Australian Government
Department of Defence
Defence Science and
Technology Organisation

A Pseudo-Reversing Theorem for Rotation and its Application to Orientation Theory

Don Koks

Electronic Warfare and Radar Division

Defence Science and Technology Organisation

DSTO–TR–2675

ABSTRACT

We state and prove a useful theorem on manipulating rotation order which, while not new, is barely present in the literature. This theorem allows the order of a sequence of rotations to be reversed, provided that the sense of the axes of rotation is changed from "body" to "space fixed" or vice versa. We use the theorem to aid calculations in geodesy (constructing a local north–east–down coordinate system) and aerospace theory (relating yaw–pitch–roll rates to vehicle angular velocity). The new notation here sheds light generally on the field of orientation theory, as well as giving insight to standard terms relating to wind direction used for treating ship motion. Although we present our analyses in the style of a tutorial in the general subject of spatial orientation theory, there is new notation here, along with alternative and novel ways of treating problems that are often seen as difficult or obscure by practitioners. This report follows on from the 2005 DSTO report DSTO–TN–0640, but is completely self contained, and DSTO–TN–0640 need not be read beforehand.

Published by

DSTO Defence Science and Technology Organisation
PO Box 1500
Edinburgh, SA 5111, Australia

Telephone: (08) 7389 5555
Facsimile: (08) 7389 6567

© Commonwealth of Australia 2012
AR No. AR–015–246
March, 2012

A Pseudo-Reversing Theorem for Rotation and its Application to Orientation Theory

Executive Summary

This report is a much-evolved follow-on from the 2005 DSTO publication DSTO–TN–0640, "Using Rotations to Build Aerospace Coordinate Systems", that explained the construction of coordinate systems used in aerospace calculations. That report followed a step-by-step approach to implement its calculations. In the current report we rephrase those calculations in a more efficient language, while incorporating a very useful theorem that is known but almost absent from the literature. This "Pseudo-Reversing Theorem" allows the order of a sequence of rotations to be reversed, provided that the sense of the axes of rotation is changed from "body" to "space fixed" or vice versa. The current report is self contained, so that familiarity with the content of DSTO–TN–0640 is not necessary.

The current report places the theorem and the reworked examples of DSTO–TN–0640 into the greater context of orientation/rotation theory. We first introduce the theorem, then establish a solid mathematical language necessary for quantifying the orientation of an object. We cover the background of how to rotate a vector, using either a matrix or a quaternion. We then rework the examples in DSTO–TN–0640: constructing a local north–east–down set of axes from a given latitude and longitude, and calculating where a pilot must look to see a distant aircraft. We also make an extended revisit to the subject of conversions within the Distributed Interactive Simulation environment for handling orientation information, since this often causes problems to practitioners who must deal with several coordinate systems at once. We end the main report by showing how the Pseudo-Reversing Theorem can be used to simplify some of the concepts behind dead reckoning an object's changing orientation.

The report ends with an appendix that applies its notation and general approach to the task of constructing the appropriate course a ship must steer in order for the wind to appear to come from some given direction with some given speed. This is a nontrivial problem that is handled well in a novel way by the orientation-matrix language of this report, although its solution doesn't require the Pseudo-Reversing Theorem.

This page is intentionally blank.

Author

Don Koks
Electronic Warfare and Radar Division

Don Koks completed a doctorate in mathematical physics at Adelaide University in 1996, with a thesis describing the use of quantum statistical methods to analyse decoherence, entropy and thermal radiance in both the early universe and black hole theory. He holds a Bachelor of Science from the University of Auckland in pure and applied mathematics and physics, and a Master of Science in physics from the same university with a thesis in applied accelerator physics (proton-induced X ray and γ ray emission for trace element analysis). He has worked on the accelerator mass spectrometry programme at the Australian National University in Canberra, and in commercial Internet development.

Currently he is a Research Scientist with the Maritime Electronic Warfare Systems group in the Electronic Warfare and Radar Division at DSTO, specialising in radar signal processing, geospatial orientation concepts, and geolocation. He is the author of the book *Explorations in Mathematical Physics: the Concepts Behind an Elegant Language* (Springer, 2006).

This page is intentionally blank.

Contents

Appendices

This page is intentionally blank.

1 Introduction

Since publishing a DSTO report [1] covering the basics of rotation/orientation theory in 2005, I have received several emails from practitioners with various questions prompted by its subject matter. These questions suggest to me that a second edition of that report would be useful.

There are several reasons to update the 2005 report. The subject of 6 degree-of-freedom modelling is a comparatively modern one, requiring the use of fast computers to update an object's state sufficiently accurately to represent the real world usefully. Perhaps because of this, straightforward information about the subject has yet to appear in nonspecialised textbooks. The calculations in [1] were written to fill a perceived gap, having both pedagogical and logical content. But they were certainly longer than necessary. Only after writing that report did I become more aware of a known theorem that would have simplified some of its analysis. I think that this theorem is not given the significance it deserves in the literature; in fact, while some practitioners are aware of it, a statement of it with a discussion of its use is hard to find anywhere—perhaps because having no widely used name makes it difficult to search for. I have called it the Pseudo-Reversing Theorem here, or PRT. It has been called the Rodrigues Transposition Theorem in [2], but no provenance is given there for the choice of name, which doesn't appear to be used anywhere else. An example of the theorem is given in [3], which gives a proof in the context of orthogonal axes, although the theorem doesn't actually require such axes.

The Pseudo-Reversing Theorem can often be invoked to give a different pedagogical basis to the many analyses and recipes abounding in orientation theory that can, for some, seem quite opaque. Part of the subject's difficulty is due to a divide in the orientation community: some books base their analysis on the concept of an "active rotation", whereas others use a "passive rotation". I suggest that knowledge of the Pseudo-Reversing Theorem forms a good bridge between these two approaches, so that one can more easily appreciate the reason why both approaches have historically been used.

Confusion over whether a sequence of rotations is active or passive can result in that sequence being written in the wrong order, possibly with wrong signs of the angles turned through. But orientation theory is a subject in which a procedure's lack of validity will quickly become obvious when it's implemented on a computer. It does not seem to me that a great deal of pedagogical effort has found its way into the proofs of some of the subject's recipes; and when they work well, there is little incentive for anyone to highlight the places where the analyses are unclear and to provide an explanation for why, nonetheless, those analyses work. I have tried to address that in this report by explaining some representative analyses in detail. And although I use only active rotations here and in [1], I do discuss how passive rotations can be used, and where they fit into the general scheme of the subject.

Another reason for this follow-up report is to establish a good set of notation that helps simplify rotation/orientation analysis. I rework the examples in [1] using this notation. I have added an appendix that uses the notation to aid calculations of wind velocity as perceived by a ship. This is a standard task in navigation that is rendered more transparent by an appreciation of the difference between proper vectors and coordinate vectors, as discussed in Section 3.

Finally, Section 5.4 on "DIS conversion" was prompted by a question that I've been asked several times regarding the DIS example in Section 4.3 of [1], an example that has been useful for DIS practitioners. That original calculation was certainly correct, but readers were apt to confuse the roles of Earth-centred Earth-fixed and local north–east–down and so arrive at a wrong answer. To address this, I've rewritten the discussion of [1] differently and in far more detail here.

2 The Pseudo-Reversing Theorem

The following thought experiment encapsulates the core of the calculations in this report.

Consider the viewpoint of an audience watching the famous mime Marcel Marceau on stage. Marcel orientates his body in some way, and then rotates his hand about his wrist. We, the audience, seek to describe that hand's final spatial orientation from a knowledge of these two procedures. Orientating Marcel's body can be described by some action, an "operator", which we write as O and assume known. To incorporate the new orientation of Marcel's rotated hand, we can write another operator that rotates that hand about the wrist. But simply saying "the rotation was about the wrist" is useful mathematically only if we know where the wrist currently points. So we wish to specify the vector, independent of Marcel's body, along which his wrist points. The act of bodily orientating himself has changed the direction of Marcel's wrist, in which case we must distinguish between the *initial* direction (vector) of his wrist, here denoted by the subscript "wrist", and the *final* direction vector of his wrist, here denoted by the subscript "⟨wrist⟩".

If we write the operation of Marcel orientating himself bodily as O, and follow this with a rotation $R_{\langle \text{wrist} \rangle}$ of his hand about the *latest* direction of his wrist, then the entire procedure of orientating Marcel's hand is written

$$\text{final orientation} = O \to R_{\langle \text{wrist} \rangle} \,. \tag{2.1}$$

Now realise that Marcel can achieve exactly the same result by first rotating his hand about his wrist, and then orientating himself bodily as before. We still describe the bulk orientation with O, but we must now rotate the hand about a different vector: the *initial* direction of Marcel's wrist. Write this rotation as R_{wrist} to obtain

$$\text{same final orientation} = R_{\text{wrist}} \to O \,. \tag{2.2}$$

We have "almost" swapped the two procedures in the mathematical description of Marcel's movements, provided we understand the operator that rotates the hand to be different in each description. This simple result forms the core of what we'll soon call the Pseudo-Reversing Theorem:

$$O \to R_{\langle \text{wrist} \rangle} = R_{\text{wrist}} \to O \,. \tag{2.3}$$

In the descriptions to follow, we'll use a phrase such as "the latest snapshot of the vector describing Marcel's wrist" to reinforce the fact that there can be two wrist vectors being discussed, which refer to the direction of Marcel's wrist at different times in an orientation procedure. The vector "wrist" describes where his wrist was initially and is unchanging; the vector "⟨wrist⟩" describes where his wrist is now, and is subject to change as he orientates his body. We might consider the initial vector as being rotated to become the final vector, but it's useful to view the initial vector as set in stone, and a *snapshot* of this vector is then physically rotated to become the final vector.

More generally, consider a sequence S of rotations about the latest snapshots of vectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$. We wish to describe the following set of rotations. First, rotate snapshots of $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$ around $\boldsymbol{u}_1$ to give a new set labelled $\boldsymbol{u}_1', \boldsymbol{u}_2', \boldsymbol{u}_3'$ respectively. Now rotate snapshots of each of these vectors around $\boldsymbol{u}_2'$ to give a new set labelled $\boldsymbol{u}_1'', \boldsymbol{u}_2'', \boldsymbol{u}_3''$ respectively. Now rotate snapshots of each of these vectors around $\boldsymbol{u}_3''$ to give a new set labelled $\boldsymbol{u}_1''', \boldsymbol{u}_2''', \boldsymbol{u}_3'''$ respectively. Finally, rotate snapshots each of these vectors around $\boldsymbol{u}_1'''$ to give a new set labelled $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$ respectively. We can write the sequence of rotations between initial and final vectors as

$$\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\} \to R_{\boldsymbol{u}_1} \to R_{\boldsymbol{u}_2'} \to R_{\boldsymbol{u}_3''} \to R_{\boldsymbol{u}_1'''} \to \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}\,. \tag{2.4}$$

S is the sequence of four rotations taking initial to final vectors. But this linguistic and notational description of the sequence is tedious—and we almost certainly will not need the intermediate vectors—so instead we'll describe the same procedure as follows. Rotate snapshots of $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$ first around $\boldsymbol{u}_1$, then around the latest snapshot of $\boldsymbol{u}_2$, then around the latest snapshot of $\boldsymbol{u}_3$, and finally around the latest snapshot of $\boldsymbol{u}_1$, to give a new set $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$. We will write this as

$$\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\} \to R_1 \to R_{\langle 2\rangle} \to R_{\langle 3\rangle} \to R_{\langle 1\rangle} \to \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}\,. \tag{2.5}$$

The above discussion of Marcel Marceau encapsulated in (2.3) allows us to replace the $R_1 \to R_{\langle 2\rangle}$ in (2.5) with $R_2 \to R_1$. This doesn't affect the sense of the next vectors $\langle 3\rangle$ and $\langle 1\rangle$, since the procedures $R_1 \to R_{\langle 2\rangle}$ and $R_2 \to R_1$ have the same effect—they produce the same final orientation. In that case, (2.5) becomes

$$S = R_2 \to R_1 \to R_{\langle 3\rangle} \to R_{\langle 1\rangle}\,. \tag{2.6}$$

Now again use the same logic of applying (2.3): think of $R_2 \to R_1$ as the orientation O, and swap it with $R_{\langle 3\rangle}$, remembering to change $R_{\langle 3\rangle}$ to R_3:

$$S = R_3 \to R_2 \to R_1 \to R_{\langle 1\rangle}\,. \tag{2.7}$$

Finally apply the same idea again to arrive at

$$S = R_1 \to R_3 \to R_2 \to R_1\,. \tag{2.8}$$

Notice that we cannot combine $R_1 \to R_{\langle 1\rangle}$ in (2.7) into one rotation: R_1 refers to a rotation around $\boldsymbol{u}_1$, while $R_{\langle 1\rangle}$ refers to a rotation around the very latest "incarnation" of $\boldsymbol{u}_1$, which in general is a *completely* different vector to $\boldsymbol{u}_1$.

An inductive argument should make it clear that the same reversing procedure holds regardless of the length of the sequence. There can be repeated subscripts, and the initial set of vectors need not be mutually orthogonal. For any number of rotations, then, we have proved the *Pseudo-Reversing Theorem*:

$$R_1 \to R_{\langle 2\rangle} \to R_{\langle 3\rangle} \to R_{\langle 4\rangle} \to \ldots \;=\; \ldots \to R_4 \to R_3 \to R_2 \to R_1\,. \tag{2.9}$$

We can equally well express the right-hand side of (2.9) as a purely operator expression, in which case it uses no arrows because operators act from right to left:

$$\boxed{R_1 \to R_{\langle 2\rangle} \to R_{\langle 3\rangle} \to R_{\langle 4\rangle} \to \ldots \;=\; R_1\, R_2\, R_3\, R_4\, \ldots} \tag{2.10}$$

Equation (2.10) is a very useful statement of the Pseudo-Reversing Theorem. It says that when rotating about latest axes, we need only write down—from *left to right*—the sequence of rotations nominally around *latest* axes, but writing them as operators that rotate about *space-fixed* axes, which will then be automatically understood to act from right to left. We'll do this in the examples to follow.

2.1 A Related Scenario in Aerospace Kinematics

The above discussion of Marcel Marceau is actually central to quantifying aerospace kinematics, and in the interests of clarity, we will reword it using the more complex language of aircraft axes.

Suppose an aircraft is flying straight and level with its nose and wings pointing in some given directions relative to north, east, and down. The pilot executes three quick manoeuvres. First, a yaw through 10°, followed by a pitch around the wing through 20°, followed by a roll around the nose through 30°. Assuming the directions of north, east, and down haven't changed appreciably during the manoeuvres, what is the final aircraft orientation relative to north, east, and down?

It suffices to specify this orientation by giving coordinates that quantify the aircraft's three "body" *basis vectors* for its nose, starboard wing, and onboard-down direction. These vectors can be envisaged as arrows embedded in the aircraft's body. Call these pre-manoeuvre vectors "nose", "starboard wing", and "down", $\boldsymbol{n}_0, \boldsymbol{s}_0, \boldsymbol{d}_0$, respectively. Their coordinates relative to north, east, and down are specified in the scenario. We will be extra clear here by using the above laboured description of intermediate rotations once more. The first manoeuvre rotates these vectors around $\boldsymbol{d}_0$ to produce $\boldsymbol{n}_1, \boldsymbol{s}_1, \boldsymbol{d}_1$. (Of course, $\boldsymbol{d}_1 = \boldsymbol{d}_0$, but a good naming convention helps prevent confusion.) The second manoeuvre rotates $\boldsymbol{n}_1, \boldsymbol{s}_1, \boldsymbol{d}_1$ around $\boldsymbol{s}_1$ to produce $\boldsymbol{n}_2, \boldsymbol{s}_2, \boldsymbol{d}_2$. The third manoeuvre rotates $\boldsymbol{n}_2, \boldsymbol{s}_2, \boldsymbol{d}_2$ around $\boldsymbol{n}_2$ to produce $\boldsymbol{n}_3, \boldsymbol{s}_3, \boldsymbol{d}_3$. We require the coordinates of $\boldsymbol{n}_3, \boldsymbol{s}_3, \boldsymbol{d}_3$ relative to north, east, and down.[1] This sequence of three rotations can be written

$$\{\boldsymbol{n}_0, \boldsymbol{s}_0, \boldsymbol{d}_0\} \to R_{\boldsymbol{d}_0}(10^\circ) \to R_{\boldsymbol{s}_1}(20^\circ) \to R_{\boldsymbol{n}_2}(30^\circ) \to \{\boldsymbol{n}_3, \boldsymbol{s}_3, \boldsymbol{d}_3\}\,. \tag{2.11}$$

Each of these rotations requires computational effort because each rotation (except the first) requires a vector produced by the previous rotation, and so cannot be constructed in advance of the manoeuvres; the intermediate vectors must be calculated. This sort of calculation was done in [1]. Now, however, we can use the PRT to write the final result as the simpler sequence of three rotations solely around the *initial* basis vectors:

$$\{\boldsymbol{n}_0, \boldsymbol{s}_0, \boldsymbol{d}_0\} \to R_{\boldsymbol{n}_0}(30^\circ) \to R_{\boldsymbol{s}_0}(20^\circ) \to R_{\boldsymbol{d}_0}(10^\circ) \to \{\boldsymbol{n}_3, \boldsymbol{s}_3, \boldsymbol{d}_3\}\,. \tag{2.12}$$

This sequence is easier to implement than (2.11) because no intermediate basis vectors need be calculated.

The Pseudo-Reversing Theorem has connected these two descriptions of a sequence of rotations of the aircraft. Both rotation sequences are necessary and useful for application in geodesy and flight modelling. Rotations about carried-along axes closely match our physical experience: they are what a pilot flies; but such rotations are not always economical to describe mathematically because of the need to calculate intermediate vectors to implement the rotations. Rotations about space-fixed axes are mathematically economical because intermediate vectors need not be calculated, but these rotations do not match our physical experience, which can make them more difficult to visualise. (For example, a pilot performing aerial manoeuvres doesn't concentrate on rotating the aircraft about the east

[1] In practice, the final directions of north, east, and down can change during the scenario, but this has no real bearing on the discussion; we need only choose a coordinate system to express everything in, and the original north–east–down directions suffice as its axes. We could easily accommodate any change in the final directions of north, east, and down using Section 3.

direction; that would require a very complex interaction with the flight controls.) The PRT allows the best of both worlds: easy mathematics and easy visualisation. It says that any sequence of rotations in one description is converted to the equivalent sequence in the other by reversing the order of the rotations and changing their sense to that of the other description.

3 Language of Vectors and Rotations

Like its position, the orientation of an object must be specified relative to some unchanging reference orientation. *Euler's theorem of rotation* states that any orientation of an object can always be reached by applying a *single rotation* to that base orientation [4]. Hence we can quantify any orientation by writing down this single rotation. Alternatively, we might prefer to restrict all rotations to a prescribed set of axes, in which case more than one rotation will generally be needed, and we'll have to specify the order in which the rotations are performed.

These procedures necessitate rotating one vector about another, together with good notation that keeps track of the various coordinate sets that might be involved. We will begin with a *proper vector* $\boldsymbol{v}$, being an arrow that signifies a directed number such as velocity. Note the distinction between a proper vector—an arrow that might be attached to one point in space, such as those comprising the velocity field of a moving fluid, and a *position vector* (or *radius vector*), which is an arrow whose tail is anchored to the origin, with its head serving to quantify a *point* in space. For example, one position vector minus another position vector gives a proper vector. We'll consider this distinction further in Section 3.2.4.

A *coordinate vector* $[\boldsymbol{v}]_S$ is a column of numbers that quantifies $\boldsymbol{v}$ as a linear combination of a set of *basis (proper) vectors* S. A proper vector can be described using any one of an infinite number of bases $S, S', \dots$, being represented by coordinate vectors $[\boldsymbol{v}]_S, [\boldsymbol{v}]_{S'}, \dots$ respectively. These coordinate vectors are all distinct columns of numbers, each associated with its basis, but all describe the *same* proper vector $\boldsymbol{v}$. Proper vectors can be manipulated in the usual way by "adding" arrows, but a numerical representation demands a basis, with each proper vector represented by its appropriate coordinate vector in that basis. To avoid notational clutter, we'll use e.g. S to refer to a set of axes, and to the basis vectors of that set.

3.1 Notation of Coordinate Vectors

Orientation analyses simplify when the basis vectors are orthonormal (mutually orthogonal and of unit length). Although the PRT applies also to non-orthogonal axes, our geodesy/aerospace focus assumes orthonormal bases S and S', with basis vectors written as

$$S \text{ basis} = \{\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z\}, \quad S' \text{ basis} = \{\boldsymbol{e}_{x'}, \boldsymbol{e}_{y'}, \boldsymbol{e}_{z'}\}, \tag{3.1}$$

and so on. Now, because a proper vector can be written as a linear combination of basis vectors

$$\begin{aligned}\boldsymbol{v} &= v_x\boldsymbol{e}_x + v_y\boldsymbol{e}_y + v_z\boldsymbol{e}_z \\ &= v_{x'}\boldsymbol{e}_{x'} + v_{y'}\boldsymbol{e}_{y'} + v_{z'}\boldsymbol{e}_{z'},\end{aligned} \tag{3.2}$$

the coordinate vectors with respect to bases S, S' are

$$[\boldsymbol{v}]_S = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}\cdot\boldsymbol{e}_x \\ \boldsymbol{v}\cdot\boldsymbol{e}_y \\ \boldsymbol{v}\cdot\boldsymbol{e}_z \end{bmatrix}, \quad [\boldsymbol{v}]_{S'} = \begin{bmatrix} v_{x'} \\ v_{y'} \\ v_{z'} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}\cdot\boldsymbol{e}_{x'} \\ \boldsymbol{v}\cdot\boldsymbol{e}_{y'} \\ \boldsymbol{v}\cdot\boldsymbol{e}_{z'} \end{bmatrix}. \tag{3.3}$$

Central to orientation theory is the description of one coordinate system's orientation relative to another. For example, we might require the orientation of an aircraft in the Earth-centred Earth-fixed coordinate system (ECEF) defined in Section 5.2. The position of a point such as the aircraft axes' origin in these ECEF coordinates is not relevant to specifying its orientation, so the relevant proper vectors can be envisaged as attached to the aircraft, and are unchanged if the aircraft is merely translated. The orientation of S in S' is quantified by an *orientation matrix* $\mu^S_{S'}$, whose columns are the S' coordinate vectors of the S basis vectors:[2]

$$\mu^S_{S'} \equiv \Big[[\boldsymbol{e}_x]_{S'} \;\; [\boldsymbol{e}_y]_{S'} \;\; [\boldsymbol{e}_z]_{S'} \Big] = \begin{bmatrix} \boldsymbol{e}_x\cdot\boldsymbol{e}_{x'} & \boldsymbol{e}_y\cdot\boldsymbol{e}_{x'} & \boldsymbol{e}_z\cdot\boldsymbol{e}_{x'} \\ \boldsymbol{e}_x\cdot\boldsymbol{e}_{y'} & \boldsymbol{e}_y\cdot\boldsymbol{e}_{y'} & \boldsymbol{e}_z\cdot\boldsymbol{e}_{y'} \\ \boldsymbol{e}_x\cdot\boldsymbol{e}_{z'} & \boldsymbol{e}_y\cdot\boldsymbol{e}_{z'} & \boldsymbol{e}_z\cdot\boldsymbol{e}_{z'} \end{bmatrix}. \tag{3.4}$$

Because basis vectors suffice to quantify any other vector, the matrix $\mu^S_{S'}$ suffices to quantify the orientation of an object. The dot products in (3.4) equal the cosines of the angles between the various basis vectors, so $\mu^S_{S'}$ is also called the *direction cosine matrix* relating S and S'. Not surprisingly, μ^S_S is the identity matrix. Also, some straightforward linear algebra can be used to show that

$$\mu^{S'}_S = \left(\mu^S_{S'}\right)^t = \left(\mu^S_{S'}\right)^{-1}, \tag{3.5}$$

where the superscript t denotes the matrix transpose. When the transpose of a matrix equals its inverse, it's known as *orthogonal*; its rows will be mutually orthonormal, and so will its columns.

The mathematics of orientation theory can be constructed from the answers to two fundamental questions.

First Fundamental Question: Given $[\boldsymbol{v}]_S$, what is $[\boldsymbol{v}]_{S'}$?

$$[\boldsymbol{v}]_{S'} = \begin{bmatrix} \boldsymbol{v}\cdot\boldsymbol{e}_{x'} \\ \vdots \\ \boldsymbol{v}\cdot\boldsymbol{e}_{z'} \end{bmatrix} = \begin{bmatrix} (v_x\boldsymbol{e}_x + \dots + v_z\boldsymbol{e}_z)\cdot\boldsymbol{e}_{x'} \\ \vdots \\ (v_x\boldsymbol{e}_x + \dots + v_z\boldsymbol{e}_z)\cdot\boldsymbol{e}_{z'} \end{bmatrix} = \begin{bmatrix} v_x\boldsymbol{e}_x\cdot\boldsymbol{e}_{x'} + \dots + v_z\boldsymbol{e}_z\cdot\boldsymbol{e}_{x'} \\ \vdots \\ v_x\boldsymbol{e}_x\cdot\boldsymbol{e}_{z'} + \dots + v_z\boldsymbol{e}_z\cdot\boldsymbol{e}_{z'} \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{e}_x\cdot\boldsymbol{e}_{x'} & \dots & \boldsymbol{e}_z\cdot\boldsymbol{e}_{x'} \\ \vdots & & \\ \boldsymbol{e}_x\cdot\boldsymbol{e}_{z'} & \dots & \boldsymbol{e}_z\cdot\boldsymbol{e}_{z'} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \mu^S_{S'}\,[\boldsymbol{v}]_S\,. \tag{3.6}$$

So the orientation matrix transforms one coordinate vector to another. Equation (3.6) is an important equation of this report, and we highlight it by putting it in a box:

$$\boxed{[\boldsymbol{v}]_{S'} = \mu^S_{S'}\,[\boldsymbol{v}]_S\,.} \tag{3.7}$$

[2]The symbol "≡" denotes a definition: $a \equiv b$ means that a is defined to equal b. We will occasionally use the "≡" symbol last in a string of equalities, in which case we mean that the *last* quantity is being defined.

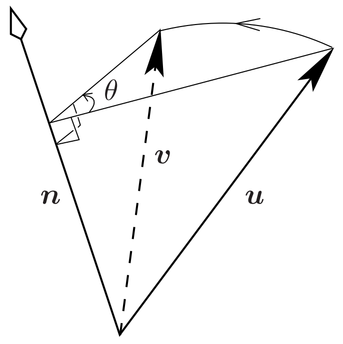

Figure 1: *Rotating a vector $\boldsymbol{u}$ in a right-handed sense around the vector $\boldsymbol{n}$ to produce $\boldsymbol{v}$.*

Notice that

$$[\boldsymbol{v}]_A = \mu_A^B\,[\boldsymbol{v}]_B = \mu_A^B\,\mu_B^C\,[\boldsymbol{v}]_C\,, \tag{3.8}$$

but we also know that $[\boldsymbol{v}]_A = \mu_A^C\,[\boldsymbol{v}]_C$. In that case we conclude

$$\mu_A^C = \mu_A^B\,\mu_B^C\,, \tag{3.9}$$

which shows how orientation matrices chain together. The process can be continued indefinitely of course: $\mu_A^D = \mu_A^B\,\mu_B^C\,\mu_C^D$, and so on.

Appendix A discusses an application of orientation matrices to quantifying the various terms used in discussions of wind motion over a ship.

The "Second Fundamental Question" belongs under the umbrella of how to rotate vectors, which we look at next.

3.2 Rotation Mathematics

Second Fundamental Question: Suppose S' was originally coincident with S, but now has been orientated according to $\mu_S^{S'}$. S' took a snapshot of a vector $\boldsymbol{u}$ and carried it along to make a new vector $\boldsymbol{v}$. We ask: how is $[\boldsymbol{v}]_S$ related to $[\boldsymbol{u}]_S$? First, by construction, the components of $\boldsymbol{v}$ in S' are just the components of $\boldsymbol{u}$ in S:

$$[\boldsymbol{v}]_{S'} \equiv [\boldsymbol{u}]_S\,. \tag{3.10}$$

In that case we can immediately write

$$[\boldsymbol{v}]_S = \mu_S^{S'}\,[\boldsymbol{v}]_{S'} = \mu_S^{S'}\,[\boldsymbol{u}]_S\,, \tag{3.11}$$

so that $[\boldsymbol{v}]_S$ relates to $[\boldsymbol{u}]_S$ via the orientation matrix.

3.2.1 How to Rotate a Vector using a Matrix

Now refer to Figure 1, in which a vector $\boldsymbol{u}$ (or rather, a snapshot of $\boldsymbol{u}$) is rotated in a right-handed sense about an axis vector $\boldsymbol{n}$ to produce $\boldsymbol{v}$. This rotation is independent of any coordinate system. Write it as $R_{\boldsymbol{n}}^{\theta}$:

$$\boldsymbol{v} = R_{\boldsymbol{n}}^{\theta}\,\boldsymbol{u}\,. \tag{3.12}$$

Working with vectors generally requires their components to be specified, and calculating the components of a rotated vector requires a coordinate system. The core result of rotation analysis is that in S, the rotation (3.12) can be performed by a matrix multiplication using the *matrix representative* $\left[R^\theta_{\boldsymbol{n}}\right]_S$ of the rotation:

$$[\boldsymbol{v}]_S = \left[R^\theta_{\boldsymbol{n}}\right]_S [\boldsymbol{u}]_S\,, \tag{3.13}$$

where the rotation matrix is, for unit-length $\boldsymbol{n}$,

$$\left[R^\theta_{\boldsymbol{n}}\right]_S = 1 + \sin\theta\,[\boldsymbol{n}]^\times_S + (1-\cos\theta)\left([\boldsymbol{n}]^\times_S\right)^2. \tag{3.14}$$

Here the first "1" is the 3×3 identity matrix. The *cross matrix* $[\boldsymbol{n}]^\times_S$ appears widely in rotation theory, and implements the vector cross product. For any two vectors $\boldsymbol{a}, \boldsymbol{b}$, we have $[\boldsymbol{a}]^\times_S\,[\boldsymbol{b}]_S \equiv [\boldsymbol{a}\times\boldsymbol{b}]_S$, where

$$[\boldsymbol{a}]^\times_S = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}^\times \equiv \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \tag{3.15}$$

The rotation matrix (3.14) is well known and can be derived by combining equations (4.22) and (4.25) of [4]. (There, the "$[\ ]_S$" notation of this report is suppressed for simplicity because only one coordinate system is used.) See also [5], whose (1) is equivalent to our (3.14).

A rotation can be inverted (undone) with a reverse turn, or by reversing the axis:

$$\left(R^\theta_{\boldsymbol{n}}\right)^{-1} = R^{-\theta}_{\boldsymbol{n}} = R^\theta_{-\boldsymbol{n}}\,, \tag{3.16}$$

and it can be shown easily from (3.14) that rotation matrices are orthogonal, meaning

$$\left[R^\theta_{\boldsymbol{n}}\right]^{-1}_S = \left[R^\theta_{\boldsymbol{n}}\right]^t_S. \tag{3.17}$$

For brevity, we write a rotation about an axis, say $R^\theta_{\boldsymbol{e}_x}$, as R^θ_x.

A rotation R_1 followed by R_2 produces

$$\boldsymbol{v} = R_2(R_1\boldsymbol{u}) = (R_2R_1)\boldsymbol{u}\,, \tag{3.18}$$

so that the result is the same as rotating with a single matrix R_2R_1.

Rotations around the coordinate axes are particularly simple and known as *Euler matrices*:

$$E^\theta_1 \equiv \left[R^\theta_x\right]_S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \qquad E^\theta_2 \equiv \left[R^\theta_y\right]_S = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix},$$

$$E^\theta_3 \equiv \left[R^\theta_z\right]_S = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.19}$$

We use subscripts 1, 2, 3 for the Euler matrices because they depend not on any particular basis, but only on the numerical ordering of the relevant axes within the basis set. That is,

$$E^\theta_1 = \left[R^\theta_x\right]_S = \left[R^\theta_{x'}\right]_{S'} = \left[R^\theta_{x''}\right]_{S''} = \dots \tag{3.20}$$

Given two sets of axes S, S', the orientation matrix of one relative to the other is precisely a rotation matrix:

$$\mu_S^{S'} = [R_{\boldsymbol{n}}^{\theta}]_S\,. \tag{3.21}$$

(We use this identity later when dealing with direction-cosine representations.) Prove this by starting with the fact that $\mu_S^{S'}$ gives the orientation of the axes S' relative to the axes S. Now consider forming the S' axes by rotating a snapshot of S by $R_{\boldsymbol{n}}^{\theta}$, so that the x' axis is the result of rotating the x axis, and similarly for y and z. The orientation matrix is

$$\mu_S^{S'} = \Big[[\boldsymbol{e}_{x'}]_S \;\ldots\; [\boldsymbol{e}_{z'}]_S\Big]. \tag{3.22}$$

Matrix multiplication proceeds column by column—as explained ahead in (5.12)—and each column of $\mu_S^{S'}$ was constructed by rotating the corresponding coordinate vector of S. That is, because

$$\boldsymbol{e}_{x'} = R_{\boldsymbol{n}}^{\theta}\,\boldsymbol{e}_x\,, \quad \boldsymbol{e}_{y'} = R_{\boldsymbol{n}}^{\theta}\,\boldsymbol{e}_y\,, \quad \boldsymbol{e}_{z'} = R_{\boldsymbol{n}}^{\theta}\,\boldsymbol{e}_z\,, \tag{3.23}$$

we can immediately write

$$\mu_S^{S'} = \Big[[\boldsymbol{e}_{x'}]_S \ldots [\boldsymbol{e}_{z'}]_S\Big] = [R_{\boldsymbol{n}}^{\theta}]_S \underbrace{\Big[[\boldsymbol{e}_x]_S \;\ldots\; [\boldsymbol{e}_z]_S\Big]}_{\text{identity matrix!}} = [R_{\boldsymbol{n}}^{\theta}]_S\,. \tag{3.24}$$

So the matrix $\mu_S^{S'}$ that gives the orientation of the S' axes in S coordinates is precisely the rotation matrix that *produced* the S' axes from the S axes. This is an important fact that aids in relating orientation and rotation matrices, depending on which is more easily constructed in the problem we wish to solve.

The intricacies and richness of rotation/orientation theory arise because rotations around *different axes* don't commute—meaning they cannot be swapped without changing the result. (They *do* commute when around the same axis.) The extent of the noncommutivity is given by examining the value of

$$\big[R_{\boldsymbol{m}}^{\theta}, R_{\boldsymbol{n}}^{\phi}\big] \equiv R_{\boldsymbol{m}}^{\theta}\,R_{\boldsymbol{n}}^{\phi} - R_{\boldsymbol{n}}^{\phi}\,R_{\boldsymbol{m}}^{\theta}\,. \tag{3.25}$$

If that value is zero, the rotations commute, but in general $\big[R_{\boldsymbol{m}}^{\theta}, R_{\boldsymbol{n}}^{\phi}\big]$ won't be zero. We can calculate its value using (3.14), omitting the "$[\;]_S$" notation for simplicity since this analysis uses a single coordinate system. Write

$$\begin{aligned}
\big[R_{\boldsymbol{m}}^{\theta}, R_{\boldsymbol{n}}^{\phi}\big] &= \big[1 + \sin\theta\,\boldsymbol{m}^{\times} + (1-\cos\theta)\boldsymbol{m}^{\times 2}\big]\big[1 + \sin\phi\,\boldsymbol{n}^{\times} + (1-\cos\phi)\boldsymbol{n}^{\times 2}\big] \\
&\qquad - [\text{swap: } \boldsymbol{m} \leftrightarrow \boldsymbol{n},\, \theta \leftrightarrow \phi] \\
&= \sin\theta\sin\phi\,\big[\boldsymbol{m}^{\times}, \boldsymbol{n}^{\times}\big] + (1-\cos\theta)\sin\phi\,\big[\boldsymbol{m}^{\times 2}, \boldsymbol{n}^{\times}\big] \\
&\qquad + \sin\theta\,(1-\cos\phi)\big[\boldsymbol{m}^{\times}, \boldsymbol{n}^{\times 2}\big] + (1-\cos\theta)(1-\cos\phi)\big[\boldsymbol{m}^{\times 2}, \boldsymbol{n}^{\times 2}\big].
\end{aligned} \tag{3.26}$$

As θ and ϕ become small, this reduces to

$$\big[R_{\boldsymbol{m}}^{\theta}, R_{\boldsymbol{n}}^{\phi}\big] = \theta\phi\,\big[\boldsymbol{m}^{\times}, \boldsymbol{n}^{\times}\big] + \text{higher-order terms in the angles.} \tag{3.27}$$

That is, the error we make in swapping two rotations is second order in the angles. So rotations commute to first order; the phrase "infinitesimal rotations commute" is often used, meaning that we can swap rotations freely when working with infinitesimals. (See Section 6 for more on this.) Note that *both* rotations must be infinitesimal if we wish to swap them.

In fact, (3.27) can be derived more quickly than we have just done in the last paragraph. The key is to realise that (3.14) can be written in the compact form

$$R^{\theta}_{\boldsymbol{n}} = e^{\theta \boldsymbol{n}^{\times}}, \tag{3.28}$$

where again we are suppressing the "$[\ ]_S$" notation. This exponential is defined as a series in the usual way. If we now calculate $\left[R^{\theta}_{\boldsymbol{m}}, R^{\phi}_{\boldsymbol{n}}\right]$ by writing each of its rotations as an exponential series, then the final small-angle result in (3.27) follows in a straightforward way, although the exact expression of (3.26) is not so easily obtained.

According to (3.27), whereas rotations through small angles can be swapped with ever-increasing impunity as the angles are made still smaller, rotations through large angles cannot be swapped without incurring a non-negligible error. We might attempt to swap two large rotations while avoiding the above error of order $\theta\phi$, by dividing each into infinitely many infinitesimal rotations, then swapping adjacent rotations to "bubble" one set through the other. In fact, this procedure fails.

To see why, start with two large rotations through θ and ϕ around distinct axes, and divide each into N tiny rotations θ/N and ϕ/N (where N is large). Now perform the required N^2 swaps, where each swap adds an error $\sim \theta\phi/N^2$. Thus the final error is of order $\theta\phi$, and we have gained nothing. The same argument shows that a large rotation cannot even be swapped with an infinitesimal rotation. The PRT's pseudo-swap of large rotations works only because it changes the rotation axes.

3.2.2 How to Rotate a Vector using a Quaternion

Euler and Rodrigues are credited with having invented a set of four numbers that can be used to rotate a vector. These numbers turn out to be identical to a set discovered later by Hamilton, known as unit-length *quaternions*. We will briefly cover how quaternions are used; the relevant proofs can be found in [4]. A quaternion can be written as a 4-element vector, typically denoted $(a_0, \boldsymbol{a})$, where the last three elements are naturally represented by a 3-element vector.[3] In fact, quaternions predate vectors, being invented in the mid 19th century by Hamilton as he sought to generalise the idea of complex numbers $a + b\mathrm{i}$. He eventually obtained the expression $a_0 + a_1\mathrm{i} + a_2\mathrm{j} + a_3\mathrm{k}$, where $\mathrm{i}, \mathrm{j}, \mathrm{k}$ had the necessary algebraic properties to allow this generalisation to use a minimum of new ideas.[4] This notation eventually split into a "scalar" part—the first component—and a "vector" part—the last three components, and this split gave rise to the modern notation $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ for generic cartesian basis vectors. So Hamilton's $a_0 + a_1\mathrm{i} + a_2\mathrm{j} + a_3\mathrm{k}$ has become the modern $(a_0, \boldsymbol{a})$. We must use more descriptive names than $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ for our basis vectors in order to accommodate more than one coordinate system: $\boldsymbol{e}_X, \boldsymbol{e}_Y, \boldsymbol{e}_Z$ for ECEF, $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$ for an aircraft, and so on.

[3] We are using the symbol for a proper vector $\boldsymbol{a}$ to represent the coordinate vector that the quaternion's last three elements can be defined to be. The distinction is understood but not worth dwelling on in this context, because our discussion of the basic properties of quaternions uses only one set of coordinates. After all, writing our quaternions using a pre-defined S as $\left(a_0, [\boldsymbol{a}]_S\right)$—or really $\left(a_0, [\boldsymbol{a}]^t_S\right)$—would just be tedious.

[4] For Hamilton, the i in this expression generalised the complex number i, but from the point of view of using quaternions for rotation, it's actually the quaternion k that generalises the complex number i, because both of these implement a rotation in the xy plane.

Addition and scalar multiplication of quaternions is defined element by element, just as for coordinate vectors. Quaternion multiplication is associative and defined by

$$(a_0, \boldsymbol{a})\,(b_0, \boldsymbol{b}) \equiv (a_0 b_0 - \boldsymbol{a}\cdot\boldsymbol{b},\ a_0\boldsymbol{b} + b_0\boldsymbol{a} + \boldsymbol{a}\times\boldsymbol{b})\,. \tag{3.29}$$

The quaternion for a rotation around unit vector $\boldsymbol{n}$ through angle θ is

$$Q^{\theta}_{\boldsymbol{n}} = (\cos{}^{\theta}\!/_2,\ \boldsymbol{n}\sin{}^{\theta}\!/_2)\,. \tag{3.30}$$

It can be shown [4] that a vector $\boldsymbol{u}$ is rotated through θ around unit vector $\boldsymbol{n}$ to produce vector $\boldsymbol{v}$ via a double quaternion multiplication:

$$(0, \boldsymbol{v}) = Q^{\theta}_{\boldsymbol{n}}\ (0, \boldsymbol{u})\ Q^{-\theta}_{\boldsymbol{n}}. \tag{3.31}$$

[We see in (3.31) the beginnings of just why it was that the last three elements of quaternions were eventually split off to form a new entity called a vector: because the quaternions that are used in place of vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ are $(0, \boldsymbol{u})$ and $(0, \boldsymbol{v})$.]

Why are there *two* multiplications in (3.31)? They are the price to be paid for the simple structure of $Q^{\theta}_{\boldsymbol{n}}$, versus the more complicated structure of the matrix $R^{\theta}_{\boldsymbol{n}}$ in (3.14). Notice that the elements of $\boldsymbol{n}$ occur only singly in $Q^{\theta}_{\boldsymbol{n}}$; contrast this with $R^{\theta}_{\boldsymbol{n}}$, where the last term in (3.14) has *products* of the elements of $\boldsymbol{n}$ sitting inside $\boldsymbol{n}^{\times 2}$. Quaternions' sparseness gives them simplicity, but the price to be paid is that two multiplications are needed to incorporate those required products of the elements of $\boldsymbol{n}$, in order to achieve the same as a matrix multiplication by $R^{\theta}_{\boldsymbol{n}}$. This double multiplication is also why (3.30) uses just half the rotation angle.

A quaternion is defined to have length

$$|(a_0, \boldsymbol{a})| \equiv \sqrt{a_0^2 + |\boldsymbol{a}|^2}\ , \tag{3.32}$$

in which case it follows that all rotation quaternions—that is, of the sort (3.30)—have unit length. This corresponds to the unit determinant of rotation matrices.

The identity quaternion for rotation—corresponding to no rotation at all—is $Q^{0}_{\boldsymbol{n}} = (1, \boldsymbol{0})$, being a rotation through zero angle around any axis. Because (3.30) makes it trivial to build the quaternion associated with an angle and an axis, quaternions have traditionally come to be treated as *the* way to implement the angle–axis approach to quantifying an orientation. But we should not forget that all they do is rotate a vector, just as does the matrix in (3.14).

The inverse of $Q^{\theta}_{\boldsymbol{n}}$ is

$$\left(Q^{\theta}_{\boldsymbol{n}}\right)^{-1} = Q^{\theta}_{-\boldsymbol{n}} = Q^{-\theta}_{\boldsymbol{n}}, \tag{3.33}$$

and, just as for matrices, the product of quaternions Q_1 and Q_2 satisfies

$$(Q_1 Q_2)^{-1} = Q_2^{-1} Q_1^{-1}. \tag{3.34}$$

Equations (3.31)–(3.34) show how successive rotations are written with quaternions: a rotation Q_1 followed by Q_2 produces

$$(0, \boldsymbol{v}) = Q_2 Q_1\ (0, \boldsymbol{u})\ Q_1^{-1} Q_2^{-1} = Q_2 Q_1\ (0, \boldsymbol{u})\ (Q_2 Q_1)^{-1}\,, \tag{3.35}$$

so, just as in (3.18), the result is the same as rotating with a single quaternion $Q_2 Q_1$.

3.2.3 The Rotation Matrix and Rotation Quaternion in Quantum Mechanics

The exponential form (3.28) of the rotation matrix is derived in some quantum mechanics textbooks, because it's used in that subject to deal with angular momentum, which is a property of fundamental particles. If you do refer to a quantum mechanics text, you'll find a different notation, perhaps accompanied by an obfuscated way of deriving (3.28). We'll pay a brief visit to that notation here to demystify it.

First, omit the explicit $[\ ]_S$ on all vectors for simplicity, and use the linearity of the cross operator to expand the cross matrix into three component matrices:[5]

$$\boldsymbol{n}^{\times} = (n_x \boldsymbol{e}_x + n_y \boldsymbol{e}_y + n_z \boldsymbol{e}_z)^{\times} = n_x \boldsymbol{e}_x^{\times} + n_y \boldsymbol{e}_y^{\times} + n_z \boldsymbol{e}_z^{\times}\,. \tag{3.37}$$

In quantum mechanics the matrices $\boldsymbol{e}_x^{\times}, \boldsymbol{e}_y^{\times}, \boldsymbol{e}_z^{\times}$ are denoted $-\mathrm{i}J_x, -\mathrm{i}J_y, -\mathrm{i}J_z$ respectively, and the matrix triplet (J_x, J_y, J_z) is written as $\boldsymbol{J}$, as though it were a vector. Hence $\boldsymbol{n}^{\times}$ is written $-\mathrm{i}\boldsymbol{n}\cdot\boldsymbol{J}$, and the rotation matrix becomes

$$R_{\boldsymbol{n}}^{\theta} = \mathrm{e}^{\theta \boldsymbol{n}^{\times}} = \mathrm{e}^{-\mathrm{i}\theta \boldsymbol{n}\cdot\boldsymbol{J}}. \tag{3.38}$$

The *only* reason for the introduction of i is that the matrices J_x, J_y, J_z eventually become identified with angular momentum in a quantum mechanical sense.

It's notationally efficient to derive (3.28) as $\mathrm{e}^{\theta \boldsymbol{n}^{\times}}$, and *only then* to branch off into quantum mechanics through the use of the angular momentum matrices. Instead of doing that, the standard quantum mechanical treatment introduces J_x, J_y, J_z at the very start of the subject—before the rotation matrix has even been derived—by writing $\boldsymbol{e}_x^{\times}$ as $-\mathrm{i}J_x$ and so on; it thus presents a real, geometrical, analysis using complex numbers that all cancel internally (since the J matrices are pure imaginary). The effect is rather like writing $2 \times 3 = 2\mathrm{i} \times -3\mathrm{i} = 6$. There is no pedagogical value in such a procedure. Of course, quantum mechanics requires that J_x, J_y, J_z involve complex numbers; that is not the issue.[6] Rather, why introduce i into a rotational analysis where it's not needed? There is nothing deep or elegant about writing a rotation as $\mathrm{e}^{-\mathrm{i}\theta \boldsymbol{n}\cdot\boldsymbol{J}}$ *before* introducing the concept of angular momentum, and I suspect that most users of the quantum mechanical notation would be hard pressed to actually rotate a vector using $\mathrm{e}^{-\mathrm{i}\theta \boldsymbol{n}\cdot\boldsymbol{J}}$—assuming they were aware that this is what $\mathrm{e}^{-\mathrm{i}\theta \boldsymbol{n}\cdot\boldsymbol{J}}$ actually *does*—despite the pages of obscure analysis that you'll find on the subject in quantum mechanics textbooks. If you do refer to the quantum mechanics approach for some rotation theory, recognise this unnecessary complex-number baggage for what it is, and the analyses will become a little less confusing.

Similar comments apply to the quaternions used in quantum mechanics. It turns out that a quaternion can be represented by a 2×2 complex matrix [4], and you will find them

[5]Equation (3.37) shows the compactness of the cross notation. We could also have used its full matrix form to get the same result:

$$\boldsymbol{n}^{\times} = \begin{bmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{bmatrix} = n_x \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}}_{=\,\boldsymbol{e}_x^{\times}} + n_y \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}}_{=\,\boldsymbol{e}_y^{\times}} + n_z \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{=\,\boldsymbol{e}_z^{\times}}. \tag{3.36}$$

[6]On this note, given that quantum mechanics is based on the complex Fourier transform, and given that the complex Fourier transform can always be replaced by the real Hartley transform [6], perhaps the last word has yet to be said on this subject. But that doesn't affect my comments above.

written that way in quantum mechanics where the fundamental matrices used, analogous to the basic quaternions $1, \mathsf{i}, \mathsf{j}, \mathsf{k}$, are called *Pauli matrices*. Just as for the J_x, J_y, J_z matrices in the previous paragraphs, Pauli matrices are highly useful in describing angular momentum. But being complex, they need to be associated with an extra factor of i (the square root of -1, not the quaternion) in order to be used to rotate a vector, so it's best to avoid this quantum mechanics notation if you want to learn rotation theory.

3.2.4 Rotating One Point About Another

Finally, consider rotating the point $(10, 0, 0)$ by $90°$ about an axis $(0, 0, 1)$ that passes through the point $(9, 0, 0)$. The result is the point $(9, 1, 0)$, which is easily seen from the simple geometry involved. How might we use the theory of rotation to rotate points in general?

To see how, we should be precise in describing the foregoing rotation using vector language. We rotated the point described by the *position vector* $(10, 0, 0)$ (necessarily quantified as a *coordinate vector*, a notion that we'll take as given!) by first forming a *proper vector*: the arrow from $(9, 0, 0)$ to $(10, 0, 0)$. This arrow, $(1, 0, 0)$, was then rotated by $90°$ about the axis proper vector $(0, 0, 1)$ to give $(0, 1, 0)$. Finally *this* arrow (proper vector) was added to the *position vector* $(9, 0, 0)$ to give the *position vector* for the sought-after point $(9, 1, 0)$.

The general procedure is straightforward to write down. We require to rotate the point $\boldsymbol{x}$ by angle θ about the point $\boldsymbol{p}$, through which runs the axis $\boldsymbol{n}$ (a unit vector). The resulting sought-after point is denoted $\boldsymbol{x}'$. We simply rotate the proper vector $\boldsymbol{x} - \boldsymbol{p}$ to produce the proper vector $\boldsymbol{x}' - \boldsymbol{p}$:

$$\boldsymbol{x}' - \boldsymbol{p} = R^{\theta}_{\boldsymbol{n}} (\boldsymbol{x} - \boldsymbol{p}) \,, \tag{3.39}$$

so that the required point is $\boldsymbol{x}' = \boldsymbol{p} + R^{\theta}_{\boldsymbol{n}} (\boldsymbol{x} - \boldsymbol{p})$.

4 Quantifying Orientation

We've seen that an object's orientation can only be expressed relative to some given base orientation, and is conveniently written within some chosen coordinate system S. It can be quantified by embedding a set of axes within the object, calling these the S' axes, and specifying the coordinates of the S' basis vectors in S. These coordinate vectors are written as the columns of the orientation matrix $\mu^{S'}_{S}$, also called the direction-cosine matrix.

But in (3.24) we saw that $\mu^{S'}_{S}$ is in fact a rotation matrix: we can interpret that equation as saying that $\mu^{S'}_{S}$ rotates snapshots of the S axes to produce the S' axes. This allows for alternative methods of quantifying an object's orientation, by constructing the orientation through imagining that the object's axes were originally coincident with those of S, and we rotated it in some chosen way to arrive at its final orientation. The three methods that are used commonly are described below. It's important to understand that they—and any others that could be added—all ultimately *do the same thing*: they construct the S' basis vectors from the S basis vectors. The choice of one method over another is partly governed by the parameters of the problem being solved, and partly by the nuances of any necessary computer coding.

Orientation or Direction-cosine matrix: As noted after (3.4), these are alternative names for the matrix whose columns are the basis vectors of S' coordinated in S. This matrix completely specifies the object's orientation.

Angle–axis: Here we specify an angle θ and a unit-length axis vector $\boldsymbol{n}$: rotating snapshots of the S basis vectors through θ around $\boldsymbol{n}$ produces the S' basis vectors.

Euler angles: We specify the angles of a *sequence* of rotations that rotates a snapshot of the S axes onto the S' axes; typically 3 rotations are used around the coordinate axes; these angles are then called *Euler angles.* The need for three rotations rather than the angle–axis method's single rotation is due to the fact that the Euler angles specify rotations around a prescribed set of cartesian axes. In contrast, the angle–axis method manages with just one axis—but that axis will seldom coincide with any "simple" axis such as the cartesian axes.

The first method is specifically a matrix method. The orientation matrix is also a rotation matrix, as we saw in (3.24). The last two methods involve rotation, and a rotation can be done using either a matrix or a quaternion. The choice of which of these two tools to use is ours; the methods stand above the tools used to implement them. But the important point to remember is that the methods all do the same thing, and in particular the matrices that can be produced from all three of them will be *identical* (up to numerical inaccuracies).

Comparing methods: Given that the orientation matrix is identical to using a single rotation to describe an orientation, the choice of method generally comes down to using either one or three rotations to rotate the base orientation. Using a single rotation is as lean and simple as things can be. It presents no great numerical difficulties and has now become a proven approach. Using three Euler angles does have its difficulties: the rotations' lack of commutivity means that users of Euler angles often find them confusing, and there are numerical problems with using three angles that we'll investigate in Section 5.4. Nevertheless, rotating by Euler angles corresponds to the yaw–pitch–roll language and manoeuvres familiar to pilots, and this means that there will always be a place for Euler angles in orientation theory, at least when applied to flight.

Comparing tools: The price we pay for the simplicity of matrix multiplication is that rotation matrices have redundancy in their elements. A quaternion can be viewed as a way of "rendering down" a rotation matrix to its core information, but this comes with a more complicated form of multiplication than that of matrices.

There is an important computational point to be considered here: if we must extrapolate orientation information (as in Section 6), we'll be updating matrices or quaternions using numerical algorithms. Throughout such procedures, we must always ensure that the matrices produced have determinant 1, as all rotation matrices do, or that the quaternions produced have length 1, as all rotation quaternions do. This mitigates problems due to numerical round-off error; if we didn't at least ensure our matrices or quaternions had this property, then the results of rotating vectors would soon begin to look skewed.

A suitable replacement for a quaternion Q whose length has strayed from 1 is to replace Q by itself divided by (meaning each of its elements by) its length, calculated in (3.32): $Q \to Q/|Q|$. This is a simple numerical exercise. In contrast, re-orthogonalising a rotation matrix R requires significantly more computation. A suitable definition for how "close" its replacement R' should be leads to $R' = R\,(R^t R)^{-1/2}$, which can be evaluated using singular value decomposition. That is, first use that technique to write $R = USV^t$ for matrices U, S, V as outlined in singular value decomposition theory; their properties quickly produce the simple expression $R' = UV^t$.

There persists a view in the orientation community that angle–axis goes hand in hand with quaternions, whereas Euler angles are involved with matrices. Because of this, the *tool* (quaternions) used to implement the angle–axis description of an orientation has gradually come to be compared with the Euler angles *method* (three rotations). But that's comparing apples and oranges; the three *methods* described above should be compared on one level, and the two *tools* (matrices and quaternions) should be compared on another level.

5 Alternative Approaches to the Applications of Reference [1]

Sections 4.1–4.3 of [1] gave examples of using rotations to construct orientation scenarios. We'll now rework those calculations using the notation of this report. The result will be a more streamlined approach to the problems being solved.

5.1 Rotation and Flight over a Sphere

Any translation that a body's centroid undergoes while the body's orientation is changing does not affect that orientation. An aircraft that rolls to fly inverted *is* inverted, irrespective of the fact that it's moving sideways as it turns upside down. We can use this fact to our advantage when visualising rotation. For example, picture how an aircraft's orientation changes as it pitches nose down, rotating through $-90°$ about its starboard wing. We can arrive at the same final orientation by imagining the aircraft to be anywhere on a sphere—say, a point on the sphere's equator. Now allow the aircraft to move over the sphere (like a car being driven, always touching the sphere so that it's straight and level at every point) until it has flown to the North Pole. This change in orientation is precisely a rotation through $-90°$ around its starboard wing; the motion over the sphere is of no consequence. This is a simple example, but we can picture a sequence of more complicated rotations in the same way, by tracing the aircraft's path over the sphere.

5.2 Constructing Local North–East–Down Coordinates

Section 4.1 of [1] posed the question "If we are in Adelaide and Earth is transparent, what is the compass bearing of Brussels if we are looking straight at it through Earth?". We can answer this question by finding the local north and east components of the vector $\boldsymbol{r}_{BA}$ pointing from Adelaide A to Brussels B, and using these with some simple trigonometry to find the required angle. Calculate these components by constructing the local north–east–down (NED) axes at Adelaide; this is done by calculating the coordinates of those axes' basis vectors in Earth-centred, Earth-fixed coordinates (ECEF), defined shortly.

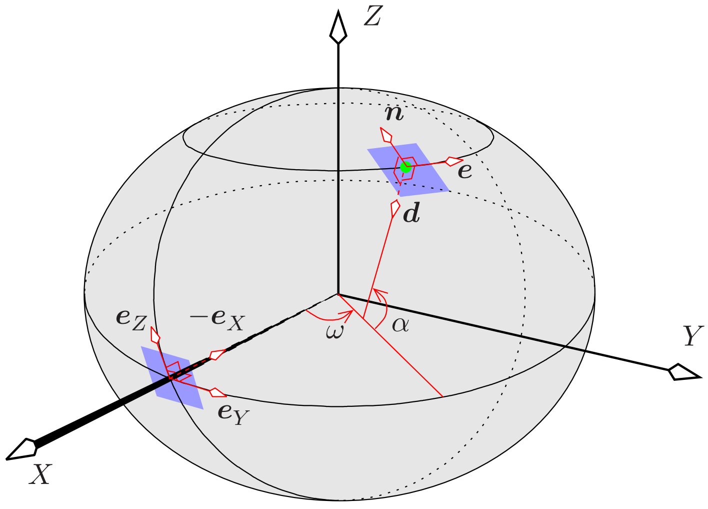

Figure 2: *The local north–east–down vectors at a given point on Earth's surface can be constructed by rotating the ordered set of vectors* $\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X$ *(at zero lat/long, lower left) to produce the ordered set* $\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}$ *(upper right)*

First, we'll construct the vector $\boldsymbol{r}_{BA}$ pointing from Adelaide A to Brussels B, specifying its components in ECEF coordinates E. ECEF coordinates are cartesian with origin at Earth's centre, using the X, Y, Z axes of Fig. 2. They often simplify geodesy calculations. The ECEF system has the following unit-length basis vectors:

1. $\boldsymbol{e}_X$ points from Earth's centre through the Equator at $\alpha = \omega = 0°$,

2. $\boldsymbol{e}_Y$ points from Earth's centre through the Equator at $\alpha = 0°, \omega = 90°$, and

3. $\boldsymbol{e}_Z$ points from Earth's centre through the North Pole.

The proper vector $\boldsymbol{r}_{BA}$ pointing from Adelaide to Brussels is found from Brussels' ECEF position vector $\boldsymbol{r}_{BC}$ (which is relative to Earth's centre C) and Adelaide's ECEF position vector $\boldsymbol{r}_{AC}$:

$$\boldsymbol{r}_{BA} = \boldsymbol{r}_{BC} - \boldsymbol{r}_{AC}\,, \tag{5.1}$$

where each of these vectors is calculated in the following way. Given a point P's latitude α, longitude ω, and height h above the WGS-84 ellipsoid, its ECEF position, $[\boldsymbol{r}_{PC}]_E$, is the (X, Y, Z) triplet of the point using the axes convention of Fig. 2. Using Earth's semi-major and semi-minor axis lengths in the WGS-84 scheme,

$$\begin{aligned} \text{semi-major: } a &= 6{,}378{,}137 \text{ m}, \\ \text{semi-minor: } b &= 6{,}356{,}752.3142 \text{ m}, \end{aligned} \tag{5.2}$$

an analysis of the shape of an ellipsoid yields P's ECEF coordinates as

$$X = \left(\frac{a^2}{\sqrt{a^2\cos^2\alpha + b^2\sin^2\alpha}} + h\right)\cos\alpha\ \cos\omega\,,$$

$$Y = \left(\frac{a^2}{\sqrt{a^2\cos^2\alpha + b^2\sin^2\alpha}} + h\right)\cos\alpha\ \sin\omega\,,$$

$$Z = \left(\frac{b^2}{\sqrt{a^2 \cos^2 \alpha + b^2 \sin^2 \alpha}} + h \right) \sin \alpha \,. \tag{5.3}$$

(This equation is equivalent to equation (2.2) of [1], but is written here in a way that treats a and b more symmetrically.) Assume the two cities are at $h = 0$ and use

Adelaide:	latitude $\alpha = -34.9°$,	longitude $\omega = 138.5°$,
Brussels:	latitude $\alpha = 50.8°$,	longitude $\omega = 4.3°$.

Equation (5.3) converts these parameters into two position vectors of the cities with respect to Earth's centre C. These positions of Adelaide and Brussels are, respectively,

$$[\boldsymbol{r}_{AC}]_E = \begin{bmatrix} -3.922 \\ 3.470 \\ -3.629 \end{bmatrix} \times 10^6 \text{ metres,} \quad [\boldsymbol{r}_{BC}]_E = \begin{bmatrix} 4.028 \\ 0.303 \\ 4.920 \end{bmatrix} \times 10^6 \text{ metres,} \tag{5.4}$$

and a subtraction gives the vector pointing from Adelaide to Brussels (with all internal calculations in this report taken to a large number of decimal places and then rounded to 3 decimal places when written down):

$$[\boldsymbol{r}_{BA}]_E = [\boldsymbol{r}_{BC} - \boldsymbol{r}_{AC}]_E = [\boldsymbol{r}_{BC}]_E - [\boldsymbol{r}_{AC}]_E = \begin{bmatrix} 7.950 \\ -3.167 \\ 8.548 \end{bmatrix} \times 10^6 \text{ metres.} \tag{5.5}$$

We have $[\boldsymbol{r}_{BA}]_E$, but require the first two components of $[\boldsymbol{r}_{BA}]_N$, since these are the north and east components needed to calculate the required bearing of Brussels as seen from Adelaide. These two coordinate vectors are related by

$$[\boldsymbol{r}_{BA}]_N \overset{(3.7)}{=\!=} \mu^E_N \, [\boldsymbol{r}_{BA}]_E \,. \tag{5.6}$$

The ECEF is particularly easy to calculate within, so we might calculate μ^E_N by first finding μ^N_E. For this, we need the NED coordinate system N at a given point on Earth at latitude α and longitude ω, also shown in Figure 2. The NED coordinate axes N are defined by unit basis vectors $\boldsymbol{n}$ (pointing north), $\boldsymbol{e}$ (pointing east), $\boldsymbol{d}$ (pointing down, $= \boldsymbol{n} \times \boldsymbol{e}$). We require to find these basis vectors, expressed in ECEF coordinates. That is, we require $[\boldsymbol{n}]_E, [\boldsymbol{e}]_E, [\boldsymbol{d}]_E$.

The flying-aircraft analogy of Section 5.1 can be used here. Earth is approximately an oblate spheroid—not exactly a sphere; but the *geodetic latitude* α in Figure 2 is defined in such a way that we *can* treat Earth as exactly spherical when using α to construct NED axes. The reason is because the "down" vector at any latitude α is *defined* to be the result of rotating the "down" vector at the Equator through the angle α (around local *west*, which allows latitude to increase towards the North Pole, purely by historical convention). This is the same resulting orientation of the "down" vector that would result by rotating it simply by an angle α—or by flying an aircraft through an arc of angle α in a great circle around a spherical Earth.

The vectors $\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}$ are constructed by rotating $\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X$ respectively twice, corresponding to implementing the changes in latitude and longitude:

$$\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\} \to \{\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}\} \,. \tag{5.7}$$

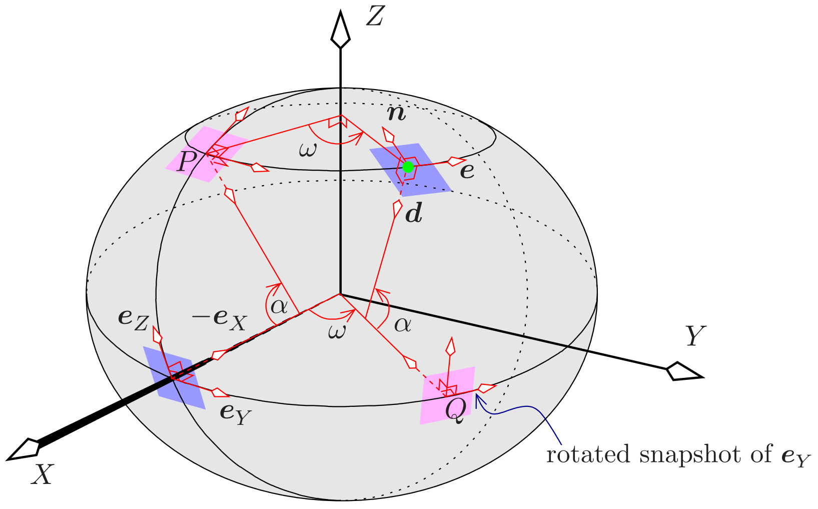

Figure 3: *Two methods for implementing the vector construction of* (5.7)*. We can rotate the original ordered set* $\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\}$ *first around* $-\boldsymbol{e}_Y$ *through* α *(or, equivalently, around* $\boldsymbol{e}_Y$ *through* $-\alpha$*) to visit point* P*, and then around* $\boldsymbol{e}_Z$ *through* ω*. Alternatively, we can rotate the original ordered set* $\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\}$ *first around* $\boldsymbol{e}_Z$ *through* ω *to visit* Q*, and then around the* rotated snapshot *of* $\boldsymbol{e}_Y$ *through* $-\alpha$*, and then use the Pseudo-Reversing Theorem to avoid having to calculate the rotated snapshot of* $\boldsymbol{e}_Y$*.*

Working in coordinates E, the relevant coordinate vectors are

$$\left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \right\} \to \left\{ [\boldsymbol{n}]_E, [\boldsymbol{e}]_E, [\boldsymbol{d}]_E \right\}. \tag{5.8}$$

Two ways of doing the rotations are immediately apparent in Figure 2. Either of these can be used, and we'll do the calculation for each way to contrast them.

First Way to Implement (5.7)**:** Recalling the positive conventions for latitude and longitude, rotate each vector of the set $\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\}$ first around $-\boldsymbol{e}_Y$ through α (or, equivalently, around $\boldsymbol{e}_Y$ through $-\alpha$) to arrive at point P in Figure 3, and then around $\boldsymbol{e}_Z$ through ω to become its corresponding vector in $\{\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}\}$. The relevant rotations are

$$\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\} \to R_Y^{-\alpha} \to R_Z^{\omega} \to \{\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}\}. \tag{5.9}$$

Second Way to Implement (5.7)**:** Alternatively, we can rotate each initial vector first around $\boldsymbol{e}_Z$ through ω to arrive at point Q in Figure 3, and then around the *rotated snapshot* of $\boldsymbol{e}_Y$ through $-\alpha$:

$$\{\boldsymbol{e}_Z, \boldsymbol{e}_Y, -\boldsymbol{e}_X\} \to R_Z^{\omega} \to R_{\langle Y\rangle}^{-\alpha} \to \{\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}\}. \tag{5.10}$$

But the Pseudo-Reversing Theorem allows us to avoid having to calculate the rotated snapshot of $\boldsymbol{e}_Y$! That is, applying it to the rotations in (5.10) produces (5.9) straightaway. You might say that there was no need to use the Pseudo-Reversing Theorem here, since we could always have chosen the First Way above of doing the rotations. That's true for this simple example, but when more than just two rotations are being used, the Second Way is usually the most natural way of approaching the rotations—and almost certainly the only easily visualised way; and then the Pseudo-Reversing Theorem becomes indispensable to avoid having to calculate the many intermediate snapshots that would be required if we were to implement the Second Way directly.

The Second Way actually implements a procedure from differential geometry called *parallel transport*: when we slide the north and east vectors from zero lat/long via point Q in Figure 3, we are parallel transporting them; the notion doesn't apply to the down vector, but it's always the cross product of north and east anyway. Parallel transport is all about movement along geodesics, which are the great circles on a sphere. Because great circles are the tracks we would fly (or drive) if we didn't turn a steering wheel, they are natural tracks on a surface, and this almost certainly accounts for their ease of use in visualising rotations.

The two rotations of (5.9) now allow us to implement (5.8) using matrices. The relevant matrices are just rotations about the axes, which are, of course, the Euler matrices of (3.19):

$$\left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \right\} \rightarrow E_2^{-\alpha} \rightarrow E_3^{\omega} \rightarrow \left\{ [\boldsymbol{n}]_E, [\boldsymbol{e}]_E, [\boldsymbol{d}]_E \right\}. \tag{5.11}$$

The three vectors can be multiplied together, somewhat in parallel. This is because matrix multiplication proceeds column by column. For suppose we have a matrix A for which

$$A \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \text{and} \quad A \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{5.12}$$

Then we can immediately write

$$A \begin{bmatrix} 1 & 10 \\ 2 & 20 \\ 3 & 30 \end{bmatrix} = \begin{bmatrix} a & p \\ b & q \\ c & r \end{bmatrix}. \tag{5.13}$$

This "block multiplication" allows us to condense the three multiplications of coordinate vectors into one matrix multiplication by coalescing each set of three coordinate vectors into a matrix:

$$\begin{aligned} \begin{bmatrix} [\boldsymbol{n}]_E & [\boldsymbol{e}]_E & [\boldsymbol{d}]_E \end{bmatrix} &= E_3^{\omega} \, E_2^{-\alpha} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} \cos\omega & -\sin\omega & 0 \\ \sin\omega & \cos\omega & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} -\cos\omega \, \sin\alpha & -\sin\omega & -\cos\omega \, \cos\alpha \\ -\sin\omega \, \sin\alpha & \cos\omega & -\sin\omega \, \cos\alpha \\ \cos\alpha & 0 & -\sin\alpha \end{bmatrix}. \end{aligned} \tag{5.14}$$

The ECEF coordinate vectors for the local north, east, and down vectors at a specific location on Earth, such as Adelaide, are now easily found. This city has latitude $\alpha = -34.9°$, longitude $\omega = 138.5°$. Use these numbers to write (5.14) as

$$\mu_E^N = \begin{bmatrix} [\boldsymbol{n}]_E & [\boldsymbol{e}]_E & [\boldsymbol{d}]_E \end{bmatrix} = \begin{bmatrix} -0.429 & -0.663 & 0.614 \\ 0.379 & -0.749 & -0.543 \\ 0.820 & 0 & 0.572 \end{bmatrix}, \tag{5.15}$$

so that the ECEF coordinates of the unit vector pointing to local north at Adelaide are $(-0.429,\ 0.379,\ 0.820)$, and similarly for east and down.

Remember now that the transpose of μ_E^N is μ_N^E, so that we have now also obtained the ECEF's basis vectors in NED coordinates:

$$\Big[\, [\boldsymbol{e}_X]_N \;\; [\boldsymbol{e}_Y]_N \;\; [\boldsymbol{e}_Z]_N \,\Big] = \mu_N^E = \left(\mu_E^N\right)^t = \Big[\, [\boldsymbol{n}]_E \;\; [\boldsymbol{e}]_E \;\; [\boldsymbol{d}]_E \,\Big]^t . \tag{5.16}$$

So, for example, the NED coordinates of $\boldsymbol{e}_X$ are $(-0.429,\ -0.663,\ 0.614)$. Equation (5.15) has the same information as equation (4.8) in [1].

Alternative methods for producing the orientation matrix (5.14) from rotations exist, depending on what vectors we take as our initial ordered set that map onto $\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}$. For example, choosing initial vectors $\boldsymbol{e}_X, \boldsymbol{e}_Y, \boldsymbol{e}_Z$ to map respectively to $\boldsymbol{n}, \boldsymbol{e}, \boldsymbol{d}$ alters the matrix multiplication of (5.14) to $E_3^\omega\, E_2^{-\alpha-90^\circ}$, which is completely consistent with the rightmost matrix in the first line of (5.14) being, in fact, $E_2^{-90^\circ}$.

Finally, the required NED coordinates of the vector pointing from Adelaide to Brussels are

$$\begin{aligned}
[\boldsymbol{r}_{BA}]_N &\overset{(3.7)}{=\!=} \mu_N^E\, [\boldsymbol{r}_{BA}]_E \overset{(3.5)}{=\!=} \left(\mu_E^N\right)^t\, [\boldsymbol{r}_{BA}]_E \\
&\overset{(5.15)}{=\!=} \begin{bmatrix} -0.429 & 0.379 & 0.820 \\ -0.663 & -0.749 & 0 \\ 0.614 & -0.543 & 0.572 \end{bmatrix} \begin{bmatrix} 7.950 \\ -3.167 \\ 8.548 \end{bmatrix} \times 10^6 \text{ metres} \\
&= \begin{bmatrix} 2.403 \\ -2.896 \\ 11.495 \end{bmatrix} \times 10^6 \text{ metres.}
\end{aligned} \tag{5.17}$$

The first two components of this vector were given in equation (4.9) of [1], and we can see that the calculations of the two reports agree. The bearing of Brussels as seen in a straight line through a transparent Earth is given by

$$-\tan^{-1}\frac{2.896}{2.403} \simeq -50.3^\circ\,. \tag{5.18}$$

5.3 The Direction in which a Pilot Sights an Aircraft

The example in Section 4.2 of [1] placed an aircraft over Adelaide, flying north-east, pitched up at 20° at an altitude of 30,000 metres. The following question was posed. We who pilot the aircraft sight another aircraft flying over Sydney at the same altitude. In which direction in our cockpit must we look to see that aircraft?

This calculation is very similar to that of Section 5.2. We construct the vector pointing from us to the other aircraft, then produce its coordinates using our aircraft's axes as opposed to the NED axes of Section 5.2. The coordinate vector pointing from us to the other aircraft is constructed in the same way as was done for Adelaide and Brussels in the previous section: begin with the locations of the Adelaide and Sydney aircraft:

Adelaide:	latitude $\alpha = -34.9^\circ$,	longitude $\omega = 138.5^\circ$,	height 30,000 m
Sydney:	latitude $\alpha = -33.9^\circ$,	longitude $\omega = 151.2^\circ$,	height 30,000 m.

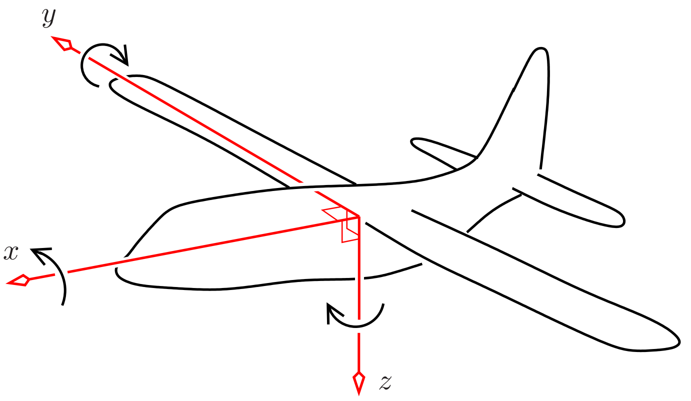

Figure 4: *Conventional labels of the cartesian body axes of an aircraft, with directions of positive rotation indicated*

Now apply (5.3) to find the position vector of the Adelaide aircraft A with respect to Earth's centre C, in ECEF coordinates E, and likewise for the Sydney aircraft S:

$$[\boldsymbol{r}_{AC}]_E = \begin{bmatrix} -3.941 \\ 3.486 \\ -3.646 \end{bmatrix} \times 10^6 \text{ metres}, \quad [\boldsymbol{r}_{SC}]_E = \begin{bmatrix} -4.666 \\ 2.565 \\ -3.554 \end{bmatrix} \times 10^6 \text{ metres}. \tag{5.19}$$

The vector pointing from us in the Adelaide aircraft to the Sydney aircraft, in ECEF coordinates E, is

$$[\boldsymbol{r}_{SA}]_E = [\boldsymbol{r}_{SC}]_E - [\boldsymbol{r}_{AC}]_E = \begin{bmatrix} -7.252 \\ -9.213 \\ 0.920 \end{bmatrix} \times 10^5 \text{ metres}. \tag{5.20}$$

We need only convert this vector to our aircraft's coordinates. Call these coordinates "*Us*", so that we require

$$[\boldsymbol{r}_{SA}]_{Us} = \mu^E_{Us} \, [\boldsymbol{r}_{SA}]_E \,. \tag{5.21}$$

Given (5.20), it remains to find

$$\mu^E_{Us} = \Big[\, [\boldsymbol{e}_X]_{Us} \;\; [\boldsymbol{e}_Y]_{Us} \;\; [\boldsymbol{e}_Z]_{Us} \, \Big]. \tag{5.22}$$

As before, label the ECEF's axes X, Y, Z, and now label our aircraft's axes x, y, z. These are shown in Figure 4. There are two main ways that we can list the procedure to construct μ^E_{Us}: one is easy to visualise but needs the Pseudo-Reversing Theorem, while the other is perhaps harder to visualise, but doesn't need the Pseudo-Reversing Theorem. There are other ways that permute the steps we outline, but they are just more of the same and needn't be considered here.

First Way to Construct μ^E_{Us}: Work in the ECEF, so invoke $\mu^E_{Us} = \left(\mu^{Us}_E\right)^t$ to allow us first to calculate μ^{Us}_E:

$$\mu^{Us}_E = \Big[\, [\boldsymbol{e}_x]_E \;\; [\boldsymbol{e}_y]_E \;\; [\boldsymbol{e}_z]_E \, \Big]. \tag{5.23}$$

This might be more naturally constructed than μ^{E}_{Us}, because μ^{Us}_{E} refers directly to the aircraft's axis vectors $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$, and these are easily constructed by rotating snapshots of the ECEF's X, Y, Z axes to "build" our aircraft's x, y, z axes in the following way. Imagine the aircraft initially at zero lat/long, with its x, y, z axes parallel to the ECEF's X, Y, Z axes respectively, balancing with its tail on the ground and with starboard wing pointing east. In the following rotations, imagine rotating the actual aircraft to bring it to the required orientation at the required place on Earth—recalling the comments of Section 5.1 and the discussion of geodetic latitude in Section 5.2. Start with snapshots of the ECEF's basis vectors $\boldsymbol{e}_X, \boldsymbol{e}_Y, \boldsymbol{e}_Z$ which will be embedded in the aircraft's body to eventually become the required $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$ vectors. Now:

1. rotate this "base" aircraft through $-90°$ around its starboard wing, being the ECEF's Y axis. This produces a new set of "latest" basis vectors, orientated so that the latest (rotated) snapshot of $\boldsymbol{e}_X$ (embedded in the aircraft's nose) points north, the latest (rotated) snapshot of $\boldsymbol{e}_Y$ (the starboard wing) points east, and the latest (rotated) snapshot of $\boldsymbol{e}_Z$ points down.

2. Now rotate these new axes (that is, the aircraft!) 138.5° around the latest snapshot of $\boldsymbol{e}_X$ (north, and the aircraft's nose), which brings them to the longitude of Adelaide, then

3. rotate them +34.9° around the latest snapshot of $\boldsymbol{e}_Y$ (starboard wing), which brings them to the correct latitude of Adelaide and now makes them the basis vectors of an aircraft flying level north over Adelaide, then

4. rotate the axes (the aircraft) 45° around the latest snapshot of $\boldsymbol{e}_Z$ (down), which yaws the aircraft to head north-east, then finally

5. rotate the axes (the aircraft) 20° around the latest snapshot of $\boldsymbol{e}_Y$ (starboard wing), which pitches the aircraft up 20°.

This sequence of rotations is

$$\mu^{Us}_{E} = R^{20°}_{\langle Y\rangle}\ R^{45°}_{\langle Z\rangle}\ R^{+34.9°}_{\langle Y\rangle}\ R^{138.5°}_{\langle X\rangle}\ R^{-90°}_{Y}\ \begin{array}{ccc} \left[\begin{matrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{matrix}\right. & & \left.\right] \\ \nearrow \quad \uparrow \quad \nwarrow & & \\ [\boldsymbol{e}_X]_E\ \ [\boldsymbol{e}_Y]_E\ \ [\boldsymbol{e}_Z]_E & & \end{array}. \tag{5.24}$$

Equation (5.24) is a recipe for constructing the basis vectors $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$. The three initial coordinate vectors $[\boldsymbol{e}_x]_E, [\boldsymbol{e}_y]_E, [\boldsymbol{e}_z]_E$ can be block multiplied by the relevant rotation matrices, and so written as columns of a matrix in (5.24)—which is just the identity matrix, and so can be excluded from the product:

$$\mu^{Us}_{E} = R^{20°}_{\langle Y\rangle}\ R^{45°}_{\langle Z\rangle}\ R^{+34.9°}_{\langle Y\rangle}\ R^{138.5°}_{\langle X\rangle}\ R^{-90°}_{Y}. \tag{5.25}$$

Now, to avoid having to calculate the intermediate "latest" vectors as was done in [1], we invoke the Pseudo-Reversing Theorem to convert all the rotations to be around the original ECEF axes, thus allowing us to use Euler matrices throughout:

$$\begin{aligned}\mu_E^{Us} &= R_{\langle Y\rangle}^{20°}\ R_{\langle Z\rangle}^{45°}\ R_{\langle Y\rangle}^{34.9°}\ R_{\langle X\rangle}^{138.5°}\ R_Y^{-90°} \\ &= R_Y^{-90°}\ R_X^{138.5°}\ R_Y^{34.9°}\ R_Z^{45°}\ R_Y^{20°} \quad \text{(Pseudo-Reversing theorem!)} \\ &= E_2^{-90°}\ E_1^{138.5°}\ E_2^{34.9°}\ E_3^{45°}\ E_2^{20°} \\ &\simeq \begin{bmatrix} -0.935 & -0.166 & 0.313 \\ -0.060 & -0.798 & -0.600 \\ 0.349 & -0.580 & 0.736 \end{bmatrix}. \end{aligned} \tag{5.26}$$

The required orientation matrix is the transpose of this:

$$\mu_{Us}^E = \begin{bmatrix} -0.935 & -0.060 & 0.349 \\ -0.166 & -0.798 & -0.580 \\ 0.313 & -0.600 & 0.736 \end{bmatrix}. \tag{5.27}$$

Second Way to Construct μ_{Us}^E: This approach calculates μ_{Us}^E directly. It doesn't need the Pseudo-Reversing Theorem, and is the same sequence described in the First Way above, but now as seen by the aircraft itself: we describe events from our viewpoint in the cockpit. This requires us to imagine that Earth (actually, the whole universe) rotates around us. We now rotate snapshots of our aircraft's x, y, z axes to the ECEF's X, Y, Z axes by doing the following.

As in the First Way above, start with the aircraft initially at zero lat/long, with its x, y, z axes parallel to the ECEF's X, Y, Z axes respectively. In the following steps, imagine rotating *Earth* to bring it to the required orientation with Adelaide below the aircraft; the aircraft is considered to be at rest throughout the procedure. Start with snapshots of the aircraft's basis vectors $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$ which will be embedded in Earth itself to eventually become the required $\boldsymbol{e}_X, \boldsymbol{e}_Y, \boldsymbol{e}_Z$ vectors. Then follow the motions of these snapshots as they rotate with Earth around us.

1. Again, the aircraft has started out balanced on its tail, so rotate Earth through $+90°$ around the aircraft's y axis. This sets the ground squarely under the aircraft.

2. Now rotate Earth by $-138.5°$ around the aircraft's nose $\boldsymbol{e}_x$, which brings the longitude of Adelaide underneath the aircraft, then

3. rotate Earth by $-34.9°$ around $\boldsymbol{e}_y$, which brings Adelaide exactly under the aircraft, then

4. rotate Earth by $-45°$ around $\boldsymbol{e}_z$, which lines north-east up ahead through the windscreen, then

5. rotate Earth by $-20°$ around $\boldsymbol{e}_y$, which pitches Earth's surface down by $20°$.

This is probably harder to visualise than the First Way above! It produces

$$\{\boldsymbol{e}_X, \dots, \boldsymbol{e}_Z\} = R_y^{-20°}\ R_z^{-45°}\ R_y^{-34.9°}\ R_x^{-138.5°}\ R_y^{90°}\ \{\boldsymbol{e}_x, \dots, \boldsymbol{e}_z\}, \tag{5.28}$$

so that

$$\begin{aligned}\mu^E_{Us} &= \Big[\, [\boldsymbol{e}_X]_{Us}\ [\boldsymbol{e}_Y]_{Us}\ [\boldsymbol{e}_Z]_{Us}\,\Big] \\ &= R_y^{-20°}\ R_z^{-45°}\ R_y^{-34.9°}\ R_x^{-138.5°}\ R_y^{90°}\ \underbrace{\Big[\, [\boldsymbol{e}_x]_{Us}\,[\boldsymbol{e}_y]_{Us}\,[\boldsymbol{e}_z]_{Us}\,\Big]}_{=\text{ identity matrix}} \\ &= E_2^{-20°}\ E_3^{-45°}\ E_2^{-34.9°}\ E_1^{-138.5°}\ E_2^{90°},\end{aligned} \tag{5.29}$$

which will give the same result as (5.26). Why? Because to invert the last expression in (5.29) in order to compare it with (5.26), we must reverse the order of its matrix factors and invert each (that is, $[ABCDE]^{-1} = E^{-1}D^{-1}C^{-1}B^{-1}A^{-1}$), and each inverts by changing the sign of its rotation angle.

Whichever way we choose to construct μ^E_{Us}, we can now write (5.21) as

$$\begin{aligned}[\boldsymbol{r}_{SA}]_{Us} &= \mu^E_{Us}\,[\boldsymbol{r}_{SA}]_E \\ &\simeq \begin{bmatrix}-0.935 & -0.060 & 0.349\\ -0.166 & -0.798 & -0.580\\ 0.313 & -0.600 & 0.736\end{bmatrix}\begin{bmatrix}-7.252\\ -9.213\\ 0.920\end{bmatrix}\times 10^5 \text{ metres} \\ &\simeq \begin{bmatrix}7.654\\ 8.016\\ 3.933\end{bmatrix}\times 10^5 \text{ metres.}\end{aligned} \tag{5.30}$$

This last coordinate vector gives the other aircraft's position relative to us in our coordinates: the other aircraft lies at a point roughly 765 km ahead, 802 km in the starboard direction, and 393 km down. From these numbers it's easy to see that we sight the aircraft at azimuth ϕ (direction positive from north to east) and elevation θ (positive upwards), where

$$\sin\phi = \frac{802}{\sqrt{765^2+802^2}}\,,\quad \cos\phi = \frac{765}{\sqrt{765^2+802^2}}\,,\quad \theta = \tan^{-1}\frac{-393}{\sqrt{765^2+802^2}}\,. \tag{5.31}$$

So $\phi \simeq 46°$ and $\theta \simeq -20°$: we in the cockpit must look to our right by 46° and down 20° to sight the aircraft flying over Sydney.

5.3.1 A Comment on Active and Passive Rotations

All the rotations of this report have been *active*, meaning that we are rotating vectors. The discussion of Section 5.3 above is essentially worded differently in some books on rotation theory, and is called the *passive rotation* approach. The idea is to narrate the sequence of rotations of the First Way on page 22 that take the aircraft from its datum orientation at zero lat/long and produce the required orientation over Adelaide, while accompanying this with a merger of the Second Way's (5.29) with (5.30). This approach allows one to ignore the fact that the rotations are being done around latest axes on page 22, in that it *requires* a sequence of rotations to always use latest axes. It rewrites each of these rotations as about a space-fixed axis through *minus* the angle turned through, giving the Euler matrices in (5.29)—which has the effect of introducing a *left*-handedness into its conventions. The resulting sequence multiplies (5.20) to give the required $[\boldsymbol{r}]_{Us}$ in (5.30).

Passive rotations are all about rotating a set of axes, in contrast to the active approach's explicitly rotating vectors. Of course, a set of axes can be considered as a set of basis vectors, so that rotating them can be done actively; this is precisely what I have done in this report. The passive approach can certainly be used to produce (5.30). But I think it becomes strained when we really wish to model the rotation of a physical object described by a set of coordinate vectors. The active approach that I have used throughout this report really rotates each vector, and we have a physical picture of the object rotating. In contrast, the passive approach doesn't explicitly rotate the object. Instead, it rotates the coordinate axes the other way, producing all coordinate vectors of the *unrotated* object in the *rotated* axis set. These now must be recognised as equalling the required coordinate vectors of the *rotated* object in the *unrotated* axis set. Users of the passive approach may or may not be aware that this is what they are doing, but newcomers to the subject can find it all very confusing.

The active-rotation approach of this report is aimed at explicitly constructing coordinate sets—of which there might be several—and then forming required coordinate vectors by finding components using the dot product. This is all neatly encapsulated in (3.7).

5.4 DIS Conversion

In the precursor [1] to the current report, I gave details of how to convert position–orientation information into and out of the IEEE standard [7] known as the *Distributed Interactive Simulation environment*, or DIS. In this section I present an alternative approach to the DIS calculations of [1], which perhaps will better disentangle the set of calculations necessary to implement the standard.

DIS position–orientation information of an object—say, an aircraft—is recorded in the following way. The aircraft's position is given by its ECEF (X, Y, Z) coordinates. Its orientation is specified by the set of three Euler angles that enable that orientation to be constructed by three rotations. These rotate snapshots of the ECEF's XYZ axes onto the aircraft's xyz axes. The three rotations are, first, by the angle ϕ around the ECEF's X, then by θ around Y, and finally by ψ around Z. The Pseudo-Reversing Theorem tells us that this is identical to rotating first by ψ around Z, then by θ around the *rotated* snapshot of Y, and finally by ϕ around the latest rotated snapshot of X; this is the sequence as it's described in the DIS standard itself, so that the three angles are usually specified in the order of ψ, θ, ϕ. Writing the orientation of the aircraft A relative to ECEF E as μ_E^A, this all means that

$$\mu_E^A = E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}\,. \tag{5.32}$$

Often, the orientation of the aircraft is specified by its location along with three non-DIS Euler angles: the angles through which snapshots of the local NED axes would be rotated to become the aircraft axes. How do we convert these Euler angles to the DIS Euler angles? We start by showing how to extract Euler angles from any orientation matrix; to be completely general for the moment, we won't yet insist on conforming to the DIS convention on the ranges that the Euler angles must take. Remember that an orientation matrix *is* a rotation matrix, as we showed in (3.24). In fact, it's not hard to show by construction [4] that any orientation can be built from three rotations conforming to the DIS standard, although we'll take such a construction as a given here.

To extract the DIS Euler angles from a rotation matrix, let's write a general rotation matrix resulting from the three DIS Euler-angle rotations above. If a snapshot of axes S is rotated through ψ, θ, ϕ in the DIS way to produce axes S', the orientation matrix is

$$\mu_S^{S'} = E_3^\psi\, E_2^\theta\, E_1^\phi = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$= \begin{bmatrix} \cos\psi\cos\theta & \substack{-\sin\psi\cos\phi \\ +\cos\psi\sin\theta\sin\phi} & \substack{\sin\psi\sin\phi \\ +\cos\psi\sin\theta\cos\phi} \\ \sin\psi\cos\theta & \substack{\cos\psi\cos\phi \\ +\sin\psi\sin\theta\sin\phi} & \substack{-\cos\psi\sin\phi \\ +\sin\psi\sin\theta\cos\phi} \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{5.33}$$

Call this matrix R, with ij^{th} element R_{ij}. We calculate each angle by stipulating its sine and cosine.

Remember this oft-neglected point: to specify an angle α, we *always* need two pieces of trigonometric information. In general, it's necessary and sufficient to specify one member of the set $\{\sin\alpha, \cos\alpha, \tan\alpha\}$, together with the *sign* of another member of this set. Usually the sine and cosine are specified, and many computer languages have a function that takes these two numbers (or any positive multiple of them) as arguments and returns α. This function is typically called `atan2`, but there is no universally agreed order for its arguments, so you should always check. Note that `atan2` is a computer term and not a mathematical function. Many people make the mistake of writing $\alpha = \tan^{-1}\frac{\sin\alpha}{\cos\alpha}$, but the function $\tan^{-1}$ is not quite defined in this way, and it will only return the correct angle if $-90° < \alpha < 90°$.

If $\cos\theta \neq 0$ (i.e. R_{31} does not equal 1 or -1), then we can see by inspecting (5.33) that

$$\begin{aligned} \sin\psi &= \frac{R_{21}}{\cos\theta}\,, \quad \cos\psi = \frac{R_{11}}{\cos\theta}\,; \\ \sin\theta &= -R_{31}\,, \quad \cos\theta = s\sqrt{1-\sin^2\theta}\,, \quad \text{where } s \in \{\pm 1\}\,; \\ \sin\phi &= \frac{R_{32}}{\cos\theta}\,, \quad \cos\phi = \frac{R_{33}}{\cos\theta}\,. \end{aligned} \tag{5.34}$$

There are two triplets of angles here, depending on an arbitrary choice of s. Also, if $\cos\theta = 0$ (i.e. R_{31} equals 1 or -1), then $\sin\theta$ can be 1 or -1 corresponding to $\theta = \pm 90°$:

$$\theta = 90°, \text{ implying } R = \begin{bmatrix} 0 & -\sin(\psi-\phi) & \cos(\psi-\phi) \\ 0 & \cos(\psi-\phi) & \sin(\psi-\phi) \\ -1 & 0 & 0 \end{bmatrix},$$

$$\text{or } \theta = -90°, \text{ implying } R = \begin{bmatrix} 0 & -\sin(\psi+\phi) & -\cos(\psi+\phi) \\ 0 & \cos(\psi+\phi) & -\sin(\psi+\phi) \\ 1 & 0 & 0 \end{bmatrix}. \tag{5.35}$$

In other words,

$$\begin{aligned} R_{31} = +1 &\Longrightarrow \theta = -90°,\ \sin(\psi+\phi) = -R_{12} = -R_{23},\ \cos(\psi+\phi) = -R_{13} = R_{22}\,, \\ R_{31} = -1 &\Longrightarrow \theta = +90°,\ \sin(\psi-\phi) = -R_{12} = +R_{23},\ \cos(\psi-\phi) = +R_{13} = R_{22}\,. \end{aligned} \tag{5.36}$$

Both of these cases yield an infinite number of solutions for ψ and ϕ, but they are *all* valid; all yield the same orientation of the aircraft.

For DIS, what of the choice of sign s in (5.34)? The two choices of s yield two sets of Euler angles, ψ_1, θ_1, ϕ_1, and ψ_2, θ_2, ϕ_2. It's not hard to show that each set is as good as the other, although we won't do that here. They are related by

$$\begin{aligned} \psi_2 &= 180^\circ + \psi_1 \,, \\ \theta_2 &= 180^\circ - \theta_1 \,, \\ \phi_2 &= 180^\circ + \phi_1 \,. \end{aligned} \tag{5.37}$$

The DIS standard actually requires

$$-180^\circ \leqslant \psi, \phi \leqslant 180^\circ, \quad -90^\circ \leqslant \theta \leqslant 90^\circ. \tag{5.38}$$

This means that $\cos\theta \geqslant 0$, which corresponds to choosing $s = 1$ in (5.34). If R_{31} does equal 1 or -1, we can solve (5.36) for whatever ψ and ϕ obey (5.38).

This last case of $\theta = \pm 90^\circ$ allows for multiple choices of ψ and ϕ, and generally any numerical algorithm that computes the Euler angles will become difficult to manage as the aircraft nears that orientation, especially if the algorithm is predicting the Euler angles. Having $\theta = \pm 90^\circ$ corresponds to the nose pointing toward one of the celestial poles—generally not an extreme orientation to fly.[7] For example, such would be the case for an aircraft on the Equator flying level and pointing north or south.

> The fact that an infinite number of such triplets will suffice for $\theta = \pm 90^\circ$ means that Euler angles can present difficulties to numerical algorithms, because they are capable of changing too quickly for any algorithm to cope with when θ nears either of these two values. As a result, they are often viewed with suspicion by their users, as if something mathematical just isn't quite right. But Euler angles do exactly what they have been set up to do, and they behave completely as they should—once we understand them, even though this behaviour can conflict with initial expectations that might misinterpret what the angles were designed to do.

Note that our extraction of the Euler angles in (5.34) didn't use all of the matrix $\mu_S^{S'}$. This is fine; being an orientation matrix, $\mu_S^{S'}$ has internal symmetry that means its elements are not all independent of one another. The omitted elements *can* be used to add some numerical strength to the calculation, but such a study in numerical stability is outside the scope of this report.

With the extraction of the DIS Euler angles under our belt, we can now construct DIS orientation information. We'll rework the example of [1] here differently to how it was done in that report. Here is the original question posed in [1]:

Convert Lat–Long–Heading–Pitch–Roll to DIS Position–Orientation Coordinates: *An aircraft is flying 10,000 m above Adelaide, pointing south-east, climbing at a 20° pitch, and holding a 30° roll. What are the two sets of triplets that the DIS standard requires to specify the aircraft's location and orientation?*

The aircraft's DIS location is its ECEF (X, Y, Z) coordinate triplet. This is found by applying (5.3) to Adelaide (lat -34.9°, long 138.5°, aircraft height 10,000 m), resulting in the first DIS triplet

$$(X, Y, Z) = (-3.928,\ 3.475,\ -3.634) \times 10^6 \text{ m}. \tag{5.39}$$

[7] I thank David Sambell of DSTO for pointing this out.

The DIS Euler angles ψ, θ, ϕ will be extracted from the orientation matrix μ_E^A of the aircraft A in the ECEF E, using (5.32) and the equations immediately after it. The ECEF's axes are X, Y, Z and the aircraft's axes are x, y, z. Remember that from (3.11), for example, $[\boldsymbol{e}_x]_E = \mu_E^A\,[\boldsymbol{e}_X]_E$, so that μ_E^A is the rotation matrix rotating snapshots of the ECEF's basis vectors onto the aircraft's basis vectors, and this is precisely the matrix from which we must extract the Euler angles using (5.34); these will be the DIS Euler angles.

We construct μ_E^A from the following sequence of rotations. Align the aircraft with its x, y, z axes coincident with the ECEF's X, Y, Z axes. Now:

1. rotate snapshots of $\boldsymbol{e}_X, \boldsymbol{e}_Y, \boldsymbol{e}_Z$ around $\boldsymbol{e}_Y$ by $-90°$,
2. rotate the resulting three vectors around the rotated version of $\boldsymbol{e}_X$ by the longitude $\omega = 138.5°$,
3. rotate the resulting three vectors around the rotated version of $\boldsymbol{e}_Y$ by *minus* the latitude, $-\alpha = +34.9°$,
4. rotate the resulting three vectors around the rotated version of $\boldsymbol{e}_Z$ by heading 135°,
5. rotate the resulting three vectors around the rotated version of $\boldsymbol{e}_Y$ by pitch 20°,
6. and finally rotate the resulting three vectors around the rotated version of $\boldsymbol{e}_X$ by roll 30°.

This is

$$\mu_E^A \;\bigl(\text{which} = E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}\bigr) = R_{\langle X\rangle}^{30°}\; R_{\langle Y\rangle}^{20°}\; R_{\langle Z\rangle}^{135°}\; R_{\langle Y\rangle}^{+34.9°}\; R_{\langle X\rangle}^{138.5°}\; R_Y^{-90°}, \tag{5.40}$$

which the Pseudo-Reversing Theorem converts to

$$\begin{aligned} \mu_E^A &= E_2^{-90°}\; E_1^{138.5°}\; E_2^{34.9°}\; E_3^{135°}\; E_2^{20°}\; E_1^{30°} \\ &= \begin{bmatrix} -0.366 & 0.928 & 0.065 \\ -0.564 & -0.165 & -0.809 \\ -0.741 & -0.333 & 0.584 \end{bmatrix}. \end{aligned} \tag{5.41}$$

The $(3, 1)$ element of this matrix is nonzero, so apply (5.34)—with $s = +1$ to select the DIS convention—to give (retaining more decimal places internally for the final calculation of the angles)

$$\begin{aligned} \sin\psi &= \frac{-0.564}{\cos\theta}\,, & \cos\psi &= \frac{-0.366}{\cos\theta}\,; \\ \sin\theta &= 0.741\,, & \cos\theta &= \sqrt{1-\sin^2\theta}\,; \\ \sin\phi &= \frac{-0.333}{\cos\theta}\,, & \cos\phi &= \frac{0.584}{\cos\theta}\,. \end{aligned} \tag{5.42}$$

These have enough information to yield (—refer to the small print on page 26)

$$\psi = -122.97°, \quad \theta = 47.79°, \quad \phi = -29.67°, \tag{5.43}$$

which are the required DIS angles. Note that we could apply (5.37) to produce the alternative set of Euler angles that correspond to taking $s = -1$ in (5.34):

$$\psi_{\text{alt}} = 57.03°, \quad \theta_{\text{alt}} = 132.21°, \quad \phi_{\text{alt}} = 150.33°. \tag{5.44}$$

As expected, θ_{alt} is not in the required DIS range of (5.38) here. (A negative cosine will place an angle into the range $90° \to 270°$. A value of $\theta = 90°$ *is* allowed by DIS, but $\cos 90° = 0$, so in that trivial case either choice of s can be used.) So the angles in (5.44) don't conform to the DIS convention. But they are *just as correct* as those of (5.43) as far as giving the aircraft's orientation is concerned.

The second DIS example that was done in [1] was the inverse of the above procedure, and we'll repeat it in greater detail now.

Convert the Above DIS Position–Orientation Coordinates of (5.39) **and** (5.43) **Back to Lat–Long–Heading–Pitch–Roll:** *An aircraft is at a location and orientation given by the above DIS coordinates*

$$\begin{aligned}(X, Y, Z) &= (-3.928,\ 3.475,\ -3.634) \times 10^6 \text{ m},\\ (\psi, \theta, \phi) &= (-122.97°,\ 47.79°,\ -29.67°)\,.\end{aligned} \tag{5.45}$$

What are its lat–long–height on Earth, and what is its orientation relative to local north–east–down in terms of the heading–pitch–roll that a pilot flies?

(Again, I have carried through more decimal places from above and have rounded them here.) The aircraft's lat–long–height comes from (X, Y, Z) and was calculated in [1], but we will cover some of the ground again here. We require to solve (5.3) for latitude α, longitude ω, and height h. There are various methods for solving these equations. The approach given in [1] is again used here: first, calculate ω from

$$\sin\omega = \frac{Y}{\sqrt{X^2+Y^2}}\,, \quad \cos\omega = \frac{X}{\sqrt{X^2+Y^2}}\,. \tag{5.46}$$

Now iterate to find α. Begin with a first estimate

$$\alpha \simeq \tan^{-1}\left(\frac{a^2}{b^2}\frac{Z}{\sqrt{X^2+Y^2}}\right), \tag{5.47}$$

with a and b from (5.2). Now refine this estimate using

$$\alpha = \tan^{-1}\left(\frac{a^2\sin^2\alpha}{b^2\sin\alpha\cos\alpha + \left(\sqrt{X^2+Y^2}\,\sin\alpha - Z\cos\alpha\right)\sqrt{a^2\cos^2\alpha + b^2\sin^2\alpha}}\right) \tag{5.48}$$

(which assumes $Z \neq 0$, but $Z = 0$ is a trivial case anyway). This expression for α converges very quickly, after which we calculate the aircraft's height h. We can use either of

$$h = \frac{\sqrt{X^2+Y^2}}{\cos\alpha} - \frac{a^2}{\sqrt{a^2\cos^2\alpha + b^2\sin^2\alpha}} \tag{5.49}$$

or

$$h = \frac{Z}{\sin\alpha} - \frac{b^2}{\sqrt{a^2\cos^2\alpha + b^2\sin^2\alpha}}\,, \tag{5.50}$$

depending on which of these formulae gives the best numerical stability; that is, if $\sin\alpha \approx 0$, use (5.49); otherwise use (5.50). Using this procedure with (X, Y, Z) from (5.45) returns the values

$$\text{latitude } \alpha = -34.90^\circ, \quad \text{longitude } \omega = 138.50^\circ, \quad \text{height } h = 10{,}000.00 \text{ m}\,, \tag{5.51}$$

meaning 10,000 metres above Adelaide, as expected.

We now find the aircraft's heading–pitch–roll relative to local north–east–down. These are precisely the three Euler angles that would be required to rotate snapshots of the north–east–down axes onto the aircraft's axes, in DIS order. They will be extracted from the orientation matrix of the aircraft relative to local north–east–down using the same approach as in the first DIS example above. The orientation matrix is μ_N^A, where A is the aircraft and N is the NED axis set. The DIS numbers refer to the ECEF E, so first use (3.9) to write

$$\mu_N^A = \mu_N^E\,\mu_E^A = \left(\mu_E^N\right)^{-1}\mu_E^A\,. \tag{5.52}$$

We calculate μ_E^N from the just-found latitude and longitude (5.51), and calculate μ_E^A directly from the DIS ψ, θ, ϕ (5.32). We showed how to construct μ_E^N in Section 5.2, but will do it again here in an abbreviated way. Take copies of the ECEF's basis vectors and rotate them first around the Y axis by -90°, then around the latest X by the longitude, then around the latest Y by minus the latitude:

$$\begin{aligned}\mu_E^N &= R_{\langle Y\rangle}^{-\alpha}\,R_{\langle X\rangle}^{\omega}\,R_Y^{-90^\circ}\\ &= E_2^{-90^\circ}\,E_1^{\omega}\,E_2^{-\alpha} \text{ by the Pseudo-Reversing Theorem,}\end{aligned} \tag{5.53}$$

remembering that this is only one of many different ways in which we could construct μ_E^N.

Equation (5.32) gives $\mu_E^A = E_3^{\psi}\,E_2^{\theta}\,E_1^{\phi}$, so (5.52) becomes

$$\begin{aligned}\mu_N^A &= \left(\mu_E^N\right)^{-1}\mu_E^A\\ &= E_2^{\alpha}\,E_1^{-\omega}\,E_2^{90^\circ}\,E_3^{\psi}\,E_2^{\theta}\,E_1^{\phi}\\ &= E_2^{-34.9^\circ}\,E_1^{-138.5^\circ}\,E_2^{90^\circ}\,E_3^{-122.97^\circ}\,E_2^{47.79^\circ}\,E_1^{-29.67^\circ}\\ &= \begin{bmatrix}-0.664 & -0.733 & 0.144\\ 0.664 & -0.491 & 0.563\\ -0.342 & 0.470 & 0.814\end{bmatrix}.\end{aligned} \tag{5.54}$$

The heading, pitch, and roll are extracted from this matrix by our realising that these three angles correspond, respectively, to the ψ, θ, ϕ that would construct this matrix in the DIS rotation order: you can picture the process as the pilot beginning with nose pointed north, straight and level, then yawing the aircraft onto a heading of ψ, pitching it up around the starboard wing by θ, then rolling it about its nose by ϕ. So we extract ψ, θ, ϕ from μ_N^A and relabel them heading, pitch, and roll respectively. Be careful not to confuse the new ψ, θ, ϕ here with the ones calculated in (5.42): they were the answer to a different question!

The $(3,1)$ element of μ_N^A is nonzero, so apply (5.34) with $s = +1$:

$$\begin{aligned}
\sin\psi &= \frac{0.664}{\cos\theta}\,, & \cos\psi &= \frac{-0.664}{\cos\theta}\,;\\
\sin\theta &= 0.342\,, & \cos\theta &= \sqrt{1-\sin^2\theta}\,;\\
\sin\phi &= \frac{0.470}{\cos\theta}\,, & \cos\phi &= \frac{0.814}{\cos\theta}\,. \qquad (5.55)
\end{aligned}$$

These lead to the expected original angles (where, as usual, I have used more decimal places internally and then rounded)

$$\text{heading} = \psi = 135.00^\circ, \quad \text{pitch} = \theta = 20.00^\circ, \quad \text{roll} = \phi = 30.00^\circ. \qquad (5.56)$$

So the exercise is finished: we have calculated the aircraft's position in (5.51) and its orientation as flown by a pilot in (5.56). We also have its orientation matrix in (5.54), which is available for use in other calculations.

6 Dead-Reckoning Aircraft Orientation

We can use the Pseudo-Reversing Theorem as an alternative way to derive the rate of change of an aircraft's orientation, given its angular velocity as a function of time. This angular velocity is typically measured by gyroscopes onboard the aircraft that make all measurements relative to its body axes. The aircraft's orientation at time t is specified by a procedure that constructs this orientation starting from some base orientation. We ask how the orientation parameters evolve as functions of the angular velocities around each of the body axes, as measured by the gyros. We'll answer the question for three different ways of quantifying orientation: the orientation matrix in Section 6.1, angle–axis using quaternions in Section 6.2, and Euler angles in Section 6.3.

For this analysis it proves useful to show that, unlike angular displacement, angular velocity is a vector, so that angular velocities can be added in any order to give a combined angular velocity. This can be proved in the following way. First, the determination of angular velocity, and the subsequent integration to calculate orientation, is a first-order process, meaning that exact expressions for rates of increase result even though we need "only" calculate to order $\mathrm{d}t$. This first-order approach greatly reduces the labour of the necessary calculations, but why does it work? When calculating aircraft orientation over time, we can work with time-dependent quantities by using Taylor's Theorem. In the more familiar language of position $s(t)$ and velocity $v(t)$ in one dimension, note that velocity is a first-order increase in position with respect to time. While we might Taylor-expand $s(t+\Delta t)$ to write

$$\begin{aligned}
v(t) &\equiv \lim_{\Delta t\to 0} \frac{s(t+\Delta t)-s(t)}{\Delta t}\\
&= \lim_{\Delta t\to 0} \frac{s(t)+s'(t)\,\Delta t+\frac{1}{2!}s''(t)\,\Delta t^2+\dots-s(t)}{\Delta t}\\
&= \lim_{\Delta t\to 0} s'(t)+\frac{1}{2!}s''(t)\,\Delta t+\dots\\
&= s'(t)\,, \qquad (6.1)
\end{aligned}$$

we can write this analysis in précis as

$$v(t) = \frac{s(t+\mathrm{d}t) - s(t)}{\mathrm{d}t} = \frac{s(t) + s'(t)\,\mathrm{d}t - s(t)}{\mathrm{d}t} = s'(t)\,. \tag{6.2}$$

Equation (6.2) is a shorthand way of writing (6.1). Writing "dt" is a way of indicating that we are effectively writing Δt + terms of order Δt^2 and higher, along with the proviso that at some stage we will divide by Δt and take a limit as $\Delta t \to 0$. In that case, terms of order Δt^2 and higher won't survive if there is a Δt present, and so we needn't bother writing those higher-order terms. We signal that we're following this procedure by writing the Δt as dt in (6.2), which is by definition *exact*.

Position is found by integrating velocity; that is, multiplying velocity by an infinitesimal time step and incrementing the previous position value,

$$s(t) = s(t_0) + \int_{t_0}^{t} v(t)\ \mathrm{d}t\,, \tag{6.3}$$

so that the process of integrating can be viewed as simple multiplication by an infinitesimal and adding.

For infinitesimal rotations expressed in some arbitrary coordinates S, (3.14) or (3.28) both give

$$\left[R_{\boldsymbol{n}}^{\mathrm{d}\theta}\right]_S = 1 + \mathrm{d}\theta\,[\boldsymbol{n}]_S^{\times}\,, \tag{6.4}$$

where "1" is the unit matrix, and we'll omit the $[\ \ ]_S$ notation for brevity in the next few lines. As noted a few paragraphs up, the way that infinitesimal notation is defined means that this is an *exact* expression: there are no higher-order terms to be written. The key point to notice is that we are now able to represent $R_{\boldsymbol{n}}^{\mathrm{d}\theta}$ by the vector $\mathrm{d}\theta\,\boldsymbol{n}$, because this allows a double rotation $R_{\boldsymbol{m}}^{\mathrm{d}\alpha}\,R_{\boldsymbol{n}}^{\mathrm{d}\beta}$ to be written as[8]

$$\begin{aligned} R_{\boldsymbol{m}}^{\mathrm{d}\alpha}\,R_{\boldsymbol{n}}^{\mathrm{d}\beta} &= (1 + \mathrm{d}\alpha\,\boldsymbol{m}^{\times})(1 + \mathrm{d}\beta\,\boldsymbol{n}^{\times}) \\ &= 1 + \mathrm{d}\alpha\,\boldsymbol{m}^{\times} + \mathrm{d}\beta\,\boldsymbol{n}^{\times} \\ &= 1 + (\mathrm{d}\alpha\,\boldsymbol{m} + \mathrm{d}\beta\,\boldsymbol{n})^{\times}, \end{aligned} \tag{6.6}$$

which can thus be represented by $\mathrm{d}\alpha\,\boldsymbol{m} + \mathrm{d}\beta\,\boldsymbol{n}$, as required if the vector representation is to be meaningful. So infinitesimal rotations behave as vectors, and combining rotations is represented by adding their vectors. Non-infinitesimal rotations don't behave as vectors because they don't commute, so they don't conform to the behaviour of vector addition, which *is* commutative. This corresponds to attempting to use $\Delta\theta\,\boldsymbol{n}$ plus higher-order terms in (6.6): those higher-order terms will prevent any identification of the rotation with $\Delta\theta\,\boldsymbol{n}$.

We're now in a position to add angular velocities by examing the effect of each over an infinitesimal time dt. Dividing the corresponding infinitesimal rotations $\mathrm{d}\theta_1\,\boldsymbol{n}_1$ and $\mathrm{d}\theta_2\,\boldsymbol{n}_2$ by dt retains their vector character; but this just produces vectors whose lengths are the corresponding angular velocities:

$$\boldsymbol{\omega}_1 \equiv \frac{\mathrm{d}\theta_1}{\mathrm{d}t}\,\boldsymbol{n}_1\,, \quad \boldsymbol{\omega}_2 \equiv \frac{\mathrm{d}\theta_2}{\mathrm{d}t}\,\boldsymbol{n}_2\,. \tag{6.7}$$

[8]It's useful in (6.6) to remember that the operation (3.15) of "applying the cross" to a vector is linear, meaning that for scalars α, β,

$$(\alpha\boldsymbol{m} + \beta\boldsymbol{n})^{\times} = \alpha\boldsymbol{m}^{\times} + \beta\boldsymbol{n}^{\times}. \tag{6.5}$$

So angular velocity is indeed a vector. Given $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$, over a time dt their combined infinitesimal rotation is represented by

$$\begin{aligned} \mathrm{d}\theta_1\, \boldsymbol{n}_1 + \mathrm{d}\theta_2\, \boldsymbol{n}_2 &= \boldsymbol{\omega}_1\, \mathrm{d}t + \boldsymbol{\omega}_2\, \mathrm{d}t && (6.8)\\ &= (\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2)\, \mathrm{d}t\,, && (6.9) \end{aligned}$$

so that the combined angular velocity is $\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2$, as expected.

With this in mind, consider the three angular velocities measured by the gyros onboard an aircraft. These make their measurements in an inertial frame that momentarily shares the aircraft's linear velocity. They report that the aircraft's angular velocity about its x axis (nose) is $p(t)$ (we'll omit the t for brevity), about its y axis (starboard wing) is $q(t)$, and about its z axis (the below-fuselage direction) is $r(t)$. In body coordinates B, the angular velocity coordinate vector about the nose is thus $(p, 0, 0)$, the angular velocity coordinate vector about the starboard wing is $(0, q, 0)$, and the angular velocity coordinate vector about the below-fuselage direction is $(0, 0, r)$. The combined angular velocity must then be the sum of these:

$$[\boldsymbol{\omega}]_B = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{6.10}$$

The Pseudo-Reversing Theorem clears up any reservations we might have about how the angular velocities are to be understood, for suppose that these velocities are visualised as rotations that act first through angle $p\,\mathrm{d}t$ around the nose $\boldsymbol{e}_x$, then through angle $q\,\mathrm{d}t$ around the *latest* wing $\boldsymbol{e}_{\langle y\rangle}$, then through angle $r\,\mathrm{d}t$ around the *latest* below-fuselage direction $\boldsymbol{e}_{\langle z\rangle}$:

$$R_x^{p\,\mathrm{d}t} \to R_{\langle y\rangle}^{q\,\mathrm{d}t} \to R_{\langle z\rangle}^{r\,\mathrm{d}t}\,. \tag{6.11}$$

The PRT says that this is equivalent to $R_x^{p\,\mathrm{d}t}\, R_y^{q\,\mathrm{d}t}\, R_z^{r\,\mathrm{d}t}$. We see that any sense of "latest" axes has now been discarded, and furthermore, these three infinitesimal rotations commute, so can be applied in any order. This is just as well, since the gyros do make their measurements simultaneously and so no rotation is necessarily "later" than any other.

We're now in a position to calculate the rate of change of an aircraft's orientation with time, as a function of $p(t), q(t), r(t)$.

6.1 Calculating Rate of Change of an Orientation Matrix

The aircraft's orientation can be specified by the matrix $\mu_W^B(t)$ whose columns are the aircraft body (B) basis vectors specified in some world coordinates W. Write the body axes as x, y, z. The body basis vectors are rotated over a time interval dt by the angular velocity $\boldsymbol{\omega} = \omega \boldsymbol{n}$, so that

$$\begin{aligned} \mu_W^B(t+\mathrm{d}t) &= \Big[[\boldsymbol{e}_x(t+\mathrm{d}t)]_W \dots [\boldsymbol{e}_z(t+\mathrm{d}t)]_W \Big] \\ &= [R_{\boldsymbol{n}}^{\omega\,\mathrm{d}t}]_W \Big[[\boldsymbol{e}_x(t)]_W \dots [\boldsymbol{e}_z(t)]_W \Big] = \Big(1 + [\boldsymbol{\omega}]_W^\times\, \mathrm{d}t\Big)\, \mu_W^B(t)\,. \end{aligned} \tag{6.12}$$

We conclude that the time derivative of the orientation matrix is

$$\dot{\mu}_W^B(t) = \frac{\mu_W^B(t+\mathrm{d}t) - \mu_W^B(t)}{\mathrm{d}t} = [\boldsymbol{\omega}]_W^\times\, \mu_W^B(t)\,. \tag{6.13}$$

But, referring to (6.10), the aircraft gyroscopes give us $[\boldsymbol{\omega}]_B$, not $[\boldsymbol{\omega}]_W$; these two coordinate vectors are related by the orientation matrix μ^B_W, but this matrix is precisely what we are trying to calculate. Address this by interchanging B and W in (6.13), remembering that reversing the viewpoint requires $\boldsymbol{\omega} \to -\boldsymbol{\omega}$, since the aircraft sees the world spin oppositely to how the world sees the aircraft spin:

$$\dot{\mu}^W_B(t) = -[\boldsymbol{\omega}]^\times_B \, \mu^W_B(t)\,. \tag{6.14}$$

But we require $\dot{\mu}^B_W$, not $\dot{\mu}^W_B$! Remedy this by noting that $\mu^W_B\,\mu^B_W = 1$, which differentiates to $\dot{\mu}^W_B\,\mu^B_W + \mu^W_B\,\dot{\mu}^B_W = 0$, or

$$\dot{\mu}^W_B\,\mu^B_W = -\mu^W_B\,\dot{\mu}^B_W\,. \tag{6.15}$$

That is, we can shift the time derivative from one factor to the other in (6.15) at the cost of a minus sign. Now post-multiply (6.14) by μ^B_W to give

$$\dot{\mu}^W_B\,\mu^B_W = -[\boldsymbol{\omega}]^\times_B\,, \tag{6.16}$$

which (6.15) changes to

$$-\mu^W_B\,\dot{\mu}^B_W = -[\boldsymbol{\omega}]^\times_B\,. \tag{6.17}$$

Premultiplying this by $-\mu^B_W$ gives

$$\boxed{\dot{\mu}^B_W = \mu^B_W\,[\boldsymbol{\omega}]^\times_B\,.} \tag{6.18}$$

[Compare this with (6.13).] Equation (6.18) is the standard expression for the time evolution of the aircraft's orientation matrix, using the gyro-supplied rotation rates p, q, r. That is, an inertial navigation system can take these rotation rates and use them to update its knowledge of the aircraft's orientation over time. An alternative derivation of (6.18) can be found in [8]. (Remember that the orientation matrix is usually called the direction-cosine matrix in the literature.) Relatively long time intervals between gyro updates can be accommodated by solving (6.18) using an efficient differential-equation solver that maintains long-term accuracy.

6.1.1 An Alternative Derivation of Equation (6.18)

Equation (6.18) can be produced in a more generic way with a careful application of the Pseudo-Reversing Theorem. Suppose $A(t)$ is some mathematical object that quantifies the orientation of the aircraft at time t by specifying how to rotate the world basis vectors W to the body basis vectors B. In practice the world basis vectors are all we have to work with, so we require $[A(t)]_W$. It might be the orientation matrix μ^B_W, but it could also be something more exotic and novel. We require the dependence of $[A(t)]_W$ on p, q, r.

By definition, the aircraft's orientation at time t is produced by having $A(t)$ act on a base orientation. In the next time step $\mathrm{d}t$, the aircraft rotates around $\boldsymbol{\omega} = p\boldsymbol{e}_x + q\boldsymbol{e}_y + r\boldsymbol{e}_z$ (remember that $\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z$ are body basis vectors), and the new orientation is specified by $A(t + \mathrm{d}t)$:

$$A(t + \mathrm{d}t) = A(t) \to \text{rotate around } \boldsymbol{\omega} = p\boldsymbol{e}_x + q\boldsymbol{e}_y + r\boldsymbol{e}_z \text{ by } \omega\,\mathrm{d}t\,. \tag{6.19}$$

The PRT converts this to rotations around the W axes x_W, y_W, z_W:

$$A(t+\mathrm{d}t) = A(t)\ \big(\text{rotate around } p\boldsymbol{e}_{x_W} + q\boldsymbol{e}_{y_W} + r\boldsymbol{e}_{z_W} \text{ by } \omega\,\mathrm{d}t\big)\,. \tag{6.20}$$

In W coordinates this is

$$[A(t+\mathrm{d}t)]_W = [A(t)]_W\ \big[\text{rotate around } p\boldsymbol{e}_{x_W} + q\boldsymbol{e}_{y_W} + r\boldsymbol{e}_{z_W} \text{ by } \omega\,\mathrm{d}t\big]_W\,. \tag{6.21}$$

The theorem doesn't alter the sense of rotation of $A(t)$, as you can see by recalling (2.10). Now, the key point to notice is that

$$\big[p\boldsymbol{e}_{x_W} + q\boldsymbol{e}_{y_W} + r\boldsymbol{e}_{z_W}\big]_W = \big[p\boldsymbol{e}_x + q\boldsymbol{e}_y + r\boldsymbol{e}_z\big]_B \tag{6.22}$$

since both equal $\begin{bmatrix} p \\ q \\ r \end{bmatrix}$. This converts (6.21) to

$$\boxed{[A(t+\mathrm{d}t)]_W = [A(t)]_W\ \big[\text{rotate around } \boldsymbol{\omega} \text{ by } \omega\,\mathrm{d}t\big]_B\,.} \tag{6.23}$$

For example, if $[A(t)]_W$ is chosen to be μ^B_W, then (6.23) becomes

$$\mu^B_W(t+\mathrm{d}t) = \mu^B_W(t)\ \Big(1 + [\boldsymbol{\omega}]^\times_B\,\mathrm{d}t\Big)\,. \tag{6.24}$$

This leads very quickly to (6.18), as expected. Equation (6.23) is a generic equation that can be applied to any choice of $[A(t)]_W$, as we'll see next when we use a quaternion to quantify the aircraft's orientation.

6.2 Calculating Rate of Change of Quaternions for Angle–Axis Representation

Set $[A(t)]_W$ to be the quaternion $Q^B_W(t)$ that rotates the W basis to the B basis. Equation (6.23) becomes a quaternion multiplication

$$Q^B_W(t+\mathrm{d}t) = Q^B_W(t)\ Q^{\omega\,\mathrm{d}t}_{[\boldsymbol{n}]_B} \tag{6.25}$$

(where $\boldsymbol{\omega} = \omega\boldsymbol{n}$), because combining multiple rotations using quaternions is accomplished by multiplying the quaternions, as we saw in (3.35). Now refer to (3.30) to write the quaternion for the infinitesimal rotation as

$$Q^{\omega\,\mathrm{d}t}_{[\boldsymbol{n}]_B} = \big(1, [\boldsymbol{\omega}]^t_B\,\mathrm{d}t/2\big) = \big(1, [p,q,r]\,\mathrm{d}t/2\big) = \underbrace{(1,0,0,0)}_{\text{identity quaternion}} + \underbrace{(0,p,q,r)}_{\equiv\,\Omega(t)}\,\mathrm{d}t/2\,. \tag{6.26}$$

Defining a *body angular velocity quaternion* $\Omega(t) \equiv (0,p,q,r)$ (that uses the angular velocities in *body* coordinates) allows (6.25) to be written using the identity quaternion $(1,0,0,0)$ (the quaternion that gives zero rotation, which can just be written as "1"):

$$\begin{aligned} Q^B_W(t+\mathrm{d}t) &= Q^B_W(t)\ \big[1 + \Omega(t)\,\mathrm{d}t/2\big] \\ &= Q^B_W(t) + Q^B_W(t)\,\Omega(t)\,\mathrm{d}t/2\,. \end{aligned} \tag{6.27}$$

This reduces to

$$\boxed{\dot{Q}^B_W(t) = \frac{1}{2}\,Q^B_W(t)\,\Omega(t)\,.} \tag{6.28}$$

An alternative derivation of this well-known differential equation is in [8].

6.3 Calculating Rate of Change of Euler Angles using Matrices

The aircraft's orientation can be specified by giving a sequence of three rotations through Euler angles that rotate the base orientation. These rotations are conventionally taken to be around either two or three different cartesian axes. $[A(t)]_W$ is usually expressed as a matrix when Euler angles are used; sometimes it's written as a quaternion. We'll use matrix language in what follows.

Choosing x, y, z to refer to e.g. ECEF axes or some other base set, and invoking the DIS convention on page 25, the rotation order becomes

$$A(t) = R_z^{\psi(t)} \to R_{\langle y \rangle}^{\theta(t)} \to R_{\langle x \rangle}^{\phi(t)}. \tag{6.29}$$

(We'll omit the explicit t dependence of ψ, θ, ϕ.) The PRT converts this to

$$A(t) = R_z^{\psi}\, R_y^{\theta}\, R_x^{\phi}, \tag{6.30}$$

so that

$$[A(t)]_W = E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}, \tag{6.31}$$

where we have replaced the generic rotations with the Euler matrices of (3.19). Equation (6.23) now becomes

$$E_3^{\psi+\mathrm{d}\psi}\, E_2^{\theta+\mathrm{d}\theta}\, E_1^{\phi+\mathrm{d}\phi} = E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}\Big(1 + [\boldsymbol{\omega}]_B^{\times}\, \mathrm{d}t\Big). \tag{6.32}$$

This equation must be solved for the time derivatives $\dot\psi, \dot\theta, \dot\phi$ (where e.g. $\dot\psi \equiv \mathrm{d}\psi/\mathrm{d}t$) in terms of p, q, r. A straightforward but tedious approach inserts the Euler matrices (3.19) into (6.32) and then multiplies the matrices to extract $\dot\psi, \dot\theta, \dot\phi$. Much easier is to use (6.4) to write

$$E_3^{\psi+\mathrm{d}\psi} = E_3^{\psi}\, E_3^{\mathrm{d}\psi} = E_3^{\psi}\left(1 + \mathrm{d}\psi \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^{\times}\right), \tag{6.33}$$

and similarly for the other matrices of the left-hand side of (6.32). These expressions allow $E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}$ to cancel from both sides of (6.32), greatly reducing further labour. The remaining manipulations are straightforward and yield (as in [8], which uses an alternative approach)

$$\boxed{\begin{bmatrix} \dot\psi \\ \dot\theta \\ \dot\phi \end{bmatrix} = \begin{bmatrix} 0 & \sec\theta\,\sin\phi & \sec\theta\,\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 1 & \tan\theta\,\sin\phi & \tan\theta\,\cos\phi \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}.} \tag{6.34}$$

This way of calculating $\dot\psi, \dot\theta, \dot\phi$ easily accommodates a change to the Euler order. Suppose that convention (6.29) was changed to a rotation about two space-fixed axes with one repeated, defining a different set of angles α, β, γ:

$$A(t) = R_z^{\alpha} \to R_y^{\beta} \to R_z^{\gamma}. \tag{6.35}$$

The required matrix multiplication is immediately written

$$[A(t)]_W = E_3^{\gamma}\, E_2^{\beta}\, E_3^{\alpha} \tag{6.36}$$

without needing the PRT, after which the calculations proceed in the same way as those following (6.31)—although the final result need not resemble (6.34).

Newcomers to the subject of aerospace orientation sometimes confuse the angular velocities p, q, r measured by gyroscopes with the Euler-angle rates of increase $\dot{\phi}, \dot{\theta}, \dot{\psi}$. They ask why the matrix in (6.34) is so complicated; shouldn't $\mathrm{d}\phi$ just equal $p\,\mathrm{d}t$, so that $\dot{\phi} = p$, and similarly for $\dot{\theta}, \dot{\psi}$?

Repeating some of the previous analysis in different language makes the situation clear. Using the DIS Euler order, the orientation at time t is produced by having $E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}$ act on the base orientation. In the next time step $\mathrm{d}t$, the aircraft simultaneously rolls about its nose (the latest x) by angle $p\,\mathrm{d}t$, pitches about the latest y by $q\,\mathrm{d}t$, and yaws about the latest z by $r\,\mathrm{d}t$. Using the PRT, the new orientation—equation (6.32)—becomes

$$E_3^{\psi}\, E_3^{\mathrm{d}\psi}\, E_2^{\theta}\, E_2^{\mathrm{d}\theta}\, E_1^{\phi} E_1^{\mathrm{d}\phi} = E_3^{\psi}\, E_2^{\theta}\, E_1^{\phi}\, E_3^{r\,\mathrm{d}t}\, E_2^{q\,\mathrm{d}t}\, E_1^{p\,\mathrm{d}t}. \tag{6.37}$$

Both sides of this equation contain the Euler rotations $E_3^{\psi}, E_2^{\theta}, E_1^{\phi}$; but whereas its left-hand side interleaves these rotations with infinitesimal rotations such as $E_3^{\mathrm{d}\psi}$, its right-hand side implements all of its infinitesimal rotations $E_1^{p\,\mathrm{d}t}, E_2^{q\,\mathrm{d}t}, E_3^{r\,\mathrm{d}t}$ first, *before* doing the three Euler rotations. But infinitesimal rotations commute only with each other and not with non-infinitesimal ones (see page 10), so we are obliged to retain the given order throughout (6.37). This implies that the infinitesimal angles are generally different on each side of that equation, so that $\mathrm{d}\phi$ does not simply equal $p\,\mathrm{d}t$. The same idea applies to $\mathrm{d}\theta$ with $q\,\mathrm{d}t$, and $\mathrm{d}\psi$ with $r\,\mathrm{d}t$. And that explains why the matrix in (6.34) is so complicated.

7 Concluding Comments

The use of the Pseudo-Reversing Theorem is part of a larger context within which vectors exist independently of coordinate systems, and yet must be expressed using coordinates for use in numerical calculations. Such expressions require the machinery of coordinate changes. We are free to calculate using any coordinates we choose, but rotations that are easy to describe and implement by an aircraft pilot are not always easy to implement mathematically, and vice versa. The PRT connects these two rotation schemes. It validates choices of coordinates and rotation order that can sometimes appear mysterious in the literature when they are accompanied by little or no explanation.

8 Acknowledgements

I thank David Sambell of DSTO for many hours of discussion, for simplifying the proof of the Pseudo-Reversing Theorem, and for other suggestions that made this report more readable; he also experimented with and demonstrated some of the above theory in various computer languages. I also thank Alice von Trojan of DSTO for her careful reading and comments on the manuscript.

References

1. D. Koks (2005, slightly updated August 2008), Using Rotations to Build Aerospace Coordinate Systems. DSTO–TN–0640, Melbourne, Vic., Defence Science and Technology Organisation (Australia).

2. M.D. Shuster (1993) A Survey of Attitude Representations, *Journal of the Astronautical Sciences*, **41** (4), pp. 439–517. See equation (117).

3. J.J. Sakurai (1985) *Modern Quantum Mechanics*, Addison-Wesley. The theorem is proved in Section 3.3 for the case of orthogonal axes, and stated in equation (3.3.18).

4. D. Koks (2006) *Explorations in Mathematical Physics*, Springer, New York. See Chapter 4. Two alternative expressions for $[R^{\theta}_{\boldsymbol{n}}]_S$ are given in this text: equations (4.22) and (4.33). The $[\ ]_S$ notation is suppressed for simplicity because only one set of coordinates is used.

5. M.W. Soijer (2009) Rotations as Double Reflections and Geometrical Derivation of Euler–Rodrigues Parameters, *Journal of Guidance, Control, and Dynamics*, **32** (1). See equation (1).

6. R.N. Bracewell (1986) *The Hartley Transform*, Oxford University Press, Inc., New York.

7. IEEE Standard for Distributed Interactive Simulation—Application Protocols, IEEE Standard 1278.1-1995 (1995)

8. P.H. Zipfel (2000) *Modeling and Simulation of Aerospace Vehicle Dynamics*, AIAA Inc., Virginia. Our equations (6.18), (6.28), and (6.34) are listed on pages 121 and 128 of this book.

Appendix A An Application of Orientation Matrices to Measured Wind in Ship Motion

Section 3 of this report stressed the distinction between proper vectors and coordinate vectors, and used (3.7) to convert one coordinate representation to another. This view of vectors is very useful for untangling the concepts involved with the wind that sailors feel to be flowing over their ship. In this appendix we'll use that notation to answer the following often-asked question:

Calculating a Desired Ship Course: *A ship is sailing at some given speed on some given compass bearing. It measures the speed and direction of the wind flowing over it. What new course should it sail so that the new speed and direction of the wind over the ship become some given values?*

Use the following notation.

– The compass bearing is always relative to local north–east–down, so let N stand for these coordinates and for the general reference frame attached to these: e.g., think of N as also representing some nearby island.

– W stands for the wind.

– S stands for the current ship and its coordinates. We'll take the x axis to point forward of the prow, and the y axis to point to starboard.

– S' stands for the "new ship" and its coordinates: those of the new orientation that we require to sail. The axes of this "new ship" will be just as for the current ship, but labelled with primes: x' pointing forward of the prow and y' pointing starboard.

– $\boldsymbol{v}_{AB}$ is the velocity of object A relative to object B, and is a proper vector. So for objects A, B, C,

$$\boldsymbol{v}_{AB} = \boldsymbol{v}_{AC} + \boldsymbol{v}_{CB}\,. \tag{A1}$$

It's worthwhile covering some established terminology relating to the wind.

True wind is $\boldsymbol{v}_{WN}$, the actual motion of the wind over Earth's surface, and is a proper vector. We can think of it as the motion of the wind relative to anything at rest in the local north–east–down system, such as a nearby island.

Measured wind is $\boldsymbol{v}_{WS}$, the velocity of the wind relative to the ship: a proper vector.

Apparent wind is $[\boldsymbol{v}_{WS}]_N$, the measured wind in north–east–down coordinates: a coordinate vector.

Relative wind is $[\boldsymbol{v}_{WS}]_S$, the measured wind in current ship coordinates: a coordinate vector.

Note that there are only two proper wind vectors here: the true wind (velocity relative to an island) and the measured wind (velocity relative to the ship). The apparent and relative winds are *coordinate* vectors: just two different ways of quantifying the measured wind.

Now, both the "current ship" and "new ship" agree on what the true wind is (assuming it didn't change during the manoeuvre, of course), because the true wind is a proper vector and so has a reality independent of any ship. So we can write the true wind vector as a sum of terms using the current ship or the new ship, using (A1):

$$\underbrace{\boldsymbol{v}_{WN}}_{\text{true wind}} = \underbrace{\boldsymbol{v}_{WS}}_{\substack{\text{current wind}\\\text{over ship}}} + \underbrace{\boldsymbol{v}_{SN}}_{\substack{\text{current ship}\\\text{velocity}}} = \underbrace{\boldsymbol{v}_{WS'}}_{\substack{\text{desired wind}\\\text{over ship}}} + \underbrace{\boldsymbol{v}_{S'N}}_{\substack{\text{new ship}\\\text{velocity}}} . \tag{A2}$$

To answer the main question posed above, we seek the new ship course $\boldsymbol{v}_{S'N}$, which comes from (A2):

$$\boldsymbol{v}_{S'N} = \boldsymbol{v}_{WS} + \boldsymbol{v}_{SN} - \boldsymbol{v}_{WS'} . \tag{A3}$$

This is the basic vector sum that will be coordinatised with respect to either north–east–down or the ship, depending on what information we have been given. The main question above requires us to coordinatise it as follows, using (3.7):

$$\begin{aligned}[\boldsymbol{v}_{S'N}]_N &= [\boldsymbol{v}_{WS}]_N + [\boldsymbol{v}_{SN}]_N - [\boldsymbol{v}_{WS'}]_N \\ &= \mu^S_N \underbrace{[\boldsymbol{v}_{WS}]_S}_{\text{given}} + \underbrace{[\boldsymbol{v}_{SN}]_N}_{\text{given}} - \mu^{S'}_N \underbrace{[\boldsymbol{v}_{WS'}]_{S'}}_{\text{given}} .\end{aligned} \tag{A4}$$

The first "given" term in (A4) is the current relative wind, which is supplied in the question. The second "given" term is the current ship course. The third "given" term is the desired relative wind.

Equation (A4) is not straightforward to solve, because the desired ship course on the left-hand side of (A4) establishes the new orientation via S', but this orientation is required in the last term on the right-hand side of (A4). However, we can find an exact solution.

Begin by specifying all coordinates. We who stand on the ship measure the wind coming at us with speed v_1 from an azimuth β_1 measured clockwise from the prow:

$$[\boldsymbol{v}_{WS}]_S = \begin{bmatrix} -v_1 \cos\beta_1 \\ -v_1 \sin\beta_1 \end{bmatrix} . \tag{A5}$$

The velocity of our ship in the north–east–down coordinates is speed v_2 along compass bearing β_2:

$$[\boldsymbol{v}_{SN}]_N = \begin{bmatrix} v_2 \cos\beta_2 \\ v_2 \sin\beta_2 \end{bmatrix} . \tag{A6}$$

We require to turn such that we'll measure the wind to be coming at us with speed v_3 from an azimuth β_3 measured clockwise from the prow:

$$[\boldsymbol{v}_{WS'}]_{S'} = \begin{bmatrix} -v_3 \cos\beta_3 \\ -v_3 \sin\beta_3 \end{bmatrix} . \tag{A7}$$

The sought-after new ship velocity is described by some number v along compass bearing β:

$$[\boldsymbol{v}_{S'N}]_N = \begin{bmatrix} v \cos\beta \\ v \sin\beta \end{bmatrix} . \tag{A8}$$

There is nothing to prevent v from being negative, so we'll refer to it as a velocity rather than as a speed. For shorthand, set for $i = 1, 2, 3$,

$$\begin{aligned} c &\equiv \cos\beta\,, & s &\equiv \sin\beta\,, \\ c_i &\equiv \cos\beta_i\,, & s_i &\equiv \sin\beta_i\,. \end{aligned} \tag{A9}$$

Equation (A4) requires the orientation matrices μ_N^S and $\mu_N^{S'}$. We need to know the ship's orientation, so will assume it moves without side-slipping. The notation can handle any orientation at all, but ships do usually move this way. If the ship currently points into the bearing direction β_2, then

$$\mu_N^S = \Big[[\boldsymbol{e}_x]_N \;\; [\boldsymbol{e}_y]_N\Big] = \begin{bmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{bmatrix}. \tag{A10}$$

Likewise, we'll assume the "new ship" points into the bearing direction β:

$$\mu_N^{S'} = \Big[[\boldsymbol{e}_{x'}]_N \;\; [\boldsymbol{e}_{y'}]_N\Big] = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}. \tag{A11}$$

With these, (A4) becomes

$$v\begin{bmatrix} c \\ s \end{bmatrix} = \underbrace{-v_1 \begin{bmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{bmatrix}\begin{bmatrix} c_1 \\ s_1 \end{bmatrix} + v_2\begin{bmatrix} c_2 \\ s_2 \end{bmatrix}}_{\text{call this } \begin{bmatrix} a \\ b \end{bmatrix}} + \underbrace{v_3 \begin{bmatrix} c & -s \\ s & c \end{bmatrix}\begin{bmatrix} c_3 \\ s_3 \end{bmatrix}}_{= v_3 \begin{bmatrix} c_3 & -s_3 \\ s_3 & c_3 \end{bmatrix}\begin{bmatrix} c \\ s \end{bmatrix}}, \tag{A12}$$

which can be rewritten as

$$\left(v.1 - v_3 \begin{bmatrix} c_3 & -s_3 \\ s_3 & c_3 \end{bmatrix}\right)\begin{bmatrix} c \\ s \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix}, \tag{A13}$$

where by "$v.1$" we mean v times the 2×2 identity matrix. This last equation becomes

$$\begin{bmatrix} v - v_3 c_3 & v_3 s_3 \\ -v_3 s_3 & v - v_3 c_3 \end{bmatrix}\begin{bmatrix} c \\ s \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix}. \tag{A14}$$

This must be solved for v, c, and s. To do so, equate the squared lengths of each side of (A14), remembering that for a matrix A and a column (coordinate vector) $\boldsymbol{x}$, $|A\boldsymbol{x}|^2 = \boldsymbol{x}^t A^t A \boldsymbol{x}$. (Remember too that the elements of $A^t A$ are just dot products of the columns of A with themselves in turn.) Writing the squared length of $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right]$ as k^2, we obtain

$$\begin{bmatrix} c & s \end{bmatrix} \begin{bmatrix} (v - v_3 c_3)^2 + v_3^2\, s_3^2 & 0 \\ 0 & (v - v_3 c_3)^2 + v_3^2\, s_3^2 \end{bmatrix} \begin{bmatrix} c \\ s \end{bmatrix} = k^2. \tag{A15}$$

In other words,

$$(v - v_3 c_3)^2 + v_3^2\, s_3^2 = k^2, \tag{A16}$$

which is easily rewritten to give the desired new ship velocity as

$$v = v_3 c_3 \pm \sqrt{k^2 - v_3^2\, s_3^2}\,. \tag{A17}$$

There could be any of zero, one or two possible real values of v resulting here. Now we need only use whatever values of v we have obtained here to invert (A14), making use of (A16):

$$\begin{aligned}\begin{bmatrix} c \\ s \end{bmatrix} &= \begin{bmatrix} v - v_3 c_3 & v_3 s_3 \\ -v_3 s_3 & v - v_3 c_3 \end{bmatrix}^{-1} \begin{bmatrix} a \\ b \end{bmatrix} \\ &= \frac{1}{k^2} \begin{bmatrix} v - v_3 c_3 & -v_3 s_3 \\ v_3 s_3 & v - v_3 c_3 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}. \end{aligned} \tag{A18}$$

This last equation gives us $\cos\beta$ and $\sin\beta$ and hence β itself, remembering the comment on page 26 that two pieces of trigonometric information are always needed to specify an angle. So the calculation is finished.

As an example, suppose we on the ship measure the wind coming at us with speed 15 units (it doesn't matter which units we use, as long as they're all the same) from an azimuth 140° measured clockwise from the prow:

$$v_1 = 15\,, \quad \beta_1 = 140°. \tag{A19}$$

Our ship has speed 20 over the ocean, along compass bearing 270°:

$$v_2 = 20\,, \quad \beta_2 = 270°. \tag{A20}$$

We require to turn such that we'll measure the wind to be coming at us with speed 10 from an azimuth 44° measured clockwise from the new position of the prow:

$$v_3 = 10\,, \quad \beta_3 = 44°. \tag{A21}$$

The required ship velocity is v along compass bearing β. Use (A12) to calculate $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right]$, then find its squared length k^2, then apply (A17) to produce v. Finally calculate β from its cosine and sine in (A18). Two pairs of v, β result:

$$v = 39.4\,, \quad \beta = 265.2°, \tag{A22}$$

and

$$v = -25.0\,, \quad \beta = 60.8°. \tag{A23}$$

The first pair describes the ship moving with speed 39.4 into compass direction 265.2°, and is one solution to the original question posed.

The second pair cannot be read as describing a speed of 25.0 into a compass direction $180° + 60.8° = 240.8°$. Rather, we used the new direction of ship travel to specify its orientation in (A11): we assumed that the ship points in the direction of its velocity, or compass direction β. That means the second pair of numbers (A23) must describe a ship travelling *backwards* with speed 25, while pointing in the compass direction 60.8°. Such a ship would indeed feel the wind coming with speed 10 from an azimuth 44° measured clockwise from its prow; but no ship is going to move in reverse like this. Even so, (A23) *is* a valid solution to the question posed.

Index

Italicised numbers denote where the entry is defined.

This page is intentionally blank.

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA	1. CAVEAT/PRIVACY MARKING

2. TITLE A Pseudo-Reversing Theorem for Rotation and its Application to Orientation Theory	3. SECURITY CLASSIFICATION Document (U) Title (U) Abstract (U)
4. AUTHOR Don Koks	5. CORPORATE AUTHOR Defence Science and Technology Organisation PO Box 1500 Edinburgh, SA 5111, Australia

6a. DSTO NUMBER DSTO–TR–2675	6b. AR NUMBER AR–015–246	6c. TYPE OF REPORT Technical Report	7. DOCUMENT DATE March, 2012

8. FILE NUMBER 2011/1121995/1	9. TASK NUMBER N/A	10. TASK SPONSOR DSTO	11. No. OF PAGES 43	12. No. OF REFS 8

13. URL OF ELECTRONIC VERSION `http://www.dsto.defence.gov.au/publications/scientific.php`	14. RELEASE AUTHORITY Chief, Electronic Warfare and Radar Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DSTO RESEARCH LIBRARY THESAURUS

Orientation	Axes of rotation	Coordinates
Space navigation	Ship navigation	Quaternions

19. ABSTRACT

We state and prove a useful theorem on manipulating rotation order which, while not new, is barely present in the literature. This theorem allows the order of a sequence of rotations to be reversed, provided that the sense of the axes of rotation is changed from “body” to “space fixed” or vice versa. We use the theorem to aid calculations in geodesy (constructing a local north–east–down coordinate system) and aerospace theory (relating yaw–pitch–roll rates to vehicle angular velocity). The new notation here sheds light generally on the field of orientation theory, as well as giving insight to standard terms relating to wind direction used for treating ship motion. Although we present our analyses in the style of a tutorial in the general subject of spatial orientation theory, there is new notation here, along with alternative and novel ways of treating problems that are often seen as difficult or obscure by practitioners. This report follows on from the 2005 DSTO report DSTO–TN–0640, but is completely self contained, and DSTO–TN–0640 need not be read beforehand.

Page classification: UNCLASSIFIED