Module 2, B-set 1: Orientation of a Body and Particle Motion

Adjusting Your Attitude

November 3, 2016

Context

In order to carry out its mission, a satellite will undoubtedly have to regularly change its orientation and position. For example, it may have to rotate to keep its solar panels facing the sun or an antenna linked to a ground station. It may have to change orbits to take measurements. In order to control position and orientation, we first need to understand how to represent these quantities and how they can be changed.

Key Concepts

By the end of this exercise you should be familiar with the following ideas

- Using Euler Angles to describe the orientation of a body
- Gimbal lock-what is it, when it occurs, and problems associated with it
- Body-fixed vs.space-fixed reference frames
- The process for calculating the 3D-motion of a particle
- Velocity and acceleration of a particle written with respect to a cylindrical coordinate system or reference frame.

Euler Angle Review (15 minutes)

Let's start by reviewing the concept of Euler Angle by deriving the rotation matrix for the 3-2-1 (yaw-pitch-roll), body-fixed set of Euler (or more precisely, Tait-Bryant) Angles. These Euler Angles are commonly used with aircraft. The first rotation is ψ about the z(3-yaw, or down from class) axis. The second rotation θ about the (new) y'(2-pitch, or right in class) axis. And the third rotation is ϕ about the (new) x''(1-roll, front from class) axis. Note that there is unfortunately no single, consistent notion for 3-2-1 Euler Angles. In some references, the names of the angles are changed so that the first rotation is called ϕ and the third is called ψ (which is the order used for the 3-1-3 rotations!).

1. **What's the Matrix?** Following the procedure that you used in class, find $[R]_{3-2-1}$ which transforms an unprimed reference frame to the triple-prime reference frame as

$$\begin{bmatrix} x''' \\ y''' \\ z''' \end{bmatrix} = [R]_{3-2-1} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

where

$$[R]_{3-2-1} = [R]_{\phi}[R]_{\theta}[R]_{\psi}. \tag{2}$$

Building a Three-axis Gimbal (2 hours)

Resources
- What is a gimbal?
- Gimbals on rockets

In order to better understand 3D rotation and orientation, you are going to build a device called a gimbal that can rotate an object mounted on it about an axis. Gimbals, invented in antiquity, are now used in camera stabilization, gyroscope, and inertial navigation systems.

If you're taking or have taken POE, this project will seem familiar. In fact, you're going to build a variation of the pan-tilt scanner from Lab #2. You should be able to modify the one that you built for POE for this project. If you have not or are not taking POE, you should partner with someone who is and work this together with them.

2. **Build It.** Working with a partner, you are to build a simple 3-axis gimbal using three servo motors controlled by an arduino (your own) that will allow you to orient your little aircraft in a specified attitude (attach your aircraft to the gimbal). Note that the servos have a limited range of travel so some orientations will not be accessible. Don't spend a lot of time building a high-fidelity mechanism. An easy-to-build demonstration unit is all you need (i.e., duct tape and cardboard is perfectly OK).

3. **Program it.** Write code for your 3-axis gimbal so that you can rotate any axis (servo) by a specified amount (within the servo's range of rotation).

4. **Play with it.** Convince yourself that you can get go from an initial orientation to a final orientation with a minimum of three sequential, single-axis rotations.

5. **Prove it.** Show (on paper) that you can go from an initial orientation to a final orientation by at least two different sets of rotations. In other words, show that there exist at least two different sets of Euler Angles matrices that take you to a given final orientation.

6. **Compare it.** Using the (incomplete) MATLAB script, rotate321.m and blockrate.m, you can you visualize rotations. You have to have to define the proper single-axis rotation matrices in blockrate.m. Using this code and your newly derived 3-2-1 rotation matrix, verify that you can predict the behavior or your 3-axis gimbal. For a couple of different rotations, run the code to produce a predicted orientation, and then take picture of your airplane on the gimbal showing that you can generate the same orientation with your gimbal.

7. **Fly it (optional).** Program your gimbal to "fly" your airplane in a smooth non-planar flight path, *e.g.*, a banked turn, by using a series of small discrete rotations. Post a video of this to youtube.

Gimbal Lock or Houston, We Have a Problem (45 minutes)

During the Apollo 11 mission when the Lunar Module was docking with the Command Module, it inadvertently went into **gimbal lock** and started to tumble. Astronauts Neil Armstrong and Buzz Aldrin had to abort the attitude control system and take manual control in order to recover.

8. **Define it.** What is gimbal lock? Discuss its impact on animating rotations using Euler angles.

This is gimbal lock

9. **Find it.** At what angles does gimbal lock occur for the 3-1-3 Euler angles? What about for the 3-2-1 Euler (Tait-Bryan) angles?

Particle Motion Process Overview

In the next section, we're going to shift gears and look at translation in space. Here is a general process outline of steps to take to determine the position of a body in space based on the forces that are applied to it.

- Start with Newton's Second Law in the form of a single vector equation $\vec{F} = m\vec{a}$.
- Choose a convenient reference frame or coordinate system in which to describe the acceleration, $\vec{a}$.
- Draw a free body diagram of the body and identify all of the forces you want to consider.
- Write out $\vec{F} = m\vec{a}$ in terms of the unit vectors of your chosen coordinate systems.

- Extract three separate scalar equations. You may need to write the vectors in terms of a common reference frame. The equations will be differential equations because we want position which is the second derivative of acceleration.
- If applicable, apply constraint conditions. You may be able to eliminate some parameters from the equations of motion. In some cases, you will have separate constraint equations.
- Solve for equations of motion for position (and velocity). Usually this will have to be done numerically. A set of initial conditions will need to be specified.
- Validate your simulation (*e.g.*, do a parameter study, compare to experiments, check against another model).
- If applicable, use the positions and velocities to find accelerations or forces of interest.
- Do a happy dance.

A More Sophisticated Projectile Simulation or Dust in the Wind. All They are is Dust in the Wind. (3 hours)

In this exercise, you will expand the capability of the projectile simulation that you started in class by including effects of aerodynamic drag and wind. As shown in the figure, the projectile is launched from point P with an initial velocity vector, $\vec{V_0} = V_x\hat{\imath} + V_y\hat{\jmath} + V_z\hat{k}$. The wind is blowing with velocity, $\vec{V_w} = W_x\hat{\imath} + W_y\hat{\jmath} + W_z\hat{k}$. The aerodynamic drag force acting on the projectile can be described as

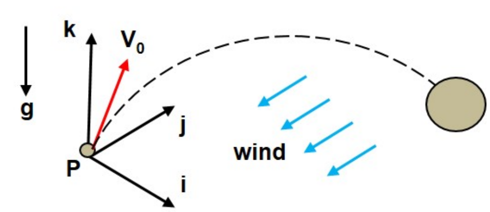

Figure 1: The projectile is launched with initial velocity, V_0, and is subject to aerodynamic drag force and the wind.

$$\vec{F_D} = -\frac{1}{2}\rho C_D \frac{\pi D^2}{4} V \left\{ (v_x - W_x)\,\hat{\imath} + \left(v_y - W_y\right)\hat{\jmath} + (v_z - W_z)\,\hat{k} \right\} \quad (3)$$

where ρ is the density of the air, D is the diameter of the projectile (assumes it is spherical), C_D is the (empirical) drag coefficient, and V is the magnitude of the projectile's velocity relative to the air

$$V = \sqrt{(v_x - W_x)^2 + \left(v_y - W_y\right)^2 + (v_z - W_z)^2} \quad (4)$$

10. Draw a **free-body diagram** of the projectile. Clearly label all of the external forces that you need to consider.
11. Write out a single, second order **vector differential equation** that governs the motion of the ball.
12. Extract three **scalar, second-order differential equations** of motion by collecting terms with the same unit vectors. For example, if you

had the first order vector differential equation $\frac{d\vec{y}}{dt} = Ay\hat{\imath} + By^2\hat{\jmath}$, you'd translate this to $\frac{dy_x}{dt} = Ay$ and $\frac{dy_y}{dt} = By^2$.

13. Rewrite those equations as set of **first order differential equations**. Recall that one second order DE can be re-written as two first orders.

14. **Implement** the differential equations by modifying your existing simulation to calculate and plot the position, velocity, and acceleration of the projectile as a function of time and the trajectory of the projectile in 3D space. Stop the simulation when the ball hits the ground, *e.g.*, use MATLAB's event function.

15. Using representative parameters (m = 0.05 kg, C_D = 0.25, D = 4.27 cm, $\rho = 1.29 kg/m^3$), **run simulations** of the projectile launched in the y-z plane with an initial speed of 130 mph at an angle of 45 degrees with respect to the y axis. Calculate the trajectory and create plots for

    (a) A 15 mph head wind (in the negative y-direction)

    (b) A 15 mph tail wind(in the positive y-direction). How much farther does the projectile travel compared to the first case?

    (c) A 15 mph side wind (in positive x-direction).

16. **Validate** your simulation by running some cases in which the initial and wind velocity vectors have components in all three directions. Show evidence that demonstrates your code obeys the right physics. For example, you can vary system parameters or inputs to see if the output behaves in ways that make sense. For example, increase the magnitude of the initial velocity. Does the projectile go farther? Another way to provide some validation is to try limiting cases. For example, turn off the wind (set $C_D = 0$). Think up some test cases and document your results with plots and supporting text.

Cylindrical Coordinate Systems (1 hour)

You were working in a rectangular coordinate system as you created your projectile simulations. At some point you have probably used a cylindrical coordinate system to define the velocity and acceleration of a mass or body. At least, you have used polar coordinates (r, θ) to define the position of a point in a plane. A cylindrical coordinate system is simply a polar coordinate system with an axial direction (called 'z'). The values of r and θ locate the projection of point P (the location of the mass) in a plane (for easy comparison let's make it the x-y plane defined by the unit vectors $\vec{i}$ and $\vec{j}$. The z value gives the normal distance of P from the plane. The point P′ is the normal projection of point P on the x-y plane.

The cylindrical coordinate system is defined using the three unit vectors, $\widehat{e}_r$, $\widehat{e}_\theta$, and $\widehat{e}_z$ where

- $\widehat{e}_r$ is aligned along the line OP' in the direction of P′.
- $\widehat{e}_\theta$ lies in the x-y plane and is perpendicular to $\widehat{e}_r$in the direction of increasing θ.
- $\widehat{e}_z$ is normal to both $\widehat{e}_r$, $\widehat{e}_\theta$ forming a right-hand coordinate system.

It is important to note that $\widehat{e}_r$, $\widehat{e}_\theta$ change direction with changes in θ but not r and z. So if point P moves in a radial or vertical line then the direction of the two unit vectors remain constant. If point P moves such that θ changes, then $\widehat{e}_r$, $\widehat{e}_\theta$ change direction.

The rectangular and cylindrical coordinates are related through

$$x = r\cos\theta, \qquad y = r\sin\theta, \qquad \text{and} \qquad z = z \tag{5}$$

Keep in mind that, in general, r, θ, and z can be functions of time. The **position** of the mass at point P with respect to point O is defined by the vector

$$\vec{r}_{P/O} = r\widehat{e}_r + z\widehat{e}_z \tag{6}$$

17. **Explain the math.** Why is there no $\widehat{e}_\theta$ term in Equation (6)? Don't you need three coordinates to specify a location in 3D space? Where's the third piece of location information?

Taking the time derivative of $\vec{r}_{P/O}$ with respect to the reference frame centered at point O, we get the velocity of point P. Note, that the reference frame in which the derivative is taken is specified. Here there is only one reference frame so the notation is not necessary but in the future we'll be dealing with multiple reference frames and this notation will be critical.

$$\left.\frac{d}{dt}\right|_O \vec{r}_{P/O} = \vec{v}_{P/O} = \dot{r}\widehat{e}_r + r\dot{\widehat{e}}_r + \dot{z}\widehat{e}_z + z\dot{\widehat{e}}_z \tag{7}$$

where the overhead dots on the right-hand side denote the same time derivative as taken on the left-hand side. What do we do with the time derivatives of the unit vectors? Well, we know that $\dot{\widehat{e}}_z = 0$ because the magnitude and direction of $\widehat{e}_z$ never changes. It's magnitude is always equal to one and it always points in the vertical direction. This is not the case for $\widehat{e}_r$ because, in general, its direction changes with time. Can it be written in terms of a unit vector without the time derivative? Let's use the transformation relationship

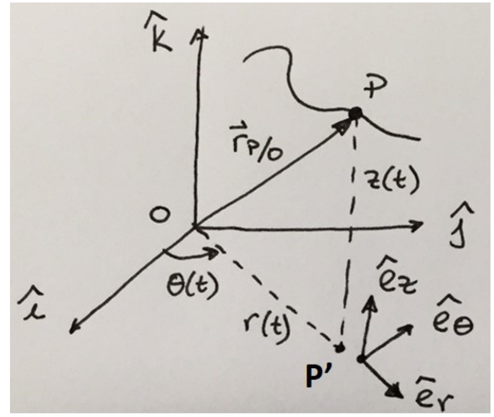

Figure 2: The position of the particle located at point P can be described in terms of a rectangular coordinate system (with unit vectors, $\vec{i}$, $\vec{j}$, and $\vec{k}$) or a cylindrical coordinate system (with unit vectors $\widehat{e}_r$, $\widehat{e}_\theta$, and $\widehat{e}_z$). The vertical projection of point P onto the x-y plane, is P′. Its location can be specified by r and θ.

The following videos regarding derivatives of unit vectors in polar coordinates may be useful:

- Vector Time Derivatives in Cartesian Coordinates
- Vector Time Derivatives in Polar Coordinates I
- Vector Time Derivatives in Polar Coordinates II

between the rectangular and cylindrical coordinate systems. From Figure 3, we can see that the rectangular and cylindrical coordinate systems are related through a 2D rotation which can be written as

$$\begin{bmatrix} \hat{e}_r \\ \hat{e}_\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \hat{i} \\ \hat{j} \end{bmatrix} \tag{8}$$

$$\hat{e}_r = \cos\theta\hat{i} + \sin\theta\hat{j} \quad \text{and} \quad \hat{e}_\theta = -\sin\theta\hat{i} + \cos\theta\hat{j} \tag{9}$$

taking the time derivative on both sides

$$\dot{\hat{e}}_r = -\dot{\theta}\sin\theta\hat{i} + \dot{\theta}\cos\theta\hat{j} = \dot{\theta}\hat{e}_\theta \tag{10}$$

$$\dot{\hat{e}}_\theta = -\dot{\theta}\cos\theta\hat{i} - \dot{\theta}\sin\theta\hat{j} = -\dot{\theta}\hat{e}_r \tag{11}$$

Substituting Equation 10 into Equation 7 gives the **velocity**

$$\left.\frac{d}{dt}\right|_O r_{P/O} = v_{P/O} = \dot{r}\hat{e}_r + r\dot{\theta}\hat{e}_\theta + \dot{z}\hat{e}_z \tag{12}$$

18. **Show** that the acceleration of point P is

$$\left.\frac{d^2}{dt^2}\right|_O r_{P/O} = a_{P/O} = (\ddot{r} - r\dot{\theta}^2)\hat{e}_r + (2\dot{r}\dot{\theta} + r\ddot{\theta})\hat{e}_\theta + \ddot{z}\hat{e}_z \tag{13}$$

by taking the derivative of Equation 12. The term $(2\dot{r}\dot{\theta}\hat{e}_\theta)$ is called the *Coriolis* acceleration. Note that it is a sum of terms arising from two separate sources. **Explain** what this terms represents. Describe some physical situations in which Coriolis acceleration is important.

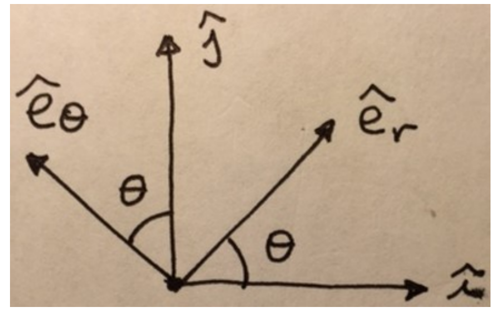

Figure 3: For easy comparison, the rectangular and cylindrical coordinate systems can be drawn with a common origin.

Take a look at Bad Coriolis.

A Geometric Derivation.

The time derivatives of the unit vectors can also be determined geometrically. In Figure 4, a point is shown at two locations P and P' with corresponding radial unit vectors $\hat{e}_(\theta)$ and $\hat{e}_r(\theta + \Delta\theta)$, respectively. The vectors can be drawn tail-to-tail for easy comparison. The angle between them is $\Delta\theta$ and the distance between their tips is $\Delta\hat{e}_r$

Take the time derivative of $\hat{e}_r$ and apply the chain rule

$$\dot{\hat{e}}_r = \frac{d\hat{e}_r}{dt} = \frac{d\hat{e}_r}{d\theta}\frac{d\theta}{dt} \tag{14}$$

where by definition

$$\frac{d\hat{e}_r}{d\theta} = \lim_{\Delta\theta\to 0}\left[\frac{\hat{e}_r(\theta+\Delta\theta) - \hat{e}_r(\theta)}{\Delta\theta}\right] = \lim_{\Delta\theta\to 0}\frac{\Delta\hat{e}_r}{\Delta\theta}. \tag{15}$$

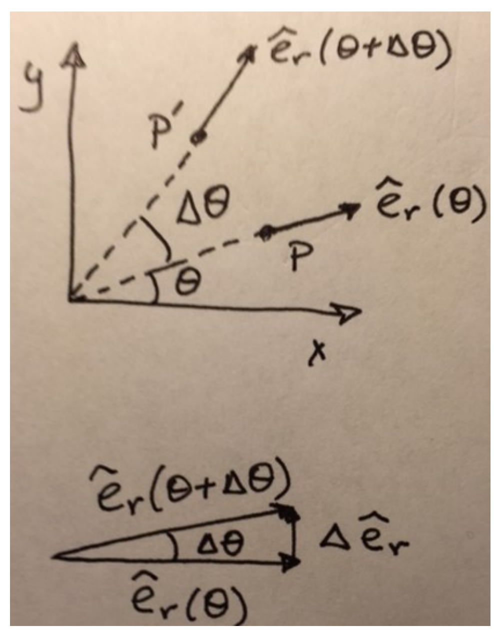

Figure 4: The unit vector, $\hat{e}_r$, changes direction as θ changes from P to P'. The two unit vectors can be drawn tail-to-tail. The distance between their tips is $\Delta\hat{e}_r$
.

For the vector, $\Delta\widehat{e}_r$, the magnitude of $\Delta\widehat{e}_r$ can be expressed (using the geometry of an isosceles triangle) as

$$|\Delta\widehat{e}_r| = 2(1)\sin(\Delta\theta/2) \tag{16}$$

and

$$\lim_{\Delta\theta\to 0}\frac{|\Delta\widehat{e}_r|}{\Delta\theta} = \lim_{\Delta\theta\to 0}\frac{2(1)\sin(\Delta\theta/2)}{\Delta\theta} \to 1. \tag{17}$$

The direction of $\Delta\widehat{e}_r$ approaches that of $\widehat{e}_\theta$ as $\Delta\theta$ approaches zero. Therefore,

$$\frac{d\widehat{e}_r}{d\theta} = \widehat{e}_\theta \tag{18}$$

Substituting this expression into Equation 14 yields $\dot{\widehat{e}}_r = \dot{\theta}\widehat{e}_\theta$. This is the same as result as found above. The procedure can be followed to verify that $\dot{\widehat{e}}_\theta = -\dot{\theta}\widehat{e}_r$.

In fact this result can be generalize to vectors with constant magnitude. They are mutually orthogonal to their time derivatives.

19. **Show that** if $\vec{A}$ has constant magnitude then

$$\vec{A}\cdot\frac{d\vec{A}}{dt} = 0 \tag{19}$$

Sliding into Spiral Learning (2 hours)

This problem asks you to develop a model and simulation of a person sliding down a helical slide. Although we'd love you to finish this problem, our guess is that you may not get to the end. That's OK. Probably the most important part of this is setting the problem up, so focus on getting that right first.

Figure 5: Whee ...

20. **Describe the helix.** Assume that the geometry of the helix is known. The pitch (vertical distance over one revolution) of a helix is defined as $p = 2\pi R\tan\beta$ where R is the radius and β is the (constant) angle between a line tangent to the helix and horizontal. Come up with a function in cylindrical coordinates that can be used to describe the path. You might think about using z as the independent variable; if you did this, you'd want to figure out how to describe $\theta(z)$ and $r(z)$.

Take a look at the MATLAB script, helix.m that makes a plot of a helix.

21. **Estimate system parameter values** (*e.g.*, helix geometry, mass of person sliding) for a realistic slide.

    Now let's abstract the person to a point mass on the slide.

22. **Identify the interactions** between the person and the rest of the universe: the slide, the earth, the air... What forces are act on the person? Do you want to neglect or include friction? Drag? Note

that there must be constraints to keep the person on the slide. Which of the forces act as constraints that keep the person on the helix?

23. **Decide on models for forces.** What direction does each force act in? What magnitude does each force have (note that for constraints, the answer to this question is "whatever value it needs to!")? If you include friction, how do you model it mathematically?

24. Capture your thoughts about the interactions by drawing a **free body diagram** for the person.

25. **How many equations of motion** do you expect? What directions (unit vectors) do they correspond to?

26. When you determined the description of the helix, you identified the constraints: the person must move along the helix. How do you incorporate your constraints into your equations of motion? Do so, and try to **reduce the number of differential equations**.

27. Identify the **the initial conditions** in both a mathematical and a physcial sense? Is the person starting from rest or do they give themselves an initial 'push'?

28. **Implement** your model using a numerical DE solver (eg `ode45`).

29. Produce one or more **figures that validate** your simulation.

For a selected case, makes plots of the trajectory; position variables (x, y, z, θ); forces from the slide acting on the person; and total energy as functions of time. Describe what each plot shows about the motion. Explain why you think your code works properly (*e.g.*, what validation cases do you run?). List some ways that your simulation could be used to design a slide for a new playground.