2 Continuous Dynamics

This chapter reviews a few of the many modeling techniques for studying dynamics of a physical system. We begin by studying mechanical parts that move (this problem is known as **classical mechanics**). The techniques used to study the dynamics of such parts extend broadly to many other physical systems, including circuits, chemical processes, and biological processes. But mechanical parts are easiest for most people to visualize, so they make our example concrete. Motion of mechanical parts can often be modeled using **differential equations**, or equivalently, **integral equations**. Such models really only work well for "smooth" motion (a concept that we can make more precise using notions of linearity, time invariance, and continuity). For motions that are not smooth, such as those modeling collisions of mechanical parts, we can use modal models that represent

distinct modes of operation with abrupt (conceptually instantaneous) transitions between modes. Collisions of mechanical objects can be usefully modeled as discrete, instantaneous events. The problem of jointly modeling smooth motion and such discrete events is known as hybrid systems modeling and is studied in Chapter 4. Such combinations of discrete and continuous behaviors bring us one step closer to joint modeling of cyber and physical processes.

We begin with simple equations of motion, which provide a model of a system in the form of **ordinary differential equations** (**ODEs**). We then show how these ODEs can be represented in actor models, which include the class of models in popular modeling languages such as LabVIEW (from National Instruments) and Simulink (from The MathWorks, Inc.). We then consider properties of such models such as linearity, time invariance, and stability, and consider consequences of these properties when manipulating models. We develop a simple example of a feedback control system that stabilizes an unstable system. Controllers for such systems are often realized using software, so such systems can serve as a canonical example of a cyber-physical system. The properties of the overall system emerge from properties of the cyber and physical parts.

2.1 Newtonian Mechanics

In this section, we give a brief working review of some principles of classical mechanics. This is intended to be just enough to be able to construct interesting models, but is by no means comprehensive. The interested reader is referred to many excellent texts on classical mechanics, including Goldstein (1980); Landau and Lifshitz (1976); Marion and Thornton (1995).

Motion in space of physical objects can be represented with **six degrees of freedom**, illustrated in Figure 2.1. Three of these represent position in three dimensional space, and three represent orientation in space. We assume three axes, x, y, and z, where by convention x is drawn increasing to the right, y is drawn increasing upwards, and z is drawn increasing out of the page. **Roll** θ_x is an angle of rotation around the x axis, where by convention an angle of 0 radians represents horizontally flat along the z axis (i.e., the angle is given relative to the z axis). **Yaw** θ_y is the rotation around the y axis, where by convention 0 radians represents pointing directly to the right (i.e., the angle is given relative to the x axis). **Pitch** θ_z is rotation around the z axis, where by convention 0 radians represents pointing horizontally (i.e., the angle is given relative to the x axis).

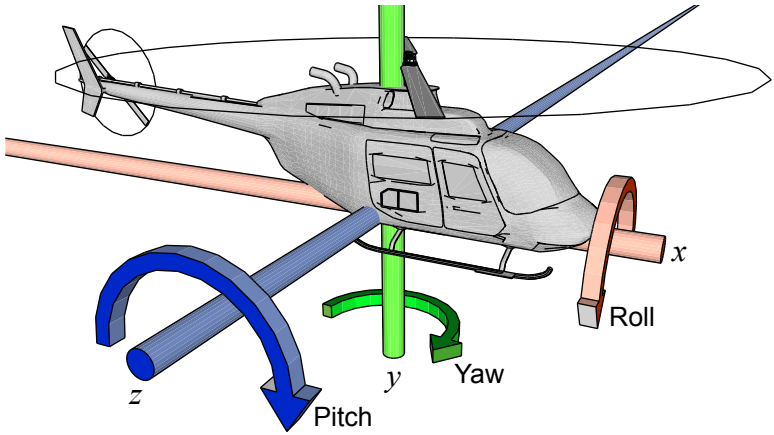


Figure 2.1: Modeling position with six degrees of freedom requires including pitch, roll, and yaw, in addition to position.

The position of an object in space, therefore, is represented by six functions of the form $f: \mathbb{R} \to \mathbb{R}$, where the domain represents time and the codomain represents either distance along an axis or angle relative to an axis.[1] Functions of this form are known as **continuous-time signals**.[2] These are often collected into vector-valued functions $\mathbf{x}: \mathbb{R} \to \mathbb{R}^3$ and $\theta: \mathbb{R} \to \mathbb{R}^3$, where $\mathbf{x}$ represents position, and θ represents orientation.

Changes in position or orientation are governed by **Newton's second law**, relating force with acceleration. Acceleration is the second derivative of position. Our first equation handles the position information,

$$\mathbf{F}(t) = M\ddot{\mathbf{x}}(t), \tag{2.1}$$

where $\mathbf{F}$ is the force vector in three directions, M is the mass of the object, and $\ddot{\mathbf{x}}$ is the second derivative of $\mathbf{x}$ with respect to time (i.e., the acceleration). Velocity is the integral

[1] If the notation is unfamiliar, see Appendix A.

[2] The domain of a continuous-time signal may be restricted to a connected subset of $\mathbb{R}$, such as $\mathbb{R}_+$, the non-negative reals, or $[0, 1]$, the interval between zero and one, inclusive. The codomain may be an arbitrary set, though when representing physical quantities, real numbers are most useful.

of acceleration, given by

$$\forall\, t > 0, \quad \dot{\mathbf{x}}(t) \;=\; \dot{\mathbf{x}}(0) + \int_0^t \ddot{\mathbf{x}}(\tau) d\tau$$

where $\dot{\mathbf{x}}(0)$ is the initial velocity in three directions. Using (2.1), this becomes

$$\forall\, t > 0, \quad \dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M} \int_0^t \mathbf{F}(\tau) d\tau,$$

Position is the integral of velocity,

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{x}(0) + \int_0^t \dot{\mathbf{x}}(\tau) d\tau \\ &= \mathbf{x}(0) + t\dot{\mathbf{x}}(0) + \frac{1}{M} \int_0^t \int_0^\tau \mathbf{F}(\alpha) d\alpha d\tau, \end{aligned}$$

where $\mathbf{x}(0)$ is the initial position. Using these equations, if you know the initial position and initial velocity of an object and the forces on the object in all three directions as a function of time, you can determine the acceleration, velocity, and position of the object at any time.

The versions of these equations of motion that affect orientation use **torque**, the rotational version of force. It is again a three-element vector as a function of time, representing the net rotational force on an object. It can be related to angular velocity in a manner similar to equation (2.1),

$$\mathbf{T}(t) = \frac{d}{dt}\left(\mathbf{I}(t)\dot{\theta}(t)\right), \tag{2.2}$$

where $\mathbf{T}$ is the torque vector in three axes and $\mathbf{I}(t)$ is the **moment of inertia tensor** of the object. The moment of inertia is a 3×3 matrix that depends on the geometry and orientation of the object. Intuitively, it represents the reluctance that an object has to spin around any axis as a function of its orientation along the three axes. If the object is spherical, for example, this reluctance is the same around all axes, so it reduces to a constant scalar I (or equivalently, to a diagonal matrix $\mathbf{I}$ with equal diagonal elements I). The equation then looks much more like (2.1),

$$\mathbf{T}(t) = I\ddot{\theta}(t). \tag{2.3}$$

To be explicit about the three dimensions, we might write (2.2) as

$$\begin{bmatrix} T_x(t) \\ T_y(t) \\ T_z(t) \end{bmatrix} = \frac{d}{dt}\left(\begin{bmatrix} I_{xx}(t) & I_{xy}(t) & I_{xz}(t) \\ I_{yx}(t) & I_{yy}(t) & I_{yz}(t) \\ I_{zx}(t) & I_{zy}(t) & I_{zz}(t) \end{bmatrix} \begin{bmatrix} \dot{\theta}_x(t) \\ \dot{\theta}_y(t) \\ \dot{\theta}_z(t) \end{bmatrix} \right) .$$

Here, for example, $T_y(t)$ is the net torque around the y axis (which would cause changes in yaw), $I_{yx}(t)$ is the inertia that determines how acceleration around the x axis is related to torque around the y axis.

Rotational velocity is the integral of acceleration,

$$\dot{\theta}(t) = \dot{\theta}(0) + \int_0^t \ddot{\theta}(\tau) d\tau,$$

where $\dot{\theta}(0)$ is the initial rotational velocity in three axes. For a spherical object, using (2.3), this becomes

$$\dot{\theta}(t) = \dot{\theta}(0) + \frac{1}{I}\int_0^t \mathbf{T}(\tau) d\tau.$$

Orientation is the integral of rotational velocity,

$$\begin{aligned} \theta(t) &= \theta(0) + \int_0^t \dot{\theta}(\tau) d\tau \\ &= \theta(0) + t\dot{\theta}(0) + \frac{1}{I}\int_0^t \int_0^\tau \mathbf{T}(\alpha) d\alpha d\tau \end{aligned}$$

where $\theta(0)$ is the initial orientation. Using these equations, if you know the initial orientation and initial rotational velocity of an object and the torques on the object in all three axes as a function of time, you can determine the rotational acceleration, velocity, and orientation of the object at any time.

Often, as we have done for a spherical object, we can simplify by reducing the number of dimensions that are considered. In general, such a simplification is called a **model-order reduction**. For example, if an object is a moving vehicle on a flat surface, there may be little reason to consider the y axis movement or the pitch or roll of the object.

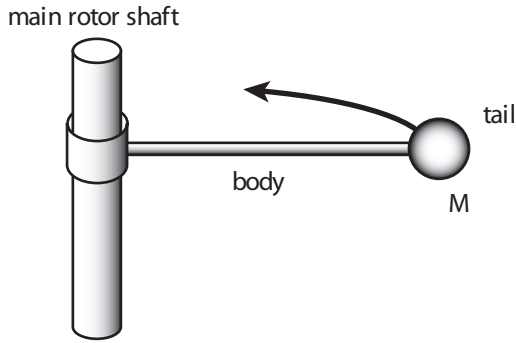


Figure 2.2: Simplified model of a helicopter.

Example 2.1: Consider a simple control problem that admits such reduction of dimensionality. A helicopter has two rotors, one above, which provides lift, and one on the tail. Without the rotor on the tail, the body of the helicopter would spin. The rotor on the tail counteracts that spin. Specifically, the force produced by the tail rotor must counter the torque produced by the main rotor. Here we consider this role of the tail rotor independently from all other motion of the helicopter.

A simplified model of the helicopter is shown in Figure 2.2. Here, we assume that the helicopter position is fixed at the origin, so there is no need to consider equations describing position. Moreover, we assume that the helicopter remains vertical, so pitch and roll are fixed at zero. These assumptions are not as unrealistic as they may seem since we can define the coordinate system to be fixed to the helicopter.

With these assumptions, the **moment of inertia** reduces to a scalar that represents a torque that resists changes in yaw. The changes in yaw will be due to **Newton's third law**, the **action-reaction law**, which states that every action has an equal and opposite reaction. This will tend to cause the helicopter to rotate in the opposite direction from the rotor rotation. The tail rotor has the job of countering that torque to keep the body of the helicopter from spinning.

We model the simplified helicopter by a system that takes as input a continuous-time signal T_y, the torque around the y axis (which causes changes in yaw). This

torque is the sum of the torque caused by the main rotor and that caused by the tail rotor. When these are perfectly balanced, that sum is zero. The output of our system will be the angular velocity $\dot{\theta}_y$ around the y axis. The dimensionally-reduced version of (2.2) can be written as

$$\ddot{\theta}_y(t) = T_y(t)/I_{yy}.$$

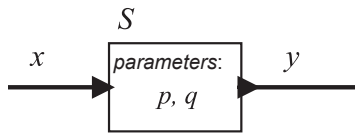
Integrating both sides, we get the output $\dot{\theta}$ as a function of the input T_y,

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau) d\tau. \tag{2.4}$$

The critical observation about this example is that if we were to choose to model the helicopter by, say, letting $\mathbf{x}: \mathbb{R} \to \mathbb{R}^3$ represent the absolute position in space of the tail of the helicopter, we would end up with a far more complicated model. Designing the control system would also be much more difficult.

2.2 Actor Models

In the previous section, a model of a physical system is given by a differential or an integral equation that relates input signals (force or torque) to output signals (position, orientation, velocity, or rotational velocity). Such a physical system can be viewed as a component in a larger system. In particular, a **continuous-time system** (one that operates on continuous-time signals) may be modeled by a box with an input **port** and an output port as follows:



where the input signal x and the output signal y are functions of the form

$$x: \mathbb{R} \to \mathbb{R}, \quad y: \mathbb{R} \to \mathbb{R}.$$

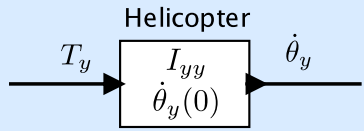
Here the domain represents **time** and the codomain represents the value of the signal at a particular time. The domain $\mathbb{R}$ may be replaced by $\mathbb{R}_+$, the non-negative reals, if we wish to explicitly model a system that comes into existence and starts operating at a particular point in time.

The model of the system is a function of the form

$$S: X \to Y, \tag{2.5}$$

where $X = Y = \mathbb{R}^{\mathbb{R}}$, the set of functions that map the reals into the reals, like x and y above.[3] The function S may depend on parameters of the system, in which case the parameters may be optionally shown in the box, and may be optionally included in the function notation. For example, in the above figure, if there are parameters p and q, we might write the system function as $S_{p,q}$ or even $S(p, q)$, keeping in mind that both notations represent functions of the form in 2.5. A box like that above, where the inputs are functions and the outputs are functions, is called an **actor**.

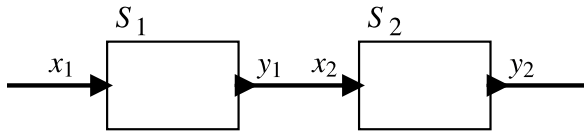
Example 2.2: The actor model for the helicopter of example 2.1 can be depicted as follows:



The input and output are both continuous-time functions. The parameters of the actor are the initial angular velocity $\dot{\theta}_y(0)$ and the moment of inertia I_{yy}. The function of the actor is defined by (2.4).

Actor models are composable. In particular, given two actors S_1 and S_2, we can form a **cascade composition** as follows:

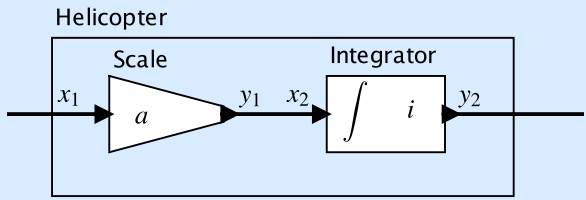
[3] As explained in Appendix A, the notation $\mathbb{R}^{\mathbb{R}}$ (which can also be written $(\mathbb{R} \to \mathbb{R})$) represents the set of all functions with domain $\mathbb{R}$ and codomain $\mathbb{R}$.



In the diagram, the "wire" between the output of S_1 and the input of S_2 means precisely that $y_1 = x_2$, or more pedantically,

$$\forall\, t \in \mathbb{R}, \quad y_1(t) = x_2(t).$$

Example 2.3: The actor model for the helicopter can be represented as a cascade composition of two actors as follows:



The left actor represents a Scale actor parameterized by the constant a defined by

$$\forall\, t \in \mathbb{R}, \quad y_1(t) = a x_1(t). \tag{2.6}$$

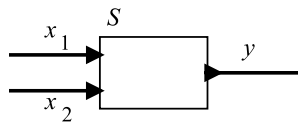
More compactly, we can write $y_1 = a x_1$, where it is understood that the product of a scalar a and a function x_1 is interpreted as in (2.6). The right actor represents an integrator parameterized by the initial value i defined by

$$\forall\, t \in \mathbb{R}, \quad y_2(t) = i + \int_0^t x_2(\tau) d\tau.$$

If we give the parameter values $a = 1/I_{yy}$ and $i = \dot{\theta}_y(0)$, we see that this system represents (2.4) where the input $x_1 = T_y$ is torque and the output $y_2 = \dot{\theta}_y$ is angular velocity.

In the above figure, we have customized the **icons**, which are the boxes representing the actors. These particular actors (scaler and integrator) are particularly useful building blocks for building up models of physical dynamics, so assigning them recognizable visual notations is useful.

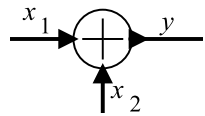
We can have actors that have multiple input signals and/or multiple output signals. These are represented similarly, as in the following example, which has two input signals and one output signal:



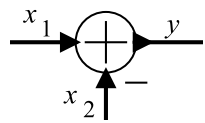
A particularly useful building block with this form is a signal **adder**, defined by

$$\forall\, t \in \mathbb{R}, \quad y(t) = x_1(t) + x_2(t).$$

This will often be represented by a custom icon as follows:



Sometimes, one of the inputs will be subtracted rather than added, in which case the icon is further customized with minus sign near that input, as below:



This actor represents a function $S\colon (\mathbb{R} \to \mathbb{R})^2 \to (\mathbb{R} \to \mathbb{R})$ given by

$$\forall\, t \in \mathbb{R},\ \forall\, x_1, x_2 \in (\mathbb{R} \to \mathbb{R}), \quad (S(x_1, x_2))(t) = y(t) = x_1(t) - x_2(t).$$

Notice the careful notation. $S(x_1, x_2)$ is a function in $\mathbb{R}^{\mathbb{R}}$. Hence, it can be evaluated at a $t \in \mathbb{R}$.

In the rest of this chapter, we will not make a distinction between a system and its actor model, unless the distinction is essential to the argument. We will assume that the actor model captures everything of interest about the system. This is an admittedly bold assumption. Generally the properties of the actor model are only approximate descriptions of the actual system.

2.3 Properties of Systems

In this section, we consider a number of properties that actors and the systems they compose may have, including causality, memorylessness, linearity, time invariance, and stability.

2.3.1 Causal Systems

Intuitively, a system is **causal** if its output depends only on current and past inputs. Making this notion precise is a bit tricky, however. We do this by first giving a notation for "current and past inputs." Consider a continuous-time signal $x : \mathbb{R} \to A$, for some set A. Let $x|_{t \leq \tau}$ represent a function called the **restriction in time** that is only defined for times $t \leq \tau$, and where it is defined, $x|_{t \leq \tau}(t) = x(t)$. Hence if x is an input to a system, then $x|_{t \leq \tau}$ is the "current and past inputs" at time τ.

Consider a continuous-time system $S : X \to Y$, where $X = A^{\mathbb{R}}$ and $Y = B^{\mathbb{R}}$ for some sets A and B. This system is causal if for all $x_1, x_2 \in X$ and $\tau \in \mathbb{R}$,

$$x_1|_{t \leq \tau} = x_2|_{t \leq \tau} \Rightarrow S(x_1)|_{t \leq \tau} = S(x_2)|_{t \leq \tau}$$

That is, the system is causal if for two possible inputs x_1 and x_2 that are identical up to (and including) time τ, the outputs are identical up to (and including) time τ. All systems we have considered so far are causal.

A system is **strictly causal** if for all $x_1, x_2 \in X$ and $\tau \in \mathbb{R}$,

$$x_1|_{t < \tau} = x_2|_{t < \tau} \Rightarrow S(x_1)|_{t \leq \tau} = S(x_2)|_{t \leq \tau}$$

That is, the system is strictly causal if for two possible inputs x_1 and x_2 that are identical up to (and *not* including) time τ, the outputs are identical up to (and including) time τ. The output at time t of a strictly causal system does not depend on its input at time t.

It only depends on past inputs. A strictly causal system, of course, is also causal. The Integrator actor is strictly causal. The adder is not strictly causal, but it is causal. Strictly causal actors are useful for constructing feedback systems.

2.3.2 Memoryless Systems

Intuitively, a system has memory if the output depends not only on the current inputs, but also on past inputs (or future inputs, if the system is not causal). Consider a continuous-time system $S\colon X \to Y$, where $X = A^{\mathbb{R}}$ and $Y = B^{\mathbb{R}}$ for some sets A and B. Formally, this system is **memoryless** if there exists a function $f\colon A \to B$ such that for all $x \in X$,

$$(S(x))(t) = f(x(t))$$

for all $t \in \mathbb{R}$. That is, the output $(S(x))(t)$ at time t depends only on the input $x(t)$ at time t.

The Integrator considered above is not memoryless, but the adder is. Exercise 2 shows that if a system is strictly causal and memoryless then its output is constant for all inputs.

2.3.3 Linearity and Time Invariance

Systems that are linear and time invariant (LTI) have particularly nice mathematical properties. Much of the theory of control systems depends on these properties. These properties form the main body of courses on signals and systems, and are beyond the scope of this text. But we will occasionally exploit simple versions of the properties, so it is useful to determine when a system is LTI.

A system $S\colon X \to Y$, where X and Y are sets of signals, is linear if it satisfies the **superposition** property:

$$\forall\, x_1, x_2 \in X \text{ and } \forall\, a, b \in \mathbb{R}, \quad S(ax_1 + bx_2) = aS(x_1) + bS(x_2).$$

It is easy to see that the helicopter system defined in Example 2.1 is linear if and only if the initial angular velocity $\dot{\theta}_y(0) = 0$ (see Exercise 3).

More generally, it is easy to see that an integrator as defined in Example 2.3 is linear if and only if the initial value $i = 0$, that the Scale actor is always linear, and that the cascade of any two linear actors is linear. We can trivially extend the definition of linearity to actors with more than one input or output signal and then determine that the adder is also linear.

To define time invariance, we first define a specialized continuous-time actor called a **delay**. Let $D_\tau : X \to Y$, where X and Y are sets of continuous-time signals, be defined by

$$\forall\, x \in X \text{ and } \forall\, t \in \mathbb{R}, \quad (D_\tau(x))(t) = x(t - \tau). \tag{2.7}$$

Here, τ is a parameter of the delay actor. A system $S : X \to Y$ is time invariant if

$$\forall\, x \in X \text{ and } \forall\, \tau \in \mathbb{R}, \quad S(D_\tau(x)) = D_\tau(S(x)).$$

The helicopter system defined in Example 2.1 and (2.4) is not time invariant. A minor variant, however, is time invariant:

$$\dot{\theta}_y(t) = \frac{1}{I_{yy}} \int_{-\infty}^{t} T_y(\tau) d\tau.$$

This version does not allow for an initial angular rotation.

A **linear time-invariant system** (**LTI**) is a system that is both linear and time invariant. A major objective in modeling physical dynamics is to choose an LTI model whenever possible. If a reasonable approximation results in an LTI model, it is worth making this approximation. It is not always easy to determine whether the approximation is reasonable, or to find models for which the approximation is reasonable. It is often easy to construct models that are more complicated than they need to be (see Exercise 4).

2.3.4 Stability

A system is said to be **bounded-input bounded-output stable** (**BIBO stable** or just **stable**) if the output signal is bounded for all input signals that are bounded.

Consider a continuous-time system with input w and output v. The input is bounded if there is a real number $A < \infty$ such that $|w(t)| \leq A$ for all $t \in \mathbb{R}$. The output is bounded if there is a real number $B < \infty$ such that $|v(t)| \leq B$ for all $t \in \mathbb{R}$. The system is stable if for any input bounded by some A, there is some bound B on the output.

Example 2.4: It is now easy to see that the helicopter system developed in Example 2.1 is unstable. Let the input be $T_y = u$, where u is the **unit step**, given

by

$$\forall\, t \in \mathbb{R}, \quad u(t) = \begin{cases} 0, & t < 0 \\ 1, & t \geq 0 \end{cases} . \tag{2.8}$$

This means that prior to time zero, there is no torque applied to the system, and starting at time zero, we apply a torque of unit magnitude. This input is clearly bounded. It never exceeds one in magnitude. However, the output grows without bound. In practice, a helicopter uses a feedback system to determine how much torque to apply at the tail rotor to keep the body of the helicopter straight. We study how to do that next.

2.4 Feedback Control

A system with **feedback** has directed cycles, where an output from an actor is fed back to affect an input of the same actor. An example of such a system is shown in Figure 2.3. Most control systems use feedback. They make measurements of an **error** (e in the figure), which is a discrepancy between desired behavior (ψ in the figure) and actual behavior ($\dot{\theta}_y$ in the figure), and use that measurement to correct the behavior. The error measurement is feedback, and the corresponding correction signal (T_y in the figure) should compensate to reduce future error. Note that the correction signal normally can only affect *future* errors, so a feedback system must normally include at least one strictly causal actor (the Helicopter in the figure) in every directed cycle.

Feedback control is a sophisticated topic, easily occupying multiple texts and complete courses. Here, we only barely touch on the subject, just enough to motivate the interactions between software and physical systems. Feedback control systems are often implemented using embedded software, and the overall physical dynamics is a composition of the software and physical dynamics. More detail can be found in Chapters 12-14 of Lee and Varaiya (2011).

Example 2.5: Recall that the helicopter model of Example 2.1 is not stable. We can stabilize it with a simple feedback control system, as shown in Figure 2.3. The input ψ to this system is a continuous-time system specifying the desired

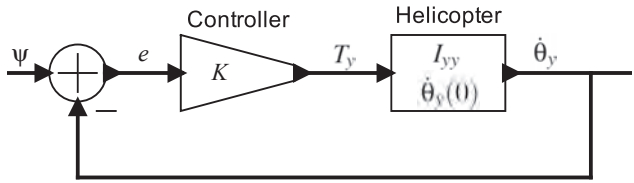


Figure 2.3: Proportional control system that stabilizes the helicopter.

angular velocity. The **error signal** e represents the difference between the actual and the desired angular velocity. In the figure, the controller simply scales the error signal by a constant K, providing a control input to the helicopter. We use (2.4) to write

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau) d\tau \tag{2.9}$$

$$= \dot{\theta}_y(0) + \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau)) d\tau, \tag{2.10}$$

where we have used the facts (from the figure),

$$e(t) = \psi(t) - \dot{\theta}_y(t), \quad \text{and}$$

$$T_y(t) = Ke(t).$$

Equation (2.10) has $\dot{\theta}_y(t)$ on both sides, and therefore is not trivial to solve. The easiest solution technique uses Laplace transforms (see Lee and Varaiya (2011) Chapter 14). However, for our purposes here, we can use a more brute-force technique from calculus. To make this as simple as possible, we assume that $\psi(t) = 0$ for all t; i.e., we wish to control the helicopter simply to keep it from rotating at all. The desired angular velocity is zero. In this case, (2.10) simplifies to

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) - \frac{K}{I_{yy}} \int_0^t \dot{\theta}_y(\tau) d\tau. \tag{2.11}$$

Using the fact from calculus that, for $t \geq 0$,

$$\int_0^t ae^{a\tau} d\tau = e^{at}u(t) - 1,$$

where u is given by (2.8), we can infer that the solution to (2.11) is

$$\dot{\theta}_y(t) = \dot{\theta}_y(0)e^{-Kt/I_{yy}}u(t). \tag{2.12}$$

(Note that although it is easy to verify that this solution is correct, deriving the solution is not so easy. For this purpose, Laplace transforms provide a far better mechanism.)

We can see from (2.12) that the angular velocity approaches the desired angular velocity (zero) as t gets large as long as K is positive. For larger K, it will approach more quickly. For negative K, the system is unstable, and angular velocity will grow without bound.

The previous example illustrates a **proportional control** feedback loop. It is called this because the control signal is proportional to the error. We assumed a desired signal of zero. It is equally easy to assume that the helicopter is initially at rest (the angular velocity is zero) and then determine the behavior for a particular non-zero desired signal, as we do in the following example.

Example 2.6: Assume that the helicopter is **initially at rest**, meaning that

$$\dot{\theta}(0) = 0,$$

and that the desired signal is

$$\psi(t) = au(t)$$

for some constant a. That is, we wish to control the helicopter to get it to rotate at a fixed rate.

We use (2.4) to write

$$\begin{aligned}
\dot{\theta}_y(t) &= \frac{1}{I_{yy}} \int_0^t T_y(\tau) d\tau \\
&= \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau)) d\tau \\
&= \frac{K}{I_{yy}} \int_0^t a d\tau - \frac{K}{I_{yy}} \int_0^t \dot{\theta}_y(\tau) d\tau \\
&= \frac{Kat}{I_{yy}} - \frac{K}{I_{yy}} \int_0^t \dot{\theta}_y(\tau) d\tau.
\end{aligned}$$

Using the same (black magic) technique of inferring and then verifying the solution, we can see that the solution is

$$\dot{\theta}_y(t) = au(t)(1 - e^{-Kt/I_{yy}}). \tag{2.13}$$

Again, the angular velocity approaches the desired angular velocity as t gets large as long as K is positive. For larger K, it will approach more quickly. For negative K, the system is unstable, and angular velocity will grow without bound.

Note that the first term in the above solution is exactly the desired angular velocity. The second term is an error called the **tracking error**, that for this example asymptotically approaches zero.

The above example is somewhat unrealistic because we cannot independently control the *net* torque of the helicopter. In particular, the net torque T_y is the sum of the torque T_t due to the top rotor and the torque T_r due to the tail rotor,

$$\forall\, t \in \mathbb{R}, \quad T_y(t) = T_t(t) + T_r(t)\ .$$

T_t will be determined by the rotation required to maintain or achieve a desired altitude, quite independent of the rotation of the helicopter. Thus, we will actually need to design a control system that controls T_r and stabilizes the helicopter for any T_t (or, more precisely,

any T_t within operating parameters). In the next example, we study how this changes the performance of the control system.

Example 2.7: In Figure 2.4(a), we have modified the helicopter model so that it has two inputs, T_t and T_r, the torque due to the top rotor and tail rotor respectively. The feedback control system is now controlling only T_r, and T_t is treated as an external (uncontrolled) input signal. How well will this control system behave?

Again, a full treatment of the subject is beyond the scope of this text, but we will study a specific example. Suppose that the torque due to the top rotor is given by

$$T_t = bu(t)$$

for some constant b. That is, at time zero, the top rotor starts spinning a constant velocity, and then holds that velocity. Suppose further that the helicopter is initially at rest. We can use the results of Example 2.6 to find the behavior of the system.

First, we transform the model into the equivalent model shown in Figure 2.4(b). This transformation simply relies on the algebraic fact that for any real numbers a_1, a_2, K,

$$Ka_1 + a_2 = K(a_1 + a_2/K).$$

We further transform the model to get the equivalent model shown in Figure 2.4(c), which has used the fact that addition is commutative. In Figure 2.4(c), we see that the portion of the model enclosed in the box is exactly the same as the control system analyzed in Example 2.6, shown in Figure 2.3. Thus, the same analysis as in Example 2.6 still applies. Suppose that desired angular rotation is

$$\psi(t) = 0.$$

Then the input to the original control system will be

$$x(t) = \psi(t) + T_t(t)/K = (b/K)u(t).$$

From (2.13), we see that the solution is

$$\dot{\theta}_y(t) = (b/K)u(t)(1 - e^{-Kt/I_{yy}}). \tag{2.14}$$

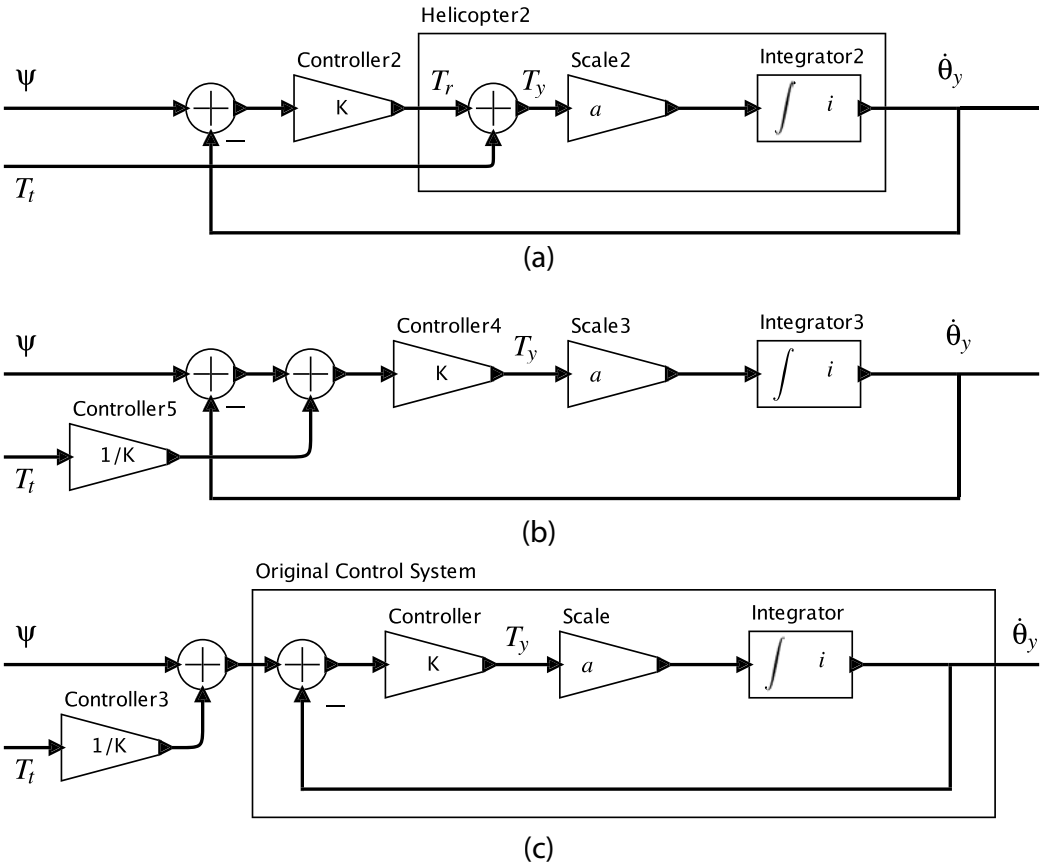


Figure 2.4: (a) Helicopter model with separately controlled torques for the top and tail rotors. (b) Transformation to an equivalent model (assuming $K > 0$). (c) Further transformation to an equivalent model that we can use to understand the behavior of the controller.

The desired angular rotation is zero, but the control system asymptotically approaches a non-zero angular rotation of b/K. This tracking error can be made arbitrarily small by increasing the control system feedback gain K, but with this controller design, it cannot be made to go to zero. An alternative controller design that yields an asymptotic tracking error of zero is studied in Exercise 7.

2.5 Summary

This chapter has described two distinct modeling techniques that describe physical dynamics. The first is ordinary differential equations, a venerable toolkit for engineers, and the second is actor models, a newer technique driven by software modeling and simulation tools. The two are closely related. This chapter has emphasized the relationship between these models, and the relationship of those models to the systems being modeled. These relationships, however, are quite a deep subject that we have barely touched upon. Our objective is to focus the attention of the reader on the fact that we may use multiple models for a system, and that models are distinct from the systems being modeled. The fidelity of a model (how well it approximates the system being modeled) is a strong factor in the success or failure of any engineering effort.

Exercises

1. A **tuning fork**, shown in Figure 2.5, consists of a metal finger (called a **tine**) that is displaced by striking it with a hammer. After being displaced, it vibrates. If the tine has no friction, it will vibrate forever. We can denote the displacement of the tine after being struck at time zero as a function $y: \mathbb{R}_+ \to \mathbb{R}$. If we assume that the initial displacement introduced by the hammer is one unit, then using our knowledge of physics we can determine that for all $t \in \mathbb{R}_+$, the displacement satisfies the differential equation

$$\ddot{y}(t) = -\omega_0^2 y(t)$$

where ω_0^2 is a constant that depends on the mass and stiffness of the tine, and where $\ddot{y}(t)$ denotes the second derivative with respect to time of y. It is easy to verify that y given by

$$\forall\, t \in \mathbb{R}_+, \quad y(t) = \cos(\omega_0 t)$$

is a solution to the differential equation (just take its second derivative). Thus, the displacement of the tuning fork is sinusoidal. If we choose materials for the tuning fork so that $\omega_0 = 2\pi \times 440$ radians/second, then the tuning fork will produce the tone of A-440 on the musical scale.

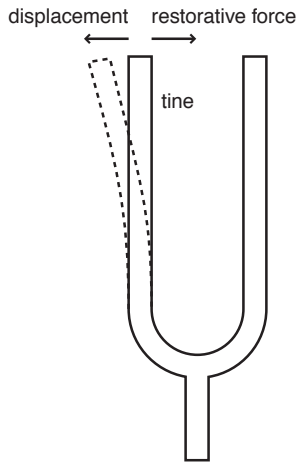


Figure 2.5: A tuning fork.

(a) Is $y(t) = \cos(\omega_0 t)$ the only solution? If not, give some others.

(b) Assuming the solution is $y(t) = \cos(\omega_0 t)$, what is the initial displacement?

(c) Construct a model of the tuning fork that produces y as an output using generic actors like Integrator, adder, scaler, or similarly simple actors. Treat the initial displacement as a parameter. Carefully label your diagram.

2. Show that if a system $S\colon A^{\mathbb{R}} \to B^{\mathbb{R}}$ is strictly causal and memoryless then its output is constant. Constant means that the output $(S(x))(t)$ at time t does not depend on t.

3. This exercise studies linearity.

(a) Show that the helicopter model defined in Example 2.1 is linear if and only if the initial angular velocity $\dot{\theta}_y(0) = 0$.

(b) Show that the cascade of any two linear actors is linear.

(c) Augment the definition of linearity so that it applies to actors with two input signals and one output signal. Show that the adder actor is linear.

4. Consider the helicopter of Example 2.1, but with a slightly different definition of the input and output. Suppose that, as in the example, the input is $T_y\colon \mathbb{R} \to \mathbb{R}$, as in the example, but the output is the position of the tail relative to the main rotor shaft. Specifically, let the x-y plane be the plane orthogonal to the rotor shaft, and let the position of the tail at time t be given by a tuple $((x(t), y(t))$. Is this model LTI? Is it BIBO stable?

5. Consider a rotating robot where you can control the angular velocity around a fixed axis.

(a) Model this as a system where the input is angular velocity $\dot{\theta}$ and the output is angle θ. Give your model as an equation relating the input and output as functions of time.

(b) Is this model BIBO stable?

(c) Design a proportional controller to set the robot onto a desired angle. That is, assume that the initial angle is $\theta(0) = 0$, and let the desired angle be $\psi(t) = au(t)$, where u is the unit step function. Find the actual angle as a function of time and the proportional controller feedback gain K. What is your output at $t = 0$? What does it approach as t gets large?

6. A DC motor produces a torque that is proportional to the current through the windings of the motor. Neglecting friction, the net torque on the motor, therefore, is this torque minus the torque applied by whatever load is connected to the motor. Newton's second law (the rotational version) gives

$$k_T i(t) - x(t) = I \frac{d}{dt}\omega(t), \tag{2.15}$$

where k_T is the motor torque constant, $i(t)$ is the current at time t, $x(t)$ is the torque applied by the load at time t, I is the moment of inertia of the motor, and $\omega(t)$ is the angular velocity of the motor.

   (a) Assuming the motor is initially at rest, rewrite (2.15) as an integral equation.

   (b) Assuming that both x and i are inputs and ω is an output, construct an actor model (a block diagram) that models this motor. You should use only primitive actors such as integrators and basic arithmetic actors such as scale and adder.

   (c) In reality, the input to a DC motor is not a current, but is rather a voltage. If we assume that the inductance of the motor windings is negligible, then the relationship between voltage and current is given by

$$v(t) = Ri(t) + k_b\omega(t),$$

   where R is the resistance of the motor windings and k_b is a constant called the motor back electromagnetic force constant. The second term appears because a rotating motor also functions as an electrical generator, where the voltage generated is proportional to the angular velocity.

   Modify your actor model so that the inputs are v and x rather than i and x.

7. (a) Using your favorite continuous-time modeling software (such as LabVIEW, Simulink, or Ptolemy II), construct a model of the helicopter control system shown in Figure 2.4. Choose some reasonable parameters and plot the actual angular velocity as a function of time, assuming that the desired angular velocity is zero, $\psi(t) = 0$, and that the top-rotor torque is non-zero, $T_t(t) = bu(t)$. Give your plot for several values of K and discuss how the behavior varies with K.

   (b) Modify the model of part (a) to replace the Controller of Figure 2.4 (the simple scale-by-K actor) with the alternative controller shown in Figure 2.6. This

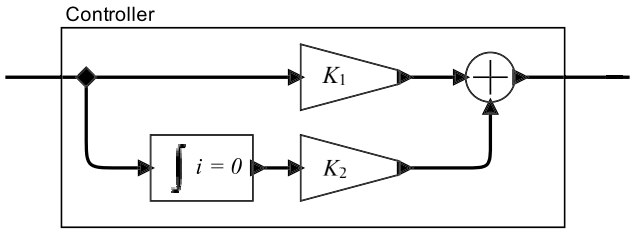


Figure 2.6: A PI controller for the helicopter.

alternative controller is called a **proportional-integrator (PI) controller**. It has two parameter K_1 and K_2. Experiment with the values of these parameters, give some plots of the behavior with the same inputs as in part (a), and discuss the behavior of this controller in contrast to the one of part (a).