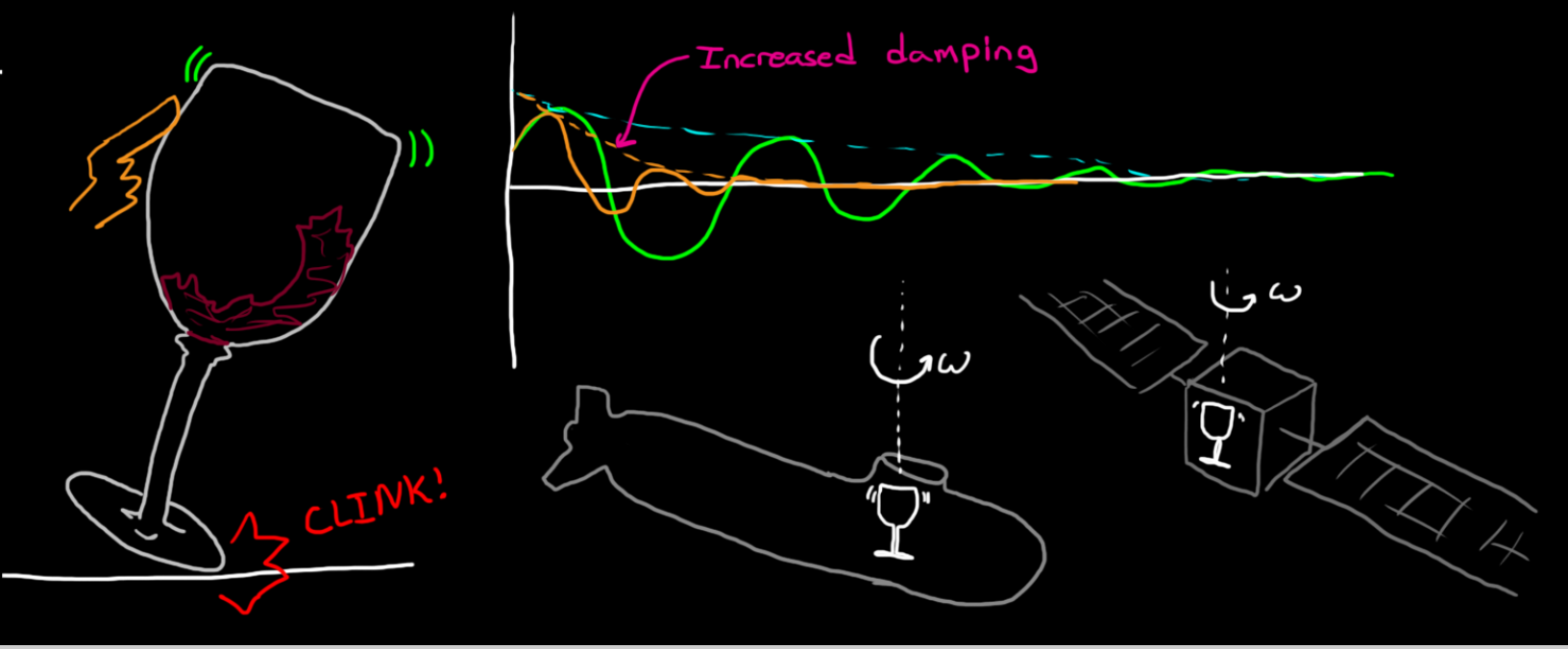
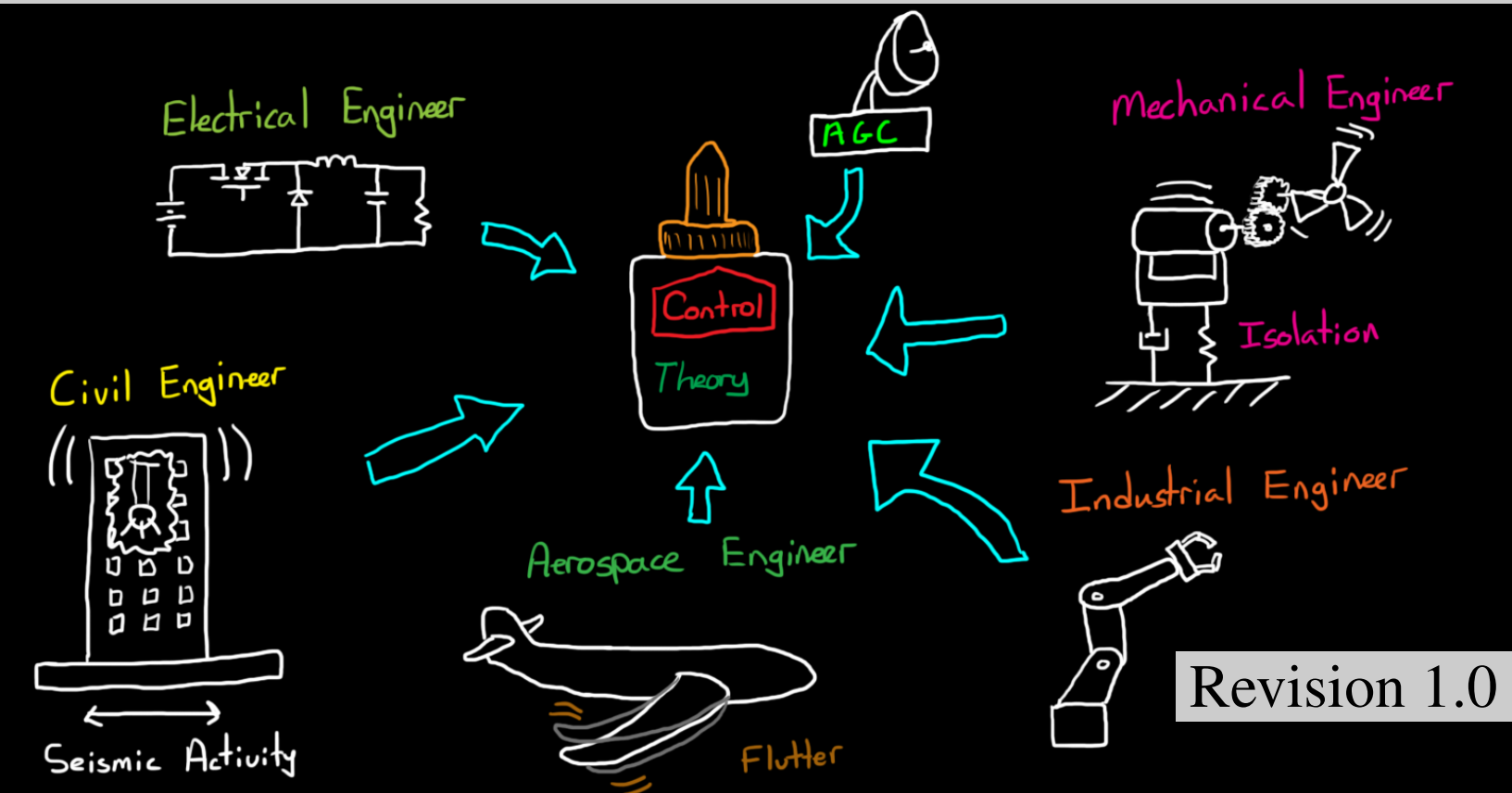


Fundamentals of Control Theory

An Intuitive Approach from the Creator of *Control System Lectures* on YouTube

Brian Douglas



Revision 1.0

Revision, 1.0

Printing Date, January 23, 2016

Contents

PREFACE

Welcome to the Fundamentals of Control Theory! This book is the direct result of my online video lectures on control system theory and the overwhelming positive feedback and encouragement I've received from my viewers to write a book. I started my YouTube channel (https://youtube.com/ControlLectures) because I was frustrated by the lack of straightforward and easy to understand videos on the topic and felt that what some students needed was a more practical and intuitive approach to understanding the material. This book is an extension of that idea.

I'm releasing this book one section at a time. Similar to how I create new videos on a monthly basis I will add new content to the book on a monthly schedule. This allows me to get the early chapters out there helping people right away while being able to have your reactions and responses influence the later chapters.

So with that in mind, as I write this book there are four goals I hope to accomplish that I think will make it a valuable resource for any aspiring controls engineer.

1. **Provide an intuitive understanding -** I want to start by saying there already exist several fantastic control system textbooks. Therefore, I don't think I would be able to write a useful book in this crowded field by presenting the same information in the same formal format. So I'm not going to try to duplicate them, instead I'm creating a book that is a bit different. The language is a little less formal - it's written as though we're having a conversation - and the mathematical proofs are a little more casual. However, I claim that what you'll learn from this book is just as useful as the existing textbooks because you'll gain an overall understanding of the problem and how to approach it.
2. **Update the book frequently -** One of the luxuries of making an eBook is that I can distribute updates quickly and cheaply. I am approaching this book more like a distribution of software, where bug fixes and minor layout changes can be updated and distributed as a point release (minor update) rather than a major new edition. With this model I can fix bugs and add content on a regular basis so that readers will always have the most up to date revision.
3. **Allow the readers to participate in improving the book -** How many times have you come across an error in a textbook or a really confusing explanation and wished you had a way of providing feedback easily to the author? I want to hear that feedback for this book! It is your feedback that will drive

the quick point releases and help me create the most useful textbook possible. This is why I provide a simple ticketing system where you can give me feedback on errors in calculations, vague or confusing explanations, and missing content. I ask that you please let me know any time you come across something in the book so that I can fix it and it won't confuse the next round of readers. For details on how to provide that feedback see Appendix A.

4. **Make the book as inexpensive as possible -** Lastly, college textbooks are expensive and if I want this book to really help students all over the world then it needs to be affordable. I understand that students don't have much money[1] and so having to buy several $180 books each semester is not high on your list of fun activities. That is why I'm releasing this book under the Creative Commons License and giving the book out for free to anyone who supports my work (which includes the videos I make) through konoz.io. You can get to my creator's page at konoz with this link.

   https://konoz.io/briandouglas

   For **any** amount of monthly support (even if it's just $1 a month) you will have continuous access[2] to the book and to all future updates. If you decide you no longer want to support me you will still get to keep and use the book you already have. So theoretically you could get the book for as little as $1. I think this is a good way of allowing people to decide how much they want to support me while not excluding people who really want to learn control theory but can't afford the book.

Engineering problems are inherently multi-disciplinary and so you have your choice of learning any number of specialized fields that will allow you to contribute to a project. But I think the best reason to learn control theory is that it is the glue that combines all other engineering fields and by understanding the fundamentals of control theory it opens the door for you to understand all of those other fields at a more basic level. It is actually a fascinating subject and through this book I hope to infect you with the same enthusiasm for the subject that I have.

Chapter 1 will describe the control problem. This chapter will set the stage for what we're trying to accomplish as control system engineers and define the terms that we'll use throughout this book.

Once written, the rest of the book will cover transfer functions, how we represent systems with block diagrams, and concepts like system stability, time, frequency, discrete domains, and system identification. We'll then cover

[1] When I was in college I had so little spending money by the end of the semester that I would buy large bags of white rice and a few condiments and then eat rice at every meal; rice and honey, rice and hot sauce, rice and mustard.

[2] Access means you get to copy the PDF onto your computer, put it on your eReader, or print it out, make copies of it, use it in your presentation, or part of your lecture, and even share it with your friends!

how we use specialized plotting tools like Root Locus, Nyquist plots, and Bode plots to analyze and understand our system. Later chapters will describe compensation techniques like lead and lag, loop shaping, and PID.

By the end of this book I hope you realize that control system theory is so much more than *just* tuning a PID controller or getting an inverted pendulum to stand upright. It's building models of your system and simulating it to make predictions, it's understanding the dynamics and how they interact with the rest of the system, it's filtering out noise and rejecting outside disturbances, it's designing or selecting proper sensors and actuators, and it's testing your system to ensure it'll perform as expected in an unexpected environment.

Now before you proceed any further I want to thank you for reading this book[3] and for supporting me to keep making improvements to this text. I hope you gain a better intuition into control theory and ultimately you become a more well-rounded engineer.

Brian Douglas

[3] and the preface! Who reads the preface anyway?

PART 1

The Big Picture

1 The Control Problem

In this chapter we'll get an overview of the big picture problem that we're trying to solve as control system engineers. This will give context to everything we'll cover in later chapters and in doing so I think will help you understand why you are learning the topics presented in this book.

1.1 What is a system?

To begin we describe exactly what a system is. The concept is really straight forward but since the term is so generic we tend to apply the word to describe just about everything. This can get confusing to someone new to the field when we refer to something called the control *system* which is then used to control the actual *system* and when put together the two parts make yet another larger *system*. As someone learning control theory the question becomes *what system am I working on?* To answer this let's start with the definition and work from there.

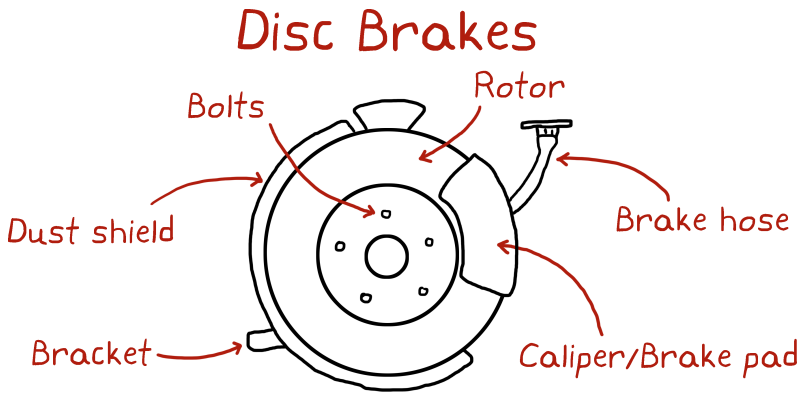
A **system** is a collection of interconnected parts that form a larger more complex whole.

Engineering projects are typically complex. Dividing complex projects into smaller pieces, or systems, simplifies the problem because it allows people to specialize in their functional area and not have to be a generalist in all areas. Therefore, as a specialist you might be working on just one of the interconnected parts that form the entire system. However, there might be many layers of complexity such that the small part you are working on is actually a complex system in its own right!

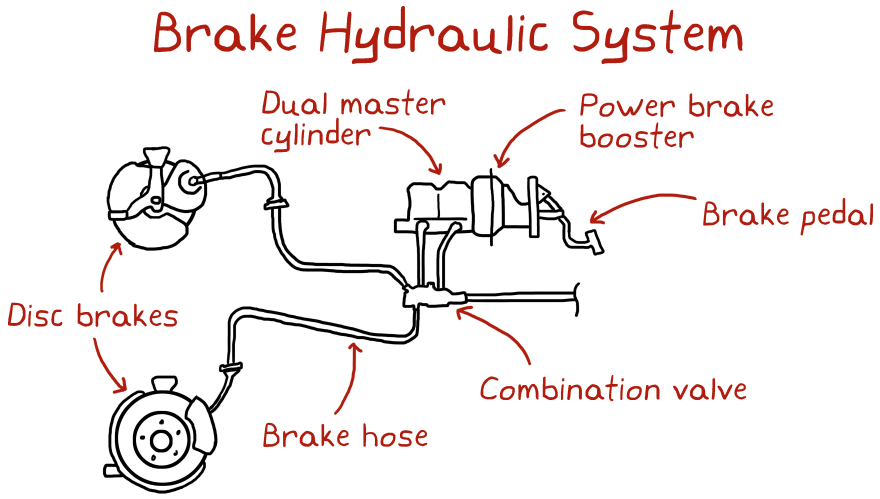
The same is true for specialists in control theory. As a control engineer your goal is to create something that meets the functional or performance requirements you set for the project. In general, we refer to the collection of the interconnected parts that are created specifically to meet these requirements as the control system. For any project other than the very simplest ones, however, the control system again might be a collection of interconnected parts that require specialists like sensor experts, actuators experts, digital signal processing experts, or state estimation experts.

To illustrate this let's imagine that you have accepted a job at an automotive company and you will be working on the braking system. At first glance you might suspect that you will be involved in all parts related to slowing the vehicle. However, there are many parts to the braking system on your car and it takes many different specialists to design the complete product.

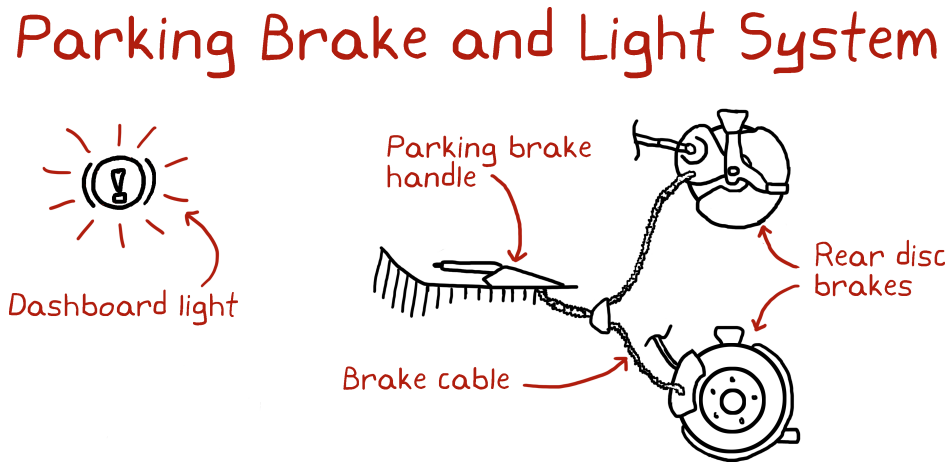
The most obvious component is the disc brake assembly in each wheel. This is the part that is actually converting the car's kinetic energy into heat energy and slowing down the vehicle. Yet the disc brakes are small systems on their own because they are made up of rotors, calipers, brackets, shielding, fasteners and hoses which allow the disc brakes to function correctly.



Engaging the brakes requires the brake hydraulic system which is responsible for transferring the pressure applied by your foot at the brake pedal through the power booster, dual master cylinder and the combination valve and finally to the brake calipers at each of the four wheels.



There is the mechanical parking brake system that bypasses the hydraulic system with a secondary cable path to the brakes and the brake light system that is responsible for lighting the tail lights and that annoying dashboard light that tells you the parking brake is engaged.



Finally there are any number of electronic brake control systems that override the human input to keep the vehicle from skidding on slick surfaces or a distracted driver from crashing into the car in front of them.

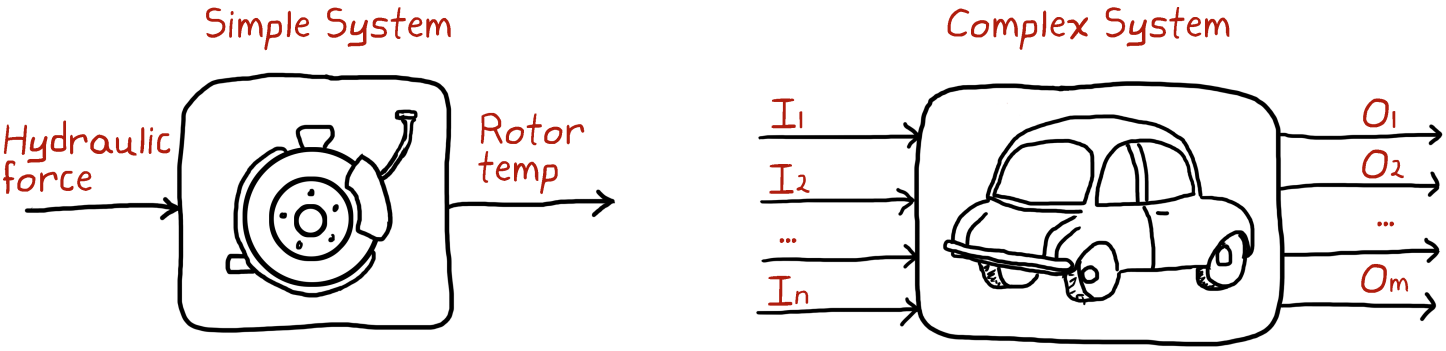
All of these smaller systems - the brakes, hydraulics, parking brake, lighting, and electronic controls - are the interconnected parts that form the larger and complete braking system. Furthermore, the braking system is just one of many interconnected parts that create the car itself.

As a control specialist in the brake department you might be responsible for writing and testing the algorithm for the electronic brake control system but have very little impact on, say, the cable routing for the parking brake.

Defining different systems allows complex projects to exist but it does create this potential confusion of everything being called a system. To mitigate this, depending on the field you work in, there is usually a term for each of the different hierarchical levels of complexity in a project. For example, a couple of parts creates a component which in turn creates a subsystem which then finally creates a system. I'm not going to try to define where the boundaries are between each of those because every industry and company does it differently. However, it is important that you recognize this and are clear what someone is specifically referring to when they ask you to design a controller for some system. In this book I will try to be explicit when I say system because what I'm referring to will change based on the context of the problem.

In general, we will represent any system graphically as a box. Arrows going into the box represent external inputs acting on the system. The system then responds over time to these inputs to produce an output - which are arrows leaving the box.

Typically we define the system in the box with a mathematical model that describes its equations of motion. At the moment we don't need to worry about the math, the most important thing is that we understand what this box means physically. For example the *system* could be really simple like a single disc brake with the inputs being the force of the hydraulic fluid and the output being the temperature of the rotor. Or the *system* could be complex like the entire car and have hundreds of inputs and thousands of outputs.



In both cases though our graphical representation would look similar, a box with arrows going into it and arrows coming out. Later we will string several systems (boxes) together to create complex block diagrams. These block

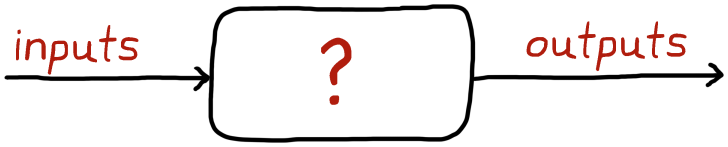
diagrams will contain the relevant interconnected parts of an even larger system. For the next section, however, this single box representation will give us some insight into three different types of problems that we'll face throughout this book and as practicing engineers.

1.2 The three different problems

You will notice that there are three parts to our simple block diagram; there is the box itself which represents a system, the inputs that are driving the system, and the outputs that the system generates. At any given time one of the three parts are unknown to you, and whichever part you don't know defines the problem that you are trying to solve.

1.2.1 The system identification problem

As a student you are usually given a mathematical model of your system at the start of your problem and then asked to perform some analysis of it or asked to design a control system around it. However, as a practicing engineer you won't always be given a model of you system[1], you'll need to determine that yourself. Determining the mathematical model is done through a process called system identification.

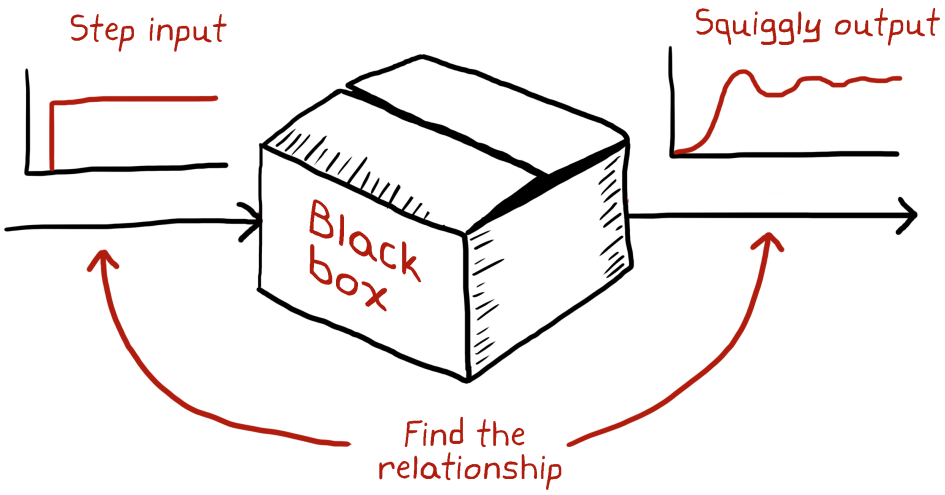


You might be doing system identification if you find yourself asking the following questions:

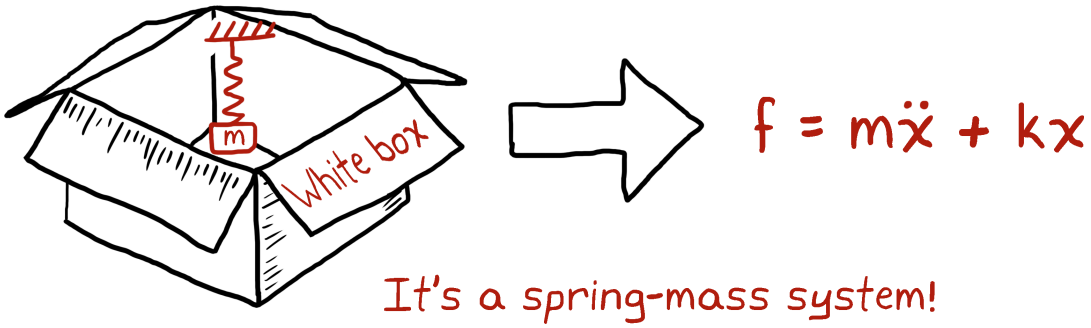
- How can I model the system that I'm trying to control?
- What are the relevant dynamics for my system (what should I model)?
- What is the mathematical equation that will convert my known inputs into my measured outputs?

There are at least two general ways that we can answer these questions. The first way is referred to as the black box method. Imagine you were given a box that you could not open but you were asked to make a model of what was inside. You could subject what is in the box to various known inputs, measure the resulting outputs and then infer what's inside the box based on the relationship between the two.

[1] In fact you'll almost never be given a model since what you're working on is very likely the first of its kind

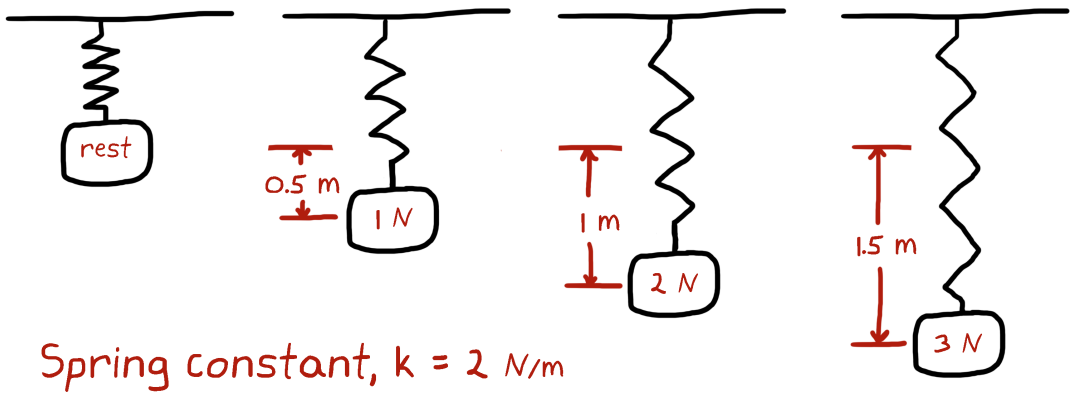


The second way to perform system identification is referred to as the white box method. Imagine now you were able to see exactly what was inside the box - all of the electronics, mechanisms, and software. Knowing the components of the system you could write the mathematical equations of the dynamics directly. This is exactly what you're doing when you use Newton's equations of motion or you are determining the equations of motion based on the energy in the system.



You might argue that you don't need to know the inputs or the outputs in order to write out some equations of motion, but that's not true. Even with this white box method there will be a need to set up a test with known inputs and measured outputs so you can get the unique parameters for your system. For example, you might need to model a linear spring - the equation of motion is well known - but will have to perform a stretch test to determine the exact spring constant for it[2].

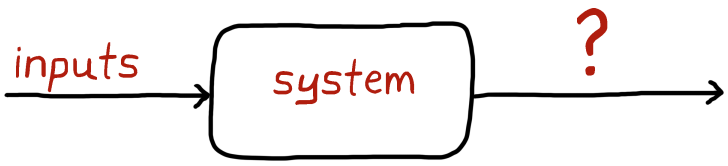
[2]I know you might be thinking, 'but I've been given the spring constant from the manufacturer so I don't have to perform that test!' And to that I have two responses, 1) are you really going to trust the manufacturer's datasheet if that parameter is important to you? and 2) if you are fine with the accuracy stated in the datasheet then this is a case where you *were* given the model of your system, just like in school!



For this test the input is the force that the mass is exerting on the spring and the output is the stretched length of the spring. We can tell from the relationship between the input forces and the output lengths that the spring constant is 2 Newtons per meter. System identification is an important part of designing a control system and so we'll discuss it in greater detail in a later chapter.

1.2.2 The simulation problem

If we know the inputs and the system dynamics then we can predict how the system will behave through simulation. This problem is interesting because you'll likely spend the majority of your design time in this stage. The trick here is figuring out what is the set of meaningful inputs and their ranges so that you have a complete idea of how your system behaves.

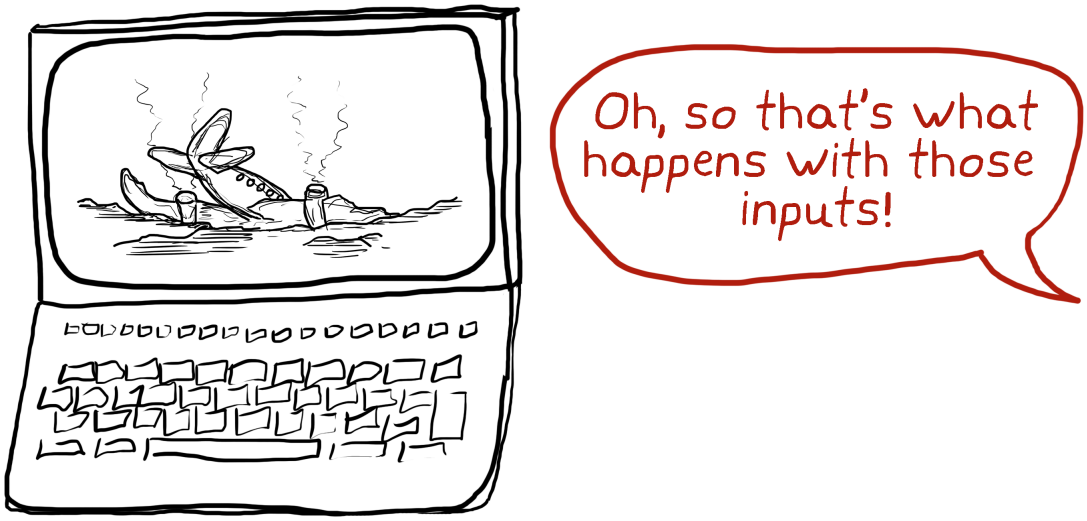


You might need to run a simulation if you find yourself asking the following questions:

- Does my system model match my test data?
- Will my system work in all operating environments?
- How does my system behave if I drive it with potentially destructive commands?

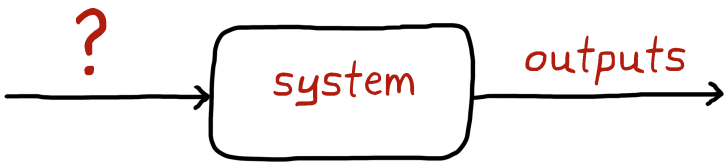
To see how simulation is important imagine you have a very good model of a passenger airplane and you're designing a pitch control system for it. You want to know how your system behaves across the entire operating

envelope and you're weighing the different approaches you could take. You could use a test airplane and fly in it every operating condition it could expect to see during its life and directly observe its behavior. The problem with this is that the operating envelope is huge[3] and flight test campaigns are expensive and so minimizing the amount of flight time could save your project a lot of money. Also flight tests can be dangerous to perform if you are trying to push the limits of your system to see how it reacts. Rather than risk project budget and possible human life it makes more sense to simulate your system.



1.2.3 The control problem

If we know the system and we know how we want the system outputs to behave then we can determine the appropriate inputs through various control methods. This is the control problem - how do we generate the appropriate system input that will produce the desired output? Control theory gives you the tools needed to design a control system which will generate the required input into the system. Without control theory the designer is relegated to choosing a control system through trial and error.



[3] You need to make sure that your pitch control system will function across all operating altitudes, wing angles of attack, air speeds, center of gravity locations, flap settings, and weather conditions.

You might need to design a control system if you find yourself asking the following questions:

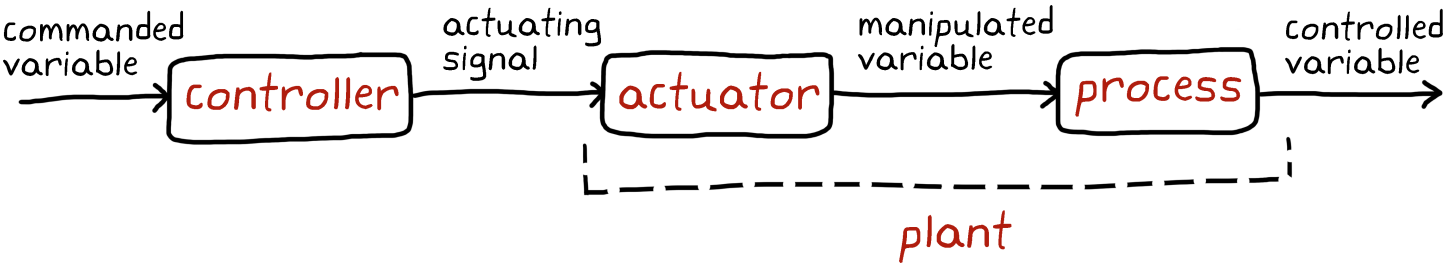
- How can I get my system to meet my performance requirements?
- How can I automate a process that currently involves humans in the loop?
- How can my system operate in a dynamic and noisy environment?

This book will lay out the fundamental tools needed to solve *the control problem* and in doing so I think you'll find that control theory can be challenging but a lot of fun and very intuitive. Before we move on to learning these tools let's take a step back and describe in more detail why we need control systems in the first place.

1.3 Why do we need a feedback control system?

Let's start with our simple system block diagram, but now the box isn't just any system, it's specifically a system that we want to *control*. From now on we'll refer to the system that is being controlled as the process[4]. The inputs into the process are variables that we have access to and can change based on whichever control scheme we choose and so we'll refer to them as the manipulated variables.

These variables are manipulated by an actuator. An actuator is a generic term that refers to a device or motor that is responsible for controlling a system[5]. Since the actuator is a physical device and is usually embedded within the process itself it can be useful to refer to the collection of both process and actuators as a single system that we'll call the plant[6].



The actuators are driven by an actuating signal that is generated by the controller. The controller is designed specifically to convert a commanded variable - which comes from someone operating this device or from a higher

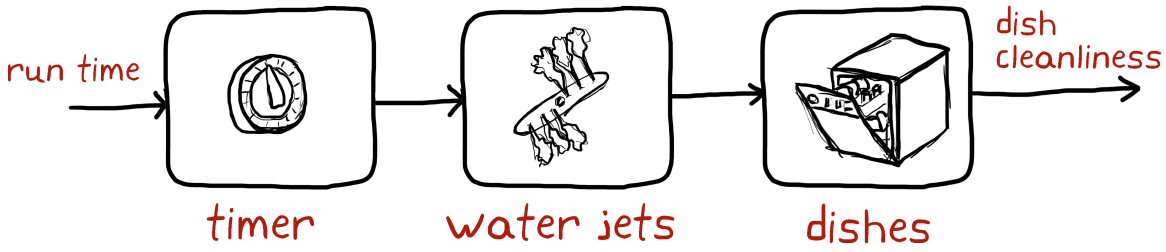
[4]Or if you lack creativity you could just call it the *controlled system*

[5]A car's engine and drive train are obvious actuators because they generate the force that manipulates the speed of the car (the process), but actuators can be less obvious as well like a voltage regulator in an electrical circuit

[6]Other textbooks and online resources might use plant and process interchangeably so be aware of that when referring to them. In this book, however, I will stick to this definition.

level control system - into appropriate actuating signals. At this point we have our first, and simplest, control system. As the operators, we could now select a set of pre-determined commands that we play through our controller. This will generate the resulting actuator commands which in turn affect the manipulated variable which then affects the process in a way that we desire. This type of control system is referred to as open-loop since there is no feedback from the output of the process.

Open-loop control systems are typically reserved for simple processes that have well defined input to output behaviors. A common household example of an open-loop control system is a dishwasher. This is an open-loop system because once the user sets the wash timer the dishwasher will run for that set time. This is true regardless of whether the dishes are actually clean or not when it finishes running. If the dishes were clean to begin with the dishwasher would still run for the prescribed time and if you filled the dishwasher full of pots with baked on grime then the set time might not be enough to fully clean them and would have to be run again. We accept this inefficiency in the run time of the dishwasher because the starting process (the dirty plates that we want cleaned) is generally well known and therefore the time it takes to clean them is pretty consistent.

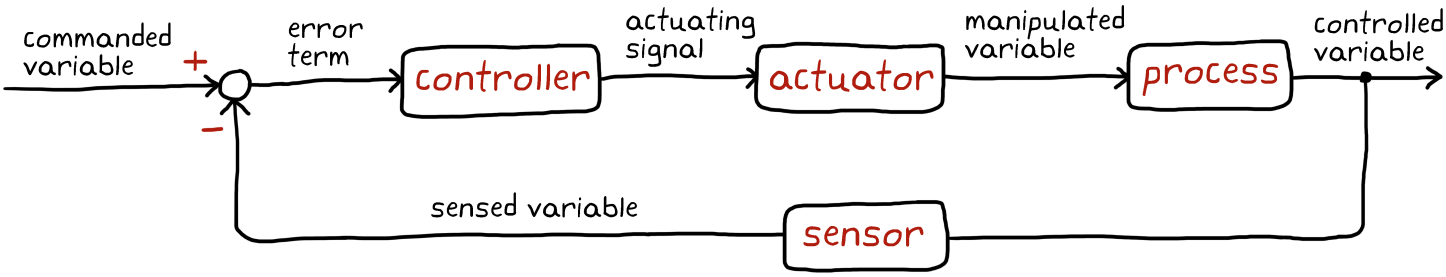


The manufacturers understand though that sometimes you will fill the dishwasher with grimy pots and want it to run for a longer period. Rather than build a complicated closed-loop system they have addressed this problem by adding additional pre-determined commands; that is they add multiple types of cleaning cycles that run for different times and at different temperatures. Then it is up to the user to select the correct set of commands to get the desired effect.

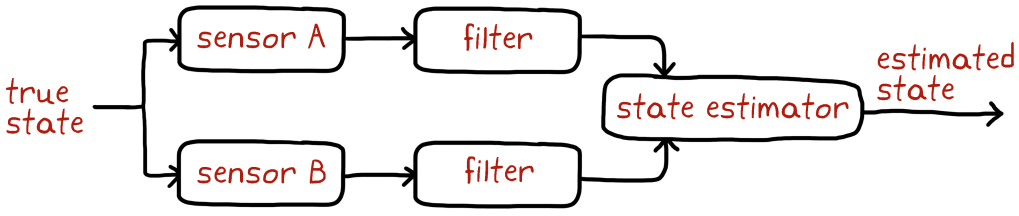
For any arbitrary process, though, an open-loop control system is typically not sufficient. This is because there are disturbances that affect your system that are random by nature and beyond your control. Additionally the process itself might have variations that you don't expect or prepare for[7]. Process variation and external disturbances will alter the behavior of your system - typically negatively - and an open-loop system will not be able to respond to them since it has no knowledge of the variation in the process output.

[7]One example of process variation is the change in electrical resistance over temperature. An open-loop control system that works at room temperature might not work when your process is extremely cold or hot due the variation in resistance throughout your electronics

So what can we do about this? We add feedback to our system! We accept the fact that disturbances and process variations are going to influence the controlled variable. However, instead of living with the resulting error we add a sensor that will measure the controlled variable and pass it along to our controller. Now we can compare the sensed variable with the commanded variable and generate an error term. The error term is a measure of how far off the process is from where you want it to be and the controller can use this to produce a suitable actuating signal which then produces a suitable manipulated variable which finally affects the process in such a way that it reduces the error term. The beauty of the feedback control system - or a closed-loop control system[8] is that it is able to react to changes to the controlled variable automatically by constantly driving the error term to zero.



The feedback structure is very powerful and robust which makes it indispensable as a control tool. Unfortunately, with the addition of the feedback structure comes new problems that we now have to address. We need to think about accuracy of the controlled variable at steady state, the speed with which the system can respond to changes and reject disturbances, and the stability of the system as a whole. Also, we've added sensors which have noise and other inaccuracies that get injected into our loop and affect the performance. To counter this last problem we can add redundant sensors that measure different state variables, we filter them to reduce the noise, and then we blend them together to create a more accurate estimate of the true state. These are some of the tools we can employ as system designers and part of what we'll cover in the rest of this book.



[8]The term closed-loop comes from the resulting loop that is formed in the block diagram when you feed back the controlled variable.

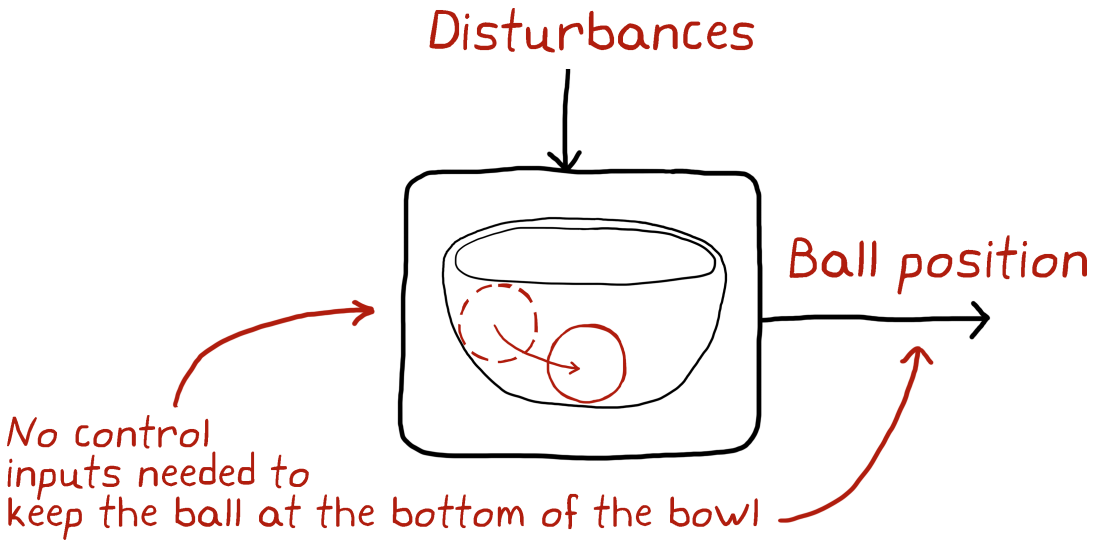
1.4 What is a control system?

From the first few sections you probably already have a vague understanding of what a control system is. You might be thinking that it is something that makes a system behave in an automated fashion or is something that allows a system to operate without human intervention. This is true, to some degree, but the actual definition is broader than you might think.

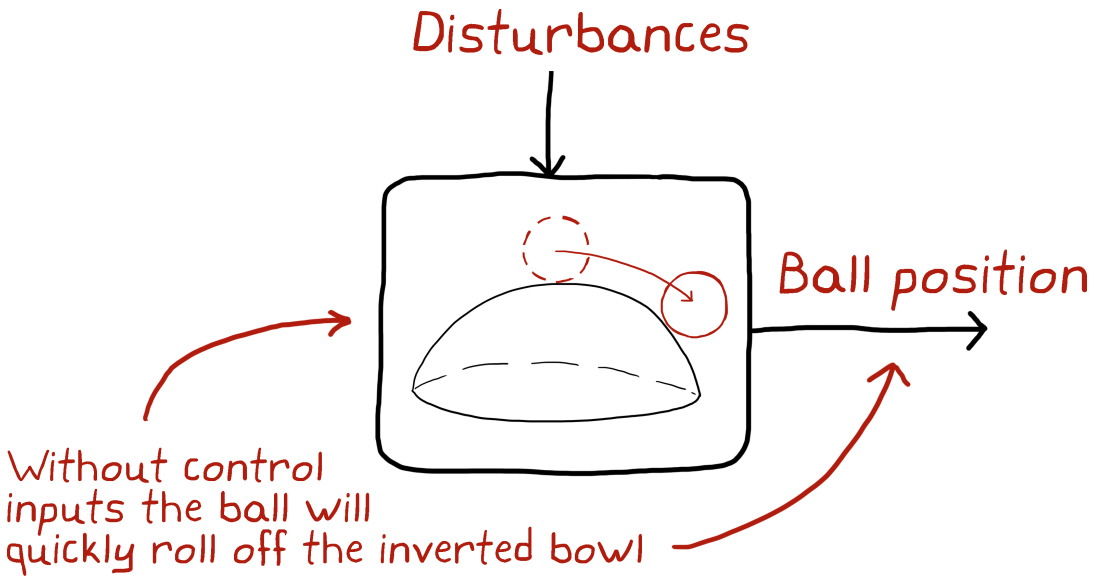
A **control system** is a mechanism that alters the behavior (or the future state) of a system.

Sounds like almost anything can be considered a control system, right? Well, one of the defining characteristics of a control system is that the future behavior of the system must tend towards a state that is desired. That means that, as the designer, you have to know what you want your system to do and then design your control system to generate that desired outcome.

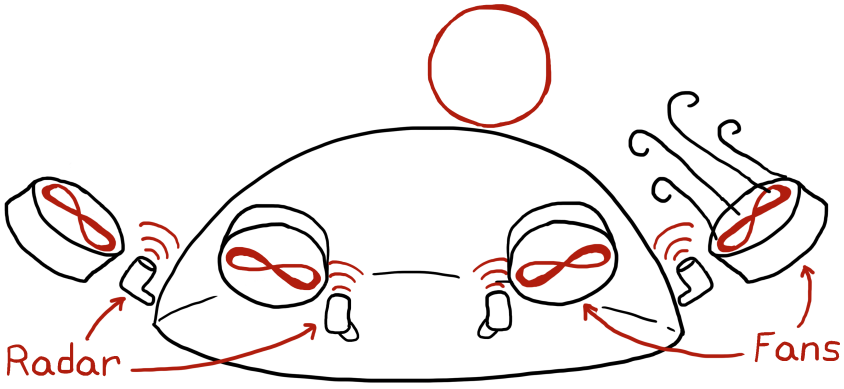
In some very rare cases the system naturally behaves the way you want it to and doesn't require any special input from the designer. For example, if you want a system that keeps a ball at the bottom of a bowl there wouldn't be a need for you to design a control system because the system performs that way naturally. When the ball is disturbed it will always roll back toward the bottom of the bowl on its own.



However, if you wanted a system that keeps a ball at the top of an *inverted* bowl, then you would need to design a control system to accomplish that. This is because when the ball is disturbed it will not roll back to the center naturally but instead continue rolling off the side of the inverted bowl.

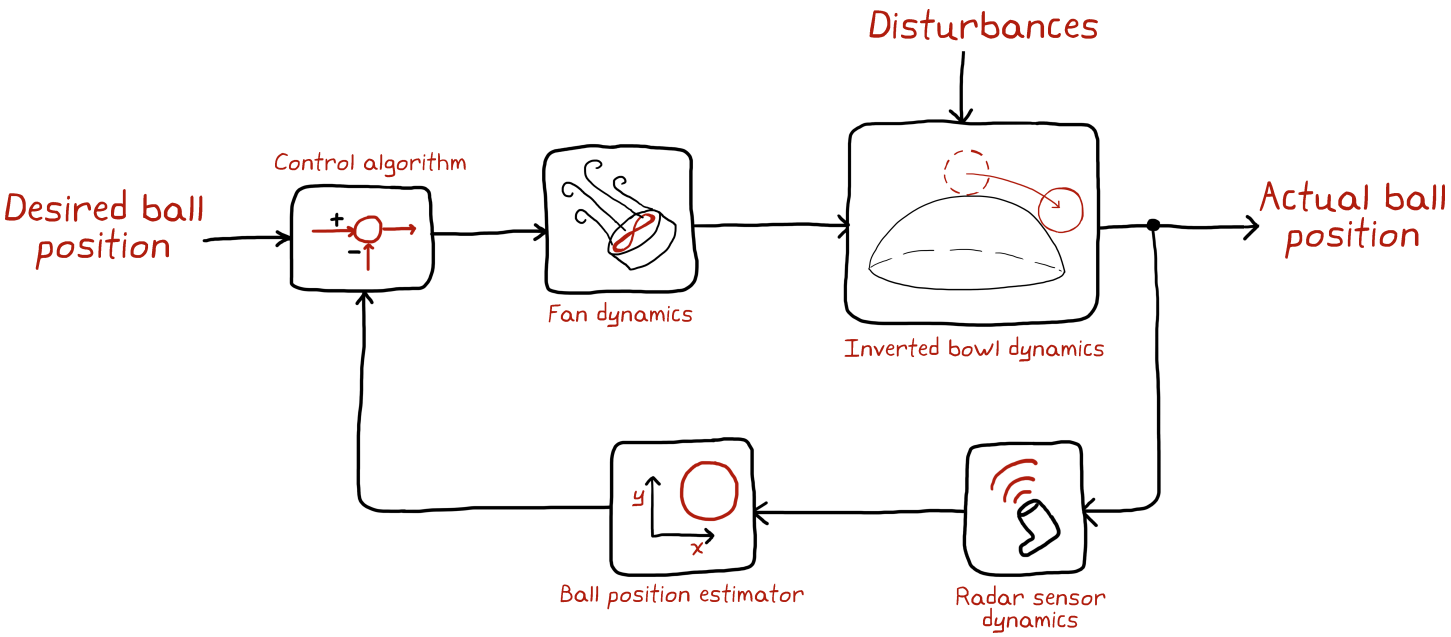


There are a number of different control systems that would work here. There is no *right answer* but let me propose a possible solution - even if it is quite fanciful and not very practical. Imagine a set of position detecting radar guns and wind generating fans that went around the rim of the bowl. As the ball deviated from the top of the bowl the fan closest to it would turn on and blow the ball back up to the top.



The set of fans, radar guns, position estimators and control algorithms would count as a control system because together they are altering the behavior of the ball and inverted bowl dynamics. More importantly, however, they're

driving the ball toward the desired state - the top of the inverted bowl. If we considered each interconnected part as its own little system, then the block diagram of the entire feedback system would look something like this:



This is a very natural way to split up the project as well because it allows multiple control specialists to work together toward a common goal. Instead of everyone trying to develop all parts of a complicated control system, each person would be responsible for designing and testing their part. Someone would be responsible for selecting the appropriate radar gun and developing the ball position estimator algorithm. Another person would be responsible for building the fans and the electronics to run them. Finally, a third person would be responsible for developing the control algorithm and setting the system requirements for the fan and radar specialists. Together these three ball balancing engineers would make up the control system team.

1.5 The First Feedback Control System

Feedback control systems exist in almost every technology in modern times, but there was a first feedback control system. Its story[9] takes place in the 3rd century BC in Alexandria, Egypt - 2000 years before the Industrial Revolution, at a time when Euclid was laying out the principles of geometry and Archimedes was yelling "Eureka!"

[9]If you're not interested in a story you can skip this section without loss of continuity in the book ... but c'mon, it's only a few pages and it's an interesting tale of the ancient ingenuity that led to modern day control theory

over discovering how the volume of displaced water could lead to the density of the object. Our protagonist is the Greek mathematician Ctesibius, inventor of the organ and widely regarded as the *father of pneumatics* due to his published work on the elasticity of air. We can overhear the conversation Ctesibius is having with one of his students, Heron, about an invention he has just completed - an invention that you will soon see is directly related to the Fundamentals of Control theory.

CTESIBIUS: I have done it!

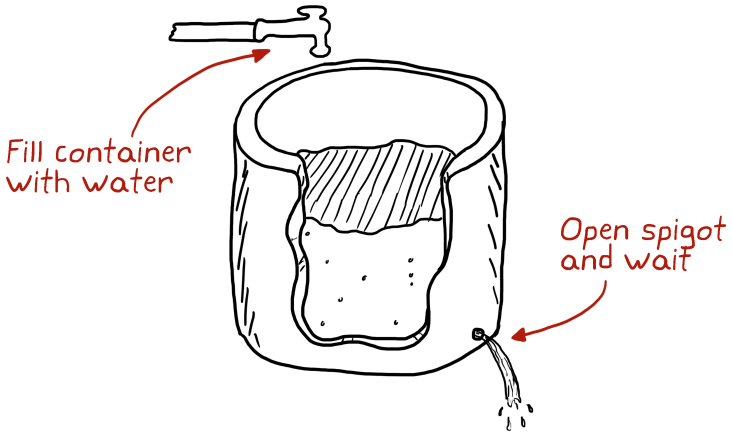
HERON: What is that Master Ctesibius?

CTESIBIUS: I have invented a mechanism - something rather ingenious I might add - that will allow *anyone* to know the time of day.

HERON: But Master, we already have a number of ways of knowing the time. As you well know I can glance over at this sundial and see from the casted shadow that it is just past 11 in the morning.

CTESIBIUS: That is true. The sundial is wonderfully simple and humans have been using it to tell time for at least 3000 years. But it has its problems. How can you tell the time at night, or if it is a cloudy day, or if you are inside a building?

HERON: That's easy! At those times we use our water clocks. We take a container that has a small spigot at the bottom and fill it with water. We let the water drain slowly through the spigot until it is empty. Since we know how long it takes to empty the container we then know how much time has passed.

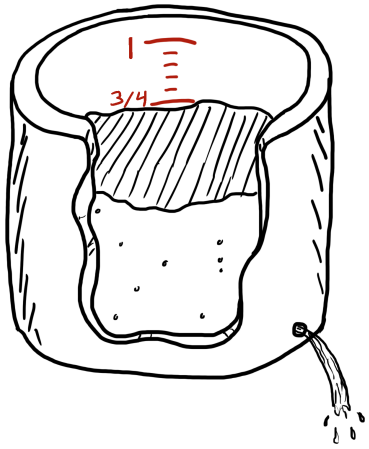


CTESIBIUS: Hmm, that is once again true, but what you have described is a *timer* and not a *clock*. There is a difference and it is very important. A timer is a device for measuring how much time has elapsed over some interval, whereas a clock's measurement is related back to the time of day. The sundial is a clock because we can see that it is 11am, but using your water container we only know that perhaps one hour has passed since we opened the spigot.

HERON: Well, if we started the spigot at 11am then when it completes we know that it is noon. Then we just start the process over again to always know the time.

CTESIBIUS: Exactly, that would make it a clock! But what if you wanted to know the time when the container was not yet empty?

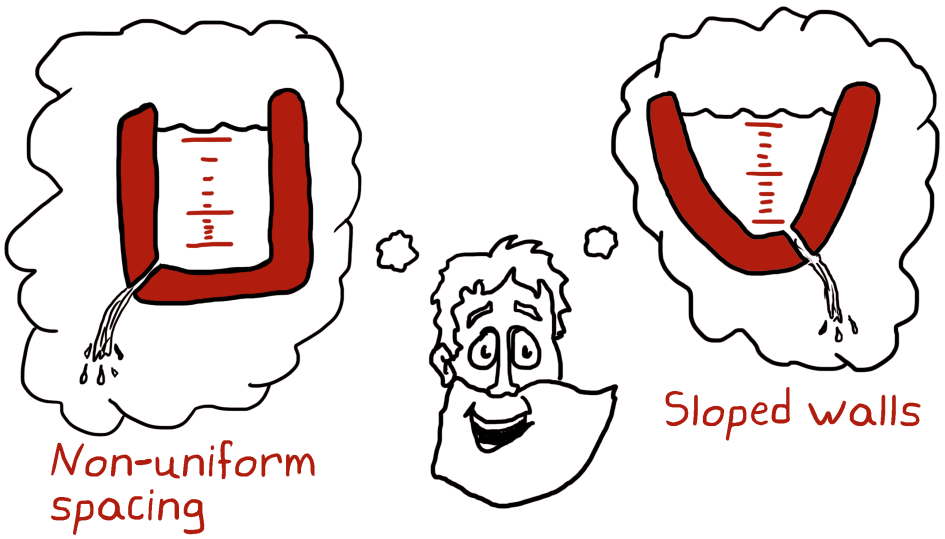
HERON: I guess we could see how full the container was by indicating the height of the water on the inside wall. Then if the container was three quarters full we would would say that one quarter of the time has elapsed.



CTESIBIUS: There is a problem with this method though. Can you see it? The water will drain from the spigot much faster when the water level is high and the flow will gradually slow down as it empties. Therefore, a three-quarter-filled container means that *less* than a quarter of the time has actually elapsed.

I can tell that you are not convinced that this is a problem, right? You are going to tell me that the markings on the inside of the container don't need to be uniformly spaced, or that

the walls of the container could slope inward so that water level will drop at a uniform pace.



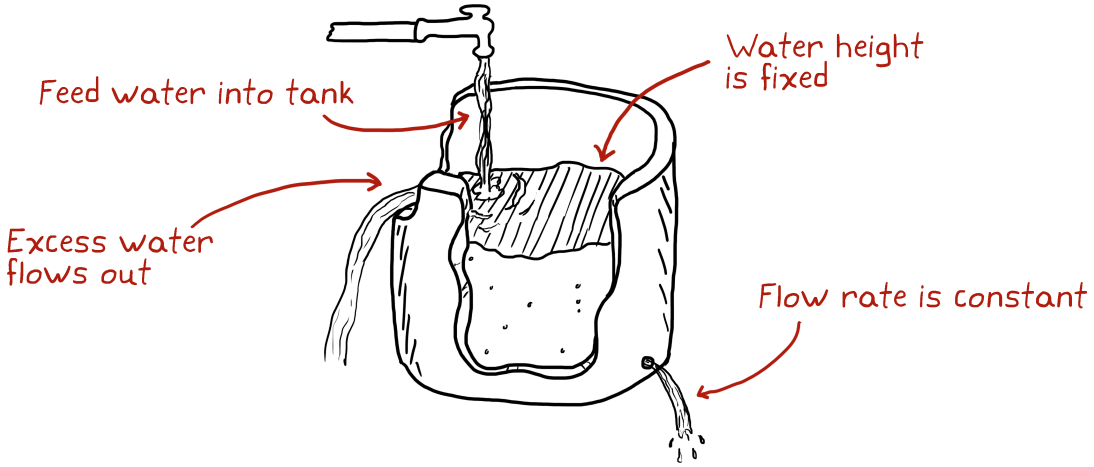
But this is hiding the real issue. By accepting that the flow rate through the spigot will change over time we need to have some extra knowledge - namely how to shape the inside of the container or where to mark the non-uniform indications on the inside wall. Once we create those markings or that container then we have no control over it. Any errors in manufacturing will generate errors in our results.

What we really need is a spigot that will have a steady and continuous flow rate. That way we don't rely on the container at all, we just need to know how much water comes out in an hour. From there, deriving *any* other time is as simple as measuring how much water has been released and comparing it to the amount released in an hour. Do you have any ideas on how to accomplish a steady flow rate from the spigot?

HERON: Let me think about it. The flow rate decreases because the water level drops in the container and the pressure exerted at the spigot is lower. So by keeping the water level in the container constant, the flow rate will be constant. Ah, but how do we keep the water level constant?

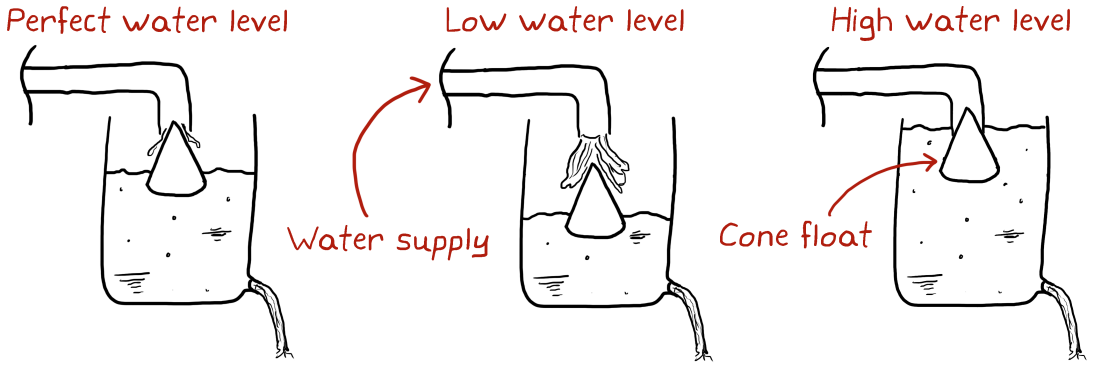
I got it! I'll modify the tank. It will have two spigots, a large one at the top and a smaller at the bottom. We'll feed water into this container at a much faster rate than the smaller

spigot can handle which will fill the container up. Once the water level reaches the larger spigot, it will begin to overflow, which will keep the water fixed at that height. Is this your invention Master Ctesibius?



CTESIBIUS: No, no, no! The overflow tank will accomplish a steady flow rate for sure but in solving our problem you've just introduced another one. Your clock will use and waste a lot of water. This will cause you to need a larger water source reservoir and that will make your clock less mobile. There is an elegant solution to this problem, and that is what I've just discovered.

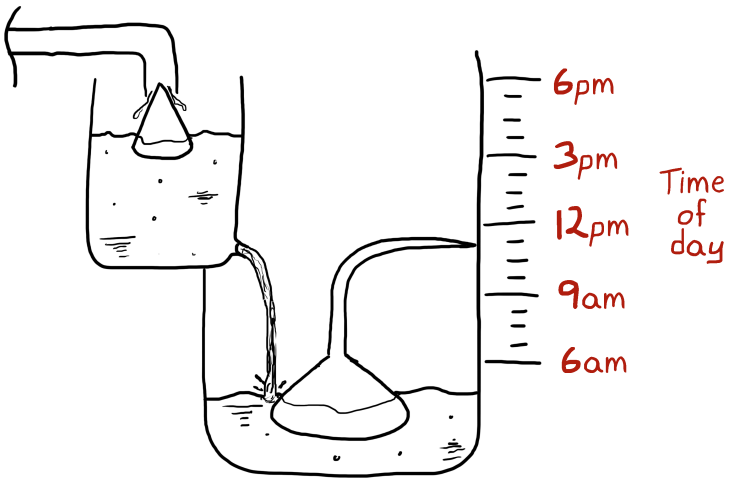
We still have the same container from before, but it is fed by a continuous water supply with a valve with a circular opening. We place a float in the shape of an inverted cone in the water container just below the water source inlet.



When the water level is high the float is pushed up into the valve such that it shuts off the water supply. As the container water level decreases, the float drops, allowing the supply

water to flow. The water level will reach a steady state once the float is low enough that the water flowing in is exactly equal to the water leaving the spigot. The beauty of this is that it is self correcting! If for some unforeseen reason the water level drops in the container, the float drops with it causing more water to be supplied to the container than it is losing through the spigot. This has the benefit of filling the container back up. It's brilliant!

Now we can add a larger container to catch the falling water and place a second float in there to mark the hour of the day. We only need as much water as it takes to fill the two containers in order to know the time all day. No wasted water!



HERON: That is brilliant! I bet there are many different applications where we could use this float regulator.

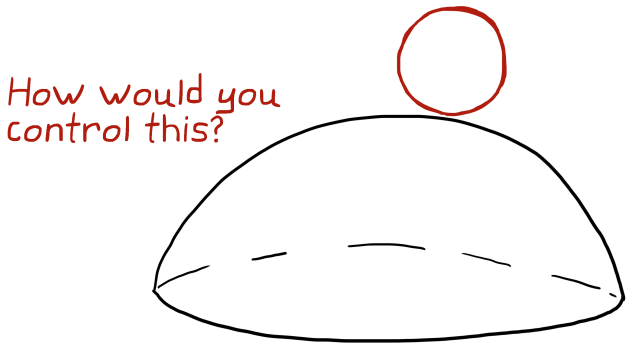
CTESIBIUS: Absolutely. Oh, is it really almost 12pm? I have to run, I'm needed at the Museum of Alexandria. Think of those other uses for this invention. I'd love to hear what you come up with.

With 2300 years of technological advancements to base your opinion on, this cone shaped float might not seem like much but the float regulator would go on to be used for many applications - in fact it's the same technology that is still used in modern day toilets. The real benefit of feedback control systems is that the system is self correcting. Now that systems could automatically adjust to the changing environment it removed the need for people to be part of the machine operation. This is similar to cruise control on your car removing *you* as the controller of your speed.

This is the goal of a control system engineer - through ingenious design and application develop systems that can regulate an output automatically. However, as you will see this opens a series of challenging and exciting problems. It will take mathematics to expose these problems and so now that we have the context of the problem we're trying to solve let's move onto the next chapter and discuss how we represent the problem mathematically.

1.6 Try This!

1. Come up with another possible control system for the inverted bowl problem.



a) What does your system use for sensors? Actuators?

b) Explain some of the benefits and drawbacks of your design? Consider things like power usage, sound volume, ease of assembly, ability to scale to larger systems, and coolness factor.

c) Make a sketch of your control system and point out how you would split up the project so that multiple people could work on it.

d) Share and discuss your ideas with the world and see what other students have created by going to `http://bit.ly/inverted_bowl`.

2. What additional applications are there for the float regulator? These can be problems that *have* been solved by a float regulator and those that *could* be solved by them.

3. How do humans act as the control system and how do they 'close the loop' with machines? As an example think about what a person is doing when they're driving a car.

4. Find examples of control systems in your everyday life (closed loop, open loop). Control systems don't always need to be machines, think about how feedback control is part of your relationships, your approach to studying for an exam, or the cycle of being hungry and eating.

5. Describe each of the following control systems. Determine the inputs, outputs and process. Describe whether it is open-loop or closed-loop.

a) Your home air conditioning system

b) Sprinkler system for your lawn

c) The lighting in a room

d) The fly-by-wire system on an aircraft

e) The population of a colony in an area with limited resources

Chapter Credits

Meg Douglas	Snohomish, Washington	Benjamin Martin	Dubai, UAE
Federico Pistono	1 AU		

Thanks for making this chapter awesome!

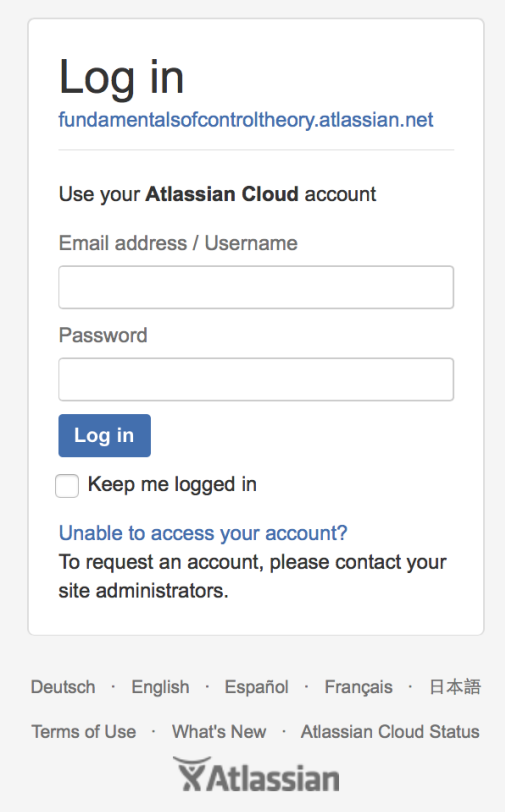
Appendices

A How to Provide Feedback

So you've found something that you'd like fixed in this book and now you want to know how to provide feedback? Well, I've set up a ticketing system using JIRA, an Atlassian product, to make it easy for you. You can access the ticketing system to write a new ticket - which they call an *issue* - or review the status of existing tickets by going to

fundamentalsofcontroltheory.atlassian.net [1]

When you click on the link the first thing you may see is the log in screen. I'm not sure why some people get it and some don't so if you don't see this screen then just move on to the next paragraph. If you do end up here just click on the link under the words "Log in" and it'll take you to the System Dashboard. You can try to log in with your Atlassian credentials if you have any but it won't work. I've disabled it for everyone except for anonymous users!



The System Dashboard is the first screen you see when you log in. It is where you can see which issues exist against each version of the book (an errata list), where you can add comments to those existing issues, or where you can create a brand new issue. When you click away from the system dashboard you can always get right back to it by clicking on the JIRA symbol in the upper left corner of any page.

To create a new issue click on the Create button in the header bar. This will bring up the Create Issue screen. Any field that is mandatory is marked with a *.

[1] Yes, this is a really long URL but it's just the title of the book so it should be easy to remember.

A.1 Filling out the Create issue screen

Project: this is where you specify which book you're writing the issue against. Since there is only a single book right now just select Fundamentals of Control Theory.

Issue Type: There is also only one type of issue that you can select; Comment. This is a generic issue type that will cover all of the different types of feedback that you might want to give. Some examples of the feedback I'm hoping to receive are as follows:

- Errors - There is something wrong with the content in the book. This could be as simple as a typo or something as grievous as an incorrect statement or a mathematical error.
- Unclear Explanations - A phrase, paragraph, or section of the book that is too hard to understand. Perhaps I've worded something strangely or I've left out some key bit of information that would clear the concept up.
- Missing Content - A section of the book is missing completely. Maybe you feel the best way to explain transfer functions is by first describing the Laplace transform but I've left it out. Or perhaps every control theory textbook worth its salt should describe Mason's rule for block diagram reduction. Let me know!

Summary: This is a short one-line description of the issue that you are writing. This will help people understand the general issue at a single glance.

This issue is against revision: This is a pull down menu where you can select which book you are writing the issue against. The book revisions will increase every release (about once a month) so it is important that you select the correct revision. You can find the revision on the cover page or the copyright page.

Description: This is a free form text field where you can write as much as you want to describe the issue you are reporting. This is not a required field (if the summary describes the problem fully) but I'd encourage you to fill this section out so I don't misinterpret anything. If you are reporting a section that is unclear you can use the description field to write out your recommendation for how it should be worded.

Would you like to be credited in the book?: I want to make sure I acknowledge all of the help I receive for creating this book ... that is if you want to be acknowledged. This is entirely up to you. If you select yes here and I use your suggestion I'll add your name and location to the chapter credits.

Appendix Credits

Meg Douglas Snohomish, Washington

Thanks for making the appendix awesome!