_\mathrm{B} &= \boldsymbol{\Omega}^\mathrm{T}(t)\alpha(t)
\end{aligned}$$

<u>Free motion case</u>:
In this case, the point P is moving with respect to both coordinate systems. In particular, $\boldsymbol{v}_{\mathrm{P},\boldsymbol{b}}(t) = \dot{\boldsymbol{b}}_\mathrm{P}(t)$ is not identically zero. We have:

$$\begin{aligned}
\boldsymbol{s}_\mathrm{P}(t) &= \boldsymbol{s}_\mathrm{C}(t) + \boldsymbol{\Omega}(t)\boldsymbol{b}_\mathrm{P}(t) \\
\boldsymbol{v}_\mathrm{P}(t) &= \boldsymbol{v}_\mathrm{C}(t) + \boldsymbol{v}_{\mathrm{P},\boldsymbol{b}}(t) + \boldsymbol{\omega}(t)\times\boldsymbol{b}_\mathrm{P}(t) \\
\boldsymbol{a}_\mathrm{P}(t) &= \boldsymbol{a}_\mathrm{C}(t) + \boldsymbol{a}_{\mathrm{P},\boldsymbol{b}}(t) + \alpha(t)\times\boldsymbol{b}_\mathrm{P}(t) + \boldsymbol{\omega}(t)\times\left(\boldsymbol{\omega}(t)\times\boldsymbol{b}_\mathrm{P}(t)\right) + 2\boldsymbol{\omega}(t)\times\boldsymbol{v}_{\mathrm{P},\boldsymbol{b}}(t)
\end{aligned}$$

where:

$$\boldsymbol{a}_{\mathrm{P},\boldsymbol{b}}(t) = \dot{\boldsymbol{v}}_{\mathrm{P},\boldsymbol{b}}(t).$$

In the expression for $\boldsymbol{a}_\mathrm{P}(t)$ the term $\boldsymbol{\omega}(t)\times\left(\boldsymbol{\omega}(t)\times\boldsymbol{b}_\mathrm{P}(t)\right)$ is identified as the *centripetal acceleration* and the term $2\boldsymbol{\omega}(t)\times\boldsymbol{v}_{\mathrm{P},\boldsymbol{b}}(t)$ is identified as the *Coriolis acceleration*.

5 Rigid body dynamics

Consider a rigid body consisting of discrete particles P_i of mass m_i and/or volume elements $\boldsymbol{V}$ with mass density function[12] ρ. $\boldsymbol{s}_i(t)$ will denote the coordinate vector at time t of point P_i in a space coordinate system with respect to the orthonormal basis $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ and $\boldsymbol{b}_i$ will denote the same point in a body coordinate system with respect to the orthonormal basis $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ attached to the body (body-fixed). Because the body is rigid, coordinate vectors $\boldsymbol{b}_i$ are independent of time.

The *total mass* of the body is: $M = \sum_i m_i + \iiint_{\boldsymbol{V}} \rho \, dV$.

The *center of mass* is located at $\boldsymbol{s}_{\text{CM}}(t) = \dfrac{1}{M}\left(\sum_i m_i \boldsymbol{s}_i(t) + \iiint_{\boldsymbol{s}\in\boldsymbol{V}} \rho(\boldsymbol{s}(t))\boldsymbol{s}(t)\, d\boldsymbol{s}\right)$.

We further require that the origin of the body coordinate system coincide with the center of mass. As a consequence, in body coordinates, the center of mass is the body coordinate system zero vector:

$$\mathbf{0} = \frac{1}{M}\left(\sum_i m_i \boldsymbol{b}_i + \iiint_{\boldsymbol{b}\in\boldsymbol{V}} \rho(\boldsymbol{b})\boldsymbol{b}\, d\boldsymbol{b}\right)$$

If $\boldsymbol{\Omega}(t)$ is the cosine matrix of $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ with respect to $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, then

$$\boldsymbol{s}_i(t) = \boldsymbol{s}_{\text{CM}}(t) + \boldsymbol{\Omega}(t)\boldsymbol{b}_i \text{ and } \boldsymbol{b}_i = \boldsymbol{\Omega}^{\mathsf{T}}(t)\left[\boldsymbol{s}_i(t) - \boldsymbol{s}_{\text{CM}}(t)\right],$$

and velocities are given by"

$$\begin{aligned}\boldsymbol{v}_i(t) &= \dot{\boldsymbol{s}}_i(t) \\ &= \boldsymbol{v}(t) + \omega(t) \times (\boldsymbol{s}_i(t) - \boldsymbol{s}_{\text{CM}}(t)) \\ &= \boldsymbol{v}(t) + \boldsymbol{\Omega}(t)(\omega_{\text{B}}(t) \times \boldsymbol{b}_i)\end{aligned}$$

where $\boldsymbol{v}(t) = \dot{\boldsymbol{s}}_{\text{CM}}(t)$ and $\omega(t)$ is the rotational velocity in space coordinates.

[12] At a point P the mass density $\rho(P)$ may expressed as either function of a space coordinate $\rho_{\text{SPACE}}(\boldsymbol{s})$ or a body coordinate $\rho_{\text{BODY}}(\boldsymbol{b})$. This notational distinction will not employed as the context makes it clear which function is intended.

The *linear momentum* of a particle is defined by $\boldsymbol{p}_i = m\dot{\boldsymbol{s}}_i = m\boldsymbol{v}_i$. The *total linear momentum* of the body is:

$$\begin{aligned}
\boldsymbol{P}(t) &= \sum_i m_i \dot{\boldsymbol{s}}_i(t) + \iiint_{\boldsymbol{s}\in\boldsymbol{V}} \rho(\boldsymbol{s}(t))\dot{\boldsymbol{s}}(t)\, d\boldsymbol{s} \\
&= \sum_i m_i \left[\dot{\boldsymbol{s}}_{\text{CM}}(t) + \dot{\boldsymbol{\Omega}}(t)\boldsymbol{b}_i\right] + \iiint_{\boldsymbol{b}\in\boldsymbol{V}} \rho(\boldsymbol{b})\left[\dot{\boldsymbol{s}}_{\text{CM}}(t) + \dot{\boldsymbol{\Omega}}(t)\boldsymbol{b}\right] d\boldsymbol{b} \\
&= \left(\sum_i m_i + \iiint_{\boldsymbol{b}\in\boldsymbol{V}} \rho(\boldsymbol{b})\, d\boldsymbol{b}\right)\dot{\boldsymbol{s}}_{\text{CM}}(t) + \dot{\boldsymbol{\Omega}}(t)\left(\sum_i m_i\boldsymbol{b}_i + \iiint_{\boldsymbol{b}\in\boldsymbol{V}} \rho(\boldsymbol{b})\boldsymbol{b}\, d\boldsymbol{b}\right) \\
&= M\,\dot{\boldsymbol{s}}_{\text{CM}}(t) + \dot{\boldsymbol{\Omega}}(t)\mathbf{0} \\
&= M\,\boldsymbol{v}(t)
\end{aligned}$$

This result:

$$\boldsymbol{P} = M\boldsymbol{v}(t)$$

shows that total linear momentum is independent of orientation or rotational velocity.

If a point $\boldsymbol{b}$ with mass m is rotating about an axis with (scalar) rotational rate ω, in a circle of radius r, then its speed is $r\omega$ and its scalar linear momentum is $p = m(r\omega)$. The scalar angular momentum L of $\boldsymbol{b}$ is its scalar linear momentum multiplied by the length r of its moment arm: $L = rp = r^2 m\omega$. The term in this expression that depends only on the geometric distribution of mass with respect to the rotation axis is the scalar moment of inertia $I = r^2 m$. We then have $L = I\omega$. If the rotational axis is determined by a unit vector $\boldsymbol{n}$ and if the angle between $\boldsymbol{n}$ and $\boldsymbol{b}$ is φ, then we note that the length of the moment arm is $r = \|\boldsymbol{b}\|\sin\varphi = \|\boldsymbol{b}\times\boldsymbol{n}\|$. This motivates the following definitions.

The rotational momentum of particle P_i is defined as $\boldsymbol{L}_i(t) = \boldsymbol{b}_i \times \boldsymbol{p}_i(t)$. Note that the rotational momentum is a vector quantity that is perpendicular to both the moment arm $\boldsymbol{b}_i$ and the momentum vector $\boldsymbol{p}_i$. The rotational momentum $\boldsymbol{L}_i$ may also be expressed directly in terms of the rotational velocity:
$\boldsymbol{L}_i(t) = \boldsymbol{b}_i \times \boldsymbol{p}_i(t) = \boldsymbol{b}_i \times m_i\boldsymbol{v}(t) = m_i\boldsymbol{b}_i \times (\boldsymbol{\omega}_{\text{B}}(t)\times\boldsymbol{b}_i)$.

The total *rotational momentum* of the body is defined as:

$$\boldsymbol{L}_{\text{B}}(t) = \sum_i m_i\boldsymbol{b}_i \times (\boldsymbol{\omega}_{\text{B}}(t)\times\boldsymbol{b}_i) + \iiint \rho(\boldsymbol{b})\boldsymbol{b}\times(\boldsymbol{\omega}_{\text{B}}(t)\times\boldsymbol{b})\, d\boldsymbol{b}$$

The subscript B indicates that the vector values of $\boldsymbol{L}_{\text{B}}(t)$ are represented in body coordinates. Since $\boldsymbol{\omega}_{\text{B}}(t)$ varies in time, all the summation and integration in the above expression needs to be re-evaluated at each time t. However, the above expression is linear in $\boldsymbol{\omega}_{\mathrm{B}}(t)$ and, as shown in Appendix D, it may be factored out to the equivalent form:

$$\boldsymbol{L}_{\mathrm{B}}(t) = \boldsymbol{I}_{\otimes\mathrm{B}}\boldsymbol{\omega}_{\mathrm{B}}(t)$$

where $\boldsymbol{I}_{\otimes\mathrm{B}}$ is the matrix defined as:

$$\boldsymbol{I}_{\otimes\mathrm{B}} = \sum_i m_i\left[\left(\boldsymbol{b}_i \bullet \boldsymbol{b}_i\right)\boldsymbol{I}_{3\times3} - \boldsymbol{b}_i \otimes \boldsymbol{b}_i\right] + \iiint_{\boldsymbol{V}} \rho(\boldsymbol{b})\left[\left(\boldsymbol{b}\bullet\boldsymbol{b}\right)\boldsymbol{I}_{3\times3} - \boldsymbol{b}\otimes\boldsymbol{b}\right]d\boldsymbol{b}$$

This matrix is the *moment of inertia tensor*[13]. The importance of this matrix is that it is time independent and needs to be computed only once and reduces the computation of $\boldsymbol{L}_{\mathrm{B}}(t)$ to nine multiplications and six additions. Appendix D expands $\boldsymbol{I}_{\otimes\mathrm{B}}$ in terms of body coordinate components:

$$\boldsymbol{I}_{\otimes\mathrm{B}} = \begin{pmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{pmatrix}$$

where:

$$I_{11} = \sum_i m_i\left(b_{i,2}^2 + b_{i,3}^2\right) + \iiint \rho\left(b_1,b_2,b_3\right)\left(b_2^2 + b_3^2\right)d\boldsymbol{b}$$

$$I_{22} = \sum_i m_i\left(b_{i,1}^2 + b_{i,3}^2\right) + \iiint \rho\left(b_1,b_2,b_3\right)\left(b_1^2 + b_3^2\right)d\boldsymbol{b}$$

$$I_{33} = \sum_i m_i\left(b_{i,1}^2 + b_{i,2}^2\right) + \iiint \rho\left(b_1,b_2,b_3\right)\left(b_1^2 + b_2^2\right)d\boldsymbol{b}$$

$$I_{12} = -\sum_i m_i b_{i,1} b_{i,2} - \iiint \rho\left(b_1,b_2,b_3\right)\left(b_1 b_2\right)d\boldsymbol{b}$$

$$I_{23} = -\sum_i m_i b_{i,2} b_{i,3} - \iiint \rho\left(b_1,b_2,b_3\right)\left(b_2 b_3\right)d\boldsymbol{b}$$

$$I_{13} = -\sum_i m_i b_{i,1} b_{i,3} - \iiint \rho\left(b_1,b_2,b_3\right)\left(b_1 b_3\right)d\boldsymbol{b}$$

$$\boldsymbol{b}_i = \left(b_{i,1}, b_{i,2}, b_{i,3}\right)^{\mathrm{T}}$$

$$\boldsymbol{b} = \left(b_1, b_2, b_3\right)^{\mathrm{T}}$$

The coordinate component values depend on the choice of the basis for the body coordinate system. The only constraint[14] imposed on the basis is that it is orthonormal

[13] The expression for $\boldsymbol{I}_{\otimes\mathrm{B}}$ is bilinear in body coordinates and may therefore be regarded as a tensor of order 2.

[14] Some applications impose additional constraints. For example, in the IEEE 1278.1-1995 standard the first axis in the entity coordinates system must point forward.

with its origin at the center of mass. The matrix $\boldsymbol{I}_{\otimes\text{B}}$ is symmetric, thus there exists some basis satisfying the constraint which will diagonalize the matrix and put the diagonal elements (the eigenvalues) in increasing order. That is, there exists a choice of basis for which:

$$\boldsymbol{I}_{\otimes\text{B}} = \begin{pmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{pmatrix}, \text{ and } I_{11} \le I_{22} \le I_{33} .$$

The coordinate axes of this basis are called the *principal axes*[15] and the diagonal elements I_{11}, I_{22}, and I_{33} are called the *principal moments of inertia*. The use of this basis reduces the computation of $\boldsymbol{L}_{\text{B}}(t)$ to three multiplications.

The total rotational momentum in space coordinates is given by:

$$\boldsymbol{L}(t) = \boldsymbol{I}_{\otimes}(t)\boldsymbol{\omega}(t).$$

In space coordinates, the moment of inertia tensor is time dependent and is computed as:

$$\boldsymbol{I}_{\otimes}(t) = \boldsymbol{\Omega}(t)\boldsymbol{I}_{\otimes,\text{B}}\boldsymbol{\Omega}^{\text{T}}(t)$$

If, in the space coordinate system, the external force acting on particle P_i is $\boldsymbol{F}_i(t)$ and/or the external force acting on a volume element is $\boldsymbol{f}(\boldsymbol{s},t)$, then the total force acting on the body is:

$$\boldsymbol{F}(t) = \sum_i \boldsymbol{F}_i(t) + \iiint_{\boldsymbol{s}\in V} \boldsymbol{f}(\boldsymbol{s},t)\,d\boldsymbol{s} .$$

If these forces are expressed in the body coordinate system as $\boldsymbol{F}_{i\text{B}}(t)$ and $\boldsymbol{f}_{\text{B}}(\boldsymbol{b},t)$, then the total force acting on the body is:

$$\boldsymbol{F}_{\text{B}}(t) = \sum_i \boldsymbol{F}_{i\text{B}}(t) + \iiint_{\boldsymbol{b}\in V} \boldsymbol{f}_{\text{B}}(\boldsymbol{b},t)\,d\boldsymbol{b} .$$

The analogue for external rotational force is *torque* and it is defined as:

$$\boldsymbol{\tau}(t) = \sum_i \left(\boldsymbol{s}_{\text{CM}}(t) - \boldsymbol{s}_i(t)\right) \times \boldsymbol{F}_i(t) + \iiint_V \left(\boldsymbol{s}_{\text{CM}}(t) - \boldsymbol{s}(t)\right) \times \boldsymbol{f}(\boldsymbol{s},t)\,d\boldsymbol{s}$$

$$\boldsymbol{\tau}_{\text{B}}(t) = \sum_i \boldsymbol{b}_i \times \boldsymbol{F}_{i\text{B}}(t) + \iiint_V \boldsymbol{b} \times \boldsymbol{f}_{\text{B}}(\boldsymbol{b},t)\,d\boldsymbol{b}$$

[15] Not to be confused with principal rotations.

In Newtonian physics the momentum of a particle is preserved unless an external force acts on it in which case the relationship between force and momentum is:

$$\boldsymbol{F}_i(t) = \frac{d}{dt}\boldsymbol{P}_i(t)$$

It follows that the relationship between total force and total momentum is:

$$\boldsymbol{F}(t) = \frac{d}{dt}\boldsymbol{P}(t) = M\frac{d}{dt}\boldsymbol{v}(t).$$

Similarly for torque we have:

$$\begin{aligned}
\boldsymbol{\tau}_{\mathrm{B}}(t) &= \sum \boldsymbol{b}_i \times \boldsymbol{F}_{\mathrm{B}i}(t) + \iiint \boldsymbol{b} \times \boldsymbol{f}_{\mathrm{B}}(\boldsymbol{b},t)\,d\boldsymbol{b} \\
&= \sum \boldsymbol{b}_i \times \frac{d}{dt}\boldsymbol{p}_i(t) + \iiint \boldsymbol{b} \times \frac{d}{dt}\boldsymbol{f}_{\mathrm{B}}(\boldsymbol{b},t)\,d\boldsymbol{b} \\
&= \frac{d}{dt}\left[\sum \boldsymbol{b}_i \times \boldsymbol{p}_i(t) + \iiint \rho(\boldsymbol{b})\,\boldsymbol{b} \times \left(\boldsymbol{\omega}_{\mathrm{B}}(t) \times \boldsymbol{b}\right)d\boldsymbol{b}\right] \\
&= \frac{d}{dt}\boldsymbol{L}_{\mathrm{B}}(t)
\end{aligned}$$

In term of rotational inertia we have:

$$\boldsymbol{\tau}(t) = \frac{d}{dt}\boldsymbol{L}(t) = \boldsymbol{I}_{\otimes}\frac{d}{dt}\boldsymbol{\omega}(t).$$

6 Use cases

6.1 DIS Euler angles

This use case is illustrated with the problem of converting aircraft orientation, as indicated by its onboard inertial system, to Tait-Bryan angles with respect to WGS 84 Geocentric (DIS Euler angles).

Consider an aircraft that at time t_0 is stationary on an airfield. Its inertial system is initialized so that the artificial horizon is level with the ground and the instrument panel compass north points in the direction of local North. For convenience, its location at t_0 shall be called ground zero. The resulting inertial system readouts of roll, pitch and yaw now indicate the orientation of the aircraft with respect to a Local-centric Euclidian Frame with origin at ground zero, and x-, y- and z-axes pointing local north, east and down respectively. This linear-space frame is denoted by $\mathbf{E}_0$. At some subsequent time, the aircraft taxies to the runway, takes off and maneuvers, and at time t_1 the roll, pitch and yaw are read out. These values are to be converted to DIS Euler angles.

The t_1 roll, pitch and yaw values correspond to the orientation of one linear-space frame with respect to another. One is the entity space (the Euclidean frame used for the aircraft), see Figure 7. The other is the $\mathbf{E}_0$ frame. Thus the roll, pitch and yaw values are the Tait-Bryan angles representation of the orientation of the aircraft coordinate frame with respect to the $\mathbf{E}_0$ frame:

$$\boldsymbol{\Omega}(t)_{\mathbf{Aircraft}\to\mathbf{E0}}.$$

Let **L** denote a range coordinate frame consisting of Local Tangent Frame Euclidean SRF. If we assume that the origin of **L** is near, or the same as, ground zero (the origin of $\mathbf{E}_0$), then $\mathbf{E}_0$ and **L** are related as shown in the Table below:

Axis	$\mathbf{E}_0$ coordinate frame	local tangent frame **L**
x	points to local North	points to local East
y	points to local East	points to local North
z	points to local down	points to local up

This relationship is expressed as the orientation $\boldsymbol{\Omega}_{\mathbf{E0}\to\mathbf{L}}$ of $\mathbf{E}_0$ with respect to **L**. Some representations of $\boldsymbol{\Omega}_{\mathbf{E0}\to\mathbf{L}}$ are:

Axis-angle: $(\boldsymbol{n},\theta) = \left(\left(\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}} \quad 0\right), \pi\right)$

Tait-Bryan: $(\phi, \theta, \psi) = \left(0, \pi, {}^{-\pi}\!/_{2}\right)$

Matrix: $\boldsymbol{\Omega}_{\mathbf{E0}\to\mathbf{L}} = \boldsymbol{\Omega}_{\boldsymbol{x}}(0)\,\boldsymbol{\Omega}_{\boldsymbol{y}}(\pi)\,\boldsymbol{\Omega}_{\boldsymbol{z}}\left(-{}^{\pi}\!/_{2}\right) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix}$

Quaternion: $\boldsymbol{\Omega}_{\mathbf{E0}\to\mathbf{L}} = \left(0, \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \end{pmatrix}^{\mathrm{T}}\right)$

Let **W** denote the WGS 84 geocentric SRF. The orientation of **L** with respect **W** is denoted as:

$$\boldsymbol{\Omega}_{\mathbf{L}\to\mathbf{W}}\,.$$

The SRM specifies the computation of the matrix representation of this orientation based on the WGS 84 geodetic coordinate for the origin of **L**.

The orientation of the aircraft with respect to **W** is then given by:

$$\boldsymbol{\Omega}(t)_{\mathbf{Aircraft}\to\mathbf{W}} = \boldsymbol{\Omega}_{\mathbf{L}\to\mathbf{W}} \circ \boldsymbol{\Omega}_{\mathbf{E0}\to\mathbf{L}} \circ \boldsymbol{\Omega}(t)_{\mathbf{Aircraft}\to\mathbf{E0}} \text{ at time } t.$$

In this equation the $\boldsymbol{\Omega}$ symbols are orientation operators. The operator compositions indicated by $\circ$ are matrix multiplications in the case of matrix representation, or quaternion multiplication in the case of quaternion representation, etc.

The DIS Euler roll, pitch and yaw values at time t are then just the Tait-Bryan angles for the orientation $\boldsymbol{\Omega}(t)_{\mathbf{Aircraft}\to\mathbf{W}}$ in the Euler angle x–y–z orientation convention.

6.2 Rigid body integration of state

In the notation of section 5, define the state of a rigid body system at time t as the ensemble of the location of the center of mass, the orientation, and the linear and angular momentums:

$$\text{Ò}(t) \equiv \left\{\boldsymbol{s}_{\mathrm{CM}}(t), \boldsymbol{\Omega}(t), \boldsymbol{P}(t), \boldsymbol{L}(t)\right\}$$

The problem is to compute $\text{Ò}(t+\Delta t)$ for some small time increment Δt given the previous state $\text{Ò}(t)$ and system specific functions for total force and torque. In the most general case, force and torque are functions of time and the state variables. That is:

$$\boldsymbol{F}(t) = \boldsymbol{F}\left(t, \text{Ò}(t)\right)$$
$$\boldsymbol{\tau}(t) = \boldsymbol{\tau}\left(t, \text{Ò}(t)\right)$$

This problem has several applications. For example, in distributed simulations, entity states are locally "dead reckoned" in time steps Δt until authoritative data has been distributed. In computer generated animations, object states are integrated in steps of Δt equal to the frame rate of the animation and covering the time interval of a scene from beginning to end.

One of many approaches to this problem, is to realize the integration step by setting each state variable $\boldsymbol{X}(t+\Delta t)$ to its approximate value $\boldsymbol{X}(t)+\Delta t\,\frac{d\boldsymbol{X}(t)}{dt}$, where $\boldsymbol{X} = \boldsymbol{s}_{\mathrm{CM}}$, $\boldsymbol{\Omega}$, $\boldsymbol{P}$, or $\boldsymbol{L}$. The error of this approximation decreases as Δt approaches zero. In particular Ò$(t+\Delta t)$ may be approximated by setting:

$$\begin{aligned}
\boldsymbol{s}_{\mathrm{CM}}(t+\Delta t) &= \boldsymbol{s}_{\mathrm{CM}}(t) + \frac{\Delta t}{M}\boldsymbol{P}(t) \\
\boldsymbol{\Omega}(t+\Delta t) &= \boldsymbol{\Omega}(t) + \Delta t\,\boldsymbol{S}_{\boldsymbol{\omega}(t)}\boldsymbol{\Omega}(t) \\
\boldsymbol{P}(t+\Delta t) &= \boldsymbol{P}(t) + \Delta t\,\boldsymbol{F}(t) \\
\boldsymbol{L}(t+\Delta t) &= \boldsymbol{L}(t) + \Delta t\,\boldsymbol{\tau}(t)
\end{aligned}$$

In the expression for $\boldsymbol{\Omega}$ the value $\boldsymbol{\omega}(t)$ is required to determine the corresponding skew-symmetric matrix $\boldsymbol{S}_{\boldsymbol{\omega}(t)}$. That requires two auxiliary computations.

$$\begin{aligned}
\boldsymbol{I}_{\otimes}^{-1}(t) &= \boldsymbol{\Omega}(t)\,\boldsymbol{I}_{\otimes,\mathrm{B}}^{-1}\boldsymbol{\Omega}^{\mathrm{T}}(t) \\
\boldsymbol{\omega}(t) &= \boldsymbol{I}_{\otimes}^{-1}(t)\,\boldsymbol{L}(t)
\end{aligned}$$

The computation for $\boldsymbol{\Omega}(t+\Delta t)$, like the other computations, is approximate. The approximate value may fail the criteria for an orientation matrix ($\det\boldsymbol{\Omega} = 1$, and $\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{\Omega} = \boldsymbol{I}_{3\times 3}$), or for a unit quaternion. This will lead to undesirable results in subsequent iterations, therefore the approximate $\boldsymbol{\Omega}(t+\Delta t)$ value needs to be adjusted to satisfy the criteria. If the $\boldsymbol{\Omega}$ operator is represented with a matrix, this adjustment can be computationally expensive. This is one reason for the popular use of unit quaternions. A quaternion is adjusted to a unit quaternion simply by dividing by its scalar modulus.

7 References

[1]
ISO/IEC 18026:2006(E) - Information technology — Spatial Reference Model (SRM), International Standard ISO/IEC 18026,
http://standards.iso.org/ittf/PubliclyAvailableStandards/

[2]
IEEE Standard for Distributed Interactive Simulation – Application Protocols, Std 1278.1-1995 standard.

[3]
David Eberly, *Rotation Representations and Performance Issues*, January 2002, Magic Software, Inc.

[4]
Ken Shoemake, *Animating rotation with quaternion curves*, Proceedings of the 12th annual conference on Computer graphics and interactive techniques table of contents, pp245 – 254,1985 ACM

[5]
James Diebel, *Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors*, Stanford University, California, diebel@stanford.edu, 20 October 2006

Appendices

Appendix A – Properties of the vector cross product

Definition:

$$\boldsymbol{u} \times \boldsymbol{v} = \left(u_2v_3 - u_3v_2,\, u_3v_1 - u_1v_3,\, u_1v_2 - u_2v_1\right)^{\mathrm{T}}$$

Note that $\boldsymbol{v} \times \boldsymbol{u} = -\boldsymbol{u} \times \boldsymbol{v}$ so that the cross production is not a commutative operation.

$$\|\boldsymbol{u} \times \boldsymbol{v}\| = \|\boldsymbol{u}\| \|\boldsymbol{v}\| \left|\sin\theta\right|$$

where θ is the angle between the two vectors.

The following identity is called Lagrange's formula:

$$\boldsymbol{u} \times (\boldsymbol{v} \times \boldsymbol{w}) = (\boldsymbol{u} \bullet \boldsymbol{w})\boldsymbol{v} - (\boldsymbol{u} \bullet \boldsymbol{v})\boldsymbol{w}$$

The cross product can be computed as a matrix multiplication. For each vector $\boldsymbol{u}$ there corresponds a skew-symmetric matrix $\boldsymbol{S}_{\boldsymbol{u}} = \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix}$ such that $\boldsymbol{u} \times \boldsymbol{v} = \boldsymbol{S}_{\boldsymbol{u}}\boldsymbol{v}$.

A special case of Lagrange's formula is:

$$\boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{w}) = (\boldsymbol{u} \bullet \boldsymbol{w})\boldsymbol{u} - (\boldsymbol{u} \bullet \boldsymbol{u})\boldsymbol{w}.$$

If $\boldsymbol{w} = (w_1, w_2, w_3)^{\mathrm{T}}$, $\boldsymbol{u} = (u_1, u_2, u_3)^{\mathrm{T}}$, then:

$$\begin{aligned}
(\boldsymbol{u} \bullet \boldsymbol{w})\boldsymbol{u} &= (u_1w_1 + u_2w_2 + u_3w_3)\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \\
&= \begin{pmatrix} u_1u_1w_1 + u_1u_2w_2 + u_1u_3w_3 \\ u_2u_1w_1 + u_2u_2w_2 + u_2u_3w_3 \\ u_3u_1w_1 + u_3u_2w_2 + u_3u_3w_3 \end{pmatrix} \\
&= \begin{pmatrix} u_1u_1 & u_1u_2 & u_1u_3 \\ u_2u_1 & u_2u_2 & u_2u_3 \\ u_3u_1 & u_3u_2 & u_3u_3 \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} \\
&= (\boldsymbol{u} \otimes \boldsymbol{u})\boldsymbol{w}
\end{aligned}$$

It follows that:

$$\begin{aligned} \boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{w}) &= (\boldsymbol{u} \otimes \boldsymbol{u})\boldsymbol{w} - (\boldsymbol{u} \bullet \boldsymbol{u})\boldsymbol{w} \\ &= \left[(\boldsymbol{u} \otimes \boldsymbol{u}) - (\boldsymbol{u} \bullet \boldsymbol{u})\boldsymbol{I}_{3\times 3}\right]\boldsymbol{w} \end{aligned}$$

Since $\boldsymbol{u} \times \boldsymbol{w} = -\boldsymbol{w} \times \boldsymbol{u}$ we also have:

$$\boldsymbol{u} \times (\boldsymbol{w} \times \boldsymbol{u}) = \left[(\boldsymbol{u} \bullet \boldsymbol{u})\boldsymbol{I}_{3\times 3} - \boldsymbol{u} \otimes \boldsymbol{u}\right]\boldsymbol{w}$$

Also, substituting $\boldsymbol{u} \times \boldsymbol{v} = \boldsymbol{S_u v}$, we have $\boldsymbol{u} \times (\boldsymbol{u} \times \boldsymbol{w}) = \boldsymbol{u} \times (\boldsymbol{S_u v}) = \boldsymbol{S_u}(\boldsymbol{S_u v}) = \boldsymbol{S}_{\boldsymbol{u}}^2 \boldsymbol{v}$.
Therefore:

$$\boldsymbol{S}_{\boldsymbol{u}}^2 = \left[(\boldsymbol{u} \otimes \boldsymbol{u}) - (\boldsymbol{u} \bullet \boldsymbol{u})\boldsymbol{I}_{3\times 3}\right].$$

Appendix B – Derivation of Rodrigues' rotation formula

Let $\boldsymbol{n}$ be a unit vector and θ a rotation angle. The point $\boldsymbol{r}$ is rotated around the axis determined by $\boldsymbol{n}$ through angle θ to the rotated point $\boldsymbol{r}'$. To compute $\boldsymbol{r}'$ in terms of $\boldsymbol{n}, \theta,$ and $\boldsymbol{r}'$, consider first the special case of a point $\boldsymbol{s}$ that is perpendicular to $\boldsymbol{n}$. A unit vector $\boldsymbol{m}$ that is perpendicular to both $\boldsymbol{s}$ and $\boldsymbol{n}$ is given by:

$$\boldsymbol{m} = \frac{1}{\|\boldsymbol{n} \times \boldsymbol{s}\|}\boldsymbol{n} \times \boldsymbol{s} = \frac{1}{\|\boldsymbol{s}\|}\boldsymbol{n} \times \boldsymbol{s} \text{ since} \|\boldsymbol{n} \times \boldsymbol{s}\| = \sin\left(\pi/2\right)\|\boldsymbol{n}\|\|\boldsymbol{s}\| = \|\boldsymbol{s}\|.$$

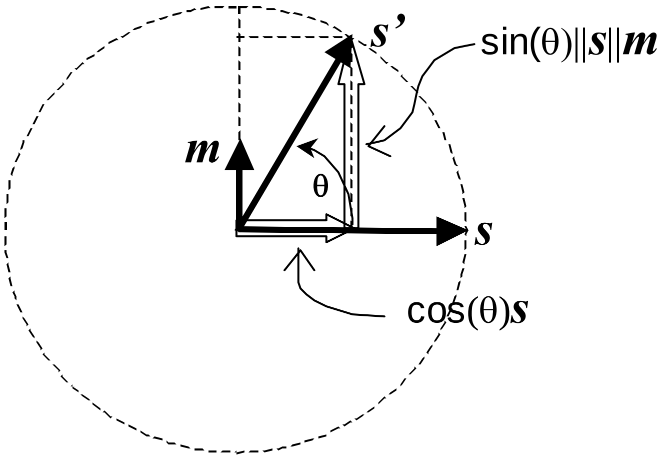


Figure B.1

The point $\boldsymbol{s}$ rotates to the point $\boldsymbol{s}'$ in the plane spanned by $\boldsymbol{m}$ and $\boldsymbol{s}$. The right triangle in Figure B.1 has a hypotenuse of length $\|\boldsymbol{s}\|$ and sides of lengths $\sin(\theta)\|\boldsymbol{s}\|$ and $\cos(\theta)\|\boldsymbol{s}\|$. It follows that

$$\boldsymbol{s}' = \cos(\theta)\boldsymbol{s} + \sin(\theta)\|\boldsymbol{s}\|\boldsymbol{m} = \cos(\theta)\boldsymbol{s} + \sin(\theta)\boldsymbol{n}\times\boldsymbol{s}.$$

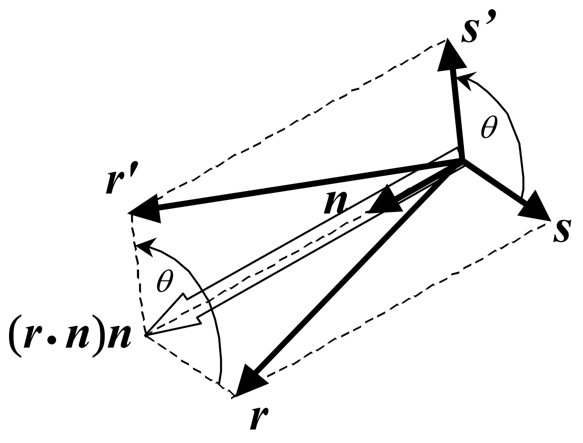


Figure B.2

In the general case, let $\boldsymbol{s} = \boldsymbol{r} - (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}$. Then, as shown in Figure B.2, $\boldsymbol{r}$ is the vector sum of $\boldsymbol{s}$ with its component in the vector $\boldsymbol{n}$ direction: $\boldsymbol{r} = (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} + \boldsymbol{s}$. Since $(\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}$ is on the $\boldsymbol{n}$ axis, it does not change under the rotation and so $\boldsymbol{r}' = (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} + \boldsymbol{s}'$. Substituting $\boldsymbol{s}' = \cos(\theta)\boldsymbol{s} + \sin(\theta)\boldsymbol{n}\times\boldsymbol{s}$ gives: $\boldsymbol{r}' = (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} + \cos(\theta)\boldsymbol{s} + \sin(\theta)\boldsymbol{n}\times\boldsymbol{s}$.

Substitute $\boldsymbol{n}\times\boldsymbol{s} = \boldsymbol{n}\times(\boldsymbol{r} - (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}) = \boldsymbol{n}\times\boldsymbol{r} - (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}\times\boldsymbol{n} = \boldsymbol{n}\times\boldsymbol{r}$, and substitute $\boldsymbol{s} = \boldsymbol{r} - (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}$ to get $\boldsymbol{r}' = (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} + \cos(\theta)(\boldsymbol{r} - (\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n}) + \sin(\theta)\boldsymbol{n}\times\boldsymbol{r}$. Simplifying the last result we have Rodrigues' rotation formula:

$$\boldsymbol{r}' = \cos(\theta)\boldsymbol{r} + (1-\cos(\theta))(\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} + \sin(\theta)\boldsymbol{n}\times\boldsymbol{r}$$

<u>Matrix form</u>:
As a consequence of Lagrange's formula, $\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}) = (\boldsymbol{n}\bullet\boldsymbol{r})\boldsymbol{n} - (\boldsymbol{n}\bullet\boldsymbol{n})\boldsymbol{r}$ and since $\boldsymbol{n}$ is a unit vector, $(\boldsymbol{n}\bullet\boldsymbol{n}) = 1$ and we have $(\boldsymbol{r}\bullet\boldsymbol{n})\boldsymbol{n} = \boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}) + \boldsymbol{r}$. Substituting for this term in Rodrigues' rotation formula yields the following alternate form:

$$\begin{aligned}\boldsymbol{r}' &= \cos(\theta)\boldsymbol{r} + (1-\cos(\theta))\left[\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}) + \boldsymbol{r}\right] + \sin(\theta)\boldsymbol{n}\times\boldsymbol{r} \\ &= \boldsymbol{r} + (1-\cos(\theta))\boldsymbol{n}\times(\boldsymbol{n}\times\boldsymbol{r}) + \sin(\theta)\boldsymbol{n}\times\boldsymbol{r}\end{aligned}$$

Substituting with the matrix form of the cross product gives the matrix form of the formula:

$\boldsymbol{r}' = \left[\boldsymbol{I}_{3\times3} + (1-\cos(\theta))\boldsymbol{S}_{\boldsymbol{n}}^2 + \sin(\theta)\boldsymbol{S}_{\boldsymbol{n}}\right]\boldsymbol{r}$, where $\boldsymbol{S}_{\boldsymbol{n}} = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$ is the skew matrix corresponding to $\boldsymbol{n}$.

Appendix C – Quaternion operators on 3D Euclidean space derivation

Given a unit quaternion $\boldsymbol{q} = (e_0, \boldsymbol{e})$ this appendix provides the derivation of the equality:

$$\boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^* = \left(0,\ \left((e_0^2 - \boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r} + 2(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e} + 2e_0\boldsymbol{e}\times\boldsymbol{r}\right)\right)$$

which shows that the quaternion operation $\boldsymbol{p} \mapsto \boldsymbol{qpq}^*$ transforms a "pure imaginary" quaternion to another "pure imaginary" quaternion and that the imaginary part $\boldsymbol{r}$ is transformed to $(e_0^2 - \boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r} + 2(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e} + 2e_0\boldsymbol{e}\times\boldsymbol{r}$ where $\boldsymbol{q} = (e_0, \boldsymbol{e})$.

Substitute $\boldsymbol{q} = (e_0, \boldsymbol{e})$:

$$\boldsymbol{q}(0,\boldsymbol{r})\boldsymbol{q}^* = (e_0,\boldsymbol{e})(0,\boldsymbol{r})(e_0,-\boldsymbol{e})$$

left multiply:

$$= (0 - \boldsymbol{e}\bullet\boldsymbol{r},\ e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r})(e_0,-\boldsymbol{e})$$

right multiply:

$$= \begin{pmatrix} -e_0\boldsymbol{e}\bullet\boldsymbol{r} - (e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r})\bullet(-\boldsymbol{e}), \\ \qquad (e_0(e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r}) + (-\boldsymbol{e}\bullet\boldsymbol{r})(-\boldsymbol{e}) + (e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r})\times(-\boldsymbol{e})) \end{pmatrix}$$

$$= \begin{pmatrix} -e_0\boldsymbol{e}\bullet\boldsymbol{r} + (e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r})\bullet\boldsymbol{e}, \\ \qquad (e_0(e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r}) + (\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e} - (e_0\boldsymbol{r} + \boldsymbol{e}\times\boldsymbol{r})\times\boldsymbol{e}) \end{pmatrix}$$

distribute terms:

$$=\begin{pmatrix} -e_0\boldsymbol{e}\bullet\boldsymbol{r}+e_0\boldsymbol{r}\bullet\boldsymbol{e}+(\boldsymbol{e}\times\boldsymbol{r})\bullet\boldsymbol{e}, \\ \left((e_0e_0\boldsymbol{r}+e_0\boldsymbol{e}\times\boldsymbol{r})+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-e_0\boldsymbol{r}\times\boldsymbol{e}-(\boldsymbol{e}\times\boldsymbol{r})\times\boldsymbol{e}\right)\end{pmatrix}$$

simplify:

$$=\left((\boldsymbol{e}\times\boldsymbol{r})\bullet\boldsymbol{e},\ \left(e_0^2\boldsymbol{r}+e_0\boldsymbol{e}\times\boldsymbol{r}+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-e_0\boldsymbol{r}\times\boldsymbol{e}-(\boldsymbol{e}\times\boldsymbol{r})\times\boldsymbol{e}\right)\right)$$

since: $(\boldsymbol{e}\times\boldsymbol{r})$ is perpendicular to $\boldsymbol{e}$, $(\boldsymbol{e}\times\boldsymbol{r})\bullet\boldsymbol{e}=0$,

and: $-(\boldsymbol{e}\times\boldsymbol{r})\times\boldsymbol{e}=\boldsymbol{e}\times(\boldsymbol{e}\times\boldsymbol{r})=+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-(\boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r}$ [Lagrange's formula]

$$=\left(0,\ \left(e_0^2\boldsymbol{r}+e_0\boldsymbol{e}\times\boldsymbol{r}+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-e_0\boldsymbol{r}\times\boldsymbol{e}+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-(\boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r}\right)\right)$$

since: $-e_0\boldsymbol{r}\times\boldsymbol{e}=+e_0\boldsymbol{e}\times\boldsymbol{r}$

$$=\left(0,\ \left(e_0^2\boldsymbol{r}+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}+2e_0\boldsymbol{e}\times\boldsymbol{r}+(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}-(\boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r}\right)\right)$$

simplify:

$$=\left(0,\ \left((e_0^2-\boldsymbol{e}\bullet\boldsymbol{e})\boldsymbol{r}+2(\boldsymbol{e}\bullet\boldsymbol{r})\boldsymbol{e}+2e_0\boldsymbol{e}\times\boldsymbol{r}\right)\right)$$

Appendix D – Moment of inertia

The definition of total angular momentum $\boldsymbol{L}$ is a summation and/or integration of terms in form $\boldsymbol{b}\times(\boldsymbol{\omega}\times\boldsymbol{b})$. As shown in Appendix A:

$$\boldsymbol{b}\times(\boldsymbol{\omega}\times\boldsymbol{b})=\left[(\boldsymbol{b}\bullet\boldsymbol{b})\boldsymbol{I}_{3\times3}-(\boldsymbol{b}\otimes\boldsymbol{b})\right]\boldsymbol{\omega}\,.$$

Substituting for this expression in the definition of total momentum yields:

$$\boldsymbol{L}(t)=\left\{\sum_i m_i\left[(\boldsymbol{b}_i\bullet\boldsymbol{b}_i)\boldsymbol{I}_{3\times3}-\boldsymbol{b}_i\otimes\boldsymbol{b}_i\right]+\iiint_V\rho(\boldsymbol{b})\left[(\boldsymbol{b}\bullet\boldsymbol{b})\boldsymbol{I}_{3\times3}-\boldsymbol{b}\otimes\boldsymbol{b}\right]d\boldsymbol{b}\right\}\boldsymbol{\omega}$$

$$=\boldsymbol{I}_{\otimes}\boldsymbol{\omega}$$

Expressed in coordinate components, the expression $\left[(\boldsymbol{b}\bullet\boldsymbol{b})\boldsymbol{I}_{3\times3}-(\boldsymbol{b}\otimes\boldsymbol{b})\right]$ is:

$$\left[\left(\boldsymbol{b}\bullet\boldsymbol{b}\right)\boldsymbol{I}_{3\times3}-\left(\boldsymbol{b}\otimes\boldsymbol{b}\right)\right]=\left[\begin{pmatrix} b_1^2+b_2^2+b_3^2 & 0 & 0 \\ 0 & b_1^2+b_2^2+b_3^2 & 0 \\ 0 & 0 & b_1^2+b_2^2+b_3^2 \end{pmatrix}-\begin{pmatrix} b_1^2 & b_1b_2 & b_1b_3 \\ b_2b_1 & b_2^2 & b_2b_3 \\ b_3b_1 & b_3b_2 & b_3^2 \end{pmatrix}\right]$$

$$=\begin{pmatrix} b_2^2+b_3^2 & -b_1b_2 & -b_1b_3 \\ -b_2b_1 & b_1^2+b_3^2 & -b_2b_3 \\ -b_3b_1 & -b_3b_2 & b_1^2+b_2^2 \end{pmatrix}$$

Appendix E – Matrix to axis-angle derivation

If $\boldsymbol{R}=\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ is a rotation matrix, then

$$\boldsymbol{R}=\left[\cos(\theta)\boldsymbol{I}_{3\times3}+\left(1-\cos(\theta)\right)\boldsymbol{n}\otimes\boldsymbol{n}+\sin(\theta)\boldsymbol{S}_{\boldsymbol{n}}\right]$$

for some unit vector and angle $(\boldsymbol{n},\theta)$ (see 3.2.1.1, alternate matrix form of Rodrigues' rotation formula).

The transpose operator is linear and since $\boldsymbol{I}_{3\times3}$ and $\boldsymbol{n}\otimes\boldsymbol{n}$ are symmetric and $\boldsymbol{S}_{\boldsymbol{n}}$ is skew-symmetric, it follows that:

$$\boldsymbol{R}-\boldsymbol{R}^{\mathrm{T}}=2\sin(\theta)\boldsymbol{S}_{\boldsymbol{n}}.$$

The trace operator is also linear and since $\mathrm{Trace}(\boldsymbol{S}_{\boldsymbol{n}})=0$ and $\mathrm{Trace}(\boldsymbol{n}\otimes\boldsymbol{n})=1$,

$$\mathrm{Trace}(\boldsymbol{R})=\cos(\theta)(3)+\left(1-\cos(\theta)\right)(1)+\sin(\theta)(0)=1+2\cos(\theta).$$

Therefore $\theta=\arccos\left(\left(\dfrac{\mathrm{Trace}(\boldsymbol{R})-1}{2}\right)\right),\ 0\le\theta\le\pi.$

If $0 < \theta < \pi$, then $\sin\theta > 0$ and

$$\boldsymbol{S_n} = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix} = \frac{1}{2\sin(\theta)}\left[\boldsymbol{R} - \boldsymbol{R}^{\mathsf{T}}\right]$$

$$= \frac{1}{2\sin(\theta)}\begin{pmatrix} 0 & a_{12}-a_{21} & a_{13}-a_{31} \\ a_{21}-a_{12} & 0 & a_{23}-a_{32} \\ a_{31}-a_{13} & a_{32}-a_{23} & 0 \end{pmatrix}$$

Therefore:

$$\boldsymbol{n} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} = \frac{1}{2\sin(\theta)}\begin{pmatrix} a_{32}-a_{23} \\ a_{13}-a_{31} \\ a_{21}-a_{12} \end{pmatrix}.$$

Alternatively, let $\boldsymbol{v} = \begin{pmatrix} a_{32}-a_{23} \\ a_{13}-a_{31} \\ a_{21}-a_{12} \end{pmatrix}$, and let $\boldsymbol{n} = \frac{1}{\|\boldsymbol{v}\|}\boldsymbol{v}$.

If $\theta = 0$, there is no rotation so $\boldsymbol{n}$ is indeterminate.

If $\theta = \pi$, then the alternate matrix form of Rodrigues' rotation formula reduces down to:

$$\boldsymbol{R} = \left[-\boldsymbol{I}_{3\times3} + 2\boldsymbol{n}\otimes\boldsymbol{n}\right]$$

$$= \begin{pmatrix} 2n_1^2-1 & 2n_1n_2 & 2n_1n_3 \\ 2n_2n_1 & 2n_2^2-1 & 2n_2n_3 \\ 2n_3n_1 & 2n_3n_2 & 2n_3^2-1 \end{pmatrix}$$

Therefore: $n_1^2 = (a_{11}+1)/2$, $n_2^2 = (a_{22}+1)/2$, and $n_3^2 = (a_{33}+1)/2$. Find the maximum of a_{11}, a_{22}, and a_{33} and use it to find one coordinate component of $\boldsymbol{n}$ and then derive the components from it.

Case: a_{11} is the maximum. Let $n_1 = \sqrt{(a_{11}+1)/2}$, $n_2 = \dfrac{a_{12}}{2n_1}$, $n_3 = \dfrac{a_{13}}{2n_1}$.

Case: a_{22} is the maximum. Let $n_2 = \sqrt{(a_{22}+1)/2}$, $n_1 = \dfrac{a_{12}}{2n_2}$, $n_3 = \dfrac{a_{32}}{2n_2}$.

Case: a_{33} is the maximum. Let $n_3 = \sqrt{(a_{33}+1)/2}$, $n_1 = \dfrac{a_{13}}{2n_3}$, $n_2 = \dfrac{a_{23}}{2n_3}$.

It is equivalent (multiply by the appropriate factor $2n_i$), but computationally more efficient, to compute $\boldsymbol{n}$ as follows:

Case: a_{11} is the maximum. Let $\boldsymbol{v} = \left(a_{11} + 1, a_{12}, a_{13}\right)^{\mathsf{T}}$.

Case: a_{22} is the maximum. Let $\boldsymbol{v} = \left(a_{21}, a_{22} + 1, a_{23}\right)^{\mathsf{T}}$.

Case: a_{33} is the maximum. Let $\boldsymbol{v} = \left(a_{31}, a_{32}, a_{33} + 1\right)^{\mathsf{T}}$.

Then: $\boldsymbol{n} = \dfrac{1}{\|\boldsymbol{v}\|}\boldsymbol{v}$.

INDEX