Nonlinear dynamic inversion

Aircraft don't always behave like linear systems. In some flight regimes, or in some (failure) scenarios, they behave in a nonlinear way. To control them, you therefore also need a nonlinear controller. And this controller should be as robust as possible, for the model never exactly matches reality.

In this file, we'll examine the technique called nonlinear dynamic inversion. It has proven to be an easy way of controlling nonlinear systems. Next to this, it offers possibilities for very robust control through an expansion called incremental nonlinear dynamic inversion.

1 Basics of nonlinear dynamic inversion

1.1 Rewriting a system for NDI

Let's examine the model of an aircraft. For simplicity, we will assume that it is a **single input single output** (SISO) model. This model has the form

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u, \tag{1.1}$$

with $\mathbf{x}$ the **state vector**. (Note that $f(\mathbf{x})$ can be a nonlinear function.) The above model can be rewritten to **companion form**

$$\begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} x_2 \\ \vdots \\ x_n \\ b(\mathbf{x}) \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ a(\mathbf{x}) \end{bmatrix} u. \tag{1.2}$$

In other words, all the nonlinear terms now only affect x_n. Also, the input only affects x_n. As a second step, we define the **virtual control input** v as

$$v = b(\mathbf{x}) + a(\mathbf{x})u \qquad \Leftrightarrow \qquad u = a^{-1}(\mathbf{x})\left(v - b(\mathbf{x})\right). \tag{1.3}$$

The virtual control input v can now be used to control the entire system in a simple linear way. This method of controlling a nonlinear system as if it is linear is called **(nonlinear) dynamic inversion** ((N)DI).

1.2 Which input to use?

The question remains on how to set v. For this, **state feedback** is often used. So,

$$v = -k_0 x - k_1 \frac{dx}{dt} - k_2 \frac{d^2x}{dt^2} - \ldots - k_{n-1} \frac{d^{n-1}x}{dt^{n-1}}. \tag{1.4}$$

Since we also have $\frac{d^n x}{dt^n} = v$, this turns the whole system into a linear closed loop system of the form

$$\frac{d^n x}{dt^n} + k_{n-1} \frac{d^{n-1}x}{dt^{n-1}} + \ldots + k_1 \frac{dx}{dt} + k_0 = 0. \tag{1.5}$$

By choosing the right k_i, the closed-loop system properties can be set. And, by then using equation (1.3), the required input u is found. The process of finding v is called the **outer loop** of NDI. Finding the corresponding value of u, and inserting it into the real system, is the **inner loop**.

In case of a **tracking task** (where x_1 has to follow some reference signal x_r), we can define $e = x - x_d$. The evident control law now is

$$v = -k_0 e - k_1 \frac{de}{dt} - k_2 \frac{d^2 e}{dt^2} - \ldots - k_{n-1} \frac{d^{n-1} e}{dt^{n-1}}. \tag{1.6}$$

This turns the system into

$$\frac{d^n e}{dt^n} + k_{n-1} \frac{d^{n-1} e}{dt^{n-1}} + \ldots + k_1 \frac{de}{dt} + k_0 = 0. \tag{1.7}$$

So this is basically the same problem as before.

2 Input-output linearization

2.1 The working principle of input-output linearization

It can occasionally be difficult to put a nonlinear system in companion form. An alternative to this is to apply **input-output linearization**. When applying this, we take the derivative of the output y until the input u appears in it. From this expression, you then derive a and b. As an example, consider the system

$$\begin{aligned} \dot{x}_1 &= x_2^2 + \sin(x_3), && (2.1) \\ \dot{x}_2 &= \cos(x_1), && (2.2) \\ \dot{x}_3 &= x_1 + u, && (2.3) \\ y &= \sin(x_1). && (2.4) \end{aligned}$$

Taking the derivative of y gives

$$\dot{y} = \dot{x}_1 \cos(x_1) = \left(x_2^2 + \sin(x_3)\right) \cos(x_1). \tag{2.5}$$

There is no input u in this expression. So we take the derivative again. This gives

$$\begin{aligned} \ddot{y} &= \left(2x_2\dot{x}_2 + \dot{x}_3 \cos(x_3)\right)\cos(x_1) - \left(x_2^2 + \sin(x_3)\right)\dot{x}_1 \sin(x_1) && (2.6) \\ &= \left(2x_2 \cos(x_1) + (x_1 + u)\cos(x_3)\right)\cos(x_1) - \left(x_2^2 + \sin(x_3)\right)^2 \sin(x_1) && (2.7) \\ &= 2\cos(x_1)^2 x_2 + x_1 \cos(x_1)\cos(x_3) - \left(x_2^2 + \sin(x_3)\right)^2 \sin(x_1) + \cos(x_1)\cos(x_3)u. && (2.8) \end{aligned}$$

As input, we can now again use $u = a^{-1}(v - b)$, where

$$\begin{aligned} a &= \cos(x_1)\cos(x_3), && (2.9) \\ b &= 2\cos(x_1)^2 x_2 + x_1 \cos(x_1)\cos(x_3) - \left(x_2^2 + \sin(x_3)\right)^2 \sin(x_1). && (2.10) \end{aligned}$$

In this way, the system can be seen as a linear system again. And, in our example case, this linear system is simply $\ddot{y} = v$.

2.2 Notes on NDI

When applying NDI, it is important to note the **singularities**. These occur when $u \to \infty$. For our example problem, they thus occur when $x_1 = \pi/2 + k\pi$ or $x_3 = \pi/2 + k\pi$, with k an integer. At such points, the system can't be sufficiently controlled.

Another important thing to note is that, to apply NDI, it is required to know the full state of the system. If the state is not known, it needs to be approximated in some away. For deterministic systems, a nonlinear observer can be used, whereas for stochastic systems a nonlinear state estimator is required.

Next to this, it is of course also required that the system model is known completely. If it is only partially known, some system identification might be required first.

2.3 Internal dynamics

The amount of times we need to differentiate the output is called the **relative degree** r of the system. In our example problem, this relative degree is thus $r = 2$. The **order** of the system (the amount of state variables) is denoted by n. We always have $r \leq n$.

If $r < n$, then part of the input-output linearization is **unobservable**. This unobservable part is called the **internal dynamics**. They have to be stable (bounded) for the controller to work properly. (In case of internal instability, the part b can become very big. Although mathematically u can become very big as well to compensate, this is physically often impossible.) However, the unobservable part is often also nonlinear. So finding out whether it's actually stable can be very difficult.

For simplicity let's examine the internal dynamics for linear systems. In linear systems, n corresponds to the amount of poles of the system, while $n - r$ is the amount of zeroes. (So r is the amount of excess poles.) It can also be shown that the internal dynamics are stable if all the zeroes are in the left half of the complex plane. In other words, the internal dynamics are stable if the system is minimum-phase.

3 State transformation

3.1 The Lie derivative

Let's examine a system of the form

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u, \tag{3.1}$$

$$y = h(\mathbf{x}). \tag{3.2}$$

Note that f and g are both vector functions (i.e. they return a vector) while h is a scalar function. The **Lie derivative** $L_f h(\mathbf{x})$ is now defined as the gradient of a certain (scalar) function $h(\mathbf{x})$ projected along a certain vector function $f(\mathbf{x})$. So,

$$L_f h(\mathbf{x}) = \nabla h(\mathbf{x}) f(\mathbf{x}) = \sum_{i=1}^{n} \frac{\partial h(\mathbf{x})}{\partial x_i} f_i(\mathbf{x}). \tag{3.3}$$

It is also possible to apply the Lie derivative multiple times. You then get the k^{th} Lie derivative, defined as

$$L_f^k h(\mathbf{x}) = L_f \left(L_f^{k-1} h(\mathbf{x}) \right) = \nabla \left(L_f^{k-1} h(\mathbf{x}) \right) f(\mathbf{x}) \qquad \text{with} \qquad L_f^0 h(\mathbf{x}) = h(\mathbf{x}). \tag{3.4}$$

3.2 The state transformation

We will use the Lie derivative to apply a state transformation. This transformation goes from the old state $\mathbf{x}$ to a new state $\mathbf{z}$ (called the **linearizing state** and puts the system in a canonical form.

First, using the Lie derivative, we define the functions $\phi_i(\mathbf{x})$ as

$$z_i = \phi_i(\mathbf{x}) = L_f^{i-1} h(\mathbf{x}), \qquad \text{with } 1 \leq i \leq r. \tag{3.5}$$

We have $L_g \phi_i(\mathbf{x}) = 0$ for all i, except for $i = r$. In fact, the definition of relative degree implies that $L_g \phi_r(\mathbf{x}) \neq 0$. (This property is often convenient to use when finding the relative degree.)

Analogously, we can also define the functions $z_i = \phi_i(\mathbf{x})$ with $r + 1 \leq i \leq n$. They also must have the property that

$$L_g \phi_i(\mathbf{x}) = 0. \tag{3.6}$$

A number $n - r$ of such additional functions always exist. However, we won't discuss here how to find them, since it's not relevant for our discussion.

The whole state transformation is now denoted by $\mathbf{z} = \Phi(\mathbf{x})$. The inverse is given by $\mathbf{x} = \Phi^{-1}(\mathbf{z})$.

3.3 Properties of the state transformation

It can be shown that, for $1 \leq i \leq r$, the new coordinates satisfy

$$\dot{z}_1 = z_2,\ \dot{z}_2 = z_3,\ \ldots,\ \dot{z}_{r-1} = z_r \ \text{ and } \ \dot{z}_r = a(\mathbf{z}) + b(\mathbf{z})u, \tag{3.7}$$

where the functions $a(\mathbf{z})$ and $b(\mathbf{z})$ (with $\mathbf{z} = \Phi^{-1}(\mathbf{x})$) are defined as

$$a(\mathbf{z}) = L_g L_f^{r-1} h(x) = L_g z_r \qquad \text{and} \qquad b(\mathbf{z}) = L_f^r h(\mathbf{x}). \tag{3.8}$$

(These functions are exactly the same as the functions a and b found in the section on input-output linearization. This method is just a somewhat more general way of finding them.)

For $r+1 \leq i \leq n$, the new coordinates satisfy

$$\dot{z}_i = L_f \phi_i(\mathbf{x}) + L_g \phi(\mathbf{x}) u = L_f \phi_i(\mathbf{x}) = L_f z_i. \tag{3.9}$$

(Remember that $L_g \phi_i(\mathbf{x}) = 0$ for $r+1 \leq i \leq n$.)

The result? We have a system where $z_1 = h(x) = y$ and

$$a(\mathbf{z}) + b(\mathbf{z})u = \frac{dz_r}{dt} = \frac{d^2 z_{r-1}}{dt^2} = \ldots = \frac{d^r z_1}{dt^r} = \frac{d^r y}{dt^r}. \tag{3.10}$$

We have put the system in canonical form! And the functions a and b are the same as what we've seen in the section on input-output linearization. So we can again control the system as if it's linear. Next to this, the states z_{r+1} to z_n don't directly affect the output y. So we have effectively split up the observable from the unobservable part.

4 MIMO systems and time scale separation

4.1 The MIMO system form

Previously, we have considered SISO systems. Now let's expand our ideas to **multiple input multiple output** (MIMO) systems. These systems have the form

$$\begin{aligned} \dot{\mathbf{x}} &= f(\mathbf{x}) + G(\mathbf{x})\mathbf{u}, & (4.1)\\ \mathbf{y} &= h(\mathbf{x}). & (4.2) \end{aligned}$$

Note that $G(\mathbf{x})$ is a matrix depending (possibly nonlinearly) on $\mathbf{x}$. Also note that we do assume that the state derivative $\dot{\mathbf{x}}$ affinely depends on the input $\mathbf{u}$. Also, the input vector $\mathbf{u}$ has size m, while the output vector $\mathbf{y}$ has size p.

4.2 The state transformation for MIMO systems

In a MIMO system, there are **individual relative degrees** $r_1, \ldots, r_p$. Together, they form the **total relative degree** $r = r_1 + \ldots + r_p$, which still satisfies $r \leq n$. There are also the ϕ functions. These are defined as

$$\phi_j^i(\mathbf{x}) = L_f^{j-1} h_i(\mathbf{x}). \tag{4.3}$$

These functions then satisfy

$$\dot{\phi}_1^i(\mathbf{x}) = \phi_2^i(\mathbf{x}),\ \ldots,\ \dot{\phi}_{r_i-1}^i(\mathbf{x}) = \phi_{r_i}^i(\mathbf{x}) \ \text{ and } \ \dot{\phi}_{r_i}^i(\mathbf{x}) = L_f^{r_i} h_i(\mathbf{x}) + \sum_{j=1}^{m} L_{g_j} L_f^{r_i-1} h_i(\mathbf{x}) u_j. \tag{4.4}$$

This holds for every i (with $1 \leq i \leq p$). We can use the above relations to find an expression for the virtual control input $\mathbf{v}$. (Note that $\mathbf{v}$ now also is a vector, having size p.) We then get

$$\mathbf{v} = b(\mathbf{x}) + A(\mathbf{x})\mathbf{u} = \begin{bmatrix} L_f^{r_1} h_1(\mathbf{x}) \\ L_f^{r_2} h_2(\mathbf{x}) \\ \vdots \\ L_f^{r_p} h_p(\mathbf{x}) \end{bmatrix} + \begin{bmatrix} L_{g_1} L_f^{r_1-1} h_1(\mathbf{x}) & L_{g_2} L_f^{r_1-1} h_1(\mathbf{x}) & \cdots & L_{g_m} L_f^{r_1-1} h_1(\mathbf{x}) \\ L_{g_1} L_f^{r_2-1} h_2(\mathbf{x}) & L_{g_2} L_f^{r_2-1} h_2(\mathbf{x}) & \cdots & L_{g_m} L_f^{r_2-1} h_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ L_{g_1} L_f^{r_p-1} h_p(\mathbf{x}) & L_{g_2} L_f^{r_p-1} h_p(\mathbf{x}) & \cdots & L_{g_m} L_f^{r_p-1} hp(\mathbf{x}) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix}. \tag{4.5}$$

Solving for the input vector $\mathbf{u}$ then gives

$$\mathbf{u} = A^{-1}(\mathbf{x})(\mathbf{v} - b(\mathbf{x})). \tag{4.6}$$

4.3 Using time-scale separation

When controlling two different parameters, a technique called time-scale separation may be applied. To know when this is the case, we first have to discuss control effectiveness.

The **control effectiveness** can be seen as the effect on a controlled parameter, due to a unity change in the controlling parameter. (Think of $\partial q / \partial \delta_e$ for aircraft pitch rate control.) Based on the control effectiveness, we can make a distinction between slow dynamics and fast dynamics. **Slow dynamics** means that the control effectiveness of a certain parameter is low. **Fast dynamics** means that the control effectiveness is high.

When there are two parameters to be controlled, one with slow dynamics and one with fast dynamics, **time-scale separation** (TSS) can be applied. This means that we split up the slow and the fast dynamics. The fast dynamics can then be seen as the inner loop, while the slow dynamics make up the outer loop. For every part, dynamic inversion is applied separately.

An example of time-scale separation occurs in an aircraft. Pitch rate control generally works a lot faster than pitch angle control. So pitch rate control forms the inner loop of the controller, while pitch angle control forms the outer loop.

When applying TSS, an assumption is made. As input, the inner loop receives a reference value. (In our example the desired pitch rate.) The outer loop assumes that this desired value is actually achieved by the inner loop. This is often a valid assumption to make, because the inner loop is much faster than the outer loop. The outer loop (the pitch angle control) then operates by simply supplying the right reference input (the desired pitch rate) to the inner loop.

5 Incremental NDI

5.1 The basic idea of INDI

There is a big downside to NDI. To apply it, the model of the system has to be known quite accurately. An aircraft model can, however, be very complicated, especially when nonlinearities start being present.

A possible solution is offered by **Incremental NDI** (INDI). This technique doesn't give the required input to control the system. It gives the required change in the input. (For example, it does not tell the elevator to deflect to 6 degrees, but to deflect (e.g.) 1 degree more or 2 degrees less.)

A big advantage of INDI is that only a small part of the model is required. Also, INDI is better able to cope with model inaccuracies like, for example, wrong coefficients. As such, INDI is more robust than NDI.

5.2 INDI applied to an aircraft – from moments to control surface deflections

Let's demonstrate the INDI idea by looking at an aircraft. When controlling the attitude of an aircraft, moments need to be applied. And, when working with moments, we usually simply use moment coefficients. The moment coefficient of L is defined as

$$C_l = \frac{L_{des}}{\frac{1}{2}\rho V^2 S b}. \tag{5.1}$$

There is a similar definition for C_m and C_n. To control the aircraft, there are also desired moment coefficients $C_{l,des}$, $C_{m,des}$ and $C_{n,des}$. Next to this, there are the coefficient derivatives. An example is

$$C_{l_{\delta_e}} = \frac{\partial C_l}{\partial \delta_e}. \tag{5.2}$$

Similar definitions hold for combinations with the other moments (M and N) and the other control inputs (δ_a and δ_r).

Based on the above definitions, we can find the required control surface deflections δ to provide the right moments. We have

$$\delta = \begin{bmatrix} \delta_e \\ \delta_a \\ \delta_r \end{bmatrix} = \begin{bmatrix} C_{l_{\delta_e}} & C_{l_{\delta_a}} & C_{l_{\delta_r}} \\ C_{m_{\delta_e}} & C_{m_{\delta_a}} & C_{m_{\delta_r}} \\ C_{n_{\delta_e}} & C_{n_{\delta_a}} & C_{n_{\delta_r}} \end{bmatrix}^{-1} \begin{bmatrix} C_{l,des} \\ C_{m,des} \\ C_{n,des} \end{bmatrix} = M_c^{-1} \mathbf{C_{lmn,des}}. \tag{5.3}$$

(Note the definitions of δ, M_c and $\mathbf{C_{lmn,des}}$ in the above equation.) This technique works well in case the moments L, M and N vary linearly with the control surface deflections δ_e, δ_a and δ_r, and if all the coefficient derivatives are accurately known. If either is not the case, we have to do something else: INDI.

When applying INDI, we look at the required change in moment. That is, we compare the current moment coefficients acting on the aircraft (C_l, C_m and C_n) to the desired moment coefficients. From this, we derive the changes in the control surface deflections. This goes according to

$$\Delta\delta = \begin{bmatrix} \Delta\delta_e \\ \Delta\delta_a \\ \Delta\delta_r \end{bmatrix} = \begin{bmatrix} C_{l_{\delta_e}} & C_{l_{\delta_a}} & C_{l_{\delta_r}} \\ C_{m_{\delta_e}} & C_{m_{\delta_a}} & C_{m_{\delta_r}} \\ C_{n_{\delta_e}} & C_{n_{\delta_a}} & C_{n_{\delta_r}} \end{bmatrix}^{-1} \begin{bmatrix} C_{l,des} - C_l \\ C_{m,des} - C_m \\ C_{n,des} - C_n \end{bmatrix} = M_c^{-1}(\mathbf{C_{lmn,des}} - \mathbf{C_{lmn}}). \tag{5.4}$$

This time, when the coefficient derivatives aren't accurate, the required moments will eventually still be reached. In this way, INDI is much more robust than NDI.

5.3 INDI applied to an aircraft – from motion to control surface deflections

When controlling an aircraft, we don't just start out with moments. Instead, we want to control the motion of the aircraft. To find out how this works, we examine the equation of motion

$$I\dot{\omega} + \omega \times I\omega = \mathbf{M} = \mathbf{M_a} + \mathbf{M_c}. \tag{5.5}$$

Here, $\mathbf{M_a}$ denotes the moments due to aerodynamics, while $\mathbf{M_c}$ denotes the moments due to the controls. (If required, we can transform the above equation, such that it uses the coefficients $\mathbf{C}$ instead of the moments $\mathbf{M}$.)

Now examine what happens when we change the control moment $\mathbf{M_c}$ by a small amount $\Delta\mathbf{M_c}$. In an infinitesimal timestep, the angular rates ω don't change yet, nor do the aerodynamic moments $\mathbf{M_a}$. Only the angular acceleration $\dot{\omega}$ changes. (Note that we are applying the principle of time-scale separation here.) So we can write

$$\Delta\mathbf{M_c} = I(\Delta\dot{\omega}) = I(\dot{\omega}_{\mathbf{new}} - \dot{\omega}_{\mathbf{old}}). \tag{5.6}$$

Using an on-board **Inertial Measurement Unit** (IMU), we can derive $\dot{\omega}_{\mathbf{old}}$. Now assume that the controlled quantity of our system (the output $\mathbf{y}$) is the angular rate ω. We can then write the system as

$$\dot{\mathbf{y}} = \dot{\omega}_{\mathbf{new}} = \dot{\omega}_{\mathbf{old}} + I^{-1}(\Delta \mathbf{M_c}). \tag{5.7}$$

We want to find the required change in control input $\Delta\delta$, such that the desired $\dot{\mathbf{y}}$ is obtained. To do that, we first substitute $\Delta\mathbf{M_c}$ by $M_c(\Delta\delta)$. (We apply linearization.) The required control surface deflection is then given by

$$\Delta\delta = M_c^{-1} I(\dot{\mathbf{y}} - \dot{\omega}_{\mathbf{cur}}). \tag{5.8}$$

Let's examine the above equation. When we apply normal NDI, the value of δ strongly depends on a lot of aircraft properties. Now, $\Delta\delta$ only depends on M_c and I. Because we use measurements of $\dot{\omega}_{\mathbf{new}}$, any other model uncertainties are cancelled. This is especially so if the measurements are very accurate and the sampling time is small. In this way, INDI is much more robust than NDI.

6 Controlling an aircraft with NDI

6.1 Aircraft attitude control

Now that we know how NDI works, we should apply it to control an aircraft. We'll discuss here how that works. Although we don't show the derivation of this technique, we will explain how to find the control surface deflections required to control the aircraft. This is done in six steps.

1. We start off with the reference flight angles ϕ, θ and β. From them, we derive the reference flight angle derivatives $\dot{\phi}$, $\dot{\theta}$ and $\dot{\beta}$. This is done using a PID controller, like

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\beta}\end{bmatrix}_{ref} = K_{P_1}\left(\begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{ref} - \begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{act}\right) + K_{I_1}\int\left(\begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{ref} - \begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{act}\right) + K_{D_1}\frac{d}{dt}\left(\begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{ref} - \begin{bmatrix}\phi\\ \theta\\ \beta\end{bmatrix}_{act}\right). \tag{6.1}$$

2. From these derivatives, the rotational rates of the aircraft should be derived. This is done using the matrix equation

$$\begin{bmatrix}p\\ q\\ r\end{bmatrix}_{ref} = \begin{bmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ \frac{w}{\sqrt{u^2+w^2}} & 0 & \frac{-u}{\sqrt{u^2+w^2}}\end{bmatrix}^{-1}\left(\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\beta}\end{bmatrix}_{ref} - \begin{bmatrix}0\\ 0\\ \frac{1}{\sqrt{u^2+w^2}}\left(\frac{-uv}{V^2}a_x + \left(1-\frac{v^2}{V^2}\right)a_y - \frac{vw}{V^2}a_z\right)\end{bmatrix}\right). \tag{6.2}$$

The coefficients a_x, a_y and a_z are, respectively, given by

$$a_x = \frac{X}{m} - g\sin\theta, \qquad a_y = \frac{Y}{m} + g\sin\phi\cos\theta \qquad \text{and} \qquad a_z = \frac{Z}{m} + g\cos\phi\cos\theta, \tag{6.3}$$

with X, Y and Z the forces in the corresponding directions. They can be measured by an IMU.

3. We now have the desired rotational rates of the aircraft. Again, using a PID controller, we can find the desired rotational accelerations of the aircraft. So,

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix}_{ref} = K_{P_2}\left(\begin{bmatrix}p\\ q\\ r\end{bmatrix}_{ref} - \begin{bmatrix}p\\ q\\ r\end{bmatrix}_{act}\right) + K_{I_2}\int\left(\begin{bmatrix}p\\ q\\ r\end{bmatrix}_{ref} - \begin{bmatrix}p\\ q\\ r\end{bmatrix}_{act}\right) + K_{D_2}\frac{d}{dt}\left(\begin{bmatrix}p\\ q\\ r\end{bmatrix}_{ref} - \begin{bmatrix}p\\ q\\ r\end{bmatrix}_{act}\right). \tag{6.4}$$

4. To find the required moments, we can use

$$\begin{bmatrix} L \\ M \\ N \end{bmatrix}_{req} = I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}_{ref} + \begin{bmatrix} p \\ q \\ r \end{bmatrix}_{act} \times I \begin{bmatrix} p \\ q \\ r \end{bmatrix}_{act}. \tag{6.5}$$

   Note that we don't use incremental NDI here, but just the normal version of NDI. If required, also the INDI version can be implemented.

5. The required moments should be normalized before we can use them. For that, we use

$$C_{l_{req}} = \frac{L_{req}}{\frac{1}{2}\rho V^2 S b}, \qquad C_{m_{req}} = \frac{M_{req}}{\frac{1}{2}\rho V^2 S \bar{c}} \quad \text{and} \quad C_{n_{req}} = \frac{N_{req}}{\frac{1}{2}\rho V^2 S b}. \tag{6.6}$$

6. Now that the required moment coefficients are known, the required elevator deflections can be calculated. This time we can use

$$\begin{bmatrix} \delta_e \\ \delta_a \\ \delta_r \end{bmatrix}_{req} = \begin{bmatrix} C_{l_{\delta_e}} & C_{l_{\delta_a}} & C_{l_{\delta_r}} \\ C_{m_{\delta_e}} & C_{m_{\delta_a}} & C_{m_{\delta_r}} \\ C_{n_{\delta_e}} & C_{n_{\delta_a}} & C_{n_{\delta_r}} \end{bmatrix}^{-1} \left(\begin{bmatrix} C_l \\ C_m \\ C_n \end{bmatrix}_{req} - \begin{bmatrix} C_l \\ C_m \\ C_n \end{bmatrix}_{computed} \right). \tag{6.7}$$

6.2 Aircraft position control – deriving equations

With the controller of the previous paragraph, we could only control the orientation of the aircraft. But how do we travel around the world then? For that, we need several equations, which we'll discuss in this paragraph. (We then use them in the next paragraph.) We will start with the equations of motion. They are given by

$$\mathbf{F_{ext}} = m\mathbf{a_C} = m\frac{d\dot{\mathbf{r}}_\mathbf{C}}{dt} = m\left(\frac{d\dot{\mathbf{r}}_\mathbf{E}}{dt} + \Omega_{EC} \times \dot{\mathbf{r}}_\mathbf{E}\right). \tag{6.8}$$

Here, the subscript C denotes the Earth-centered Earth-fixed reference frame. (We assume that this reference frame is inertial.) However, we will be using the **equation reference frame** F_E. This reference frame has its Z axis pointing up (away from Earth's center), the X axis pointing East and the Y axis pointing North. The origin is at the center of the Earth. The position of the aircraft is now simply given by

$$\mathbf{r_E} = \begin{bmatrix} 0 \\ 0 \\ r \end{bmatrix}. \tag{6.9}$$

The derivative depends on the **flight path angle** γ and the **heading angle** χ, according to

$$\dot{\mathbf{r}}_\mathbf{E} = V \begin{bmatrix} \cos\gamma \sin\chi \\ \cos\gamma\cos\chi \\ \sin\gamma \end{bmatrix}, \tag{6.10}$$

The term Ω_{EC} is the rotation rate of the F_C reference frame with respect to the F_E reference frame. As seen from the F_E reference frame, it is given by

$$\Omega_{EC} = \begin{bmatrix} -\dot{\delta} \\ \dot{\tau}\cos\delta \\ \dot{\tau}\sin\delta \end{bmatrix}, \tag{6.11}$$

where τ is the **longitude** and δ the **latitude**. Their time-derivatives depend on V, γ and χ, according to

$$\begin{aligned} \dot{r} &= V \sin\gamma, && (6.12)\\ \dot{\tau} &= \frac{V \sin\chi \cos\gamma}{r\cos\delta}, && (6.13)\\ \dot{\delta} &= \frac{V\cos\chi\cos\gamma}{r}. && (6.14) \end{aligned}$$

Based on the above relations, we can eventually derive that

$$\begin{bmatrix} \dot{V} \\ V\dot{\gamma} \\ V\dot{\chi}\cos\gamma \end{bmatrix} = \begin{bmatrix} \cos\gamma\sin\chi & -\sin\gamma\sin\chi & \cos\chi \\ \cos\gamma\cos\chi & -\sin\gamma\cos\chi & -\sin\chi \\ \sin\gamma & \cos\gamma & 0 \end{bmatrix}^{-1} \frac{\mathbf{F_E}}{m} + \begin{bmatrix} 0 \\ -\frac{V^2\cos\gamma}{r} \\ \frac{V^2\cos^2\gamma\sin\chi\tan\delta}{r} \end{bmatrix}. \tag{6.15}$$

The above equation will look a bit prettier if we transform it to the **velocity reference frame** F_V. This reference frame has the X axis pointing in the direction of the aircraft velocity vector. Now examine the plane spanned by this X axis and the local gravity vector. The Y axis is perpendicular to this plane and points to the right of the aircraft. Finally, the Z axis is perpendicular to both the X and Y axes and points roughly downward. If we use this reference frame, then we will get

$$\begin{bmatrix} \dot{V} \\ V\dot{\gamma} \\ V\dot{\chi}\cos\gamma \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} T_{VB}\frac{\mathbf{F_B}}{m} + \begin{bmatrix} 0 \\ -\frac{V^2\cos\gamma}{r} \\ \frac{V^2\cos^2\gamma\sin\chi\tan\delta}{r} \end{bmatrix}. \tag{6.16}$$

In the above equation, the rightmost term is often small enough to be neglected. But, for accuracy (and just because we can), we will keep it. Do note that the force vector $\mathbf{F}$ is now expressed in the **body reference frame** F_B. The transformation vector T_{VB} between the velocity and the body reference frame is given by

$$T_{VB} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta\cos\mu + \sin\alpha\sin\mu & \cos\beta\cos\mu & -\sin\alpha\sin\beta\cos\mu - \cos\alpha\sin\mu \\ -\cos\alpha\sin\beta\sin\mu - \sin\alpha\cos\mu & \cos\beta\sin\mu & -\sin\alpha\sin\beta\sin\mu + \cos\alpha\cos\mu \end{bmatrix}. \tag{6.17}$$

6.3 Aircraft position control – the actual plan

We now have a lot of equations concerning an aircraft flying around planet Earth. But how do we control it? First, we examine the velocity. We want to keep a certain reference velocity V_{ref}. Using PID control, we find the corresponding value of $\dot{V}_{ref}$. We have

$$\dot{V}_{ref} = K_P\left(V_{ref} - V_{act}\right) + K_I \int \left(V_{ref} - V_{act}\right) + K_D \frac{d\left(V_{ref} - V_{act}\right)}{dt}. \tag{6.18}$$

We then use this reference value to find the required thrust. We also use the top row of equation (6.16). This gives us the equation

$$\dot{V} = \frac{1}{m}\begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \end{bmatrix} \begin{bmatrix} X+T \\ Y \\ Z \end{bmatrix} = \frac{1}{m}\left((X+T)\cos\alpha\cos\beta + Y\sin\beta + Z\sin\alpha\cos\beta\right). \tag{6.19}$$

Solving for the required thrust gives

$$T_{req} = \frac{m\dot{V}_{ref} - (X\cos\alpha\cos\beta + Y\sin\beta + Z\sin\alpha\cos\beta)}{\cos\alpha\cos\beta}. \tag{6.20}$$

From this, the corresponding thrust setting δ_T should be derived.

Now let's examine the flight path. This is specified by the climb angle γ and the heading χ. For both these parameters, we have certain desired values γ_{ref} and χ_{ref} that need to be followed. We use these desired values to find

$$\dot{\gamma}_{ref} = K_D\left(\gamma_{ref} - \gamma_{act}\right) + K_I\int\left(\gamma_{ref} - \gamma_{act}\right) + K_D\frac{d\left(\gamma_{ref} - \gamma_{act}\right)}{dt}, \tag{6.21}$$

$$\dot{\chi}_{ref} = K_D\left(\chi_{ref} - \chi_{act}\right) + K_I\int\left(\chi_{ref} - \chi_{act}\right) + K_D\frac{d\left(\chi_{ref} - \chi_{act}\right)}{dt}. \tag{6.22}$$

Using these values, we can derive required values for the angle of attack α and the bank angle μ. But we do have to make some assumptions for that. We assume that the reference angle for the sideslip angle β is always zero, so $\beta_{ref} = 0$. In fact, for simplicity, we assume that the sideslip angle is always kept at zero by the autopilot, so $\beta = 0$. We also assume that, because the sideslip angle is zero, there are no side forces, so $Y = 0$. Finally, we assume that the angle of attack is small, so $\sin\alpha = \alpha$ and $\cos\alpha = 1$. If we now solve the bottom two rows of equation (6.16), we get

$$\mu_{ref} = \arcsin\left(\left(\frac{m}{X\alpha - Z}\right)\left(V\dot{\chi}\cos\gamma - \frac{V^2\cos^2\gamma\sin\chi\tan\delta}{r}\right)\right), \tag{6.23}$$

$$\alpha_{ref} = \frac{m}{X\cos\mu}\left(V\dot{\gamma} + \frac{Z}{m}\cos\mu + \frac{V^2\cos\gamma}{r}\right). \tag{6.24}$$

Using the reference angles α_{ref}, μ_{ref} and β_{ref} (which is zero), we can find

$$\dot{\mu}_{ref} = K_D\left(\mu_{ref} - \mu_{act}\right) + K_I\int\left(\mu_{ref} - \mu_{act}\right) + K_D\frac{d\left(\mu_{ref} - \mu_{act}\right)}{dt}, \tag{6.25}$$

$$\dot{\alpha}_{ref} = K_D\left(\alpha_{ref} - \alpha_{act}\right) + K_I\int\left(\alpha_{ref} - \alpha_{act}\right) + K_D\frac{d\left(\alpha_{ref} - \alpha_{act}\right)}{dt}, \tag{6.26}$$

$$\dot{\beta}_{ref} = K_D\left(\beta_{ref} - \beta_{act}\right) + K_I\int\left(\beta_{ref} - \beta_{act}\right) + K_D\frac{d\left(\beta_{ref} - \beta_{act}\right)}{dt}. \tag{6.27}$$

From the desired angle derivatives, we can find the required aircraft rotational rates p, q and r. For this, we do need to derive some more complicated kinematic equations. We won't discuss the derivation here. We'll only mention the result, which is

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix}_{ref} = \begin{bmatrix} \cos\alpha\cos\beta & 0 & \sin\alpha \\ \sin\beta & 1 & 0 \\ \sin\alpha\cos\beta & 0 & -\cos\alpha \end{bmatrix}\begin{bmatrix} \dot{\mu}_{ref} \\ \dot{\alpha}_{ref} \\ \dot{\beta}_{ref} \end{bmatrix} + T_{BV}\begin{bmatrix} -\dot{\chi}\sin\gamma \\ \dot{\gamma} \\ \dot{\chi}\cos\gamma \end{bmatrix}. \tag{6.28}$$

Once these reference values are known, we can continue with step 3 of the original orientation control plan to find the required control surface deflections. In this way, we only have to specify a desired aircraft velocity V_{ref}, flight path angle γ_{ref} and heading angle χ_{ref}, and the airplane will follow it. Who still needs a pilot?