A Picture Book for the Roboticist— Why we Should Start with Hardware, and How to Teach so it Sticks

by

Aditya (Adi) Mehrotra
S.B. Massachusetts Institute of Technology (2022)

Submitted to the **Department of Mechanical Engineering** in partial fulfillment
of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by:	Aditya Mehrotra Department of Mechanical Engineering May 10th, 2024
Certified by:	Professor Sangbae Kim Department of Mechanical Engineering
Accepted by:	Nicolas Hadjiconstantinou Professor of Mechanical Engineering Graduate Officer, Department of Mechanical Engineering

A Brief Dedication:

To my students, I can only hope you've learned as much from me as I have from you. And to Ms. Leslie Anderson, my high school chemistry teacher, thank you for teaching me what it means to learn.

And to Max, you will be missed ❤️.

A Picture Book for the Roboticist—
Why we Should Start with Hardware, and How to Teach so it Sticks

by

Aditya (Adi) Mehrotra

Submitted to the **Department of Mechanical Engineering** **o**n May 10th, 2024 in partial fulfillment of the requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

This thesis explores why and how to teach hardware design in relation to building intelligent systems. We focus on the concepts of modeling, embedded systems, and actuation, and develop a series of hands-on exercises to teach specific concepts based on previous work. We identify and explain the concept of the translation layer, which we define as the interface between high-level controls and the hardware system. We explain the importance of hardware engineering to its operation and explore the role of the hardware engineer in building this layer. We use these ideas to build an undergraduate curriculum in robotics, the syllabi of four core classes, and hands-on exercises for their associated lab components.

Along the way we focus on the science of learning that often doesn't make its way into engineering education. We present a summary of key concepts surrounding how our students learn and use this to explain why hardware engineering is a good medium for teaching. We use this to build a loose design paradigm for what 'works' in engineering teaching. And we use that design paradigm to build the aforementioned hands-on exercises.

Additional discussions include topics that should be considered when building a curriculum including providing space for low-stakes curiosity, teaching our students about the application of their work to global problems, and including narratives on learning in our teaching.

Thesis Advisor: Sangbae Kim

Title: Professor of Mechanical Engineering

Acknowledgements

To Dr. Dawn Marie Wendell. Thank you for putting the book *Make it Stick* in my hands. Thank you for teaching me about education and learning. Thank you for being a wall to bounce ideas off of and an incredible source of advice and support. Most of all, thank you for everything you did for MIT and our students.

To Professor Sangbae Kim. Thank you for taking a chance on me in graduate school. Thank you for giving me the chance to explore, and for supporting me throughout the process of an unconventional thesis. Thank you also, for letting me teach with you.

To Professor Kait Becker. Thank you for letting me TA for you in 2.008, and for everything else as well. Your kindness is a constant inspiration to me.

To Fischer. Thank you for being one of the best friends a guy could ask for. I cannot describe how much I've learned from you, and how much fun it was to teach PCB design with you. To Winnie and Will thanks for taking what we started and making it grow. To Lee and Jesse, thanks for helping us make the class what it was. Can't wait to do it again with y'all. And to everyone who supported the class along the way, thank you too.

To my friends in the MIT Biomimetic Robotics Lab. To Elijah, Andrew, and David, thank you for teaching me how to build awesome things, the fun times, and the advice and support. To Steve, thank you for helping me navigate this thesis as I struggled to put my thoughts into words. To Kendrick and Jess, thanks for being buddies for the last two years. To the rest of my lab mates, thanks for always being there.

To Ana, Heewon, Dan, Libby, and Nancy thanks for giving me a home at D-Lab. You work means so much to the communities you support, and to the students of MIT. To Emily, thanks for being an amazing mentor and friend all these years, can't wait to go back to Ghana with you. To Jack, let's go rallying shall we?

To the staff at the MIT Edgerton Center. Thanks for supporting me in all my endeavors from the beginning of MIT until now. My appreciation for you is profound. To Mark and Pat, thank you for what you do.

To my family on Solar Car and EVT. There's too many of you to name, but thank you all for making the drive a good one. To Quang, and Caroline specifically, thank you for teaching me what I know. To Jack and Cameron, thank you for all you've done for me and the team. To Diane, Annemarie, and Christopher, aerofrosh are forever!

To my friends. Anita and Cat, thank you for being the best roomies a guy could ask for. To Shaida, Stefan, Natasha, Aashini, and Andrew, from Simmons to now—you all rock. Thanks for the memories. To Ronak, Summer, Jenny, and Hannah—I appreciate you lots, and always will. To María, thank you for the adventures, and the music.

And finally to my family. Thank you Mama and Papa for being a constant source of support in everything I do. Thank you so much for all you've given up for Varun and I. To Varun, thanks for being the best 'lil bro I could have asked for. To our family friends, thank you for being the people we look forward most to spending time with when we come home.

Contents

List of Figures

List of Tables

Preface

When I first started this project, the goal was to explore the possible solutions to a loose conclusion that a number of us who have spent time teaching the subject of robotics had come to: we're not teaching what we need to be teaching, and what we are teaching we're teaching poorly.

Background

We had arrived at this opinion from a couple places.

The first side of it came from trends in industry, and academia regarding *perceived* attitudes towards the relative relevance of 'hardware design' and 'computation' (these terms being ones that describe very broad fields of study). We noticed a shift in institutions of higher education during new degree program rollouts, and this is especially relevant to electrical engineering and computer science departments, to move away from teaching the fundamentals behind 'traditional' electrical engineering and *instead* focus on teaching our hardware engineers the concepts behind artificial intelligence [2].

Despite the fact that I don't personally study, nor have much intent to study, computational methods I don't deny their utility. When applied considerately (and with a rigorous regard for both ethics and impact) these tools can be incredibly useful. What we started to take issue with was the seemingly increasing 'one or the other' attitude.

The second side of this feeling came from our observations about our students' learning outcomes. One of the first things we noticed was that there was a large difference between the ability to connect theory to practice between students who had and had not spent time on student engineering teams. The former could generally do this better than the latter. Regarding the students who did this 'less-well', a number of lab instructors that I had conversations with said that there's been a trend over recent years of our students exhibiting an increasing in-ability to use physics to design and build real things, and that this trend is accelerating. *"They're good at the math, they're pretty good at building things too, but not at the connection point. They have difficulty using one to inform the other."*

When I asked one of my professors the question: *why do you think this is the case?* He took out his phone from his pocked and waved it at me.

About a year after we had arrived at these conclusions, and while I was deep into working on this thesis, I came across Crispin Mount Miller's thesis from the MIT Mechanical Engineering Department published in 1999.

> *"I decided to develop these exercises because I have found that most of the students coming into the M.I.T. design-class sections I have taught (1988 to 1993) have not been skilled at connecting physical theory to the hardware in front of them—they may be able to handle equations very well, but they cannot look at a simple machine and see the physics in it. One place this weakness shows up most prominently is in their mistakes and bewilderment when they try to design and build rudimentary machines in the department's student shop.*
>
> *Some teachers say that this problem is a recent development; I have not been teaching long enough to know. I can think of various trends in our culture which could cause a change in this direction (decline of skilled trades as role models, passive mass-media entertainment, households full of things with "no user-serviceable parts*)-and while young people in other times may have learned the practical context of their book-learning on their own, now that many evidently do not, education's complacent disregard of it (except maybe in vocational schools) does not help."*
> *— pg. 10* [1]

I looked back at my own notes, and realized we'd come to the *same* conclusions on teaching engineering over 20 years later. The narrative he was telling was scarily similar to our own.

I don't think its fair at all to say what we *know* about teaching, and teaching engineering hasn't changed for the past 20 years, but it started to make me wonder if what we *do know* in terms of techniques that 'work' has not made it into the classroom. A few instructors I talked to said it hasn't.

Methods

I started exploring these two ideas in a breadth-first fashion.

I started with examining a number of systems[1] our students would be expected to engineer after graduating that combined the concepts of *intelligence* and *mechatronics* and analyzed their subcomponents. I went ahead and compared this to what MIT's curriculum was teaching, and what curricula at other schools were teaching to identify specific gaps between what our students *need* to know and what they are being taught. As one does in the research phase, I also looked at any previous writings I could find on the subject of 'what we should be teaching' in robotics. And I started to organize this into a curriculum for teaching robotics to the undergraduate student.

At the same time, I started doing a lot of reading on teaching and learning. First, I started with learning generally exploring cognitive science's literature on memory, retention, and application of knowledge. Then I started to look for studies that specifically applied this to teaching engineering. Having found relatively few, I started to draw my own connections between observations I made during my own experiences teaching engineering, and what the literature was saying regarding techniques for robust learning. Then I started to use what I'd learned to think about *how* I'd approach teaching the topics that I think we should be teaching, and started to develop a number of 'exercises' or 'experiences' we could provide our students with.

Initial Findings

Throughout this process I arrived at a couple of conclusions.

First, on the side of 'what' we should be teaching in robotics. A lot of work has gone into teaching what I think are the two critical sides of the field. The first side is intelligence, which is a combination of decision making algorithms, information processing, and computation. The second side is mechatronics which is a combination of actuation, mechanical design, power electronics, and engineering of the physical hardware system. These are all topics that we teach, and we can argue that current curriculum structure isn't necessarily conducive to how wide field robotics is, but courses on these topics do exist. What seemed to be a trend, however, is that we teach these topics as if they are separate and little work has been done to explore how to teach the large amount of engineering that goes into *connecting* the hardware system to the upper intelligence levels at least at the undergraduate level.

Second, on the side of 'how' we should be teaching. It's true that a lot of learning science doesn't make its way into the engineering classroom. But it's also true that it's difficult to find (I'm not saying it doesn't exist) documentation on how to translate the concepts of cognitive science into structuring our teaching *specifically* related to engineering. By this I mean taking the concepts of cognitive science, and tying them directly to the parts of engineering teaching that work and do not work in some illustrative way. There is some work that documents the effectiveness of specific techniques that teachers have performed in classroom settings, but little has been offered in the form of generalization at least in the field of electrical and electronics engineering. I assume that this is not because it's unknown to everyone rather that it's un-documented, and to some extent we are all still trying to figure it out.

So while I did structure a curriculum and a set of core classes to teach robotics (and specifically the applications of robotics to solving global problems) to the undergraduate student, I decided that the bulk of

[1] A trend has been to call systems (that aren't traditionally what we think of when we hear the word 'robotics' but rely on many of the fields key ideas) Intelligent Physical Systems (IPS) [3][4]. What we call these systems does not matter. The ideas behind them are far more important and useful if we apply them rigorously and ethically to solve problems.

the thesis should spend time contributing to the documentation of a few topics that I feel haven't gotten enough attention in the literature to date.

Organization of this Work

The recommended reading section details specific works I believe were formative to the ideas in this thesis and deserve additional callout past being references. They provide good framing.

The first chapter starts by providing an overview of the systems we're interested in teaching. This includes robots, but also robotic's application to energy systems, biomedical devices, manufacturing, agriculture, and other fields that benefit from the integration with computation and hardware. I then spend some time discussing what we came to term the 'translation' layer, which is the interface between the computation and the hardware, and describe the engineering that goes into building that layer so these systems can work robustly in the real world. I then move onto discussing the role of the hardware engineer in building these systems, and through a few stories identify specific topics of focus for teaching. These topics form part of the 'core' in the curriculum we present in Appendix B.

The second chapter of this work focuses on learning. I start by documenting what I've learned from cognitive science on both how we learn, and specific techniques that seem to lead to robust learning. I connect these learnings to specific experiences I've had teaching engineering that seemed to work well in terms of retention and practical application of knowledge from the perspective of the student. I talk about debugging, and why hardware design is a good medium for teaching engineering, and use these learnings to recall a very 'loose' design paradigm for teaching engineering. This design paradigm is then used to structure a number of exercises to teach the topics identified in Chapter 1.

The third chapter of this work includes discussions on additional areas of focus at both the curriculum and classroom level for teaching, and opportunities for future work.

Then there are various appendices with content that includes the curriculum and classes we defined, various exercises for teaching specific topics, and lots of figures that document the process.

Preface References

1. C. M. Miller, "So can you build one? : learning through designing--connecting theory with hardware in engineering education," PhD Thesis, Massachusetts Institute of Technology. Department of Mechanical Engineering, 1995. Available: https://dspace.mit.edu/handle/1721.1/11548

2. T. LONG, "Short Circuited: Electrical Engineering Degrees in the United States," Information Technology & Innovation Foundation, Apr. 2023. Available: https://www2.itif.org/2023-ee-degrees.pdf

3. K. Petersen, ECE 3400 – Intelligent Physical Systems, Cornell University Department of Electrical and Computer Engineering, 2017-2018.

4. "NSF 10-515: Cyber-Physical Systems," National Science Foundation, Dec. 2009.

Recommended Reading

The following works provide good framing and understanding for the ideas contained in this thesis.

Make it Stick—The Science of Successful Learning

Peter C. Brown, Henry L. Roediger III, Mark A. McDaniel

Make it Stick is a great summary of the research into the cognitive science behind learning and the efforts in understanding robust learning. It's quite comprehensive, and uses a number of stories to ground abstract concepts in a way that makes them concrete and actionable. It debunks a number of illusions behind how we learn which is incredibly informative to the teacher looking to understand why certain techniques 'work' and others don't. It's also very helpful for the student looking to improve their learning techniques.

—So Can you Build One?

Crispin Mount Miller

Crispin's thesis documents methods we can use to understand and evaluate student learning through detailed interaction and observation. His work is a very candid exploration of teaching as it is, especially at a time where little literature on 'how we learn' had been published. His is one of the first, comprehensive efforts to understand how our students learn and apply it to engineering teaching that I've come across. His narratives on learning and philosophy on teaching are also a very valuable read.

Chapter 1

Defining What to Teach

Many who work in the field would agree that the semantics of the definition of a robot[2] are un-important. The key product of the field is more of a combination of ideas and methods that we use to integrate the vague concepts of *intelligence* and *mechatronics* for some practical function [3]. Even more would agree that this integration causes some challenges when it comes to defining what to teach.

It's not the *complexity* of these systems that's the problem so much as it is their *interdisciplinary* nature. Historically, we're fairly ok with dealing with complex systems if you consider complexity to be sheer number of parts. A quick example of this would be an old-timey steam locomotive which, after a four-year degree, a mechanical engineering student should have all the necessary information to engineer each individual component (even though expecting them to do all of it would be largely impractical). In complex systems it's the volume of the task at hand that causes problems.

The challenge in terms of teaching in interdisciplinary fields of engineering has always been figuring out where to provide *depth* and where to provide *breadth* to the student [4]. The desire is often to avoid the *throw-it-over-the-wall* approach to engineering (depicted in Figure 1) [3]. In this way and a few others, the challenges we have in teaching robotics today are not unlike the challenges of teaching mechatronics in the 1990s-2000s [3]. The simple gluing together of mechanics and electronic lacks a cohesive theme, just as the simple gluing together of intelligence and mechatronics to build a robot [3].

> *"Often the mechanical engineers design a machine; when finished they 'throw it over the wall' to the electrical/electronic engineers to design and fit the control system and they, in turn, 'throw it over the wall' to the software engineers to write the control programs. Mechatronics is a trans-disciplinary approach, based on open communication systems and concurrent practices." — pg. 1* [3]

If this is true, then in terms of both engineering the system, and *teaching* the student to engineer the system what we're looking at is the interface or overlap between intelligence and mechatronics. In my own research, I could find relatively little literature that documented the importance or contents of this interface, and even less in terms of how to teach *at* that interface.

That brings me to this chapter which has a couple purposes.

The first is to give us a lens through which to view this work. I don't mean to detract from what I said earlier, I don't think trying to constrain robotics to a definition is useful, and I also don't think it's possible in a practical way. I do think, however, when it comes to discussing what to teach it's important for us to start with the same view of the system so that the following conversation can be more fruitful and less arbitrary.

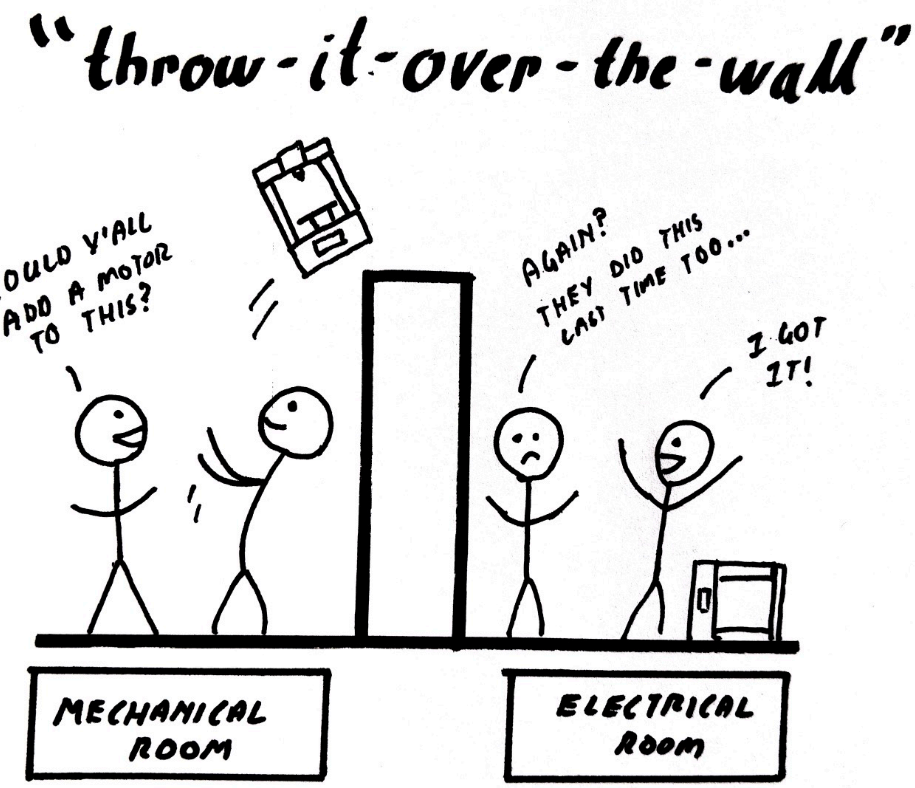
---

2 The term robot wasn't coined by engineers or scientists working in the discipline; it was science-fiction that actually proposed the term, specifically Isaac Asimov's 1942 short story 'Runaround' [1]. In the same work, the three 'laws-of-robotics' also appeared. These laws being: "(1) a robot may not injure a human being or, through inaction, allow a human being to come to harm; (2) a robot must obey the orders given it by human beings except where such orders would conflict with the First Law; (3) a robot must protect its own existence as long as such protection does not conflict with the First or Second Law."

I include this because it's somewhat clarifying to understand the history behind the terminology that defines robotics, especially to the extent that we've never truly been able to define a robot completely, and the lack of a definition beyond the images of our imaginations is not necessarily new. However, I don't wish to give Asimov more credit than he has already been given on this front. Asimov is far too often praised for his 'vision' in the field of science fiction, and his treatment of women is far too often neglected to be mentioned. I draw your attention specifically to this correspondence in Nature [2].

The second, while I'm painting this picture I mentioned above, I spend some portion of that time documenting what happens at the interface between intelligence and mechatronics. I do this for two reasons (1) it's received less attention than I believe it deserves in literature, (2) what goes on in there will become a considerable focus of our teaching efforts.

The third, is to document some of my process in determining the curriculum presented in Appendix B and the syllabus of the core classes presented in Appendix A. I spend a considerable time discussing what sub-sections of the curriculum I chose to focus on in terms of asking *how* we should be teaching. This sets up the rest of the work presented in this thesis.

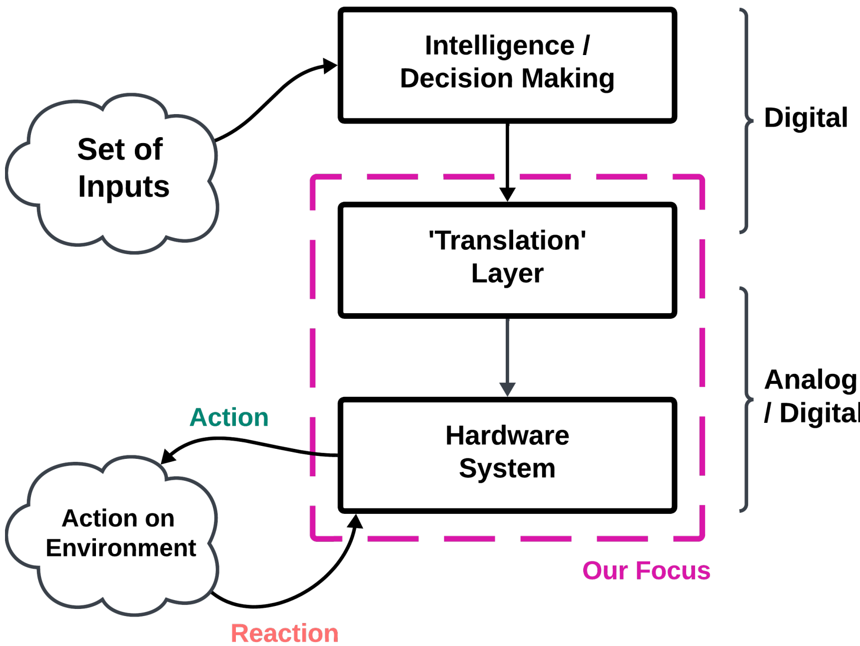


F.1.1—THE 'THROW-IT-OVER-THE-WALL' APPROACH TO ENGINEERING.

What's in a Robot?

I like to think of a robotics system as having three layers.

The upper layer is what I call *intelligence,* and by intelligence I mean command determination. Its job is to take in information (this could be in the form of senso-perceptive information, the state of the system, a task description of some sort) and then use that to make a decision about what the 'thing' should 'do' in the real world to achieve, or bring it closer to achieving the desired task. This upper layer exists purely digitally, the information is digital, the processing of that information is digital, and is kind of abstract in that sense. If we were talking about a creature, this layer would be the brain, but it's a little bit erroneous to make such comparisons.



F.1.2—A HIGH-LEVEL CARTOON OF AN INTELLIGENT PHYSICAL SYSTEM (IPS).

On the opposite end of the spectrum is what I call the *hardware* system. This is the 'thing' that exists in and interacts with the real world. It's not necessarily accurate to say the hardware system is purely *analog*. Electromechanical transduction often involves digital signals, code running on microcontrollers, etcetera. But it is a purely physical system and it's job is to apply the command determined by the upper layer to the physical world. In some cases, we can also argue that the role of this layer is to provide some feedback to the upper layer that could be used to determine how effective applying a certain command was, but I would say it depends on the nature of the command and the system.

The layer in the middle is what I call the *translation* layer which contains any interface between the hardware and the software that's needed to make the system function. What it's essentially translating is information, both in the form of signals that need to move from one layer to the other, but also representations of the physical system that the intelligence system needs to make decisions (we can call these models).

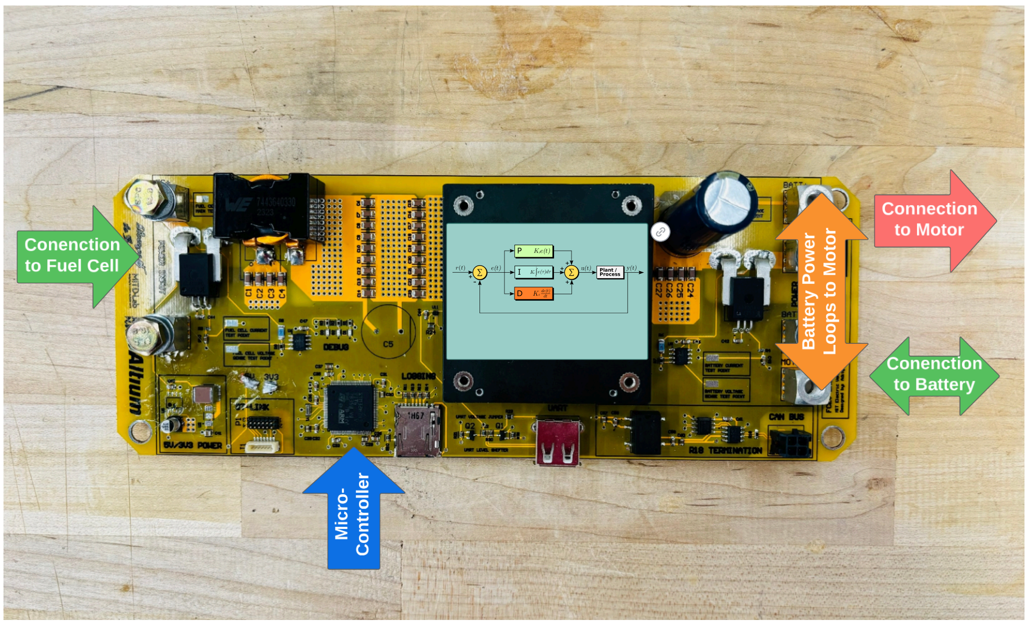
The Energy Management System of a Fuel Cell Vehicle

A relatively simple example of a system that follows this basic structure is a fuel-cell electric vehicle. Hydrogen is stored in compressed gaseous form on-board in a set of tanks. These tanks feed hydrogen to a 'fuel-cell' which combines it with oxygen from the air, and through a series of reduction-oxidation half reactions, and some clever design in the structure of the fuel-cell itself, the chemical reaction of $2H_2 + O_2 \rightarrow H_2O$ produces water and electricity (Figure D.1) [5][6][7].

Ideally, in a vehicle, we'd be able to directly use this energy to power a motor (and any other sub-systems required) to propel the vehicle along. However, fuel-cells exhibit particularly slow power-response times (Figure D.2) [8][9][10].

The solution to this is to add a storage element to the system that acts as a 'filter.' Generally this is in the form of a small battery pack or bank of capacitors. The goal is to have the fuel cell see a 'smooth' power draw while the 'spikes' and 'noise' are handled by the storage element, which is far better at taking that kind of roughness (Figure D.3).

The component responsible for 'splitting' the power between the fuel-cell and storage element is the energy management system (EMS). There are many architectures for this, but the simplest is essentially a large DC-DC converter operating in current-control mode on the output of the fuel cell. According to Kirchhoff's laws whatever current *isn't* being supplied to the motor by the fuel cell, has to be supplied by the storage element.

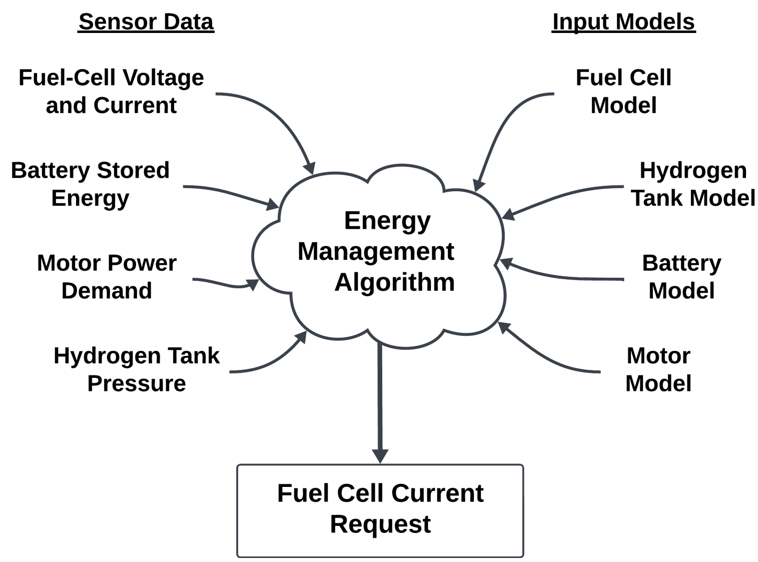


F.1.3—FUEL-CELL VEHICLE DC/DC CONVERTER WITH MICROCONTROLLER FOR ENERGY MANAGEMENT, AND FEEDBACK CONTROLLED POWER CONVERTER (CONSTANT-CURRENT). PID CONTROL DIAGRAM BY ARTURO URQUIZO.

Note that the storage element only has so much capacity and also needs to be re-charged by the fuel-cell during the periods of low-demand. Also note that the peak power of the propulsion system can exceed the peak power the fuel-cell can provide while energy-supply in the storage element lasts (Figure D.4).

On the physical power converter is a microcontroller running an algorithm which determines the set-point of the current-controller at some specified update rate. The algorithm takes in a few critical pieces of information including the current motor power demand, the total amount of energy stored in the battery, and the total amount of energy stored in the hydrogen tank. It has a few objectives. The first objective ensure that the motor is getting the power total power that it is requesting (and that the system is not under-powered if that can be avoided). The second tries to maximize the range of the vehicle by using as little hydrogen from the tank as possible. The third tries to minimize degradation of the fuel cell by minimizing any high-frequency power draw. And the fourth tries to ensure the storage element is always fully charged in-case a power spike comes a long. These objectives all 'fight' each other and the algorithm must balance them based on both current sensor readings, and by storing a history of previous system activity.

The algorithm needs to 'know' a couple things about the physical components of the system in addition to their power draw at any point in time to be able to balance these objectives. For example, it needs to know how large the hydrogen tank is. It needs to know how large the storage element is. It needs to know what the degradation *rate* of the fuel-cell is as a function of input frequency. It also needs models for how the amount of hydrogen in the tank changes with power-draw from the fuel cell and how the amount of energy stored in the storage element changes as a function of power-draw.



F.1.4—THE ENERGY MANAGEMENT SYSTEM REQUIRES A MODEL OF THE KEY COMPONENTS OF THE SYSTEM AND THE OBJECTIVES WE WANT TO BALANCE. THEN IT CAN RUN AN OPTIMIZATION BASED ON THE POWER-DEMAND AT EACH TIME-STEP.

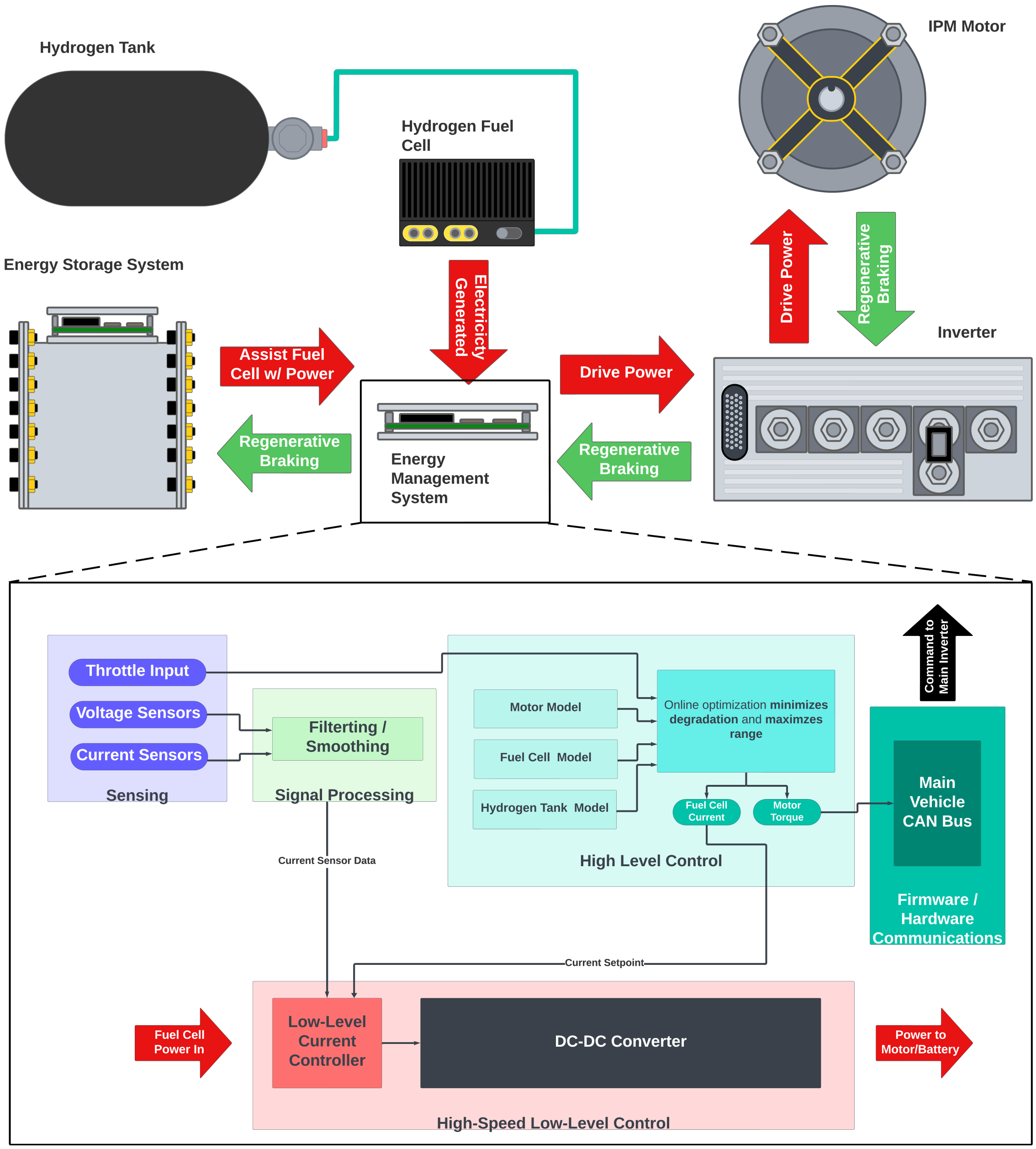
In the case of this system, the *intelligence* is the algorithm itself. Often times in a fuel-cell electric vehicle, a multi-objective optimization program in the form of cost functions and constraints can be run online to determine an optimal output to send to the hardware (or the results can then be simulated).

The *hardware system* is the power converter itself, any sensors that read the voltage and current of the fuel cell and storage element, any sensors that read the power demand of the motor, any hardware filtering needed for those sensors, and the feedback controller (or compensator) that ensures the output current of the fuel cell matches the desired set point.

The *translation layer* in this system includes the models of fuel cell degradation as a function of frequency, the model of the battery or capacitor, the model of how the hydrogen tank pressure drops with power-draw from the fuel cell, and any other models. These models are *generally experimentally determined* and carefully measured before being translated into math and then coded into the algorithm.

Also included in this translation layer is the conversion of the desired current from a floating-point number output by the algorithm to a voltage or whatever other language the converter understands. Similarly included is the translation of hardware-level signals such as current, voltage, and tank pressure to a digital signal that can be used by the algorithm to make decisions.

The diagrams for a number of other systems that follow this same paradigm are included in Appendix D.



F.1.5—SYSTEM-LEVEL DIAGRAM OF A FUEL-CELL VEHICLE. THE ‘INTELLIGENCE’ IN THIS SYSTEM IS THE ENERGY MANAGEMENT ALGORITHM. THE ‘HARDWARE’ INCLUDES THE DC/DC CONVERTER AND LOW-LEVEL CURRENT CONTROLLER (WHICH IS IMPLEMENTED WITH AN OP-AMP). THE ‘TRANSLATION LAYER’ INCLUDES THE FUEL CELL, MOTOR, HYDROGEN TANK, AND BATTERY MODEL, AS WELL AS CONVERSION OF SIGNALS FROM THE ALGORITHM TO THE POWER CONVERTER AND THE SENSORS TO THE ALGORITHM.

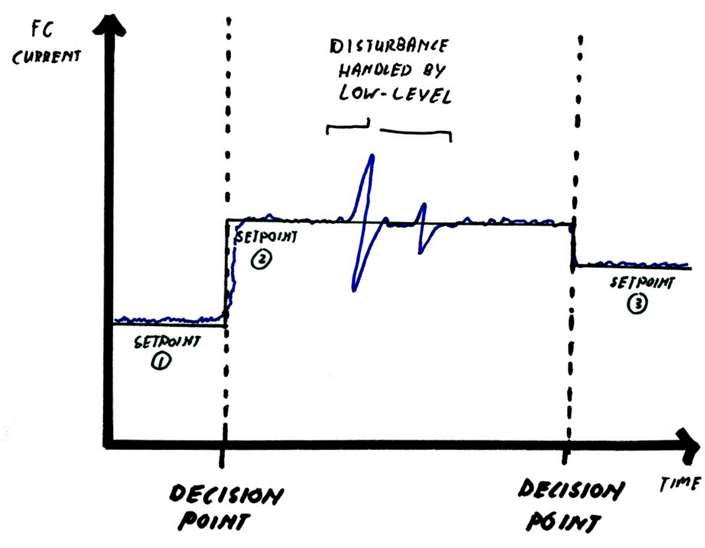
Timing is Critical

In any system with this sort of hierarchical control structure timing is critical. The low-level controller generally operates *far faster* than the high-level intelligence algorithm for a few reasons.

First, because it can. It has less to deal with. The high-level algorithm is often balancing a series of complex objectives based on detailed models. This is computationally intensive and while solve times can be low, they are still *far* longer that what we would consider 'real-time' (100Hz is not real-time, for example). The low-level controller is generally a simple feedback loop controlling a single parameter. In the case of the current controller in the fuel-cell vehicle example above, we could probably get that to run at a bandwidth of 40kHz or more with a high enough switching frequency.

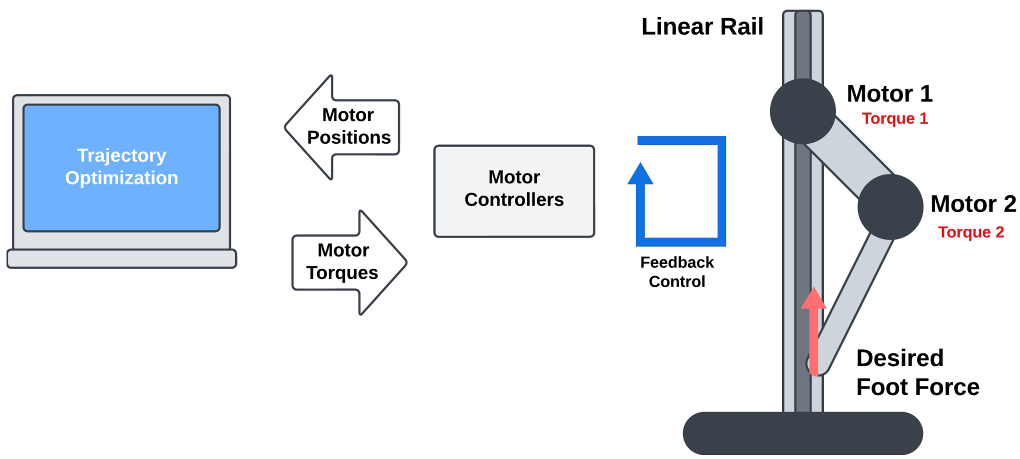
The second reason is because it *has* to, and the reasoning is two-fold.

The simpler of the two has to do with the continuous dynamics of operating a system in the physical world. While the high-level policy is busy making a decision things are *still happening*. The motor is still spinning, power is still being delivered from the fuel-cell and battery. And because the intelligence is somewhat slow (at least at the time of writing this thesis) compared to continuous time, the low-level controller needs to handle any disturbances or issues that happen between the time when the policy made it last decision and the time it makes the next one. If the motor power suddenly spikes before the energy management system has the time to react, the storage element will start to drain and the current controller will try its best to maintain the last set-point until the energy management system slowly starts ramping up the desired fuel-cell power to account for the spike and replenish the storage element.



F.1.6—SINCE THE HIGH-LEVEL OPERATES SLOWLY DUE TO COMPUTATIONAL INTENSITY, THE LOW-LEVEL MUST HANDLE ANY DISTURBANCES IN THE INTERMEDIARY BETWEEN DECISION POINTS. IN THIS CASE, THE LOAD MAY HAVE SUDDENLY CHANGED ON THE POWER SYSTEM BEFORE THE DECISION ALGORITHM HAD TIME TO REACT. INSTEAD, THE LOW-LEVEL CONTROLLER REACTS TO COUNTER THE DISTURBANCE AND MAINTAIN THE CURRENT BEFORE THE HIGH-LEVEL CONTROLLER CAN CHANGE THE SET-POINT.

For the more complex of the two it's important to think about what the high-level controller is actually trying to control. If the dynamics of the low-level controllers are significantly faster than the dynamics of the overall system or 'task' the high-level algorithm is trying to control, we can often neglect the controller's dynamics in modeling and treat it as input *directly* equals output.



F.1.7—A 2DOF JUMPING LEG HAS A COMPUTER THAT SENDS COMMANDS TO ITS MOTOR CONTROLLERS THAT CONTROL THE TORQUE TO EACH OF ITS TWO MOTORS.

A good, but simple example of this comes from motor control in dynamic robotics. And the simplest form of a dynamic robot is a 2 Degree-of-Freedom jumping leg.

Let's take the case that we want to design an algorithm that maximizes the jumping height of the 2DoF leg. How this algorithm is going to approach the problem is by controlling the force at the contact point between the foot and the ground to be purely vertical (and at a maximum based on the system configuration and constraints at any given time). To do this, it'll send two torque commands, one to each motor.

To control the torque of a motor, we need to control the current going into the motor's armature. To control the current going into the armature, we need an H-bridge motor control circuit with fast PWM switching that actively adjusts the voltage. The voltage adjustment is done via a feedback control loop that sets the duty cycle of the H-bridge based on the error between the desired current and the actual current (Figure D.4).

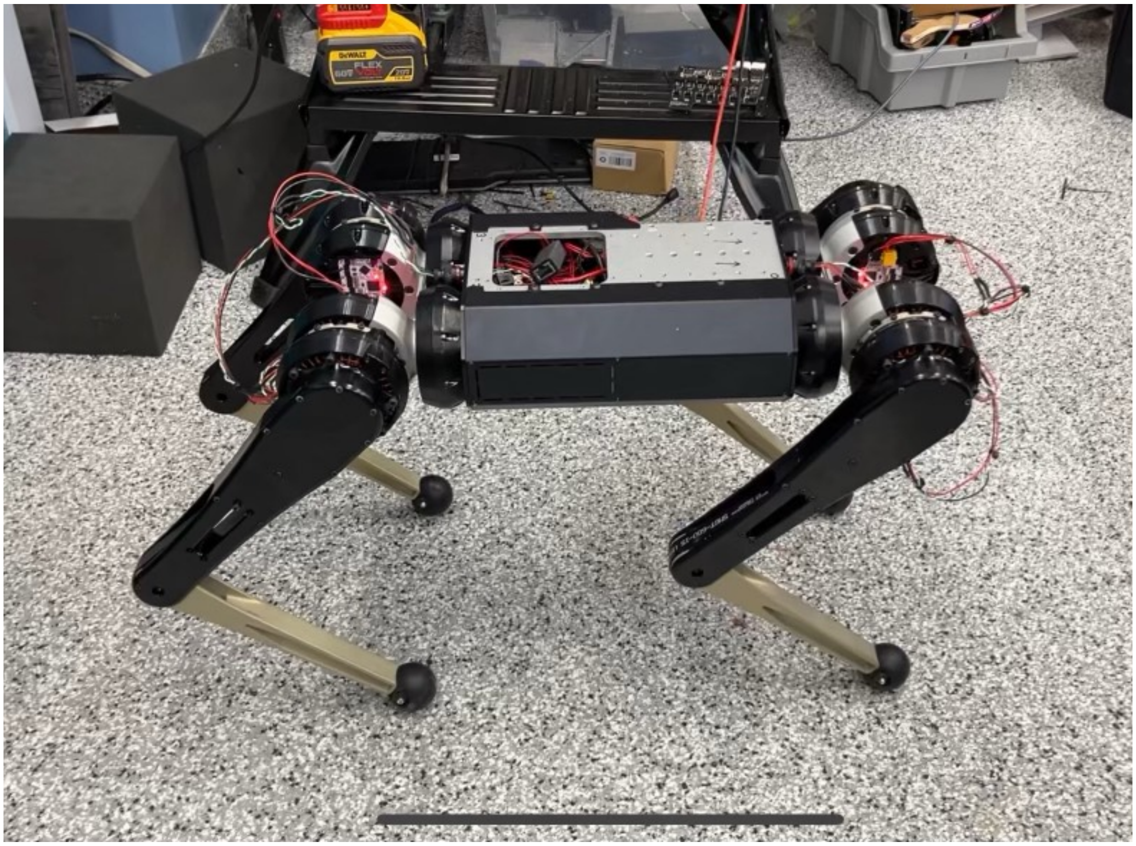
The question is, for the high-level algorithm, do we *need* to model all the fine details of how the controller adjusts voltage to generate current, and the switching in the transistor? Well, if it's slow enough we might need to. For example, if it takes one second for the controller to give us 10Nm when we ask for 10Nm, the dynamics of the controller might change the decision made by the high-level algorithm. However, if the motor supplies 10Nm of torque in 1us after the command is sent, then the dynamics might be quick enough where we don't need to model them. Input simply equals output and the high-level controller can just *think* in terms of torque.

In the case of motor control, the latter is generally the case. The electrical dynamics current in a motor are *far faster* than the mechanical dynamics the torque being applied is trying to control. In this case, we need to program the mechanical dynamics into the algorithm, but don't necessarily need to program in the electrical dynamics in the form of a model. This makes modeling simpler, and the reduced complexity makes the high-level algorithm run faster.

A Dynamic Legged Robot

I want to take a moment, before we move onto the topic of teaching, to sort of outline *just how complicated* this can get by describing the system of the MIT Mini Cheetah 'Pro.'

The Mini Cheetah Pro is an upgraded version of the original MIT Mini Cheetah platform designed by Ben Katz [11]. It's a dynamic quadrupedal robot that was designed as a research platform for controls, and it's primary functions are to walk, run, jump, stand up, sit down, and do a backflip [12].



F.1.9—THE MIT MINI CHEETAH 'PRO' (PIC BY RONAK ROK).

The physical layout of the hardware system can be divided into a communications stack and power stack. In the communications stack, the central computer (which also interfaces with any USB sensors) communicates with microcontrollers on an 'interface board' called the SPIne board over Serial Peripheral Interface (SPI). Each of these microcontrollers then sends commands and receives data back from the motors over CAN Bus. Each microcontroller has two CAN Bus interfaces and each interface communicates with three motors.

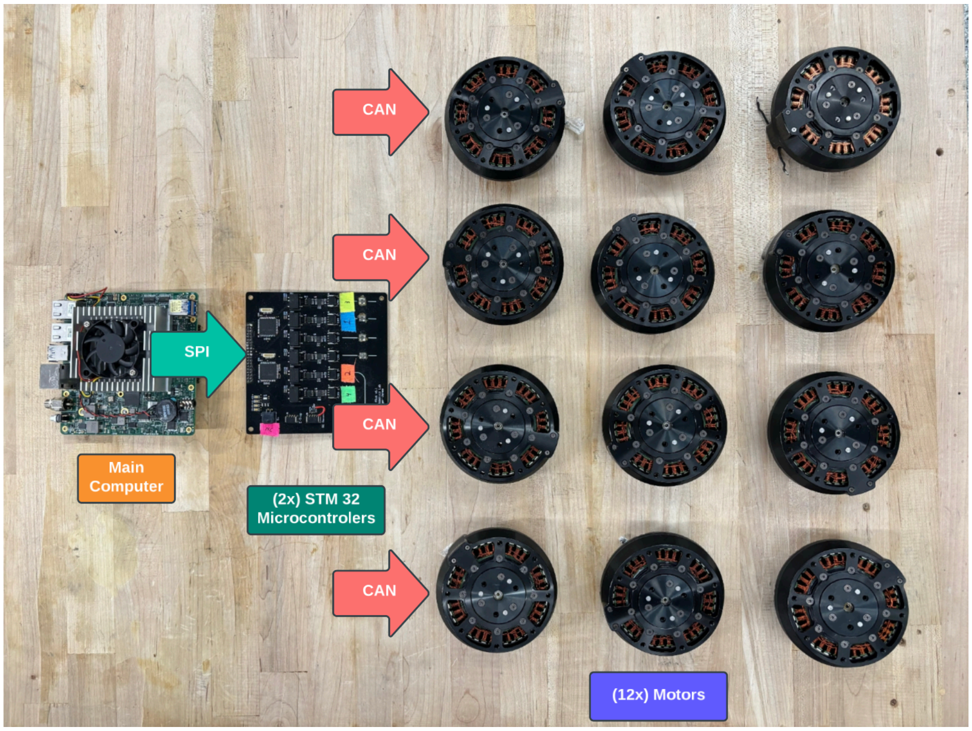
You can put up to 32 nodes on a CAN bus system and the Mini Cheetah only has 12 motors. So theoretically, if data transfer was instantaneous, we would only need one CAN Bus to communicate with all 12 motors. However, communications is *not* instantaneous, and with the CAN Bus operating at 1MBit/s, we can calculate the data transfer rate by dividing the number of bits to transfer with the data-rate.

The motors in our system take a command of 16 hexadecimal characters. That's 16 times 4 bits per hex is 64 bits. That means each data transfer to and from a motor takes 128us (64us of data going to the motor, 64us of data coming back). For 12 motors, that's 1.536ms that it would take to update all the motors if we used a single CAN Bus.

$$t_{transfer} = \frac{n_{bits,data}}{r_{transfer}}$$

So is that too slow? Well we need to look at the rest of the system to figure that out. That's the digital layout of the system.

This is a bit of a simplification, but in the digital layout, sensors including the IMU, the accelerometer, and the motors themselves send data to a Model-Predictive Controller (MPC) running at 100Hz. The whole body controller's job is to take the current dynamics of the system and roll it forward over a time horizon to make some estimates about where the system will be after a few seconds. It then uses that to determine the command to execute right now to get the system to go where we want it to.

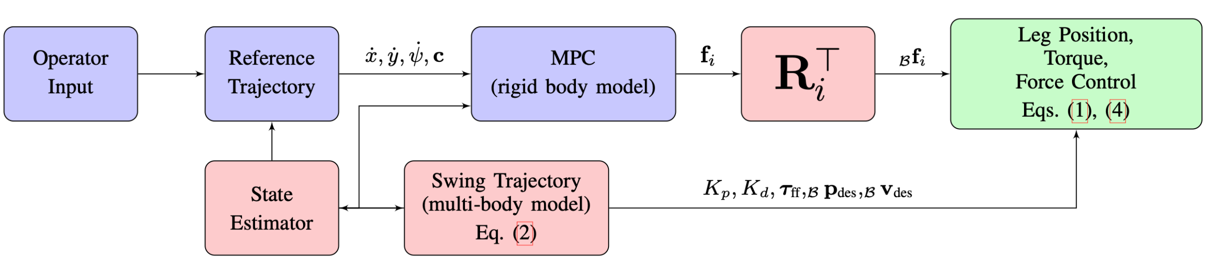


F.1.10—COMMUNICATIONS ARCHITECTURE OF THE MINI CHEETAH.

In the example of the Mini Cheetah, the MPC takes in a command from a radio-controller operated by a user. If the user pushes forward on the stick, that prescribes a forward walking 'speed' for the robot. The MPC reads the motion of the body, and knowing which legs are in contact with the ground determines the force each leg has to apply to get the body to go forward at the specified speed.

The MPC sends commands to a whole-body controller (WBC) which operates at 500Hz [13]. The WBC takes the force commands and translates them to torques to be sent to each of the motors. These commands are the sent to each of the motors through the communications interface we described above.

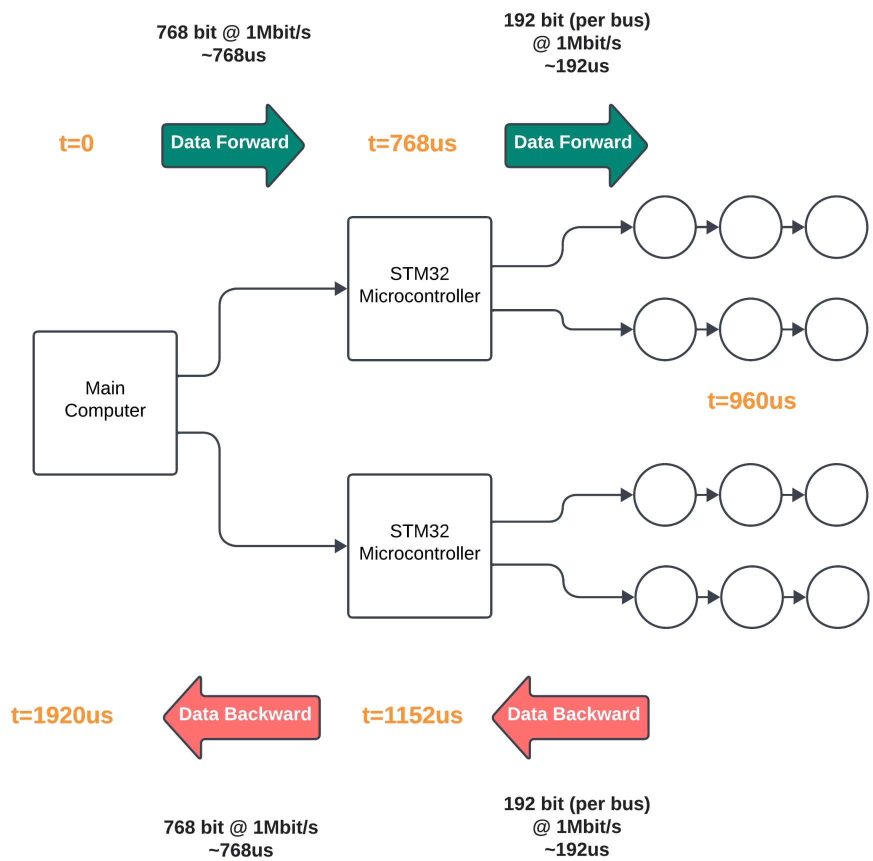
That means we want to update the motor command every 2ms (1/500Hz). So at face-value it seems like the single CAN Bus might work.



F.1.11—CONTROL ARCHITECTURE OF THE MINI CHEETAH PRO FROM [14].

But that's not all of the delays in the system. We still need to get the command from the computer to the microcontroller and *then* get the command from the microcontroller to the motors over the CAN Bus. The SPI interface handles that and operates at a baud rate of 1MBits/s as well. That means it takes 1.536ms to transfer data between the computer and the SPIne board even *before* we have the chance to use the CAN Bus. So now we have 436us to transfer the command. Guess where that leaves us, 436us/128us is 3.4, so we can transfer the data for 3 motors over a single CAN Bus in 436us.

We need four CAN Bus systems for 12 motors.



F.1.12—DATA TRANSFER RATES AND PACKET SIZE ALONG THE ROBOT COMMUNICATIONS SYSTEM.

There's more. For the MPC to work it needs information on the state of the robot (the positions of the motors, the orientation and velocity of the body, the torques of the motors). And it needs to figure this all out using nothing but a few current sensors on the motors and an IMU. Luckily it has some help. Because we know the torque constant of the motor, and since the motor is measuring its current, it can report torques directly to the state estimator. The motor controller also knows the position and velocity of the motor from the encoder connected to it so these can also be directly reported back to the state estimator. The IMU is the problem. The IMU provides 6DoF accelerations $(x, y, z, \theta_x, \theta_y, \theta_z)$ which need to be integrated into velocities and orientations of the body. There's noise on these measurements so it takes a good amount of engineering work to ensure these are accurate. In addition, the torques reported by the motors when combined with the positions need to be converted into forces on the body, which are then combined to determine the net acceleration of the body in all 6DoF. This all wrapped together is called state estimation, we don't know the state of the robot we can only estimate it.

When it comes to designing the MPC controller. This is done in a simulation environment and this environment needs an accurate model of the robot. This includes geometry, the masses and inertias of the links, and then sometimes non-idealities such as belt-friction and anything else that could affect the dynamic performance of the system. All these things are deeply difficult to measure (especially the inertias and friction). So one method for this is to estimate them using what's called system-identification. There's many ways to do this, but one way is to send somewhat random sinusoids of torques into the motor at increasing frequencies and measure the positions and velocities of the system, then linear regression can be used to fit the parameters of a dynamic model formulated using Lagrangian dynamics (Figure D.5) [15].

The list of things we have to do to make this system work goes on, and there's a number of good papers and thesis that have documented these in detail in case you're more interested [11][16]. What I've listed here are some of the major challenges that can get wrapped up into the three layers we identified earlier.

The *intelligence* would be things like the MPC that take in the model of the system, the state, and output the series of control actions to the whole body controller which converts them to torques.

The *hardware* or mechatronics systems consists of the low-level feedback control loops running on the motor controllers operating a 1kHz closed-loop current controller and 40kHz switching frequency, the power electronics system, the mechanical system of the robot, any sensors, the computer, and various other components.

The *translation layer* in this system is more detailed. It involves the whole-body controller, and the communications system that sends the torque commands from the whole-body controller to the motor. It also includes the state estimator, the model that results from system identification, and model of the motor's performance limits. It could also include things like any software engineering that needs to be done to get the algorithm to run quickly on the computer system (threading, GPU optimization, etcetera). In some cases, the translation layer is everything that's 'behind' the scenes. It plays a large role in determining the performance of the overall system, yet it is mostly hidden.

Notes on Intelligent Systems

We detailed a number of systems that follow this intelligence-translation layer-mechatronics system architecture in Appendix D. These include the system of a CNC machine, a continuous glucose monitor, a humanoid robot, and additional diagrams for the fuel-cell vehicle. These represent the areas of manufacturing, biomedical devices, and consumer products with the two system we've already presented representing energy systems and robotics. All of these are key areas of impact for the ideas that make up robotic systems. It's not worth taking the time to explain every one in the body of this thesis because that would get quite repetitive.

Some key things to point out. In both systems we've described so far the intelligence system makes decisions actively. There's a target, the system changes state, the intelligence updates its decision to hit that target. This doesn't necessarily have to be the case. For the CNC machine in Appendix D, the intelligence is the CAM done on the computer that outputs G-Code that's run feed-forward on the CNC machine. The intelligence has no way of knowing wether the commands it generated worked or not, it has no feedback. But it doesn't matter. The hardware of the CNC is so good at its job (at least within the bounds of its operation) it can remain disconnected. There are many systems where the disconnect is OK, there's a number of systems where the disconnect is not. The latter, as you might imagine, are far harder to engineer.

Motivations for a Curriculum

There were three primary motivations for exploring building a curriculum in robotic systems. The first was from our own observations in teaching, the second and third were observations about current curriculum options.

Observations from 2.74 Bio-Inspired Robotics

Our initial motivations in exploring the topics and methods for teaching robotics came from our observations from teaching *2.74 Bio-Inspired Robotics*. The course focuses on teaching the underlying concepts behind designing and building dynamic robots similar to the Mini Cheetah platform described earlier. Students develop both the hardware design and control system to explore a research topic of their choice which can range from building a robot that can flip pancakes to understanding the biomechanics of hula-hooping. The course combines the concepts of dynamic simulation, electro-mechanical design, modeling, and controls to a project that must work in the real world (as opposed to simply simulation) with measurable results. The course description is included below:

> *"Interdisciplinary approach to bio-inspired design, with emphasis on principle extraction applicable to various robotics research fields, such as robotics, prosthetics, and human assistive technologies. Focuses on three main components: biomechanics, numerical techniques that allow multi-body dynamics simulation with environmental interaction and optimization, and basic robotics techniques and implementation skills. Students integrate the components into a final robotic system project of their choosing through which they must demonstrate their understanding of dynamics and control and test hypothesized design principles. Students taking graduate version complete additional assignments. Enrollment may be limited due to laboratory capacity."* — [17]

The course has both lecture and lab components that are designed to cover different topics. Lecture focuses mostly on the modeling, simulation, and intelligence components of a dynamic robotic system. Lab focuses on the hardware.

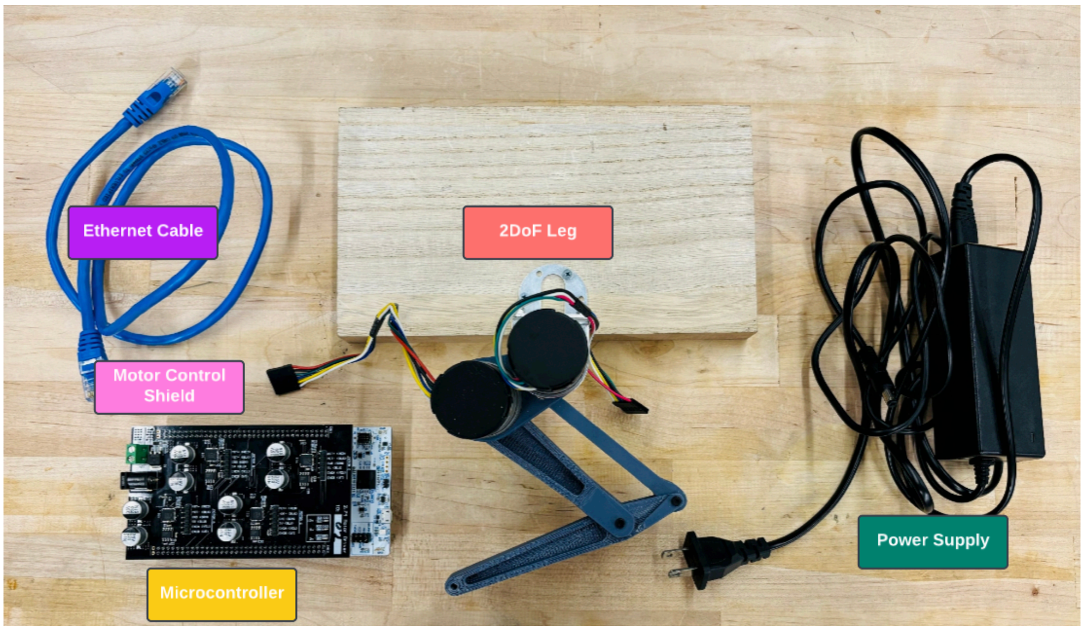
Lecture Content	Lab Content
Week 1 ▪ Introduction to bio-inspired design ▪ Generalized forces and coordinates ▪ Jacobian for dynamic control **Week 2** ▪ Motor Control ▪ Introduction to Simulation (Numerical Integration) **Week 3** ▪ Impedance Control ▪ Lagrangian Mechanics **Week 4-5** ▪ Impedance Control (Operation-Space) ▪ Contact Simulation **Week 6-7** ▪ Muscles ▪ Optimization **Week 8** ▪ Energetics ▪ Actuation **Week 9-14** ▪ Miscellaneous Topics	**Lab 1** ▪ Introduction to microcontrollers ▪ Position control of a motor (PID) **Lab 2** ▪ System ID of a motor ▪ Current control of a DC motor ▪ Feedforward control compensation **Lab 3** ▪ Impedance control of a motor **Lab 4** ▪ 2DoF leg impedance controller ▪ 2DoF trajectory following with impedance control **Lab 5** ▪ 2DoF trajectory tracking in operational space ▪ Measuring your ground-reaction force while walking and running

T.1.1—LECTURE AND LAB CONTENT FOR THE 2.74 BIO-INSPIRED ROBOTICS COURSE.

There are some key course design decisions that make teaching a student how to build an entire robotics system tractable.

First, many of the exercises and concepts are presented in simplest possible form. For example, system identification of a motor is performed in Lab 2, but only to the extent of measuring the armature resistance and the rotor inertia. Students are asked to formulate an optimization problem to send a leg to a target jump height but the leg has a single degree of freedom (one motor). I don't say any of this in a negative way.

First, a lot of engineering work was done by TAs under the hood. Much of the simulation framework is provided, students are mostly expected to 'fill in' models such as the dynamic models of their system and the contact models. A lot of the hardware engineering is done as well. Students are given the design of a 2DoF leg, a shield for a NUCLEO board that drives four motors and reads from four encoders. Students are also given the code to interface the NUCLEO board with Matlab to run their trajectory optimization on hardware. Microcontroller clock speed configuration and PWM timing optimization for maximum motor control bandwidth are *all* done by the TAs in advance. Even programming of the impedance controller is done by the TAs, students simply need to put the $\tau = K_p e_\theta + K_d e_\omega + \tau_{ff}$ formula into the right location before hitting upload. Minimal programming experience in embedded systems or Matlab is required for the course.



F.1.13—IMAGE OF THE BASIC 2.74 HARDWARE LAB KIT.

I took the class in the fall of 2021 and happened to be a TA for the class in the fall of 2022 which enabled the ability for me to observe a lot of student learning and have a number of discussions with course staff regarding what the students were and weren't learning. I'm happy to report they learned a lot.

There are some things the students had trouble with. The first was embedded systems programming and the operating limits of microcontrollers. A number of students had trouble with C-programming and flashing when starting out. Students also seemed to have trouble interfacing the microcontroller with new sensors and devices, and struggled to pinpoint the issue when the system didn't work. Lab conceptual questions also indicated some struggle in terms of understanding the 'inner workings' of a microcontroller.

None of this was entirely surprising to us. Embedded systems wasn't a prerequisite requirement for the course.

The second observation came from noticing how students approached the final project. The final project for the course is done in teams of 2-4 people. In all team-based final projects students are expected to divide the work but it was interesting to see how the students chose to do it. Students tended to divide the project amongst the interface lines of the system. What I mean by this is one person generally worked on the electronics, one generally worked on the simulation, and one generally worked on the mechanical design. Students also tended to gravitate to an area where they felt the most comfortable (those with the most experience in mechanical design did the mechanical design).

The engineering of interdisciplinary systems is always and will always be a team sport. I don't think that's a bad thing. But this approach to the final project led me to wonder if we were trying to do too much in one class. Were students only learning *parts* of the course content because there was too much content and then gravitating towards the content they were familiar with in the final project? I still haven't come to any conclusions on that front. I think much more rigorous experimentation would need to be performed to figure that out.

Accessibility of Degree Options

To clarify something outright, it's currently completely possible to achieve a comprehensive education in mechatronics at MIT. There are many who have, and I expect some criticism against this work will arise from that fact. This is not a problem, we all have differing opinions on education and sharing them allows for the discourse required to build better learning environments. I think it's important to focus on the concept of accessibility: **a mechatronics degree at MIT is not accessible to all students who attend the institute depending on their technical background, and social circles they have access to.** While my evidence of this statement is largely anecdotal, it is based in detailed observation of how our students navigate their degree plans and roadmaps. Being immersed in student life at MIT for some time, and having gone through a number of these challenges myself has led me to a lot of these 'observations' but they are additionally compiled from the larger community including professors, undergraduate, and graduate students.

Starting at a high level, I've observed that students tend to have difficulty in selecting a major when interested in pursuing mechatronics, and that once a major has been selected tend to have difficulty finding the coverage they desired within their choice of major. A common complaint amongst the undergraduate population is that a cohesive overlap of mechanical design and mechatronics does not exist as a clear degree path.

Solutions to this are wide-ranging, our students are quite creative. Some opt for the 'petition' route replacing classes from external departments with core requirements in their major. Some double-major which leads to an significantly increased course-load if structured improperly (though this can be tractable with intentional selection and petitioning of classes to count towards both majors). A majority of our students, in an effort to fit more advanced classes in their degrees, will 'skip' pre-requisites and request instructor permission to study advanced topics. In certain classes, permission is regularly given and in others the over-head of verifying that the student actually fulfills pre-requisites is too much on top of the effort already required to teach the course. I don't pretend to come to any conclusions on wether this is beneficial or not. It is simply a thing that is worth considering.

More comprehensively, it would be worth developing a rigorous survey to explore topics including the difficulties students feel when selecting a major and where those feelings originate from. Additionally, it would be worth cataloging courses that are often petitioned for various degree requirements as that could indicate overlaps that should exist but may not. That's outside the scope of this work but would be a logical next step.

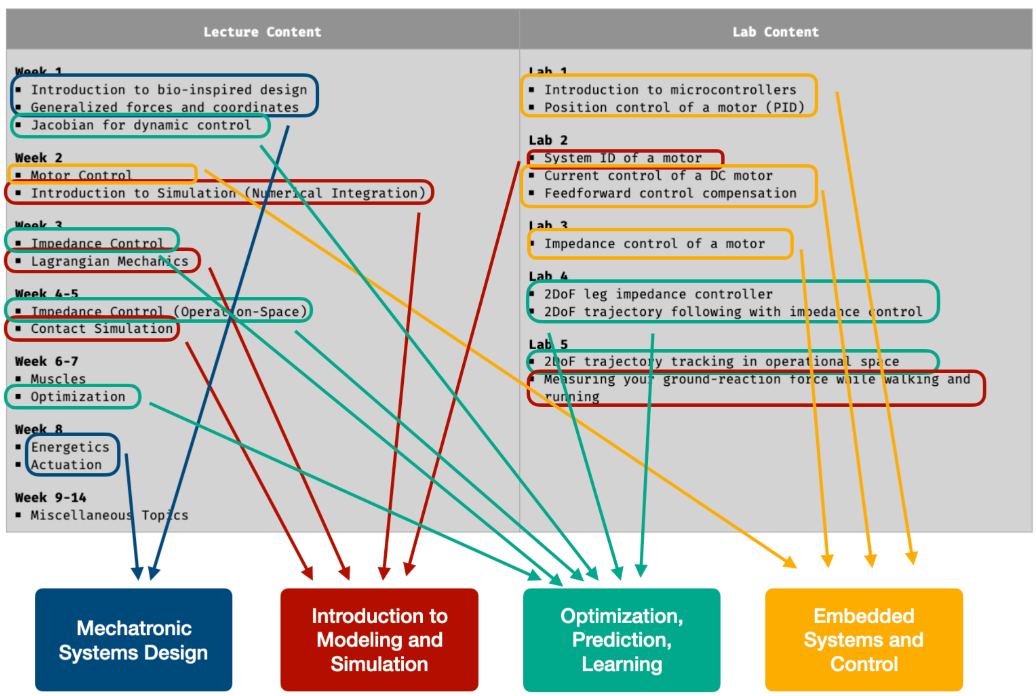
As a preliminary list of choices our students are offered from the vending machine of degrees, the most popular in the area of 'mechatronics' seems to be the dilemma between Mechanical Engineering with a concentration in EECS (2A-6), or Electrical Engineering and Computer Science (6-2, which as of 2024 fall will now be 6-5).[3] Aeronautics and Astronautics has a flexible major (16-ENG) that many of our students choose to use to study control and decision making for flight vehicle systems but the specialization in aerospace makes this option a little bit less universal (see Appendix D for degree charts) [18][19][20][21].

These degrees provide a level of flexibility that is useful with a clear vision and access to the right advice. That is if you *know* what you don't know, *understand* what you wish to learn, and you can acquire the right advice

[3] MIT EECS 2024 Curriculum Changes: https://www.eecs.mit.edu/academics/undergraduate-programs/curriculum/faq-about-fall-2024-changes/

through the social circles you have access to it may be straight-forward to structure your own degree. But the counter-example is also true that if a student lacks access to any one of these things, it may be difficult. Other examples: the student who had access to programs such as FIRST robotics in high school likely has additional awareness of the field of mechatronics and a more informed list of what they might wish to learn than the student who did not; the student who was able to apply transfer-credit has more space in their degree to take additional classes without sacrificing sleep, stress, and exploration of extracurriculars. This is where we bring up the discussion of accessibility.

If I'm honest, I'm not sure it's worth advocating for the creation of a new degree program. That sounds like a lot of overhead but I wouldn't know. Another options is to provide a scaffold. Create a visual and informative guide that helps students structure their degrees. This is the route I chose as it's easy to update as the field and offered classes change, it requires minimal alteration to current degree structures, and it's potential for 'usefulness' is high. The work we present in this section and Appendix B is preliminary to that. A representation for what this might look is '*What Every Electrical Engineering Student Must Know*' written by Ali Alqaraghuli which is publicly available through the University of Michigan Free ECE Textbook initiative (FET) [22].



F.1.14—SPLITTING 2.74 INTO FOUR 'CORE' CLASSES.

Defining Core Courses

I gave myself the unconstrained design exercise of asking: *if we could organize an entire degree with no constraints, what would we teach?*

It start with number of hypothetical conversations with 2.74 course staff members on asking: *what more could we do this class if students came in knowing "_"?* ("_" in this case being some arbitrary topic). A number of discussions arose from this (I'm paraphrasing).

"Well, if they knew brushed motor control coming in, like torque control, and they knew it well then we could maybe teach them brushless motor control, that could be cool."

"It would simplify teaching if they had increased experience with embedded systems, a class like 6.08[4] before this could help."

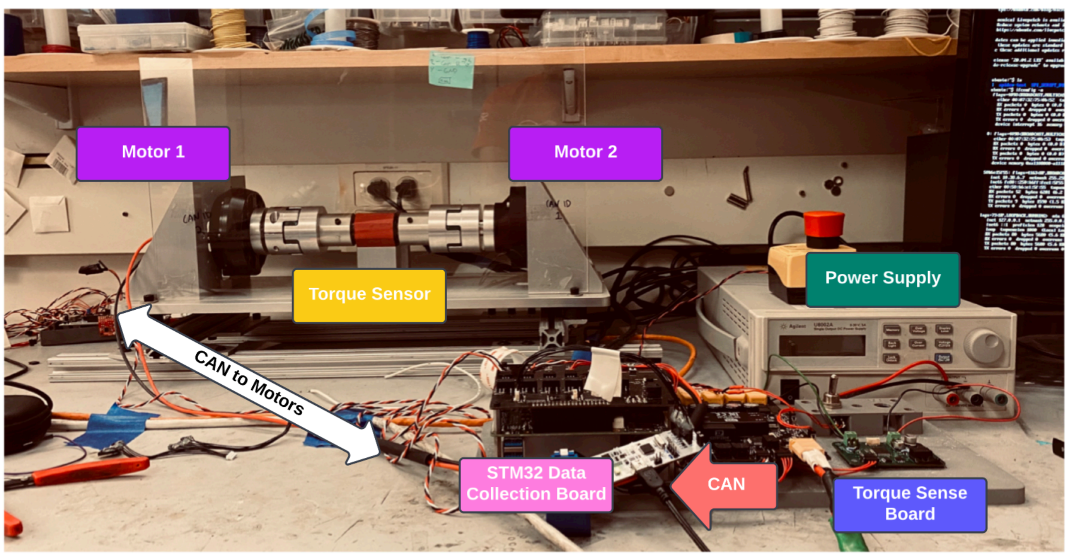
"They only learn optimization, we're doing a lot of reinforcement learning these days. Not sure if that's worth teaching but it could be interesting."

This gave me an idea, if we take the view of an intelligent system presented in the previous section, we could approach teaching by defining three courses. One on *intelligence*, one on *mechatronics,* and one that teaches this *translation layer.*

A course on the translation layer, however, didn't seem to make much sense. There's far too much in the interface between intelligence and mechatronics and we'd need to pick something to focus on. This turned out to be modeling and simulation, and embedded systems.

Embedded Systems

On the side of embedded systems, we looked to history. It's largely argued that integrated circuit[5] was the component that *brought mechanical engineering and electronics together,* by packaging electronics in such a way that it could be tightly and actively integrated with the mechanical system (rather than as an after-thought) [25][26][27]. Later, the microcontroller did the *same* thing for 'programmatic ability' and electrical systems (and therefore mechanical systems from a mechatronics perspective).[6] In the case of any hierarchical control system the embedded engineering is the critical link between the high-level and low-level controllers. Professor Edward A. Lee of Berkeley University described embedded systems as *"having the explicit role of enabling interaction with the physical world"* [28][29]. Embedded systems seemed to be a consistent theme in our personal work, as well as a key requirement for the systems presented in Appendix D.



F.1.15—TORQUE CONVERTER BOARD SETUP FOR A MOTOR DYNAMOMETER.

[4] 6.08 Introduction to EECS via Embedded Systems, was an MIT course focused on being an introduction to embedded systems using IoT devices. It was not as intensive as a course such as 6.115 Microcomputer Project Lab which was designed to dive into the deep details of how microcontrollers work. The course was deprecated in the MIT 2022 curriculum transition [23][24].

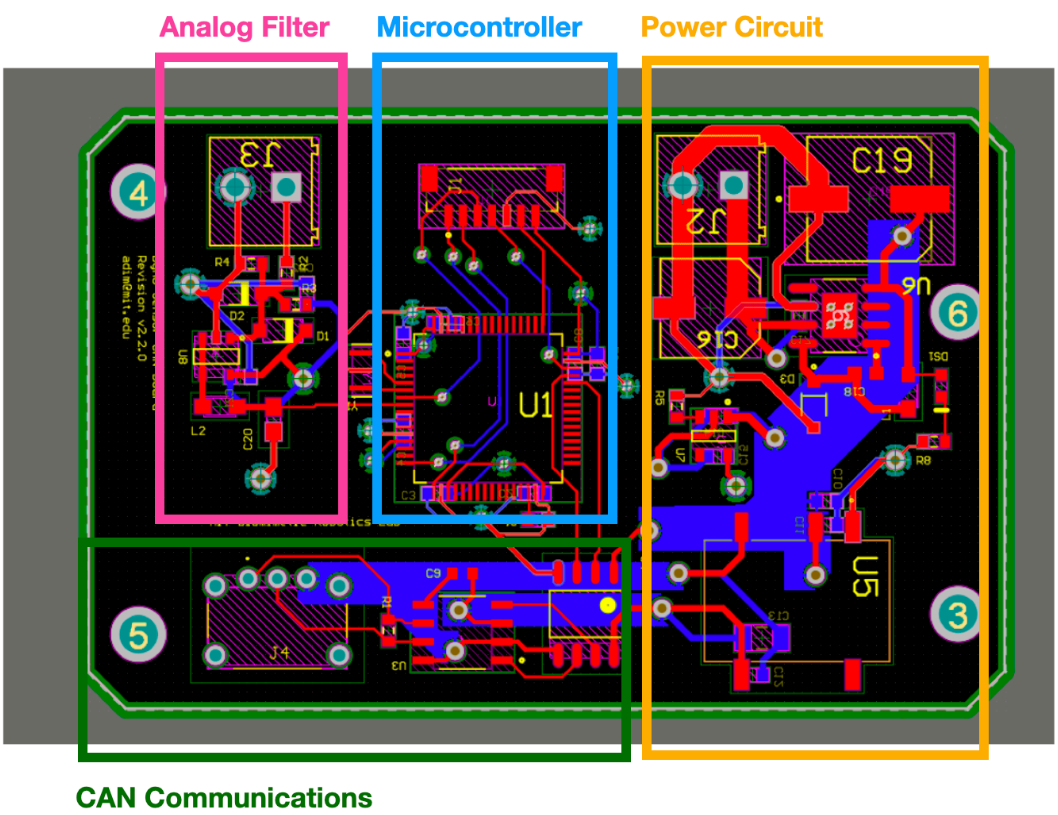
[5] The invention of which is largely credited to Jack Kibly of Texas Instruments. But Robert Noyce of Fairchild Semiconductor as well as Kurt Lehovec of Sprague Electric Company also had a hand in developing critical components of the IC [30][31].

[6] If you're immediately interested in some more historical examples check out the following references on the Apollo Guidance & Navigation (G&N) Computer [32][33] and the story of the first microprocessor [34].

One example from my own work that I think illustrates the key concepts we want to teach our students is a torque-sensing board I had to develop for lab. It came in the form of a Printed Circuit Board (PCB) and its primary function was to read from a torque sensor (the interface for which cost far too much for us to reasonable consider buying it, and even then didn't plug into the computer we were using; not an uncommon problem in hardware). The torque sensor was to be used on a motor dynamometer which we were using to build torque speed curves and efficiency maps [35].

The torque sensor outputted a +/-10V analog signal, that we had to read with the microcontroller which only takes a 0-3V3 analog input (so some level-shifting design would have to be done here). We also wanted to time-sync this data with the data coming back from the motors. To communicate with the motors, we were actually using the same stack-up that we described for the Mini Cheetah in the previous section. We were doing this mostly for ease, but that did mean if we could send the torque data to the SPIne board, time-syncing would be easier. That meant connecting both the microcontrollers over an CAN Bus slot.

The level-shifter itself needed power (24V that would split into the +/-12V rails of the op-amp, 20V was cutting it close and we wanted to use the full input voltage range). The 24V needed to be dropped to 5V to power the CAN transducer and 3V3 to power the microcontroller.



F.1.16—DIAGRAM OF KEY COMPONENTS FOR THE TORQUE CONVERTER BOARD.

The reason I like this board as an example is all the circuits and components are *super simple*, the code is also really simple, but it has many of the major intersections that are important to teach. It has analog filtration, it has programming microcontrollers with input and communications, and it has power conversion.

It's also a good example of how teaching embedded systems is useful past the field of robotics. Here was a simple experimental setup that benefited greatly, and saved us a lot of time any money because we had some embedded electronics experience. This is a skill we wanted to impart onto our students, enabling them to quickly design and implement solutions using microcontrollers to problem, starting with simple ones of course.

A few additional things I would add in terms of topics we want to teach would be low-level feedback controls and potentially some awareness of sensing and signal processing. There are good classes on this [36][37][38] but I think they could benefit from a hands-on introduction alongside some initial theory before proceeding to advanced courses.

Modeling and Simulation

Modeling is another topic that goes beyond the scope of robotics to other fields. But in this case we're a little more specific in thinking about the role of the mechatronics engineer in building intelligent systems.

Some of the inspiration for this, and indeed some detail on how teaching it might look, came from an exercise we gave to an undergraduate researcher on MIT's Electric Vehicle Team for the hydrogen motorcycle project. In this case, we were interested in developing a simulator that could predict the range of the vehicle in its current state when driving on a dynamometer platform with the given battery, fuel cell, and hydrogen tank. This initial simulation development sprint would allow us to size components accurately for future designs if the output matched the hardware experiments on the vehicle dynamometer well.

The challenge in this exercise wasn't really the simulation framework, we used Matlab Simulink so it was really more about connecting blocks together and choosing the right solver and time-step. Both of these are *important,* but they weren't the key challenge.

The hardest part was building the models of the system, and more importantly knowing when to *stop* building the models of the system.

One example that was particularly memorable was in trying to model how the pressure drained from the hydrogen tank as a function of output power from the fuel-cell. We started by trying a simple model.

$$PV = nRT$$

(The Ideal Gas Law)

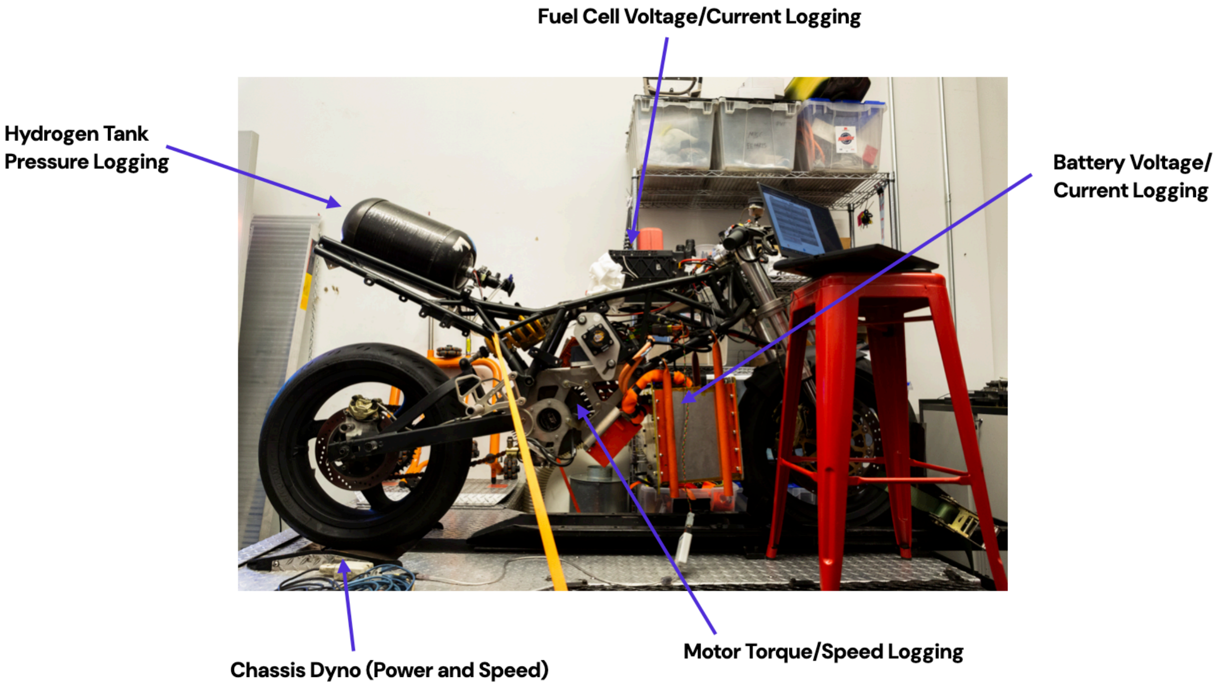
$$n = \frac{1}{M_{H_2} e_{H_2} \eta_{fc}} \int_0^t P_{fc} dt$$

(Where e is the specific-energy)

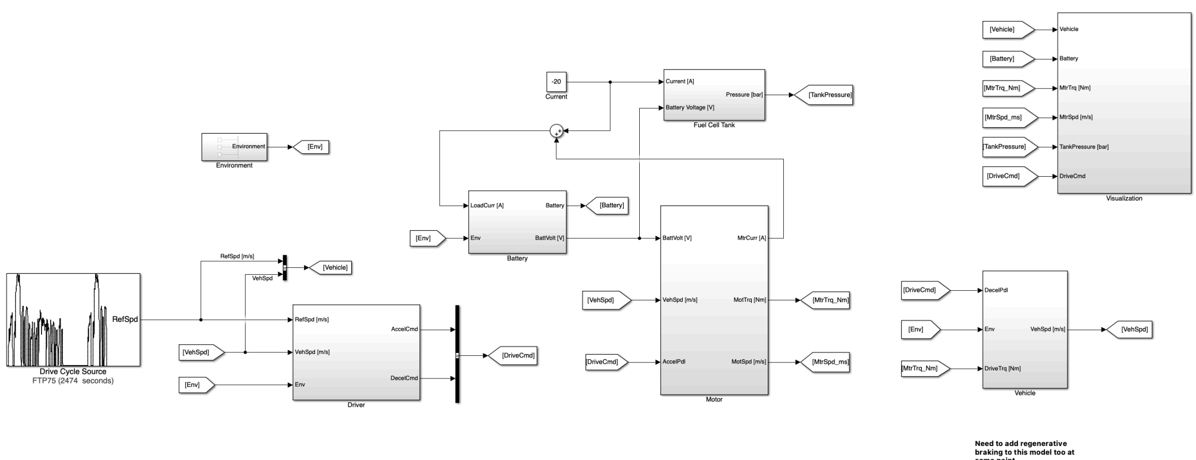
This model would be updated every time-step of the simulation. Essentially, we integrated the electrical power load on the fuel-cell to get the total energy the fuel-cell had outputted until that point. We divided this by the estimated efficiency of the fuel-cell system to get the total energy that had to be chemically stored in the hydrogen, and then divided that number by the specific-energy of hydrogen, and then divided that number by the molar mass. That gave us number of moles that we plugged into the ideal gas law.

Putting this in simulation, we got a curve of pressure reduction as the vehicle drove along. But it didn't matter, we had no idea if it was accurate or not. The efficiency was an estimate based on literature, there's no way to know that 100% of the hydrogen going into the fuel-cell was being converted in the reaction (it's not possible for that to be the case), and the specific energy would depend on the purity of Hydrogen in the tank.

We had to verify this model somehow. So we devised a little experiment that was based on what we'd learned from characterizing batteries [39][40]. In battery characterization, you hook a battery up to a constant-current load and draw energy out until the voltage hits the minimum. Then, you can plot energy on the x-axis, and battery voltage on the y-axis. In that sense you create a bit of a look-up table, if you keep track of the energy you can estimate the voltage. This is a pretty standard technique in determining electric vehicle range [41].



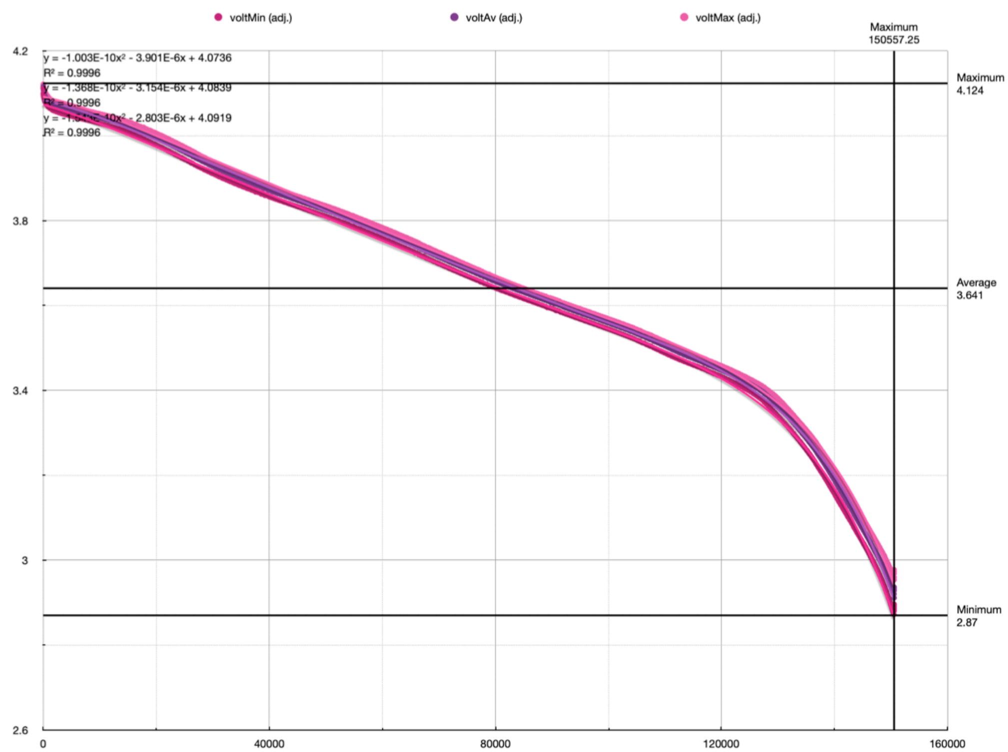
F.1.17—DYNAMOMETER HARDWARE TESTING PLATFORM FOR DEVELOPING MODELS FOR SIMULATION, AND FOR TESTING SIMULATION/REALITY GAP.



F.1.18—MATLAB SIMULATION FRAMEWORK FOR A HYDROGEN VEHICLE. THE ROLE OF THE MECHATRONICS ENGINEER IS TO MEASURE AND THEN 'INJECT' MODELS INTO THE SIMULATION FRAMEWORK. SPECIAL THANKS TO CONNIE LY.

We thought, what if we could do the same for the Hydrogen system? We hooked up the fuel-cell to a DC electronic load and drew constant power while logging the pressure in the Hydrogen tank. The goal was to generate a energy vs. pressure curve for the entire system which we could then either *plug* into the simulation as a look-up table, *or* use to fit parameters to our simple model.

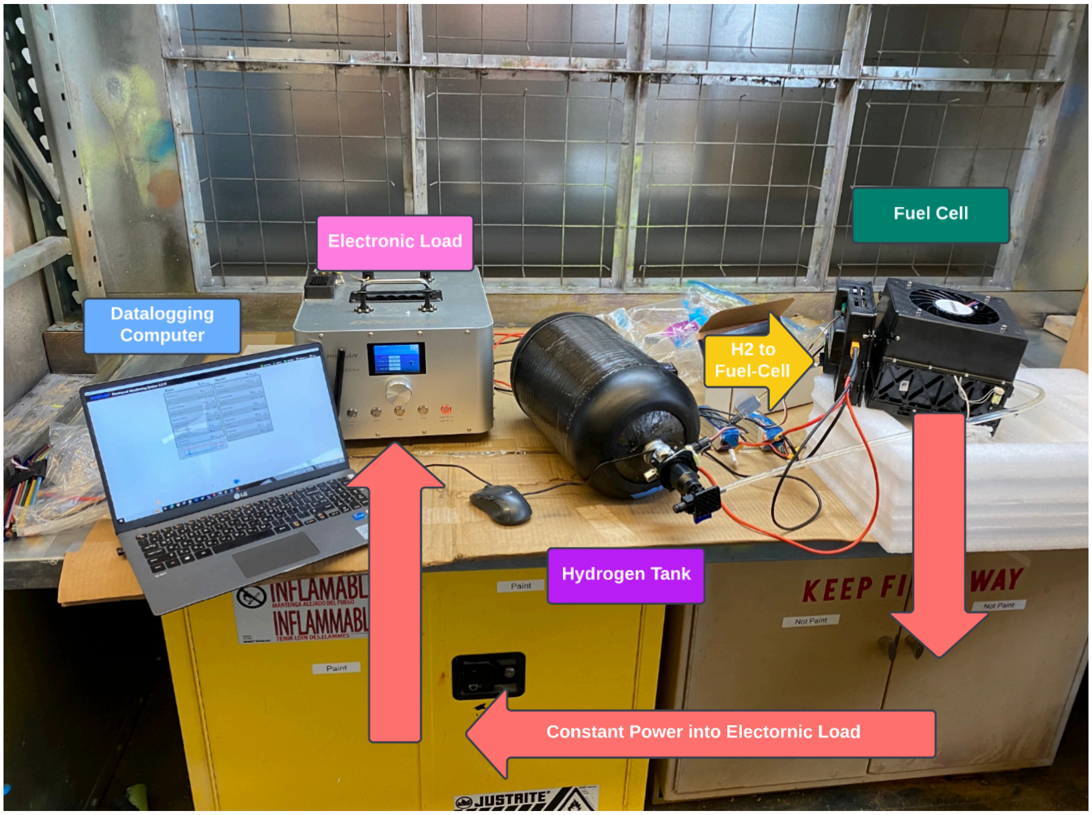
impedance-adjusted discharge graph @ 10A:



F.1.19—BATTERY CHARACTERIZATION OUTPUT. ON THE X-AXIS WE PLOT THE CAPACITY OF THE BATTERY AND ON THE Y-AXIS WE PLOT THE VOLTAGE. THE AREA UNDER THE SOC CURVE IS THE ENERGY CONTENT WHICH, IF WE KEEP TRACK OF AS THE VEHICLE DRIVES, CAN BE USED FOR RANGE ESTIMATION.

It was working really well, right until the resistors melted. Oops.

At the time of writing this thesis, we haven't been able to finish collecting the data we needed for me to present a pretty graph as a figure. But that's not the point. The point is I think it's really important to teach our students how to go through this process of creating an initial model of a system that you can use, and then devising experiments to ensure it is accurate. Often the job of mechatronics engineers goes beyond simply building the system and making it work. It is also to communicate the limitations, and the behavior in the form of models to other engineers developing different portions of the system, *and* to the simulation and intelligence frameworks.



F.1.20—HYDROGEN CHARACTERIZATION TEST SETUP.

Notes on Presented Works

We explore teaching a number of additional topics in Appendix A, and we build out a curriculum in Appendix B.

In terms of building out the curriculum, we started with what we defined above with four key areas of study. We looked around at a number of courses that already teach topics within these areas, and tried to pick parts from all of them to combine into a set of core classes.

We also then thought about the topics students would need to study before-hand in order to understand the content we were presenting in the core classes. Then we left some space for capstones and advanced subjects, and some low-stakes curiosity (which we talk about in the last chapter of this thesis). Finally, we ensured what we did was tractable in a four-year degree using MIT as a case study.

In terms of the core courses though, I want to describe briefly the intent behind them. I imagine some criticism will arise from the fact that they do not necessarily focus on depth of understanding. I think that's ok, and I think that's the point. The goal of the core is to provide a series of hands-on learning opportunities to introduce students to the wide array of topics contained in intelligent systems. To give them a reasonable amount of breadth in a way that's memorable, and leads to robust learning (we discuss this further in the next chapter). Courses before these classes provide a depth of mathematical and foundational knowledge, and courses after allow the student to dive deep into the details. But these intermediary courses provide the student with an understanding of the system, an understanding of their role in engineering the system, and the ability to build skills and vocational knowledge.

References

1. I. Asimov, Runaround. Astounding Science Fiction, Mar. 1942. Publicly Accessible: https://web.williams.edu/Mathematics/sjmiller/public_html/105Sp10/handouts/Runaround.html

2. L. Lehmann, “Isaac Asimov: dark side of a bright history,” Nature Correspondence, Mar. 2020. DOI: https://doi.org/10.1038/d41586-020-00785-z

3. M. Acar and R. M. Parkin, "Engineering education for mechatronics," in IEEE Transactions on Industrial Electronics, vol. 43, no. 1, pp. 106-112, Feb. 1996, doi: 10.1109/41.481414.

4. D. A. Bradley, “What is Mechatronics and Why Teach it?,” International Journal of Electrical Engineering Education, vol. 41, no. 4, pp. 274-291, Oct 2004, doi: https://doi.org/10.7227/IJEEE.41.4.2

5. “Fuel Cell Electric Vehicles,” Alternative Fuels Data Center, US Department of Energy.

6. M. A. Aminudin, S. K. Kamarudin, B. H. Lim, E. H. Majilan, M. S. Masdar, and N. Shaari, “An overview: Current progress on hydrogen fuel cell vehicles,” International Journal of Hydrogen Energy, vol. 48, no. 11, Nov. 2022, doi: https://doi.org/10.1016/j.ijhydene.2022.10.156.

7. “Fuel Cell Basics,” National Museum of American History, the Smithsonian Institution.

8. T. Markel, K. Wipke and D. Nelson, "Vehicle Systems Impacts of Fuel Cell Power Response Capability,” National Renewable Energy Lab, Future Car Congress: On the Road to Energy Efficient Vehicles, Jun. 2002. Accessible: https://www1.eere.energy.gov/hydrogenandfuelcells/pdfs/fcc02_finalpaper_040302.pdf

9. A. Ferrara and C. Hametner, "Impact of Energy Management Strategies on Hydrogen Consumption and Start-Up/Shut-Down Cycles in Fuel Cell-Ultracapacitor-Battery Vehicles," in IEEE Transactions on Vehicular Technology, vol. 71, no. 6, pp. 5692-5703, June 2022, doi: 10.1109/TVT.2021.3127582.

10. H. Li, A. Ravey, A. N'Diaye and A. Djerdir, "A Review of Energy Management Strategy for Fuel Cell Hybrid Electric Vehicle," 2017 IEEE Vehicle Power and Propulsion Conference (VPPC), Belfort, France, 2017, pp. 1-6, doi: 10.1109/VPPC.2017.8330970.

11. B. Katz, “A low cost modular actuator for dynamic robots,” SM Thesis, Massachusetts Institute of Technology. Department of Mechanical Engineering, 2018. Available: https://dspace.mit.edu/handle/1721.1/118671

12. B. Katz, J. D. Carlo and S. Kim, "Mini Cheetah: A Platform for Pushing the Limits of Dynamic Quadruped Control," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 6295-6301, doi: 10.1109/ICRA.2019.8793865.

13. S. H. Jeon, S. Kim and D. Kim, "Online Optimal Landing Control of the MIT Mini Cheetah," 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 2022, pp. 178-184, doi: 10.1109/ICRA46639.2022.9811796. Open Access Link: https://dspace.mit.edu/handle/1721.1/153616

14. J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt and S. Kim, "Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 1-9, doi: 10.1109/IROS.2018.8594448. Open Access Link: https://dspace.mit.edu/bitstream/handle/1721.1/138000/convex_mpc_2fix.pdf

15. P. M. Wensing, S. Kim and J. -J. E. Slotine, "Linear Matrix Inequalities for Physically Consistent Inertial Parameter Identification: A Statistical Perspective on the Mass Distribution," in IEEE Robotics and Automation Letters, vol. 3, no. 1, pp. 60-67, Jan. 2018, doi: 10.1109/LRA.2017.2729659. ArXiv Link: https://arxiv.org/pdf/1701.04395

16. P. M. Wensing, A. Wang, S. Seok, D. Otten, J. Lang and S. Kim, "Proprioceptive Actuator Design in the MIT Cheetah: Impact Mitigation and High-Bandwidth Physical Interaction for Dynamic Legged Robots," in IEEE Transactions on Robotics, vol. 33, no. 3, pp. 509-522, June 2017, doi: 10.1109/TRO.2016.2640183. Open Access Link: https://fab.cba.mit.edu/classes/865.18/motion/papers/mit-cheetah-actuator.pdf

17. S. Kim, 2.74/2.740 Bio-Inspired Robotics, Department of Mechanical Engineering, Massachusetts Institute of Technology.

18. 2-A Program, Department of Mechanical Engineering, Massachusetts Institute of Technology. Information: https://meundergrad.mit.edu/2A-degree/choosing-2A-concentration

19. 16-ENG Program, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology. Information: https://web.mit.edu/AEROASTRO/academics/undergraddocs/16eng-requirements.pdf

20. 6-2 Program, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. Information: https://www.eecs.mit.edu/academics/undergraduate-programs/curriculum/6-2-electrical-engineering-and-computer-science/

21. 6-5 Program, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. Information: https://www.eecs.mit.edu/academics/undergraduate-programs/curriculum/6-5-electrical-engineering-with-computing/

22. A. Alqaraghuli, What Every Electrical Engineering Student Must Know. 2020. Open Access: https://fet.eecs.umich.edu/What%20Every%20Electrical%20Engineering%20Student%20Must%20Know.pdf

23. J. Steinmeyer, 6.08 Intro to EECS via Interconnected Embedded Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

24. S. Leeb, 6.115 Microcomputer Project Lab, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. Information: https://web.mit.edu/6.115/www/

25. Miguel-Angel Martínez-Prado, Juvenal Rodríguez-Reséndiz, R. A. Gómez-Loenzo, K. A. Camarillo-Gómez, and G. Herrera-Ruiz, “Short informative title: Towards a new tendency in embedded systems in mechatronics for the engineering curricula,” Computer Applications in Engineering Education, vol. 27, no. 3, pp. 603–614, Feb. 2019, doi: https://doi.org/10.1002/cae.22101.

26. R. Comerford, "Mecha...what? [mechatronics]," in IEEE Spectrum, vol. 31, no. 8, pp. 46-49, Aug. 1994, doi: 10.1109/6.299539.

27. F. Harashima, M. Tomizuka and T. Fukuda, "Mechatronics - "What Is It, Why, and How?" An editorial," in IEEE/ASME Transactions on Mechatronics, vol. 1, no. 1, pp. 1-4, March 1996, doi: 10.1109/TMECH.1996.7827930.

28. Jie Liu, S. Jefferson and E. A. Lee, "Motivating hierarchical run-time models in measurement and control systems," Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148), Arlington, VA, USA, 2001, pp. 3457-3462 vol.5, doi: 10.1109/ACC.2001.946165.

29. Global surge in embedded software demand: here is why, DAC.digital, 2022. Accessible: https://dac.digital/global-surge-in-embedded-software-demand-here-is-why/

30. “Integrated Circuits,” NASA. Accessible: https://www.hq.nasa.gov/alsj/ic-pg3.html

31. A. N. Saxena, "Invention of Integrated Circuits.” World Scientific, 2009.

32. P. Parker. The Apollo On-board Computers, Apollo Flight Journal, NASA. Accessible: https://www.nasa.gov/history/afj/compessay.html

33. Integrated Circuits in the Apollo Guidance Computer, klabs.org. Accessible: https://klabs.org/history/history_docs/integrated_circuits/ic4-po.pdf

34. K. Shirriff, The Surprising Story of the First Microprocessors, IEEE Spectrum, Aug. 2016. https://spectrum.ieee.org/the-surprising-story-of-the-first-microprocessors

35. M. Chignoli, D. Kim, E. Stanger-Jones and S. Kim, "The MIT Humanoid Robot: Design, Motion Planning, and Control For Acrobatic Behaviors," 2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids), Munich, Germany, 2021, pp. 1-8, doi: 10.1109/HUMANOIDS47582.2021.9555782. Open Access: https://arxiv.org/abs/2104.09025

36. D. L. Trumper, 2.14 Feedback Control Systems, Department of Mechanical Engineering, Massachusetts Institute of Technology.

37. D. L. Trumper, 2.737 Mechatronics, Department of Mechanical Engineering, Massachusetts Institute of Technology.

38. D. L. Trumper, 2.171 Digital Controls, Massachusetts Institute of Technology.

39. “Race-pack Characterization,” MIT Solar Car. Open Access: https://www.adim.io/_files/ugd/067a81_00780811c8bb441c8df4d58ae436911a.pdf

40. “Battery Modeling + Resistance/Impedance Matching,” MIT Electric Vehicle Team, 2023. Open Source: https://github.com/Licence-to-Fab/battery-design/blob/main/Documentation/Battery_Modeling___Internal_Resistance_Matching.pdf

41. “BU-903: How to Measure State-of-Charge,” Battery University, Feb. 2011. Link: https://batteryuniversity.com/article/bu-903-how-to-measure-state-of-charge

42. K. Craig, “Mechatronics Online Courses,” Multidisciplinary Mechatronics Education. Link: https://multimechatronics.com/

43. M. Acar, "Mechatronics challenge for the higher education world," in IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C, vol. 20, no. 1, pp. 14-20, Jan. 1997, doi: 10.1109/3476.585140.

Chapter 2

Learning and Methods

There has been a lot of work, both in published literature and in examples implemented at the classroom-level, in defining hands-on exercises for teaching specific concepts in engineering. There's been a number of reasons for this, but largely the focus is to provide students with 'practical' experience in an effort to bridge the classic gap between theoretical knowledge and implementation skill in engineering education. Others have used it to the effect of better understanding our students' learning processes and in defining 'what works' when it comes to teaching.

Two key works specifically in the space of developing hands-on exercises for teaching engineering are Crispin Mount Miller's thesis from the Mechanical Engineering Department at MIT in 1995, and Katherine Ann Lilienkamp, (now Professor Katie Byl at UCSD) thesis also from the department in 2003 [1][2].

Crispin's thesis focused on developing hands-on exercises to understand, and somewhat counter the numerous studies in physics literature that indicated student's tendencies to hold onto pre-Newtonian understandings of the behavior of physical objects despite their satisfactory completion of physics courses [3][4][5]. However, the focus of his thesis went beyond the initial development of exercises.

> *"So while I do present, in Appendix 1, a description of all the exercises I have devised, with notes about how they have worked and suggestions for what might be improved, I have decided that rather than try to polish these or specify them precisely, a more important effort is to try to convey the benefit of my experiences to help other teachers develop the skill of recognizing what the students are understanding (or not) as they work on things." — Pg. 13* [1]

Professor Byl's thesis work focused on developing specific exercises for the MIT mechatronics courses taught by Professor David Trumper (including 2.737, 2.14, and 2.171 which are mechatronics, feedback controls, and digital controls respectively) [6][7][8]. Many of these exercises are still in-use today, and as such what was presented in her thesis was different than that of Crispin's. Her work painstakingly documents the engineering of each of the exercises presenting creative approaches to teaching specific concepts using specific systems. The attention to detail in the design of the exercises and demos enables almost part-to-part implementation of them in other classes looking to teach the same concepts.

One thing to note is that their work, specifically in the area of understanding *how* to teach our students engineering and documenting robust teaching methodology, occurred during a time period where learning was *far less understood* than it is today. One of my favorite books on the subject (one that I will heavily reference below and have already mentioned in the recommended reading section) *Make it Stick—the Science of Successful Learning* by Peter Brown, Henry L. Roediger III, and Mark A. McDaniel wouldn't be published until 2014 [9]. When I say this, I in no-way mean to invalidate or discredit any of their work, it's work I've learned a lot from (as you can probably tell).

It does, however, give us a distinct advantage of an increased depth of understanding in learning psychology. And more importantly, and to the end of teaching, a *design paradigm* for both educational environments and teaching exercises. To some extent, we now know far better what 'works' in teaching, and we can take advantage of applying that knowledge to our methodology.

It is also true that much of this literature doesn't make its way into the engineering classroom, and there have been numerous publications on that subject in recent years [10][11][12][13]. The reasons for this I can only speculate. But I think one, at least the one I see mentioned most often, is the high use (not necessarily reliance) on 'anecdotal' evidence in an effort to be illustrative. It feels somewhat arbitrary, and it's *sometimes* difficult to extend such evidence past the contexts in which it's presented. However, I argue that sometimes stories are far more translatable than assigning numbers to arbitrary concepts.

> *"First of all, it may bear mentioning that even in the 'hard sciences' such as physics and the engineering sciences, qualitative and subjective knowledge plays a much more central role than the apologetic or dismissive tone often used with those words might lead one to suppose.*
>
> *In saying this I do not mean at all to disregard the importance of making careful measurements where measurements can be made, or of handling rigorously the numbers that come from measurements or anywhere else. Nor am I referring to philosophical chestnuts like how do we know we're not dreaming. At least for the purpose of this discussion, I do consider careful measurement itself to be a reasonably objective affair*
>
> *But I think that the central accomplishment of a science is not the numbers it has measured but the framework of theory in which it places them (and that, similarly, the fundamental skill required of an engineer is to be able to form competent analytical models of whatever physical objects one deals with descriptions that establish a context in which data become useful), and these sorts of things are a different matter. While they may include quantitative predictions, these theories or models themselves are not measurements at all but descriptions of logical structure, complex products of imagination and choice that can never come from a measuring instrument. They are certainly not arbitrary—some models fit data much better than other ones —but nonetheless a model's connection to the world is inescapably mediated by subjective judgment." — pg. 14* [1]

The second reason that I can think of is simply that learning is very much not intuitive. *Make it Stick* spends its first chapter discussing a number of our general, and wide-spread misconceptions on how learning works and notes that there's is a critical difference between strategies that *feel* like they work, and strategies that actually produce robust, long-term learning [9]. They refer to this as *"illusions of knowing,"* and this lack-of-awareness actually exists both on the side of the student and the teacher in many cases. This makes it hard to trust the research because it *feels* so opposite than the techniques we're generally passed down.

> *"We are poor judges of when we are learning well and when we're not. When the going is harder and slower and it doesn't feel productive, we are drawn to strategies that feel more fruitful, unaware that the gains from these strategies are often temporary." — pg. 3* [9]

Another point on this that both Crispin and Professor Warren Seering (a Professor of Mechanical Engineering at MIT and someone I spoke to at length about the topic of teaching) made was the idea that despite the existence of the literature and regardless of the depth of the understanding any of us as teachers have about learning, teaching can still never happen 'open-loop.' There's no reasonable way to define an exercise to guarantee a specific learning outcome. What our students take away is a product of their past experiences. This I can only quote from my own observations from previous teaching as well as discussions with those I've taught with. But it's, I think, one of the most important lessons for the educator that student feedback, and adjusting teaching methods based on that feedback is critical and requires detailed and specific observation.

All this considered, I decided to dedicate the first half of this chapter to a summary of what I've learned about learning, both form the 'understanding psychology' perspective, as well as the practical experiences I've had in teaching hardware design. If I'm being fully honest, I always felt that electrical hardware design was a great medium for teaching, and had a number of theories on its effectiveness that I could articulate. But it wasn't until I started to look into the details of learning psychology that I could start to pin-point exactly *what parts* of it worked and *why*. I don't pretend to be an expert on this subject, but I felt that mental framework was important to document both for both in formal and informal settings for teaching engineering and connecting that to the recent literature.

In the second half of this chapter I present a series of exercises that cover the topics we presented previously as ones that we feel are 'important to teach' in robotics. I also spend some time documenting what I think the student will learn from them, and any possible pit-falls, traps, nuances, or anything else that might be interesting. For exercises that were 'pre-tested' in some capacity, I note any observations we made during those processes.

Overview of Learning[7]

Peter Brown, Henry L. Roediger III, and Mark A. McDaniel define learning in their book *Make it Stick—the Science of Successful Learning* as the following:

> *"Acquiring knowledge and skills and having them readily available from memory so you can make sense of future problems and opportunities." — pg. 2* [9]

Already, we've agreed on a few things as noted in their book. First, all learning requires memory or some form of retention. We need to *remember* the formula, method, experience, or other piece of critical information we're trying to use [9]. Second, that the goal of learning is not simply memory, rather to use the information and experiences that are stored in our memory to make sense of the world around us in whatever form that may be [9].

For engineering, we can make this definition a little more specific. When it comes to teaching undergraduate students at least, this definition also frames our teaching goals; **learning is understanding the fundamental science, mathematics, and methods pertaining to a engineering discipline and being able to apply said understanding to solve unfamiliar, real world problems in a rigorous manner *or* to explain an observed phenomenon** (I'll cover this in far more detail past this initial statement in later sections).

The Tiny Librarian in your Brain

Imagine your brain as a little library similar to the cartoon in Figure 2.1. At the front of this library is a tiny librarian. When you need information, you ask your tiny librarian. This information could be of many forms. You might ask for the specific formula to calculate the peak stresses of a statically-determinate beam in bending when loaded at one end with a point-load. In that case the librarian would likely go to the stacks, pull out the singular formula from a single book, and bring it back to the front of your brain where you're waiting to retrieve the information.

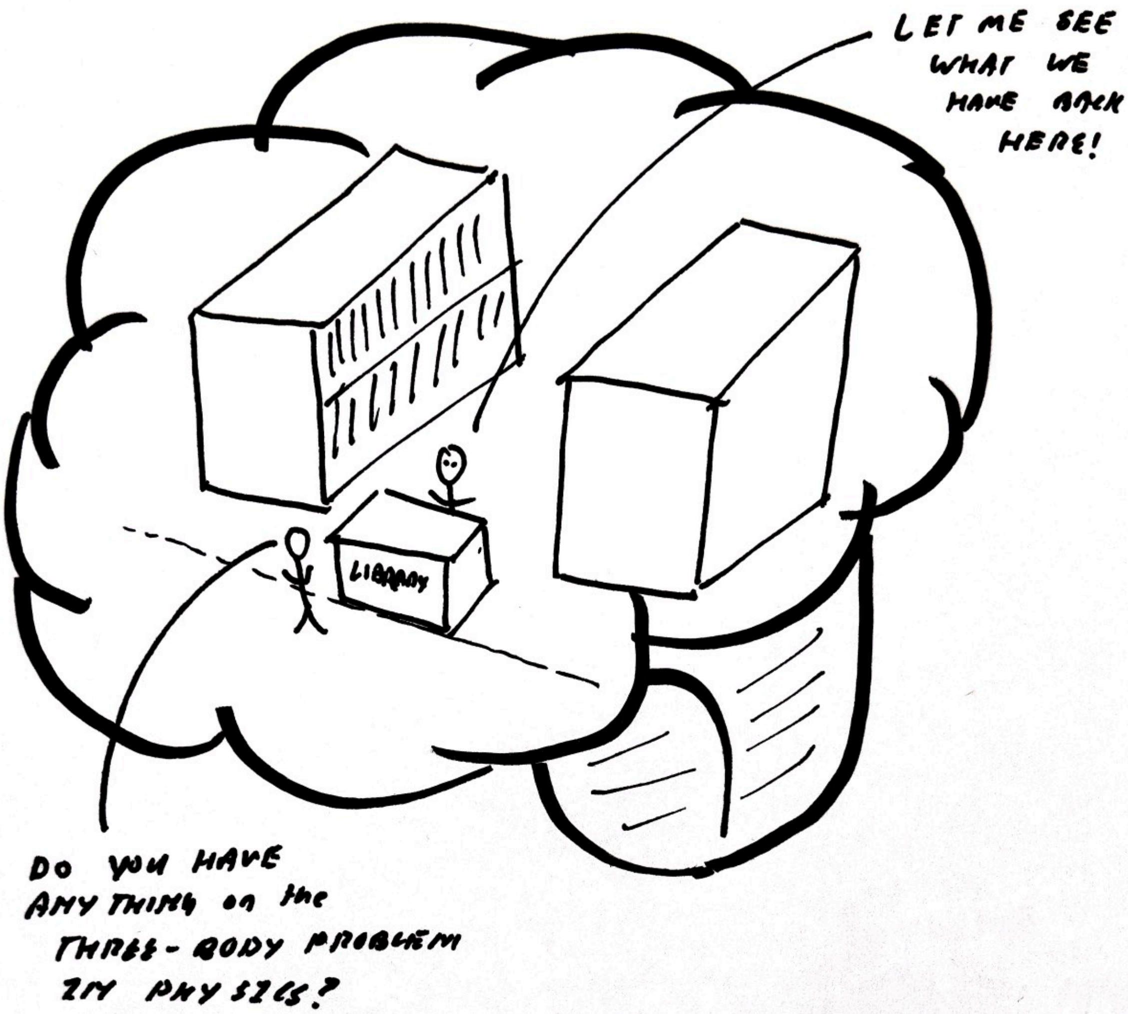
However, the information you might request could be far more general. You might ask the librarian to bring you back a whole bunch of information on the three-body-problem in physics and various attempts to solve such a problem. In which case your little librarian would go in the back, and come back with a number of books related to newton's laws of gravitation, the work of Heinrich Burns and Henry Poincaré, typical initial conditions, and attempts at scaling the two-body problem [14][15][16]. They might also return with a Medium blog post about the San Ti from Liu Chixin's *Remembrance of Earth's Past* trilogy and a number of others related concepts on nearby shelves that you didn't necessarily ask for [17][18]. All this information is spread out on the library table in front of you ready to be used and applied.

The library table is your **short-term memory**, and the stacks are your **long-term memory**.

Two ideas follow naturally from this. First, we're not going to leave all the books in the library out on the library table for constant access. They simply won't all fit. Second, the more times we ask the librarian to go access a

---

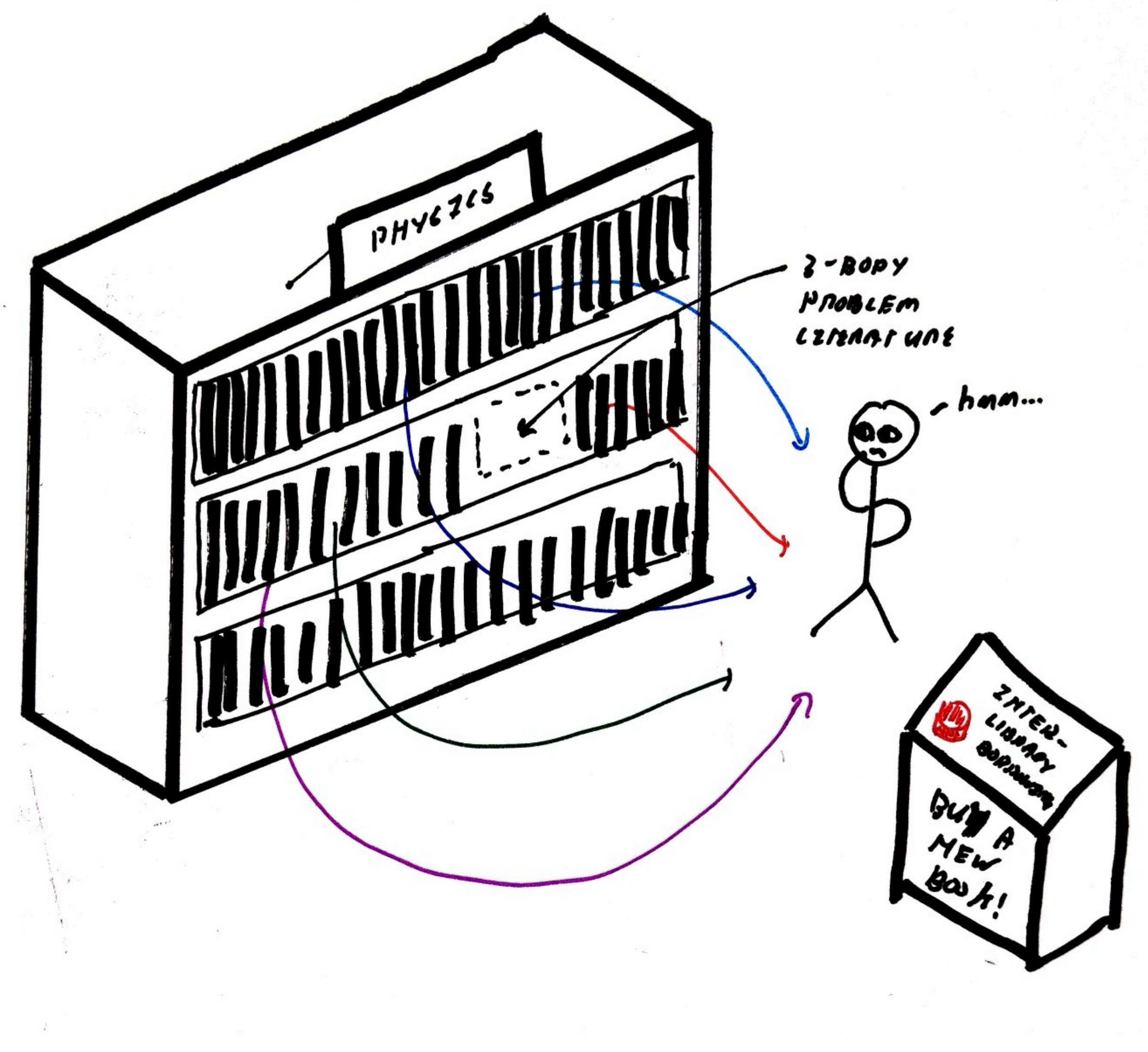
[7] If you've read the book 'Make it Stick' as suggested by the section on recommended reading, you can largely skip this section (and potentially the following one). I include it here only as a summary of key concepts presented in that work and other such works that provide framing for the later parts of this thesis.



F.2.1—INSIDE YOUR BRAIN IS A TINY LIBRARIAN.

particular set of information stored in the stacks, the quicker they'll be able to access it. The first time they might try searching by section, author, or alphabetical order. The second time they might know the author and book title but forget the specific section in the book you might need. But over enough times they'll quickly be able to point to even the specific page containing the information. This builds a map to both the location of the information, its content, and information considered *related* or adjacent.

There's another scenario, and this is the scenario we present in Figure 2.2. This is the scenario where the librarian goes back into the stacks and notices the shelf that is supposed to have books on the three-body-problem is empty.



F.2.2—THE TINY LIBRARIAN HAS IDENTIFIED A BOOK WE DO NOT HAVE BUT NEED.

In this scenario, the librarian might pull some related books off the shelf and given them to you to get you started. But if they're a good librarian, they will likely recommend that you need to go out, and find the books that you would need to fill this little hole on your library shelf. They might even recommend other people who could give you some books that could fill this space, but now both you and the librarian know there's a little bit of missing knowledge that you weren't aware of before.

This is a little preview on how memory works.[8]

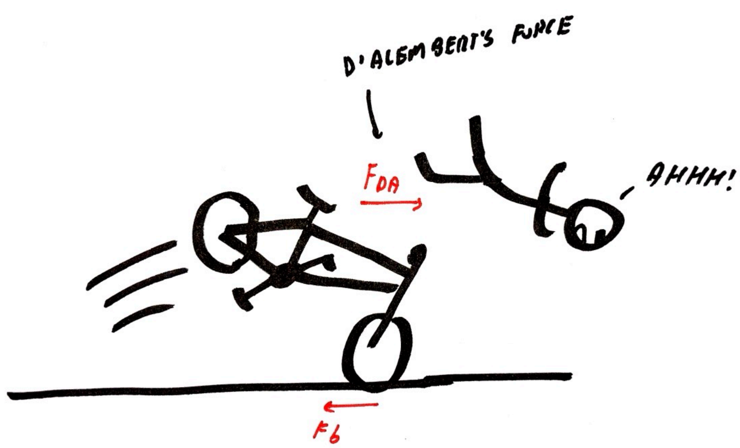
Information Storage is a Three-Stage Process

Make it Stick describes three steps to taking information and storing it in long-term memory. These steps are Encoding, Consolidation, and Retrieval [9].

Encoding is the process of converting sensory perceptions to representations that are not fully understood and storing them in short-term memory. Without further study, this information would leave our short-term memory and be forgotten [9].

Consolidation is the process of connecting these new memory traces to something concrete and specific that is already well-understood by our brains. This strengthens learning and prepares to push it into long-term memory [9].

D'Alembert's Principle is a good example of this: if a virtual force with a magnitude equivalent to the mass times net acceleration of an object not in equilibrium is placed at the center of mass of said object opposite to the direction of acceleration, the object can be analyzed as if it were in equilibrium [19].



F.2.3—A DEPICTION OF D'ALEMBERT'S PRINCIPLE.

[8] This analogy was inspired by **Professor Mary Pat Wenderoth** from the University of Washington who described memory the following way: "Our brains are like a forest, and your memory is in there somewhere. You're here, and the memory is over there. The more times you make a path to that memory, the better the path is, so the next time you need the memory, it's going to be easier to find it. But as soon as you take your notes out, you have short-circuited that path. You are not exploring for the path anymore, someone has shown you the way" — Make it Stick, pg. 229-230 [9]. It was additionally inspired by Tim Urban's Ted Talk, **Inside the Mind of a Master Procrastinator** [20].

This is quite abstract for the student who is used to solving static problems. But the way it's often explained to students in the *2.007 Design and Manufacturing I* course is to think about pressing the front brake too hard on a bicycle [21]. A decidedly more comical depiction of this is presented in Figure 2.3.

The key dilemma in this problem is why does the bike flip? The front brake imparts a force towards the back of the bike yet the rider *feels* a forward throwing force. Detailed free body diagrams of the bike, person, and wheels as individual bodies acting in a system will reveal the answer. But intuition building is key.

One way to think about this is using reference frames. Imagine the wheel suddenly gets locked to the ground at the contact patch and we now analyze the bike with respect to the locked wheel. The pivot point becomes the hub of the wheel and the rider (who's center of mass is generally higher than the height of the hub of the wheel off the ground) still has some 'rotation inertia' about the pivot that needs to be dissipated to come to a stop. With only the friction of the brakes to impart a torsional impulse to slow this rotational inertia, and if static friction is not high enough to counter the inertial force, the bike must pivot about the front wheel for a certain time t so it's inertial may be driven to zero.

By imagining (and hopefully not experiencing this feeling directly) the student can connect the physics principal to an observation and understanding they hold about the real world. This makes the principal more concrete and 'memorable.' They now imagine D'Alembert's force as the force that throws you off the front of a bike, and can now connect that to similar phenomenon they observe if that information is encoded and consolidated in the form of a mental model.

This would be like the process of 'shelving' a new book in our library example. The first thing that might happen is the librarian reads the title, maybe skims the table of contents, looks at the author, and tries to think about where to shelve it in the library. The content isn't fully understood yet, but the librarian can move that book to the area of our brains where similar information exists. Guidance in this process is critical because if the book is moved to the wrong place, then an incorrect correlation is formed between the new information, and related information.

Retrieval is the act of recalling this information from long-term memory to be used and applied [9]. *Make it Stick* spends a large portion of the book distinguishing between this and repetition. **Repetition is not the same thing as retrieval** [9]. Repetition would be like leaving a book out on the library table and constantly looking at it for the answer. Retrieval would be like closing that book, putting it back on the shelf, and then sometime later trying to find it again. Finding your way to that information without short-circuiting the effort by telling yourself the answer builds a stronger path to that information in the brain, your librarian can recall the information much quicker.

Interestingly, *Make it Stick* makes the point that this process of effortful-retrieval **requires a little bit of forgetting** [9].

> *"Since as far back as 1885, psychologists have been plotting 'forgetting curves' that illustrate just how fast our cranberries slip off the string. In very short order we lose something like 70 percent of what we've just heard or read. After that, forgetting begins to slow, and the last 30 percent or so falls away more slowly but the lesson is clear: a central challenge to improving the way we learn is finding a way to interrupt the process of forgetting." — pg. 28* [9]

But the book quickly makes the point this is *not* the same as constantly repeating something over-and-over so you don't forget it.

> *"Massed practice [repeating the same skill over and over in order in an effort to gain] gives us a warm sensation of mastery because we're looping through information through short-term memory without having to reconstruct the learning from long-term memory. But just as with rereading as a study strategy, the fluency gained through massed practice is transitory, and our sense of matter is illusionary." — pg. 82* [9]

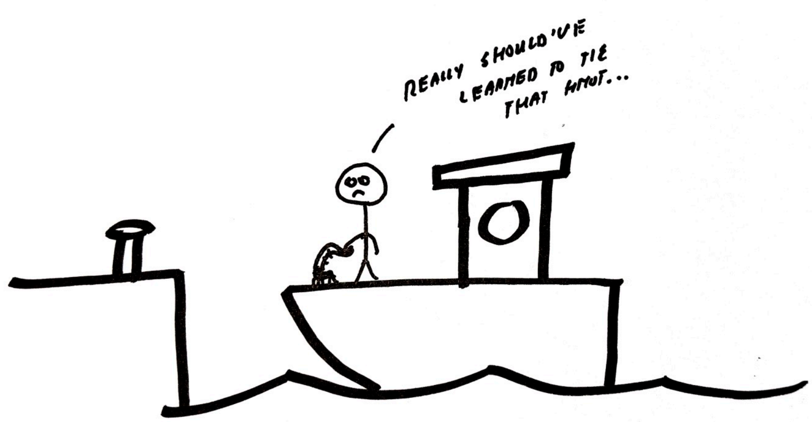
We'll touch more on this below when we talk about simulation. But to illustrate this just a little bit. This is the difference between cramming for an exam the night before, memorizing the content, and trying to apply those memorizations three weeks later to a new problem without having looked at the material [9]. We've brought a bunch of new books into the library, put them out on the front table, and we stare at them until we're done with that information. Then we leave with the books. This is different than connecting that new information to current knowledge, asking our librarian to store it in the back, and revisiting it occasionally every few days, or weeks. In that case, we'd probably do better applying the information to a new problem later.

Unfamiliar Problems Prime the Brain for Learning

Make it Stick also identifies the concept of **Generation**, which is trying to solve a problem before being presented the solution.

> *"When you're asked to struggle with a problem before being shown how to solve it, the subsequent solution is better learned and more durable remembered. When you've bought your fishing boat and are attempting to attach an anchor line, you're far more likely to learn and remember the bowline knot..." — pg. 86* [9]

Generation is what we described above in the second situation we presented to our librarian. When the librarian when back to the stacks and didn't find any books on the subject we were interested in. Your brain is actively identifying a gap in its own knowledge, and the frustration or curiosity that arises from that primes the mind for learning [9].



F.2.4—GENERATION LETS US IDENTIFY GAPS IN OUR OWN KNOWLEDGE [9].

Simulators Provide an Environment for Robust Learning

Simulators apply a combination of concepts in a structured learning environment that provides the opportunity to practice applying that information to unfamiliar problems. So we're no longer just talking about memory, which we agreed that *remembering* information was a precursor to the goal of learning but not necessarily the goal itself [9].

Similar logic is applied when we consider the reason for the increasingly popular Cooperative Education (Co-Op) requirement in higher education. Northeastern University's Co-Op program is widely regarded as one of the largest and most comprehensive examples of this with over 90% student participation [22]. Students spend 6-months at a company or research lab with a focused project applying skills they learned in the classroom to real project [23]. The stated reason for the Co-Op program at multiple universities is to enable the connection between theoretical knowledge and practical implementation of said knowledge, and to increase student's awareness about the true nature of their work enabling them to make better decisions regarding their degree paths [24][25]. A Medium blog post by Serena Wang on Northeastern University's Co-Op program illustrated some additional benefits.

> *"Some people come out of a Co-Op realizing what they don't want to do for the rest of their lives and that is just as important as knowing what you want to do for the rest of your life…"*
>
> *— Serena Wang* [26]

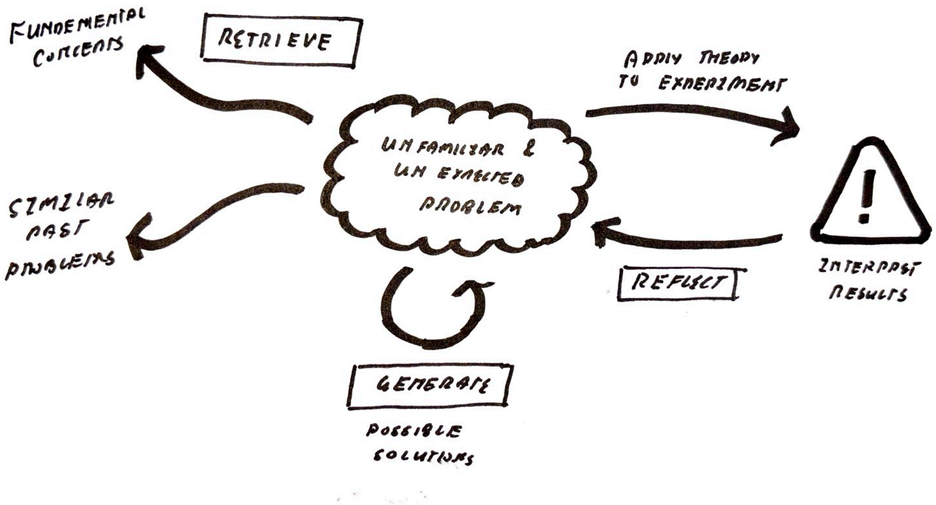
Time also seems to be an important factor. Some universities claim that Co-Ops are better than internship programs because of the increased level of 'immersion' a student can achieve, which is generally agreed upon [27]. There some literature that's come out of the Georgia Institute of Technology that's self-described as liminal that indicates the potential for Co-Ops to increase students engineering abilities, but these studies are generally difficult in terms of controlling confounding variables [28]. Other academic work in this area has suggested Co-Ops lead to increased graduation rate [29][30], as well as increased accessibility of studying engineering to under-represented minorities [31]. In my humble opinion, this alone makes the concept of implementing 'Co-Op-like experiences' in the classroom worth studying.

Make it Stick provide two very illustrative examples of this (albeit unrelated to engineering). Pages 79-81 of the book describe an experiment by the California Polytechnic State University baseball team; pages 10-12 describe one pilot's experience in flight school [9]. These sections are definitely worth skimming, but I'll provide a brief summary here.

Flight Simulators are a great example of a good environment for learning. In a simulator, the pilot generally enters with some basic knowledge of the aircraft systems, some knowledge of possible failures, and some basic piloting skills. Instructors will let the pilot fly until an unexpected time when they will randomly instigate a failure or emergency the pilot must respond to. The pilot must draw on their knowledge (without opening a book, or a reference because there's rarely time to do so) to identify the problem, and then take steps to correct it. They'll try a solution, read the response of the aircraft, and determine if that solution worked or didn't work. The consequences for failure are minor in reality (as any crash that may result is simulated) but are visual and real. It's often possible to review the series of actions that led to a negative outcome, and adjust technique accordingly [9].

Baseball Pitching Practice is a similar case study. The CalPoly team found that hitters that practiced types of pitches in-order (10 curveballs, 10 fastballs, so on) seemed to hit those balls better at the end of the streak of 10 pitches but performed worse in the game than players who were thrown, say, 10 random pitches. The latter of these methods is called **interleaving** [9]. This should make sense, in a game the player doesn't know the next pitch that will be thrown. They need to identify the pitch, the nuanced differences between the current pitch and previous pitches they've hit, apply the correct technique and actively adjust it all in just a couple milliseconds. This is not memorization, it's ingrained technique and instinct that comes from practicing in a way that's similar to the actual game [9].

Make it Stick argues that these are the environments that provide the *"kind of active engagement that leads to durable learning"* (pg. 11 [9]). Figure 2.5 shows a sort of 'map' of how this works. An unfamiliar problem is presented of which the solution is not know. This triggers **generation** of possible solutions by **retrieving** both fundamental concepts to apply to the unfamiliar problem, as well as past problems with similar symptoms from long-term memory. A possible solution is then applied in the form of an **experiment** (at least during the learning phase) and the **consequences** of applying that solution are analyzed through **reflection**. This reflection is both immediate and delayed as it can occur hours or days after the actual learning exercise [9]. Consequences of decisions or gaps in knowledge are clear, and **difficulties** instill curiosity that primes the mind for learning [9].



F.2.5—SIMULATION COMBINES THE CONCEPTS OF GENERATION, REFLECTION, RANDOMNESS, AND ELABORATION INTO A ROBUST LEARNING ENVIRONMENT.

This idea of *"practice like you play"* hasn't fully made its way into engineering education beyond a few isolated cases [32]. In engineering teaching, an internship, Co-Op, or research experience is generally as close as you can get to 'real' experience; it's generally more tractable for universities to provide resources to help students seek out these opportunities rather than implementing them at the classroom level. I don't necessarily advocate for turning every classroom into an internship, but I do advocate for understanding the parts of the internship environment that lead to skill-building, increased retention of content, and development of innate curiosity.

—A Very Loose Paradigm for Teaching Engineering

In terms of translating these ideas to engineering, and despite the fact that what we're talking about here is fairly well documented research, attempting to generalize is still somewhat prudent. So I don't meant to make this section too rigid past presenting certain observations from my own past experiences in teaching engineering.

After much discussion with co-instructors from various classes and friends in more informal learning environments, I think I can identify two loose 'features' that seem to lead to better retention of content, and then the later application of that content when it comes to engineering. The first is dealing with consequence, and the second is eliminating shortcuts.

By 'dealing with consequence' I mean providing opportunities for students to go through the engineering process in ways similar to which they would be doing so for a real system: the process of defining a problem, figuring out what to model, applying physics to develop that model, understanding when to *stop* modeling, using that model to make a decision or design choice, and verifying that decision works on hardware. Such open-ended problems enable the student to calibrate their judgment of their abilities, gaps in their mental models, and knowledge.

By 'eliminating shortcuts' I can only really think to describe this as structuring an exercise in such a way that it's easier for the student to do it the 'right way' than the 'wrong way.' This statement troubles me a bit because it brings up the question on wether there really is a right or wrong way when it comes to engineering. But for the purposes of this thesis I mean the 'right way' as the specific method that we're hoping to teach. I will explain this further in the second half of this section, but I say it now just to get you thinking a little.

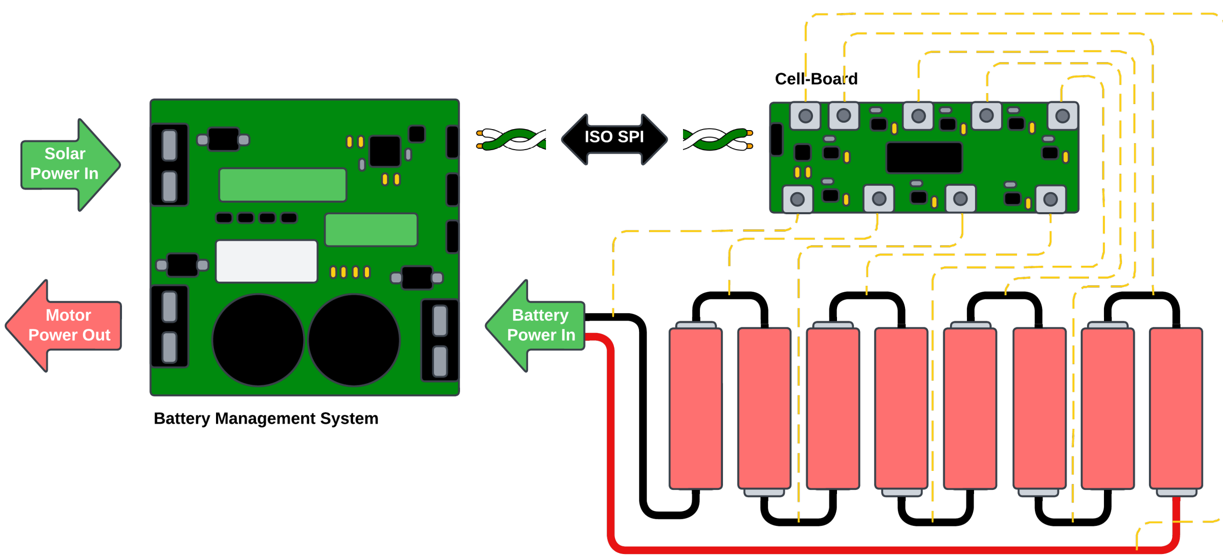
I want to note here, before we go on, that I don't believe I'm the first to notice these, nor am I pretending that I've invented them. I more seek to document them in relation to the stories I'm about to tell and the exercises I'm about to design. At any rate, the best way to understand them I think is anecdotally.

The EMI-Limiting Resistor Problem

This one came from a time where I was observing a group of people on solar car solve a strange issue we were having with the battery system. The solar car has a battery, it's charged by the sun, and it makes up any differences in power demand when the sun isn't shining brightly or the motor wants more than 1kW of power to drive up a hill.

As is likely now common knowledge, Lithium batteries are fickle things, and if one exists its safe region of operation some potentially dangerous situations can follow (most likely a fire). To avoid a fire, each row of cells in the solar car batter is monitored for temperature, and voltage. The overall battery current in monitored from the main 'headboard' or Battery Management System (BMS) which isolates the battery from the rest of the care in the event that any measured parameter goes outside the acceptable range. Individual cell currents are not monitored for simplicity, this is a design tradeoff we decided on that's worked thus-far.

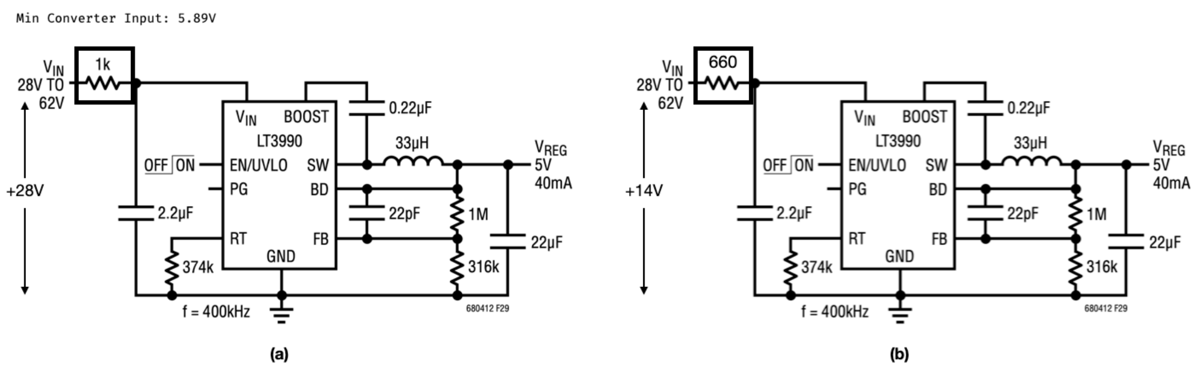
The individual cells are connected in series and parallel into modules, which are connected together to form the full battery. But we'll use a simplified image in this case (see Figure 2.6).



F.2.6—WIRES FROM EACH VOLTAGE TAP ARE CONNECTED TO THE POSITIVE AND NEGATIVE TERMINAL OF EACH SERIES CELL IN THE BATTERY. THE LTC6804 READS THE VOLTAGE BETWEEN ADJACENT TAPS, EQUIVALENT TO THE VOLTAGE OF EACH CELL. THIS DATA IS SENT OVER THE ISO-SPI BUS TO THE MAIN BATTERY-MANAGEMENT-SYSTEM (BMS) WHICH IS RESPONSIBLE FOR ISOLATING THE BATTERY FROM THE REST OF THE SYSTEM IN CASE OF ISSUES.

The cells are monitored with a cell-board, which has a series of voltage taps that get fed into an LTC6804 multi-cell batter monitor chip. This chip communicates over an ISO-SPI connection to the main headboard which reads the data from all the cell-boards (one per module) and decides wether the cells are within their operational ranges, and what to do if they're not. The simplest model of these boards is a bank of multiple sensors that communicates with an external microcontroller.

The communications system, the ADC that reads the thermistors, and other components on the board are powered by a 5V buck converter (DC-DC step-down) that is powered by the battery cells. In the original design a 1k resistor was placed in series with the input of the converter to prevent in-rush current and limit electromagnetic interference (EMI) as specified by the data sheet [33]. This design worked fine. Everything powered up properly, and the cell boards would communicate cleanly with the headboard.



F.2.7—(A) OLD CONVERTER DESIGN WITH 1K EMI-LIMINTING RESISTOR AND 8S CELL INPUT. (B) MODIFIED CONVERTER DESIGN WITH 660Ω EMI-LIMINTING RESISTOR AND 4S CELL INPUT. SEE [33].

From car to car, the solar car mentality is to generally not change things in the electrical system that worked well if possible. Therefore the designers used the same cell board in the new version of the battery. The only difference was in the new car, there were four series cells instead of 8 leading to a total input voltage to the buck converter that was half of the original. At face value this is OK. That voltage was well within the range of the buck converter (see Figure 2.7).

Fast forward to a few months later, we're all sitting around an open 120V battery on the first floor of N51 wondering why the boards were talking to the headboard. The code was the same (tested to be functional), the hardware was the same—still nothing. It wasn't even a deterministic problem. There are 8 total cell-boards that need to send data back to the car. Sometimes 4 would work, sometimes 6. 7 never worked and 8 only worked after flashing.

As is generally the case in debugging, all sorts of theories were brought up, mainly centered around the question: *what is different this time?* This is loosely how I remember the conversation going.

> *"The cables are longer, maybe it's EMI?"*
>
> *"No, ISO-SPI is a repeater architecture, so really it's the max distance from board-to-board that matters not the overall...?"*
>
> *"It says 50 meters."*
>
> *"That's with a CAT-5 cable"*
>
> *"Still, this is like 3 feet."*
>
> *"Idk, put some copper tape on it."*
>
> *"Dude the motor isn't even on, how could it be EMI?"*

That wasn't the issue. We wondered if the microcontroller was getting corrupted somehow but this was unlikely. I'll spare you the mystery puzzle and tell you the outcome.

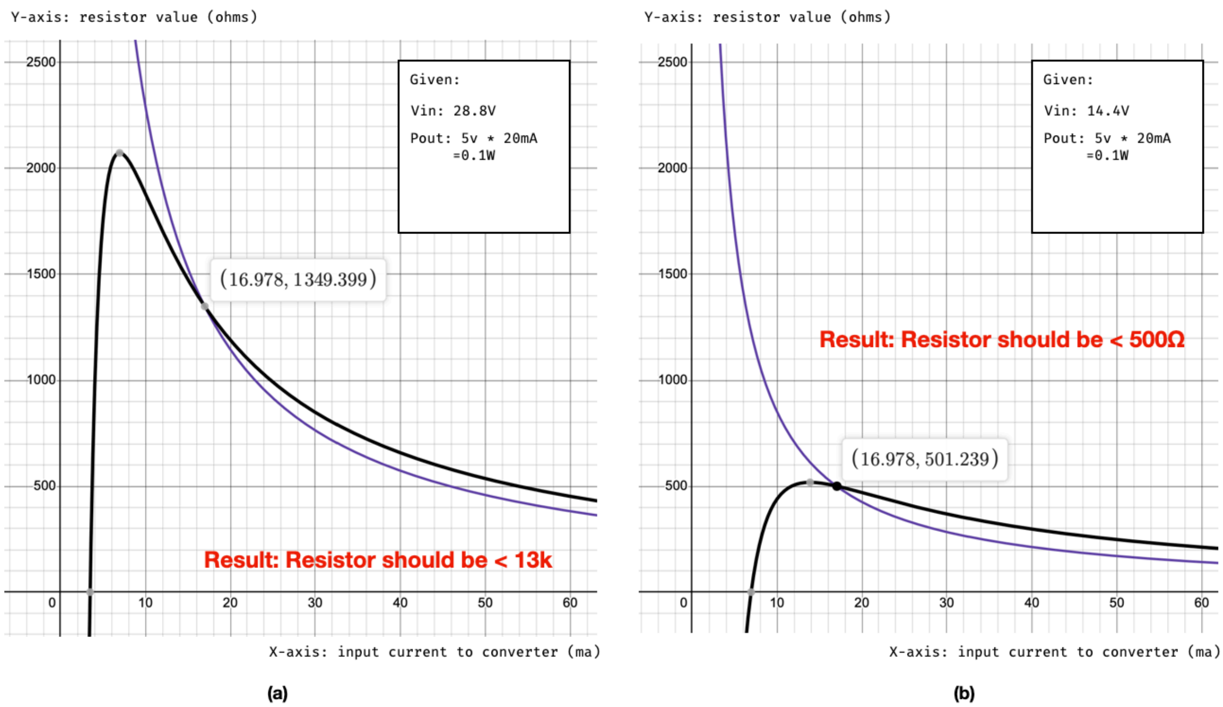
Remember that resistor? That one measly resistor? The one designed to help us with EMI? Turns out that was the problem.

$$(V_{in} - I_{in}R)I_{in} = V_{out}I_{out}$$

(Power in = Power Out)

$$(V_{in} - I_{in}R) > 5.89V$$

(Minimum voltage input to converter)



F.2.8—(A) INPUT CURRENT VS. LIMITING RESISTOR VALUE AT 8S INPUT TO THE CONVERTER. (B) INPUT CURRENT VS. LIMITING RESISTOR VALUE AT 4S INPUT TO THE CONVERTER. WE CAN SEE THAT 660Ω IS RIGHT OUTSIDE THE EDGE OF FEASIBLE.

The truth is the only thing that had changed was the voltage input to the converter. With 4 cells instead of 8, the converter input was half of what the 'reference design' was originally specified for. It turns out the 1k resistor in the data-sheet was *also* specified for a voltage range above a 7-8 cell input. As the converter started to draw more power, the voltage drop across the resistor caused the total voltage input to the converter to drop below 5.89V (the minimum input for the converter). The converter shut off every time it tried to communicate, and turned back on right after. It could provide just enough power to turn on the 'I'm alive' LED, but not much more than that (see Figure 2.8).

It was a solar car alum who figured this out, so full credit goes to them for that.[9] One of the other students, however, had actually suspected this earlier and changed out that 1k resistor to a 660Ω resistor. But this was right at the edge of where the board would turn on, but still shutoff when the board started drawing more power to communicate.

Oops.

[9] I do not include names in this story for privacy reasons.

Most students coming out of an introductory Physics Electricity and Magnetism course would be able to tell you that current through a resistor create a $V = IR$ voltage drop across it according to Ohm's law, but few would be able to tell you a practical situation where that might cause you a problem. Here was one, and it took the team 6 days to remember that—and now they'll never forget it.

The experience t taught them quite a bit. It taught them how to compare two systems to see what changes from one to the other. It taught them to devise tests to verify their hypothesis, and wether these tests probed them right or wrong, they helped them update their mental models on both the concepts they were drawing from and the physical system in front of them. They learned to read a data-sheet more closely, and that when translating one design to another attention to detail really matters. Every component value was based on something, and that something can come back to bite you later.

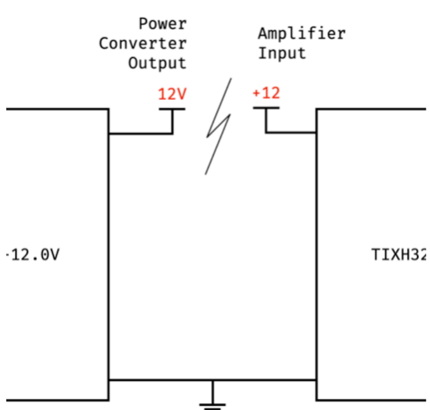
For me, it also taught me that a student can learn something despite being 'given the answer' if the situation is right. Here was a system where the design was on-paper in front of them, and the direct translation from one system to even a slightly different system didn't work and produced a lot of learning when the student had to figure out why on their own.

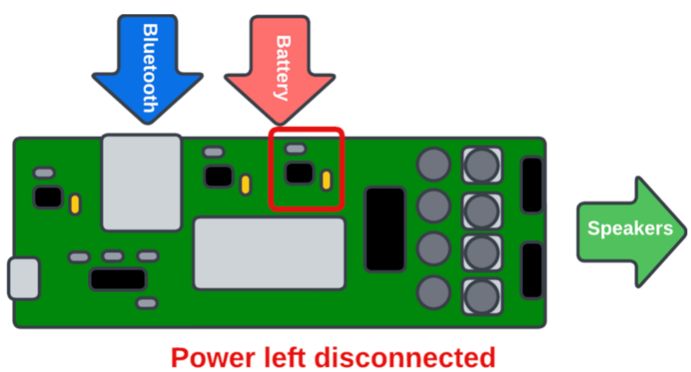
I do think it's worth mentioning a situation like the one above is largely intractable in the classroom. The problem in this case is too open-ended, the students would've taken far longer to solve it without intervention, and the hours put into the exercise and the frustration associated (while emotions that burn the knowledge into ones memory) aren't exactly the *desire* when it comes to teaching. So it's more useful for us to think of ways to apply this to smaller problems in more constrained environments.

Deliberately Creating Problems for Our Students to Find

It was Lee Zamir's idea to leave the 12V power bus disconnected on the PCB. And together, we chose not to tell the students.

It was January 2022, Fischer, myself, and a few of our friends were teaching a PCB design course over IAP [34]. Our goal for the course was to try to teach some of the things we learned about electronics in Solar Car and Formula SAE *through* PCB design into a classroom setting. It was a short, 4-week course. Students designed their own PCBs, and then made one of the instructor's design.



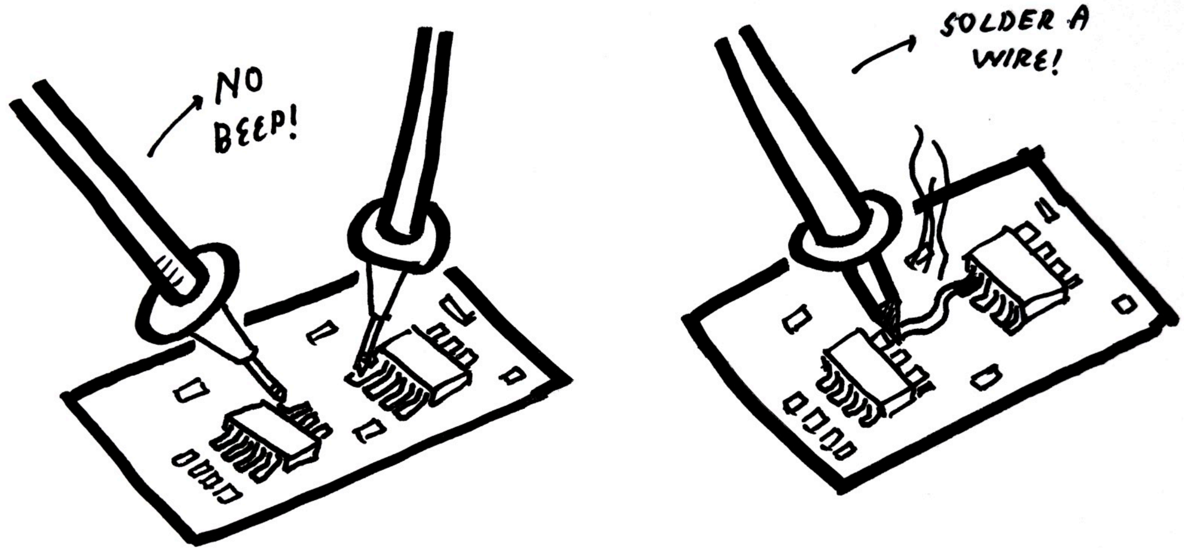


F.2.9—LEE ZAMIR HAD THE IDEA TO DISCONNECT THE 12V POWER BUS FROM THE DESIGN AND HAVE THE STUDENTS 'FIND' THE PROBLEM. FISCHER IMPLEMENTED IT ON HIS BOARD DESIGN WITH JUMPER PADS FOR EASY REPAIR.

What we were trying to teach them was the art of systematic board bring-up. In PCB design you often populate things little at a time, testing discrete parts of the circuit in isolation, ensuring they work, before connecting things together. This is especially true with power converters, you don't want to explode anything on your board by giving it too much voltage, and you don't want to hurt yourself either. We (being the instructors) had learned this lesson the hard way. But in the course it wasn't practical to keep supplying chips as the students exploded them, and it also wasn't tractable for us to debug the board designs of 60 individual students. So the first compromise was that they would debug a board we made.

The second compromise was we had to make something our board not work. A tiny error was introduced in the schematic (the renaming of two net labels that connected the output of the main power converter to the audio-amplifier on the board, one net was labeled +12 and the other 12V, since these software generally use the names to define connectivity, these two were no longer connected).

The idea was that we'd tell the students to test the components in order, starting with the power converters, then digital signals, and so on. This was to counter the student's tendency to want to populate and test the entire board at once (a common desire resulting from the excitement of building your first PCB, I've done it, I've regretted it). We wanted to show them this was a bad idea, but we didn't tell them there was a problem. We also had provided schematics and told the students *if* there was a problem these *might* be a good reference.



F.2.10—BRIEF CARTOON OF PROBING A PCB AND BRIDGING UN-CONNECTED POINTS WITH A JUMPER WIRE.

Our expectation was that the careful student would probe the 12V bus, see it not working, go back to the schematic to check for continuity (or run some other test and see it wasn't connected), fix the problem with a bridging wire, and move on. The not-so-careful student wouldn't realize this until they'd populated everything and no sound would come out of their bluetooth speaker. Debugging would take more time as the number of 'unknowns' would be higher because individual circuit components hadn't been tested before hand. A few details, the board has a 12V and 5V power bus. The 5V powers the microcontroller. So parts of the board would be working, and others wouldn't.

In figuring out if this worked or not, I think that time was not a reasonable indicator. Wether students spent more or less time in getting from a blank board to a working board had far too many confounding variables for me to develop any reasonable correlation between the exercise specifically and the time they spent debugging their board. We did observe more 'struggle' in students who approached the problem less systematically (in that they were more results-oriented than process oriented). Again, this may not be a direct result of the exercise, but it's indicative that a systematic approach on their part may have been helpful.

We additionally did observe that the exercise taught the students to observe the schematic they were handed more closely. This is purely based on their reactions to observing the different on their own, or it being indicated to them by an instructor. Though it would be worth testing their long-term retention after a few months in that topic.

Where the most learning happened, at least that I can tell through observation, was by the students observing *each other's* process in debugging and bring-up. The class was intended to be collaborative to some extent, partially due to our own concerns about the number of staff we had on hand. And students seated close-together in the lab environment seemed to encourage some level of talking (which was purely un-intentional on our part but had a nice effect). They did not *seem* intimidated but it is largely difficult to tell such things without spending more time with the students one-on-one (which is something we all agreed we'd like to improve upon for future iterations of the course).

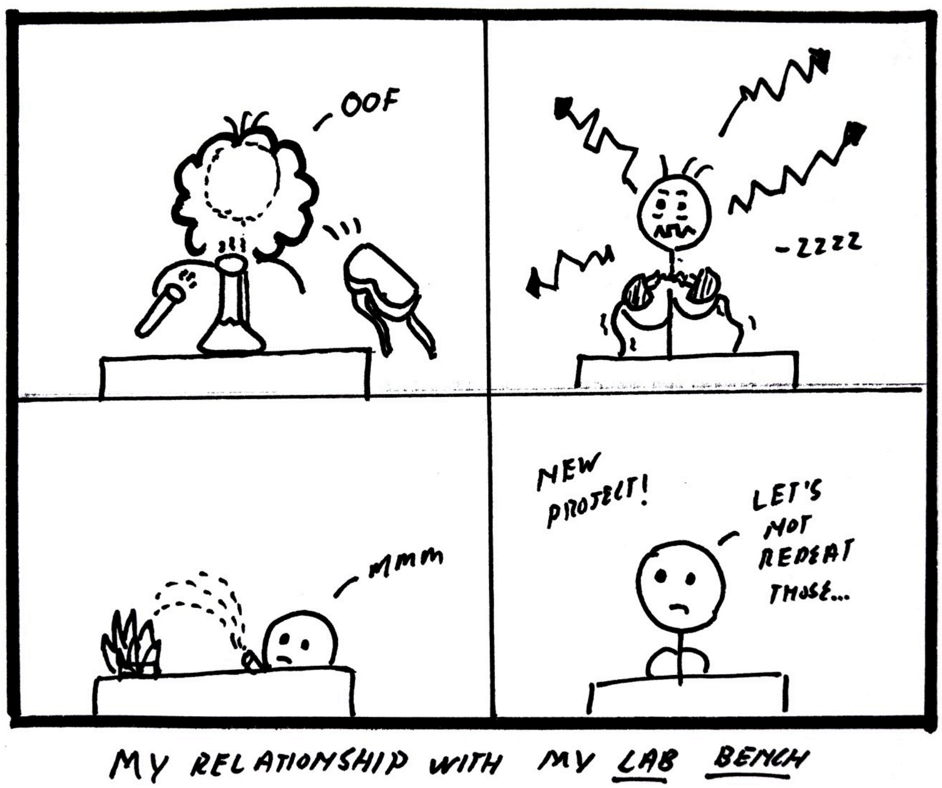
Regardless, students seemed perceptive to what others were doing if it was 'working better' than what they were doing, and sought to learn from it. They'd ask each other questions, or they would observe and then come ask the staff questions. And this exercise was one of the notable discussions points between students and students, and students and instructors (though I think it's important to note that in some cases the discussion involved telling each other the answer). I think that not telling the student there was a problem, and having them go through the process of discovering it was key to this. We observed brief confusion even in those who approached the debugging systematically, and that triggered a self-check on their understanding that was somewhat visible. To both side, the exercise at the very least seemed to *suggest* the utility of systematic debugging, or demonstrate what it meant to us as instructors.

Debugging as a Medium for Teaching

Both of the above examples I bring up are from debugging electrical hardware. Any EE will tell you that debugging is both a required skill, and a source of formative memories (all of us have set something on fire at least once, and once would be impressive). These formative memories stay with us for a while, and frankly it becomes hard to forget what the problem was even if you don't exactly remember how you solved it (I say this from experience).

I think this is due to a few reasons. First, electrical hardware is particularly unforgiving. The differences between it working and not working are very subtle, and often the symptoms of the problem are confusing and vague. Electrical systems don't scream 'the problem is here' at you the way some other systems might. On top of this you generally have to go through multiple 'layers' of the system before you find the root cause of the issue. Second, if you *don't* find the root cause, the problem will likely manifest itself again but in a new way. This is true with many engineering systems but electrical devices will exhibit wildly different symptoms for the same problem based on what you've changed. You can't really cheat. Third, you generally don't see the same problem multiple times because you usually strive to correct your past mistakes during the design phase.

This makes debugging a deeply active process, it's very engaging, and satisfying when you find the issue (though sometimes miserable during the process). You must retrieve information from your brain. You have to design and apply experiments while understanding the limitations of the equipment you have in front of you, and be able to determine wether you can trust the readings you are getting. You have to analyze there results based on your own mental models, apply the corrections you think should be applied, and then see if they worked. And if they didn't you have to do it again. Generally, you go back and forth many times before coming to even half of the answer. There are clear parallels to the active learning processes we described earlier.



F.2.11—A LIBRARY OF IMAGES OF PREVIOUS FAILURES AND EXPERIENCES IS THE BEGINNINGS OF ENGINEERING INTUITION.

What debugging provides you with, however, is a sort of library of images in the pack of your mind. This is both in the form of a list of problems you've dealt with, and sometimes vivid images of previous failures (such as a fire). These images are somewhat burned into the back of your memory because of the amount of active learning that was put into arriving at them.

When it comes to designing a new system, you often reference these images and seek to avoid them, your design decisions are affected by them. When it comes to debugging a new problem, you also draw from these images but this time in two ways. First, you attempt to connect what you're seeing to any of these previous images hoping for similarity.

Second, these images are also mini-movies, they're not still frames. You don't just remember the problem, you can kind of recall the steps you took, the process that led you to the conclusion. In debugging, you reference these as well. The tests you devised previously could be useful for the current problem, tests that were previously fruitless might be fruitful, or might be avoidable and might save you time. You're connect what you *see* to what you *saw*.

This, in my mind, is what we refer to when we say engineering 'intuition.'

Exercises for Teaching

Earlier I discussed motivations for teaching the specific topics covered in the exercises below. There are a few motivations for creating hands-on design exercises specifically.

For one, these topics (embedded systems, modeling and simulation, actuation, and power electronics) are often better learned through practical experience. You really have to struggle with the hardware for a bit to learn its limits and quirks. And while theoretical understanding *does* translate well, some of the implementation details translate less well which leads the engineer to necessarily pour over pages of documentation looking for specific 'tid-bits' of knowledge which might lead to an epiphany. This is a skill that lecture-based learning has a hard time teaching. I'm in no position to say this with any form of certainty but I believe this is a reason they're often not a focus of the core requirements of undergraduate electrical engineering, or mechatronics curriculum and generally left to industry or internships (as we noted earlier they are not as niche of fields as we sometimes would expect).

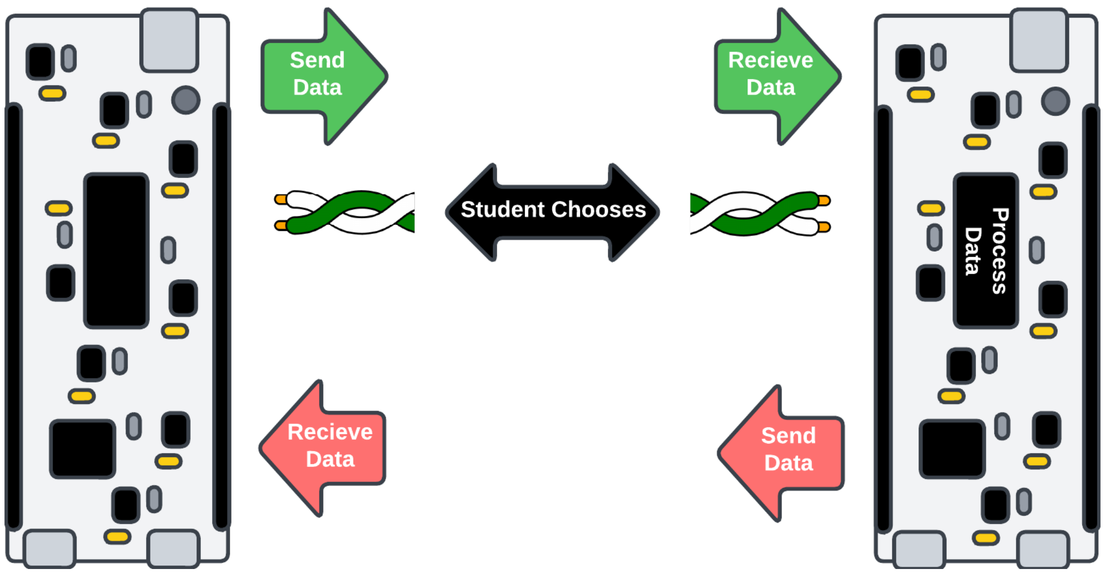
The second reason is to address the varying opinions of people I spoke to on the idea of *coverage* versus teaching *methodology* and what to focus on. There are varying opinions on this, and presenting them in detail is not so useful here, rather I'll say both are important. If we aren't teaching the topics that our students need to know, or we aren't providing the right opportunities to learn such content at the right entry-levels then we cannot possible expect them to know that material (though some may pick them up through other means). But it is also true that the method of teaching must match the topic, and exploring one without the other would be potentially interesting but may lack sufficient information to make it implementable. To that end, my goal was to *suggest* some possible methods for teaching these topics.

The last reason would be, if I had more data for it, to explore the translation of our loose design paradigm from concept into reality. Ideally, I would have had the time to implement all of the exercises in a classroom setting, test them, and document my observations. This proved beyond the scope of this thesis but I hope to do this work in the future and update accordingly. It is important to note however that we have some data. All the exercises are modified or constrained version of real problems we've observed students work on in various educational settings and are just re-worked, and more clearly defined for the classroom (so they're not arbitrary). And our observations of teaching with those do translate, and in that sense the exercises below are a second attempt.

Embedded Systems and Control

<u>E.1—A Simple Exercise in Communications:</u> students are asked to take two STM32 microcontrollers and wire either a SPI, CAN, I2C, UART, or similar hardware communications interface of their choice between them. They are then asked to send a randomly generated 32 bit number (generated by course staff) from the 'input' microcontroller to the 'processing' microcontroller. The 'processing' microcontroller has to perform a mathematical operation on the data (a divide by 2, or add 1 is more than sufficient), and send it back. This operation is also of student choice. The challenge is to do this as quickly as possible without distorting any data. Staff is responsible for measuring the timing which includes data transmission, processing, and data return.

The key design decisions in the above problem include the choice of communications interface, the clock speed of said communications interface, and the arithmetic operation performed by the secondary controller. All of these have significant affects on the timing of the system. Tradeoffs include, the faster the communications the more likely it is that the signal will be distorted, and no EMI shielding is allowed for the purposes of this exercise. Student do learn about the noted topics in class, but must also consult component data sheets to determine exact timing parameters. Data structure and memory allocation choices may also affect speed.



F.2.12—MICROCONTROLLER COMMUNICATIONS EXERCISE. STUDENTS MUST SEND A 32-BIT NUMBER FROM ONE MICROCONTROLLER TO THE OTHER, PROCESS THE DATA, AND RETURN IT AS QUICK AS POSSIBLE.

The idea for this exercise actually came from two different places. First, was my own work in lab where in the process of debugging the communications system for a piece of test equipment we wrote some code to send data back to a microcontroller, have it add 1, and have it send it back to test the robustness of the communication protocol at high-speeds over a long stretch of time. This proved to be a very useful technique and it made it far simpler to debug what was exactly wrong with the interface (separating code complexity from actual hardware issues). The second was from MIT's Electric Vehicle Team, where we had first-year students link to STM32s together to 'talk' over CAN bus as an introductory exercise to programming embedded systems. The speed optimization was omitted in this exercise.

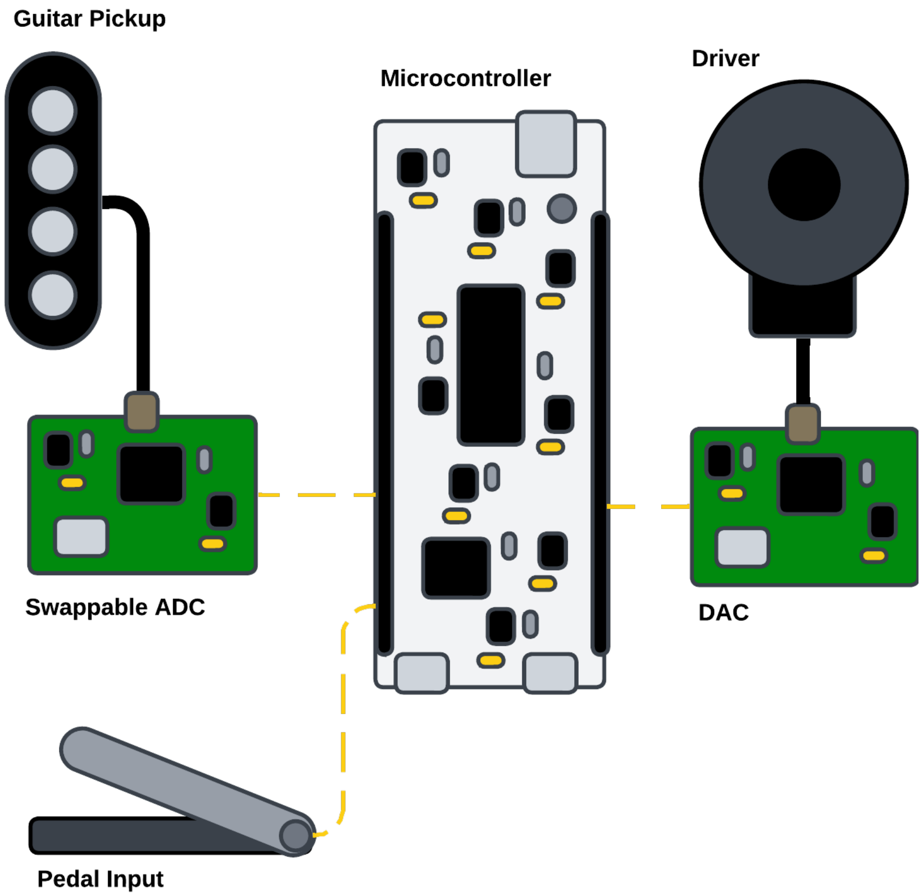
The thing I liked about this exercise in both cases was the little things it taught you about microcontrollers. On its own, teaching pin-setup, clock setup, and other 'configuration' settings in a microcontroller is a little boring. It's also sometimes difficult. With simpler functions such as 'Analog Input' and 'Print to Serial' the setup is fairly straightforward for most microcontrollers. In the case of a communications system, it's less so. Which makes the act of setup an important part of the exercise. That's something you only get good with through practice.

I also liked the use of digital signal measurement such as logic analyzers to test the output of the communications system. It gives students both hands-on experience with the equipment (and plenty of practice as debugging communications tends to be a lengthy process), but also provides them with a visual of what's actually happening on the hardware. I've observed that seeing the bits is very useful for learning how these communications systems work.

Finally, the exercise provides some initial intuition to the student on the concept of time with communications. Communications timing very much affects control bandwidth, and in hierarchical controllers this timing is critical.

<u>E.2—Electric Guitar Distortion Pedal:</u> students are asked to read the analog signal from a guitar pick-up and read a desired 'distortion' level from a potentiometer. They must then write software to perform signal-clipping proportional to the value of read on the potentiometer, leaving the signal un-distorted if the threshold is below a certain value. They must then send this signal to an ACD/Amplifier. The signal is then output to a speaker. They run this code both on their staff-provided hardware setup (which has an output ADC and input

DAC both with 24 bit-depth), and additionally on a staff setup with a 12bit, 16bit, 24bit, and 32bit 'swappable' ADC and are asked to comment on any differences.



F.2.13—STUDENTS USE A MICROCONTROLLER TO READ A GUITAR PICKUP SIGNAL FROM A DAC. THEY PROCESS THE SIGNAL WITH A DISTORTION PROPORTIONAL TO THE PRESS OF A PEDAL INPUT AND SEND THE SIGNAL TO A DAC AND AMPLIFIER COMBINATION WHICH DRIVES A SPEAKER.

This exercise gives the student the ability to practice digital signal processing, digital to analog conversion, analog to digital conversion, and sampling. One design decision (or at least a hidden design decision) in this exercise is the main loop-speed, or sampling frequency of the microcontroller from the guitar pickup.

Standard digital sampling for music is 44.1 kHz which is a little higher than the nyquist frequency of human hearing (the highest frequency we can hear is 20kHz) [35]. There are a number of other sampling standards including 48kHz, 88.2kHz, 96kHz, and 192kHz [35]. Naturally, the higher the sampling rate the more data will be generated (and then will need to be stored). At 441.kHz the data rate is around 1KBit/s for a 44.1kHz sampling frequency. But we leave this as something for the student to 'choose' as a tradeoff between exercise difficulty and performance of the final pedal.

The idea for this exercise actually came from a conversation we had amongst instructors in the PCB design course [34]. Fischer had spend a good amount of time testing various ADCs for the bluetooth speaker we were asking the students to design. One evening as we were testing various components as a staff, and we started to wonder based on the layout design if the students might be able to *hear* flaws in their PCB designs if we were to manufacture all of their boards. This conversation came after Fischer made the decision to place a shielding pour on the staff board as a faraday cage to reduce noise.

Because we never ended up fabricating individual students' boards we never found out, but the idea for having them *hear* the issue was a good one in terms of grounding their learning.

E.3—Magnetic Material BH Curve Measurement Device: this exercise was inspired by an online video from the Electromagnetic Fields and Energy course by Markus Zahn, James R. Melcher, Manuel L. Silva [36]. In the demo they show a simple Variac®-controlled BH curve device with key components shown in Figure 2.14. The Variac® drops the 120V 60Hz wall outlet voltage to a safe input still oscillating at 60Hz, and when applied to a primary coil of N_1 turns (wrapped around a toroid of material for measurement), this produces a sine-wave of current as well. This current is measured by a shunt resistor to the input of the 'x' channel of an oscilloscope.

$$H_\phi = \frac{Ni}{2\pi R}$$

(derived from Ampere's Law)

Because the H field in the core is directly proportional to the current in the windings, measuring the current gives us a representative measurement of the *magnetic field strength*.

To measure flux density, a coil of N_2 turns is wrapped on the opposing side of the coil. We know that the total flux linkage through the secondary coil is proportional to the flux density, the area, and the number of turns.

$$\lambda_2 = B\pi r^2 N_2 = \Phi N_2$$

$$\mathscr{E}_{coil} = -N_2 \frac{d\Phi}{dt}$$

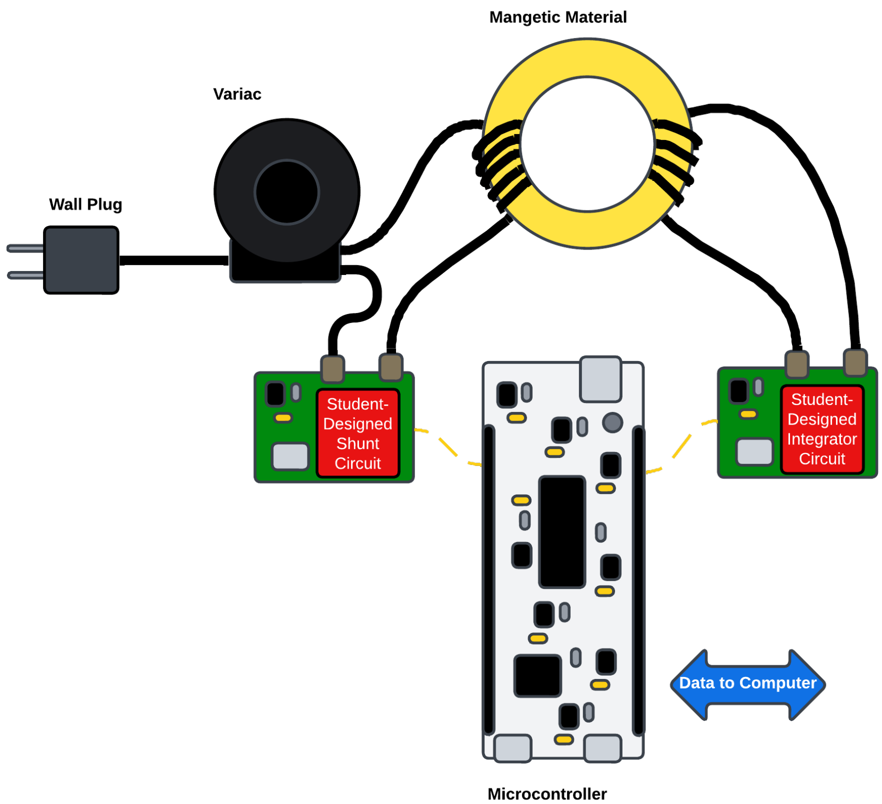
(Faraday's Law)

Assuming *no* current flows through the secondary coil, which we can achieve by terminating it in a high enough impedance. The integrated voltage across the coil is directly proportional to the B field. The voltage on the output of the secondary coil is read and passed through an RC integrating network before being plotted on the 'y' channel of the oscilloscope.

For the modification of this demo into an exercise students are asked to design the shunt resistor and RC output filter given basic circuit parameters including the expected voltage, current, frequency, and number of turns on the primary and secondary windings. Additionally, instead of passing the outputs to be plotted in X/Y display mode on an oscilloscope, they are asked to sample this data to a microcontroller and send the data to a their laptops to be saved as a (*.csv) file over serial. Of course the student can test the operation of a circuit using the scope, and compare that to the output from their microcontroller which is, in general, a good habit to get into in terms of the component-by-component testing we referred to earlier. They are also asked to perform an error analysis between their BH curve and a 'ground-truth' BH curve of the material read by an industrial measurement device.

The key design decisions in the case of the problem above include the value and power rating of the shunt resistor and filter resistor, as well as the capacitance and voltage rating of the capacitor, as well as the sampling frequency. The student must also decide wether to 'stream' the data to the computer and actively save it one-line-at-a-time or hold the data in a buffer and send a single data transfer at the end of a set 'measurement time.' The former may slow down the sampling, but avoid the issue of potentially running out of memory on the microcontroller, but the latter gives the student the chance to specify the maximum measurement time based on the amount of memory the microcontroller has, and to apply a significant safety factor.

I think there's two key advantages to this exercise. The first is the requirement to select *real* components. This is something the student often does not have the chance to do in introductory circuit classes which is to build intuition on the values of a component past a number on a data sheet. It also allows them the experience of having to adjust the values of their selection based on availability of component, and forces them to ask the questions on the necessary tolerances on those component values. Because the system is simple enough, they can test multiple, and check if their intuition holds.



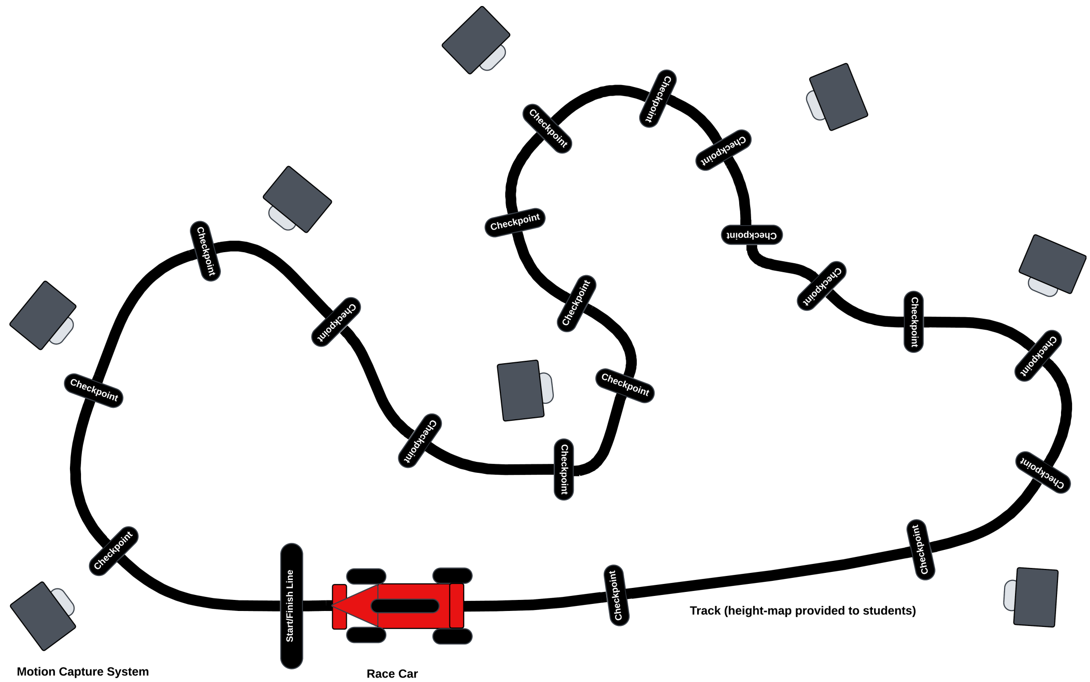
F.2.14—STUDENTS ARE ASKED TO DESIGN THE CURRENT SHUNT, AND RC INTEGRATOR CIRCUIT FOR A BH CURVE MEASUREMENT DEVICE. THEY ARE ASKED TO SELECT SAMPLING RATE, AND SEND THE RESULTANT DATA AS A (*.CSV) FILE TO A COMPUTER.

The second key advantage is a deeper understanding of magnetics and magnetic materials (or at the very least the relation between B and H). Magnetics is hard to visualize and intuitively understand to some extent. The above exercise produces help students link the concepts of magnetic flux, magnetic flux density, and magnetic field strength using direct implementation of Ampere's law and Faraday's laws. They also get to do this in a simple magnetic circuit, and may other magnetic circuits (such as motors) are far more complex and more difficult to conceptualize. Another example of a simple magnetic circuit (perhaps one of my favorites) is presented in the next chapter.

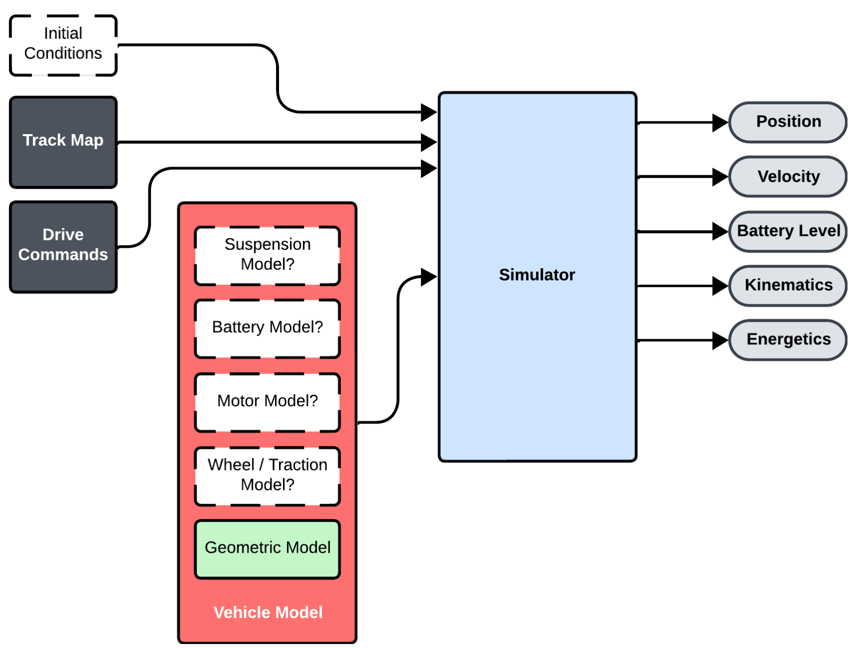
Modeling and Simulation

To teach modeling and simulation, I decided to start with a semester-long project that gets broken down into a number of sub-exercises. This was so that students could practice both the simpler modeling of individual component in a complex system, but also combine these models later to model a complex dynamic system. I start by describing the over-all project, and the breakdown into certain sub-components for modeling.

The project for this course starts with an RC car with a passive suspension system. The car is given control commands to drive around an obstacle course setup by staff in the real world. The course is equipped with a motion capture system that accurate measures Euler angles, velocity, acceleration, angular velocity, and angular acceleration of the vehicle body. Suspension compression is measured by a linear encoder on each wheel. Additional sensors read wheel speed, motor speed, battery voltage, battery current, and steering angle.



F.2.15—AN RC CAR IS RACED AROUND A PRE-DEFINED TRACK AND GROUND-TRUTH DATA IS CAPTURES ABOUT POSITION, AS WELL AS BODY ROLL, SUSPENSION COMPRESSION, VOLTAGE, CURRENT, AND VEHICLE SPEED. STUDENTS MEASURE KEY MODELS TO FIT INTO A SIMULATION FRAMEWORK TO ACCURATELY SIMULATE THE RECORDED REAL-WORLD BEHAVIOR.

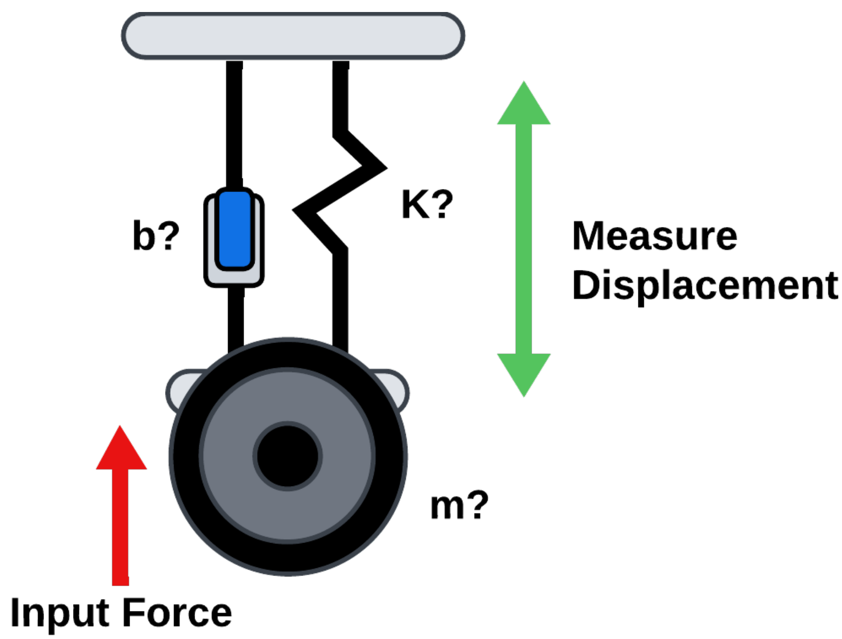


F.2.16—STUDENTS ARE PROVIDED WITH THE SIMULATOR IN MATLAB SIMULINK. A GENERAL VEHICLE MODEL IS PROVIDED BUT STUDENTS MUST MEASURE AND IMPLEMENT KEY SUB-MODELS.

Students are given a height-map of the track, control inputs, and certain ground-truth outputs. Students are asked to develop models of key components of the system to be implemented into a simulation framework. The output of the students' final simulation is graded on accuracy and reputability on the ground-truth data. Students are asked to predict the range of the vehicle, and the energy lost due to wheel slip.

The inspiration for this came from two sources. First, is the 6.141 Robotics Science and System class at MIT where students have to implement autonomous driving algorithms for a race-car to achieve the fastest lap time on a defined course [37]. The simulator is *provided* in the case of this course, we were curious to experiment with the idea of having the students implement the key components of the simulator. The second was from the discussion on simulation and model insertion presented in the previous chapter.

M.1—Suspension Model Measurement: In the first lab, students would devise an experiment to measure the parameters of a second order model for their suspension taking the form $F = m\ddot{x} + b\ddot{x} + Kx$. There's number of ways to do this. Students could measure the parameters individually by dis-assembling the suspension system and characterizing each component. Another interesting method would be to measure the bode plot of the system by injecting sine-waves of position at the wheel and measuring the position response of the suspension's connection to the 'chassis' with a known mass load.



F.2.17—STUDENTS ARE ASKED TO MEASURE THE CHARACTERISTIC PARAMETERS OF A SECOND-ORDER SPRING-MASS-DAMPER SYSTEM (THEIR SUSPENSION!) THEY WILL FIT THIS MODEL INTO THE SIMULATION FRAMEWORK PRESENTED IN FIGURE 2.16.

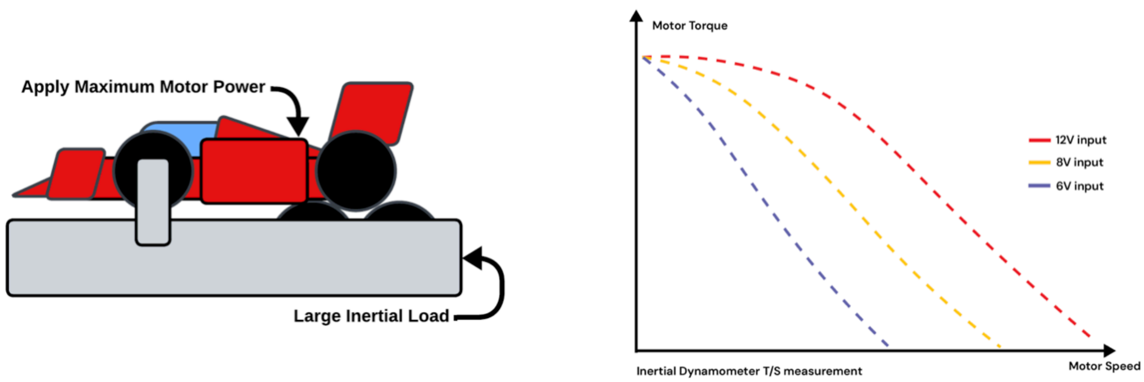
M.2—Motor Torque-Speed Measurement: In the second lab, students would measure the torque-speed curve of their motor and the efficiency of their drive-train. They would place their cars on an inertia dyno which could be bought or staff-built [38]. By applying a large enough inertia to the wheels and commanding full-power, the motor will accelerate along its operation limits according to the governing dynamics equations.

$$V = iR + L\frac{di}{dt} + k_{BEMF}\frac{d\theta}{dt}$$

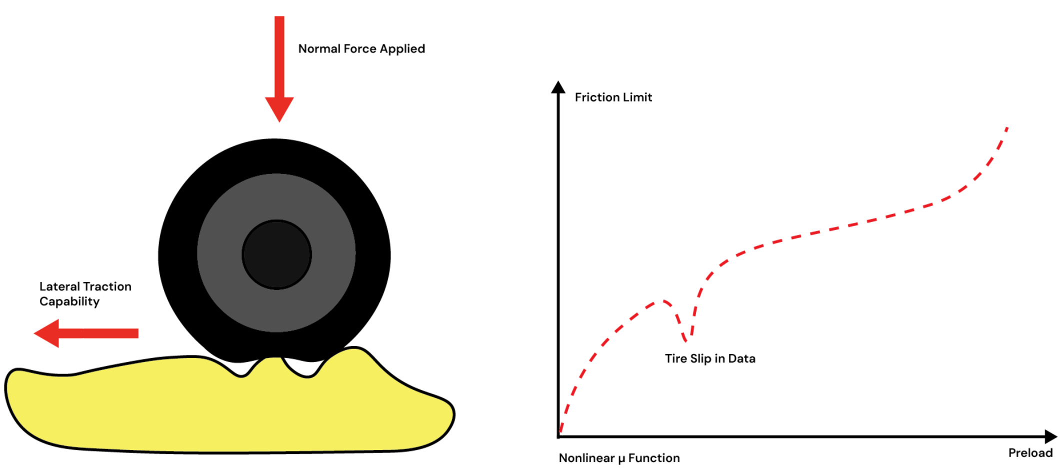
$$k_\tau i - k_d\frac{d\theta}{dt} = J\frac{d^2\theta}{dt^2}$$

This will enable us to measure the performance limits. For efficiency, the voltage and current input to the motor controller can be measured. This experiment can be repeated in a series of increasing voltage steps from 0V to the maximum battery voltage to plot a series of lines in the T/S plane to get the motor performance over the entire range of operation.

This is a modified exercise from a specific team project in 2.74 [39]. Students looking to measure the torque-speed curve of the motor attached a large weight to it and applied the full 12V the motor system was designed for. Efficient was not measured, but a clear graph of the motor operational limits was developed and students used this in their simulation framework for one of their projects. Watching them do this inspired this exercise.



F.2.18—STUDENTS USE AN INERTIA DYNAMOMETER TO MEASURE THE T/S CURVES OF THEIR MOTORS. THE DECREASING VOLTAGE MEANS STUDENTS CAN GET THE LIMITING OPERATION REGION, AS WELL AS A DATA WITHIN THE OPERATIONAL REGION.



F.2.19—STUDENTS ARE ASKED TO DEVISE EXPERIMENTS TO MEASURE THE TRACTION OF THEIR WHEEL, AND TO FIT A FUNCTION FOR (MU) AS A FUNCTION OF PRELOAD.

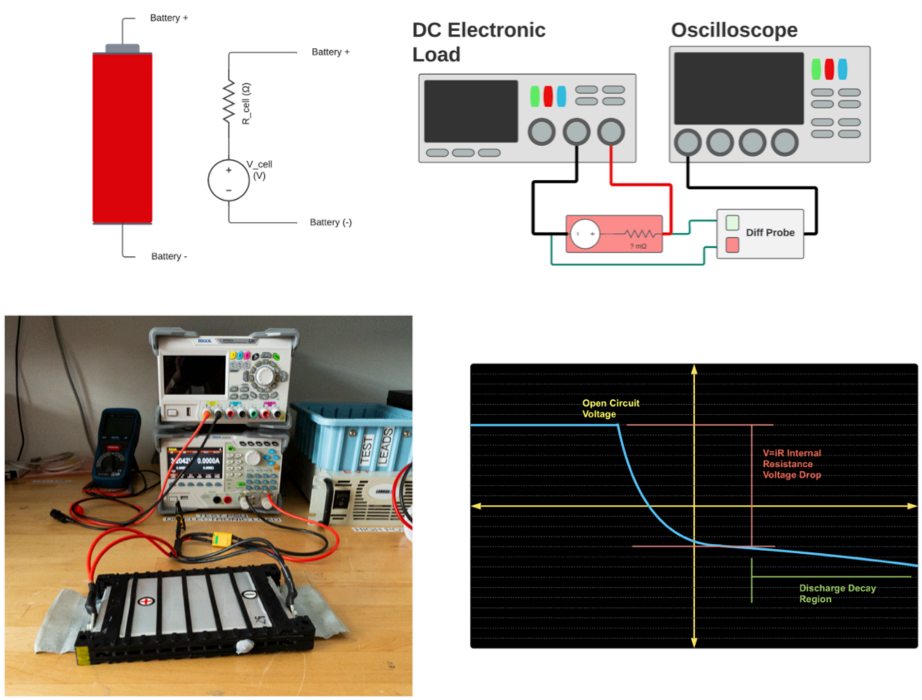
M.3—Wheel Traction Characterization: We'd like to have the students estimate wheel traction and slip. Traction is made of up both friction, which assumes rigid bodies and smooth surfaces, and 'grip' which comes from the wheel or medium conforming to the tire increasing lateral force capability and effective coefficient of friction. This is why when you let the air out of your tires, you see an effective increase in friction, an effect that one of my friends refers to as *statistical traction*. The goal here is to help students with the common misconception that friction is *area-dependent*.

This is a common confusion we see in the 2.007 course at MIT [20]. Students would measure tire traction capabilities on the track medium and be asked to diagram and explain the difference between various soft and hard tires. They then estimate the effective coefficient of friction *as a function of normal tire force* according to the following formula.

$$F_f = \mu(F_N)F_N$$

Where, in this case, the coefficient of friction is a *function* of the normal force due to the conformity of the tire and the surface.

M.4—Battery Modeling: The last exercise would focus on battery modeling. With the goal of having the student estimate the range of the vehicle. Students would use a DC electronic load to measure the parameters of a simple DCIR model. They would then also perform a State-of-Charge characterization (SoC) measurement on the cell to measure the total capacitor, and the discharge curve. This curve would enable them to estimate the total energy stored in the battery, and the remaining range in real-time.



F.2.20—STUDENTS USE AN OSCILLOSCOPE AND DC ELECTRONIC LOAD TO PERFORM AN EXPERIMENTS TO DETERMINE THE INTERNAL RESISTANCE OF A BATTERY CELL FOR THERMAL MODELING. IN ADDITION, THEY USE THE LOAD TO COLLECT STATE-OF-CHARGE CHARACTERIZATION CURVES LIKE THE ONES PRESENTED IN THE PREVIOUS CHAPTER.

We've done this exercise with students multiple times on both the Solar Car and Electric Vehicle teams, in research opportunities, and other outside-classroom settings to great effect. Students exhibit a far deeper understanding of batteries after completing these initial exercises on their modeling limitations. Prior to state-of-charge or internal resistance modeling, most students exhibit the *ideal-voltage-source* model of a battery.

I think the key advantage in all of these exercises is the ability to practice devising experiments to measure and fit parameters to known models of simple systems. In addition, students get the chance to practice rigorously collecting data with as little noise, and as high an accuracy as possible because *not* doing so would significantly affect their modeling results. Another thing I like about this is if students are given multiple cars across teams, for example, we can plot the differences in their models at the end of the class and we can use one team's model on another team's data to show the divergence of the simulation. This helps reinforce the idea that even the *same* system will be slightly different, and the slight difference will be significantly amplified in the output.

Additional modeling exercises could include Ackerman steering modeling, and inertial parameter estimation. Some ideas are included in Appendix C for these.

Counterarguments and Scaling

There's somewhat of a risk in teaching in the way I described above which is in making the problem *too* open-ended. So much so that the student doesn't know where to begin, or where the finish line is. In designing the exercises we have to be careful to make the problem *just open-ended* enough that the student has the freedom to experiment, but we also need to ensure they have the background knowledge required to solve the given problem, or the ability to find the required information within some reasonable time and effort.

This is where lecture, readings, and external resources do very much have their place in teaching engineering—I don't think it's reasonable to say they don't or that lecture in itself is a poor method for teaching. The lectures provide any background knowledge required, the readings (which could even be in the form of data sheets) contain information that don't make sense to be presented in lecture that is very much required for the lab exercises. But the *learning* happens in lab, where the theoretical gets applied to the practical. This is very much the paradigm of *flipped classroom* that's becoming increasingly popular in teaching, and MIT's 2.008 Design for Manufacturing II course uses this to great effect [40].

I want to note two things without diving into them into too much detail. It is *possible* to get robust content retention using lecture methods, just like it's possible to get *little* content retention from a poorly designed lab.

An example on the side of how these techniques can be applied to increase lecture's effectiveness can be seen in *Make it Stick* on page 37 (the *testing* effect, as they call it).

An example on the lab side that I think is illustrative, I remember from a course that will remain un-named was an exercise where we were asked to compute the LQR gains for a pendulum swing-up controller. In the lab, most of the work was formatting the dynamic equations into matrices, but at the end we simply called a staff-written function called, for example, *compute_LQR_gains()*. From what I remember, I now know a lot about numpy and linear algebra, but I still don't know how to computer an LQR gain. But it's not my place to argue if the staff believed that the matrix manipulation was more important than the computation of the gains themselves. And if that was the case, then the exercise was well designed.

That does bring up another point that I expand upon in the next chapter. The student need to *know* the intent of the exercise, Make it Stick makes this point as well [9]. Sometimes, I have noticed (and others have written about) that we tend to *hide* our reasonings on our teaching decisions from the student, or sometimes tend to assume they realize our intent more than we let on. There's a number of sources that have indicated that being more explicit in how what we ask the student to do ties into learning objectives leads to increased effectiveness of the teaching [9].

In terms of scaling this methodology beyond teaching hardware design, this is an area in which I have less experience. Of course it is possible, *Make it Stick* largely talks about robust learning with examples other than teaching engineering. This makes me think we can scale these ideas to *any* subject of teaching. I can offer some examples on how you might do this. I present a few examples of these for the subjects of optimization and learning in Appendix C.

References

1. C. M. Miller, “So can you build one? : learning through designing--connecting theory with hardware in engineering education,” PhD Thesis, Massachusetts Institute of Technology. Department of Mechanical Engineering, 1995. Available: https://dspace.mit.edu/handle/1721.1/11548

2. K. A. Lilienkamp, “Lab experiences for teaching undergraduate dynamics,” PhD Thesis, Massachusetts Institute of Technology. Department of Mechanical Engineering, 2003. Available: https://dspace.mit.edu/handle/1721.1/17007

3. J. Clement, “Students’ Preconceptions in Introductory Mechanics,” American Journal of Physics, Volume 5Q(1) pp. 66-71, 1982.

4. A. diSessa, “Unlearning Aristotelian Physics: A Study of Knowledge-Based Learning,” Cognitive Science, Volume 6 pp. 37-75, 1982.

5. A. diSessa, “Knowledge in Pieces,” Ir G. Forman and P. Putall (Eds.), Constructivism in the ComputerAge, Hillsdale, N. J.: Erlbaum, 1988.

6. D. L. Trumper, 2.14 Feedback Control Systems, Department of Mechanical Engineering, Massachusetts Institute of Technology.

7. D. L. Trumper, 2.737 Mechatronics, Department of Mechanical Engineering, Massachusetts Institute of Technology.

8. D. L. Trumper, 2.171 Digital Controls, Massachusetts Institute of Technology.

9. P. C. Brown, H. L. Roediger III, and M. A. McDaniel, Make it stick: the science of successful learning. Harvard University Press, 2014.

10. A. Bagiati and S. E. Sarma, “Merging STEM education with brain science: Breaking the silo mentality,” SEFI Annual Conference. Jan. 2018. Open Access Link: https://dspace.mit.edu/handle/1721.1/129429

11. A. Johri and B. M. Olds, “Situated Engineering Learning: Bridging Engineering Education Research and the Learning Sciences,” Journal of Engineering Education, vol. 100, no. 1, pp. 151–185, Jan. 2011, doi: https://doi.org/10.1002/j.2168-9830.2011.tb00007.x.

12. E. F. Crawley, Johan Malmqvist, Sören Östlund, D. R. Brodeur, and K. Edström, Rethinking engineering education the CDIO approach. Cham Switzerland: Springer, 2014. Open Access: https://link.springer.com/content/pdf/10.1007/978-3-319-05561-9.pdf

13. M. Borrego and J. Bernhard, “The Emergence of Engineering Education Research as an Internationally Connected Field of Inquiry,” Journal of Engineering Education, vol. 100, no. 1, pp. 14–47, Jan. 2011, doi: https://doi.org/10.1002/j.2168-9830.2011.tb00003.x.

14. Barrow-Green, June (2008). "The Three-Body Problem". In Gowers, Timothy; Barrow-Green, June; Leader, Imre (eds.). The Princeton Companion to Mathematics. Princeton University Press. pp. 726–728.

15. J. Barrow-Green, Poincaré and the three body problem. Providence, Ri: American Mathematical Society ; London, 1997. https://www.wolframscience.com/reference/notes/972d/

16. R. Montgomery, “The Three-Body Problem,” Scientific American, Aug. 01, 2019. https://www.scientificamerican.com/article/the-three-body-problem/

17. L. Cixin, Remembrance of Earth’s Past Trilogy. 2014-2019.

18. L. Cixin, “The Three-Body Problem,” Remembrance of Earth’s Past Trilogy, 2008.

19.I. C. Jong, “Teaching Students Work and Virtual Work Method in Statics: A Guided Strategy with Illustrative Examples.” University of Arkansas. Available: https://icjong.hosted.uark.edu/docu/05aseePortland.pdf

20.T. Urban. “Inside the Mind of a Master Procrastinator,” TED. Apr. 2016. Available: www.youtube.com/watch?v=arj7oStGLkU

21.2.007 Design for Manufacturing I, Department of Mechanical Engineering, Massachusetts Institute of Technology.

22.“Co-op Participation - Career Outcomes at Northeastern University.” Northeastern University College of Engineering. Available: https://careeroutcomes.northeastern.edu/coop-participation/

23.“Undergraduate Co-op,” Northeastern University College of Engineering. Available: https://coe.northeastern.edu/academics-experiential-learning/co-op-experiential-learning/co-op/undergraduate-co-op/#:~:text=Co%2Dop%20is%20central%20to

24.“10 Reasons Why You Should Pursue a Co-op Experience at Tufts Gordon Institute | Gordon Institute,” Gordon Institute at Tufts University. Available: https://gordon.tufts.edu/news-events/news/10-reasons-why-you-should-pursue-co-op-experience-tufts-gordon-institute

25.“About Co-op Programs | Wentworth,” Wentworth Institute of Technology. Available: https://wit.edu/co-ops-careers/about-co-ops

26.S. Wang, “Behind Northeastern’s Co-op Program,” Medium, Jul. 01, 2020. Available: https://medium.com/@serenawang0210/behind-northeasterns-co-op-program-f4265614127a

27.“Advantages of Co-op,” Engineering Professional Practice. Available: https://coop.utk.edu/students/coop/advantages

28.C. R. Noyes, J. Gordon, and J. Ludlum. “AC 2011-630: THE ACADEMIC EFFECTS OF COOPERATIVE EDUCA-TION EXPERIENCES: DOES CO-OP MAKE A DIFFERENCE IN ENGI-NEERING COURSEWORK?” ASEE PEER. Georgia Institute of Technology, 2011.

29.J. B. Main, B. N. Johnson, N. M. Ramirez, H. Ebrahiminejad, M. W. Ohland, and E. A. Groll, "A Case for Disaggregating Engineering Majors in Engineering Education Research: The Relationship between Co-Op Participation and Student Academic Outcomes,” International Journal of Engineering Education, Vol. 36, No. 1(a), pp. 170-185, 2020. Accessible: https://www.ijee.ie/1atestissues/Vol36-1A/14_ijee3865.pdf

30.N. M. Ramirez, J. B. Main, and M. W. Ohland, "Academic Outcomes of Cooperative Education Participation,” Purdue e-Pubs, 2015. Accessible: https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1069&context=enegs

31.T. L. Strayhorn and R. M. Johnson, “What Underrepresented Minority Engineering Majors Learn from Co-Ops & Internships,” ASEE International Forum, Jun. 2016. PaperID: #17553.

32.S. R. Brunhaver, R. F. Korte, S. R. Barley, and S. D. Sheppard, "Bridging the Gaps Between Engineering Education and Practice,” Eds. Richard Freeman and Hal Salzman, Engineering in a Global Economy, National Bureau of Economic Research. Available: https://www.nber.org/system/files/chapters/c12687/revisions/c12687.rev0.pdf

33.LTC6804-1/LTC6804-2 Multi-cell Battery Monitors. Linear Technology. Datasheet: https://www.analog.com/media/en/technical-documentation/data-sheets/680412fc.pdf

34.A. Mehrotra, F. Moseley, W. Sezto, J. Steinmeyer, W. Vu, and L. Zamir, “The Art + Science of PCB Design,” Massachusetts Institute of Technology. Course Page: https://pcb.mit.edu/

35.“Sample Rates and Audio Sampling: A Guide for Beginners,” Adobe. Accessible: https://www.adobe.com/uk/creativecloud/video/discover/audio-sampling.html

36.M. Zahn, J. R. Melcher, M. L. Silva, “9.4.1 Measurement of B-H Characteristic,” Electromagnetic Fields and Energy, Chapter 9: Magnetization, MIT Open Courseware, Jun. 2021. Accessible: www.youtube.com/watch?v=pXukVix5Pcw (but also include the original course)

37.N. Roy, D. Rus, and staff, 6.141 Robotics Science and Systems, Department of Electrical Engineering and Computer Science, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology.

38.Minipro Chassis Dyno for RC Cards. Accessible: https://www.minipro.com/products/chassis-dyno

39.S. Kim, 2.74/2.740 Bio-Inspired Robotics, Department of Mechanical Engineering, Massachusetts Institute of Technology.

40.2.008 Design and Manufacturing II, Department of Mechanical Engineering, Massachusetts Institute of Technology.

41.Y. Xie, "Mechatronics examples for teaching modeling, dynamics, and control," BS Thesis, Massachusetts Institute of Technology. Department of Mechanical Engineering, 2003. Available: https://dspace.mit.edu/handle/1721.1/29730

42.J. J. Maldonado, R. Garrido and G. Castro, "A Methodology to Teach Mechatronics through Building a Hands-on Platform," 2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico, 2018, pp. 1-6, doi: 10.1109/ICEEE.2018.8533992.

43.Miguel-Angel Martínez-Prado, Juvenal Rodríguez-Reséndiz, R. A. Gómez-Loenzo, K. A. Camarillo-Gómez, and G. Herrera-Ruiz, "Short informative title: Towards a new tendency in embedded systems in mechatronics for the engineering curricula," Computer Applications in Engineering Education, vol. 27, no. 3, pp. 603–614, Feb. 2019, doi: https://doi.org/10.1002/cae.22101.

44.B. Sunedal, Investigation into the pedagogical value of demonstrating deliberate design deficiencies in PCB design instruction. 2021. Accessed: May 21, 2024. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2:1601910&dswid=9061

45.G. L. Plett and M. D. Ciletti, "Piloting A Balanced Curriculum In Electrical Engineering Introduction To Robotics," Sep. 2020, doi: https://doi.org/10.18260/1-2--15500. Accessible: http://mocha-java.uccs.edu/dossier/RESEARCH/2005asee-.pdf

46.C. L. Magee, "Needs and Possibilities for Engineering Education: One Industrial/Academic Perspective." Jun, 2003. Accessible: https://dspace.mit.edu/bitstream/handle/1721.1/102907/esd-wp-2003-04.pdf?sequence=1&isAllowed=y

Chapter 3

Discussions

More than a Technical Understanding

In this final section, I want to touch on a few times that are important when it comes to teaching. Things I think we could be spending more time on especially in the sphere of engineering education.

In the curriculum presented in Appendix B, I purposefully left some space for what I refer to as 'elaboration' subjects, or 'beyond what you're supposed to know' subjects. There's a few reasons for this.

First, in terms of an ever-changing field like robotics. Small 6-unit classes (which are far easier to create and implement) mean its far less effort to be able to quickly fill gaps in the curriculum as the field changes. This allows us to give the student new experiences, and test wether this 'new' content should make it into the curriculum or not without a major overhaul. It also provide the opportunity to enable the student to do some career planning, by presenting applications of course work through case-studies, and bringing in individuals to talk about their career experiences, both of which have proved effective in the past.

The second reason, and the purpose of this chapter, is to give our student *more* than a technical understanding. Here's just a few things we could be using that space for.

The Power of Low-Stakes Curiosity

One of my favorite moments I've had while teaching came recently during lab time in the *D-Lab: Mobiles for Development* class we were running this past spring. A student and I were taking apart a Harbor Freight Predator gasoline engine for a project the student was doing using Arduino to monitor engine use which she and her partner predicted would let them notify the user when it was time to change the oil prevent damage.

The idea was to connect a little power converter to the alternator, so that when the engine turned on, the Arduino would turn on and start taking data. This was the preferred solution when compared to a battery or solar panel because it had less parts, and reduced cost which was critical to the partner the student were working with in Tanzania.

It had already been a fun day since the student I happened to be working with didn't have prior experience taking apart engines or using hand tools. She'd learned how to use a nut driver and ratchet and started disassembling the engine herself all the the short span of the lab time that day.

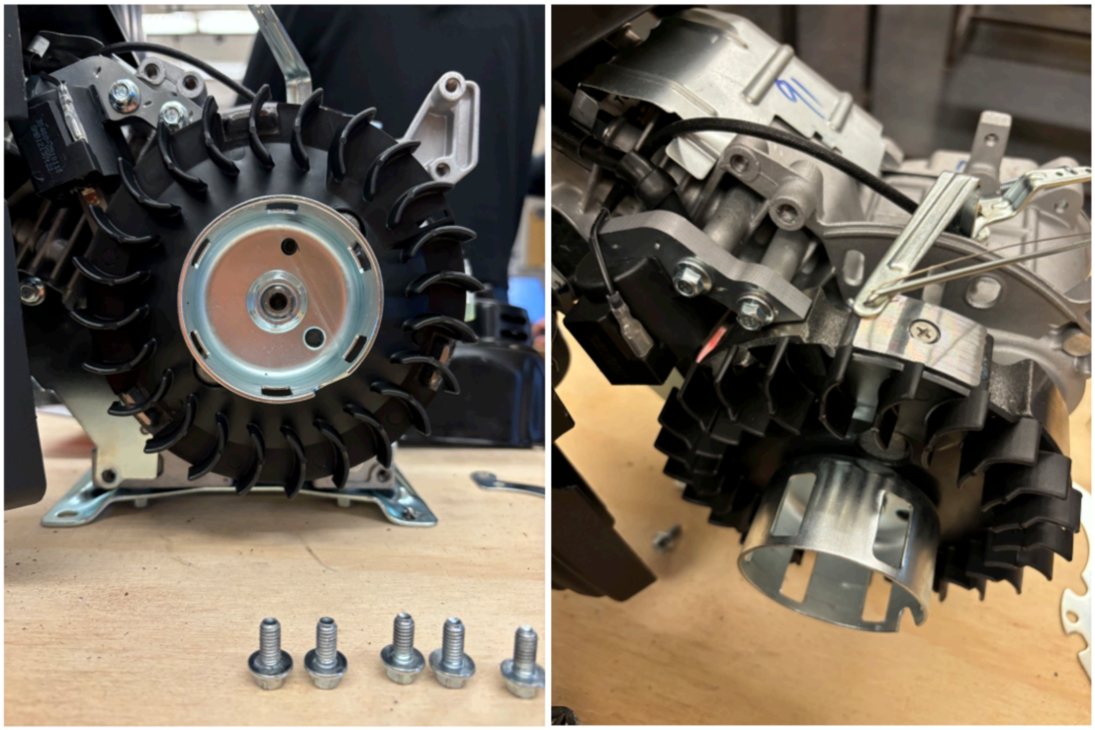
When we opened up the generator, we didn't find an alternator. We instead, found the most curious device.

It was a single magnet, attached to a wheel that passed by a series of steel laminates. The laminates had a coil wound around them that seemed to go directly to the spark plug.

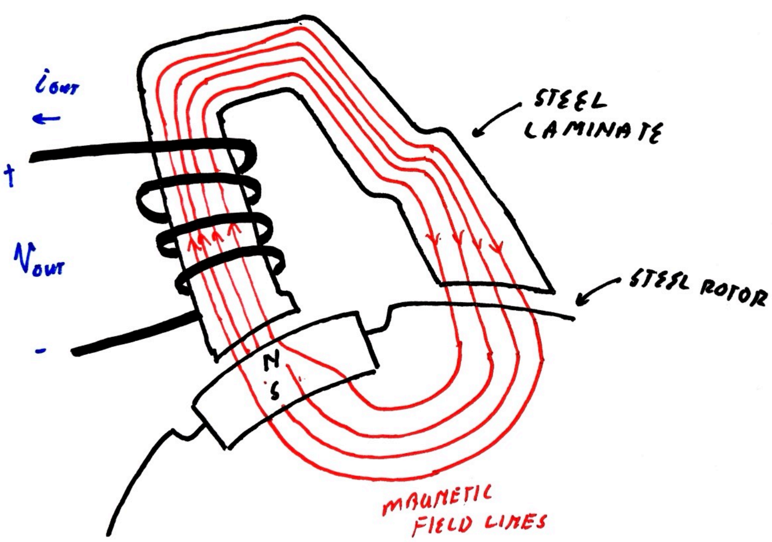
In a normal engine there's a motor that we call an alternator and as it spins it generates 12V DC. This voltage goes, in simple systems, to what's called a point system. The point system is a mechanical switch that triggers when the cylinder is at the *top* of the combustion chamber in a spark-ignition IC engine and sends 12V to the spark plug to explode the fuel and send the cylinder traveling downwards again. The point system is one of the simplest ways to deal with timing and firing the spark plugs in an IC engine.

Someone had found a way to make it even simpler.

Since there was a single magnet and a single cylinder, the magnet was position so that as it passed the steel laminate the cylinder was at the top of the combustion chamber. As the magnet passed, it induced a current in the windings (and a voltage across it) due to Faraday's law of induction. The spark plug fired.



F.3.1—THE SIMPLEST ELECTRO-MECHANICAL GENERATOR IN THE WORLD.



F.3.2—MAGNETIC CIRCUIT OF THE THE SIMPLEST ELECTRO-MECHANICAL GENERATOR IN THE WORLD.

We felt intrigue to the point that we forgot what we were supposed to be doing. As we were fiddling with the device, another D-Lab instructor came over. He brought over a visitor. Another student came over. She brought a friend. And soon we had a crowd of people poking a magnet.

It was like the moment every teacher dreams of. Everyone in the room forgot what they were doing to understand and appreciate how a device worked. It that feeling of collective curiosity that stuck with me, and want to replicate in future teaching endeavors.

A.2—The Simplest Generator in the World: If we were to turn this into an exercise, there's a few options. First, we could have students build this little contraption and measure the power output you can get from spinning the magnet at various speeds. We could also have them specify a rectifier and power converter that would allow them to extract the energy to power a small device like an Arduino which is exactly what the students are working on this summer. You could even explore wrapping extra windings around the laminate to see how *many* devices you could power, thought you could also calculate this with the basic laws of magnetics.

Later, I was having a discussion with Professor Warren Seering about curiosity and one of the things he told me was often times, classes are so intense our students *"don't have the time to be curious, and we need to provide them with more opportunities to do so."*

This got me thinking, can we provide space for that at the curriculum level. Can we provide space for classes where students are just able to explore interesting topics with their friends and instructors, without having to worry about grades, and without having to worry too much about content. First-year advising seminars at MIT are often like this. But generally classes that do this on the *technical* side stop after the first-year with the exception of a few IAP courses, and courses in the Merton C. Flemings metals processing lab [1]. I'm sure they are others, but no matter what they aren't directly baked into the curriculum. Students often have to go out of their way to take 'fun' courses like these and they often do because our students are curious people. But the argument for including this extra space in the curriculum, is so they feel more encouraged to do so.

Relatively recent work that approaches this quite well is the work of Christian Cardozo who taught a class called *ES.S10: Many Interesting Things*. He documented the work in his master's thesis [2]. The course focused on computer architecture, strobe photography, probability, quantum computing, machine learning, computer vision, and cosmology. His thesis is worth a read, and it would be interesting if someone did this for hardware engineering, robotics, or other topics.

Applications to Global Problems

A large body of internal research at MIT and some external research indicates that a significant percentage of engineering students *decrease* their attitude towards social responsibility during the time of their undergraduate degree [3]. Any reasoning for this that I would present is only speculation, but it's kind of disheartening.

In addition to this, and this evidence is also largely anecdotal, students that I've spoken to have indicated that they're less able to take classes focused on apply engineering to the United Nations Sustainable Development Goals because they need to take classes required to graduate, and courses like those 'don't count for much.' At least, that's what a portion of the student body feels.

How I came to know this wasn't through a rigorous survey data but because of a lecture we did in the PCB design course towards the end of the class called *'Hardware's Not Dead'* where, during the second half of the lecture, I presented a number of case studies I'd found on how PCB design is helping to solve various parts of the SDGs [4].

The feedback I got from that lecture was quite interesting. A number of students had never heard of the SDGs despite being interested in a number of them. Some had heard of them but knew little about their contents or our progress towards them.

A number of students came up to me after the lecture and thanked me. They said they hadn't seen engineering integrated with social responsibility in course content in this way before. Some even said they didn't know that *"that was a job they could have."*

What this indicates to me is that we need to do a few things. First, we need to increase our student's awareness on *what* metrics like the UN SDGs are and where engineering fits into solving them. Two, we need to inspire them; show them their efforts can make quantifiable positive impacts using engineering to solve global problems. Three, we need to show them what careers in the fields of social innovation look like and that they are in fact viable career paths. And four, we need to give them the time in the curriculum to learn about these topics.

One idea I had on this front in terms of integrating this with electronics, and something I hope to implement in the 2024 fall class of *D-Lab: Hardware Design for Development* is some content related to 'robust electronics design in developing world environments,' and 'repurposing old electronics to make useful things.'

A quick example of repurposing electronics could be a part of the fall class came from some previous work at MIT D-Lab that used the transformer in a microwave to make a spot-welder. I don't have a reference and didn't want to put a photo in without citing it, but you can imagine that that was a creative and interesting exercise. The spot-welder was designed so that when old microwaves break, people living in under-served communities repurpose the transformer and use the tool to make other things. Sheet-metal working is very common in the areas that D-Lab works so the spot-welder is a super useful tool.

A quick example of robust electronics design would be a PCB my project team for *D-Lab: Design for Scale* had to make to power and control (4x) UV LEDs for a disinfection box for hospitals in South Africa after the COVID-19 pandemic. Much of the effort in the design wasn't in the actual circuit. That was easy. The effort was in sourcing the parts, and laying out the circuit in such a way that we could bring the cost lower than a mercury UV bulb (which our partner didn't want to use for safety reasons). Supply chain was also a huge issue. We had to choose parts that we could get both here in the US as well as in South Africa if the device were to break. We also had to consider the assembly procedure. If it did break, how difficult was it to re-solder components and fix it? In the end we lost. The mercury bulb still beat us in cost. But we learned a lot. In the end the hospital bought some UV mercury disinfection boxes and we were able to help them make that decision. That felt good and I want more of our students to be able to have that feeling.

Narratives on Learning

Something I alluded to in previous chapters but didn't expand on was the idea of including narratives surrounding learning in our teaching. This include *both* explaining the purpose and the design of our own teaching to our students so they can more actively understand and engage with their own learning processes, *and* helping them understand how they learn and what strategies are most effective for them. I think this is not best done as a single subject in the elaboration section, rather an underlying narrative that exists through the classes our students take as they go through their degrees.

Doing this has a few key advantages.

First, it's been proven to lead to more active and engaged learning. *Make it Stick* uses the work of Professor Mary Pat Wenderoth as an example in terms of injecting narratives on learning into education. Professor Wenderoth specifically teachers her students about the testing effect, desirable difficulties, and illusions of knowing and how to avoid them [5].

> *"I model it in class for them. Every five minutes or so I throw out a question on the material we just talked about. I can see them start to look through their notes. I say 'Stop. Do not look at your notes. Just take a minute to think about it yourself.' I tell them our brains are like a forest, and your memory is in there somewhere. You're here, and the memory is over there. The more times you make a path to that memory, the better the path is, so that next time you need the memory,*

> *it's going to be easier to find it. But as soon as you get your notes out, you have short-circuited the path... someone has told you the way." — pg. 230* [5]

The book also indicates that after injecting these little narratives around teaching, Professor Wenderoth has seen positive results in terms of students seeking active learning opportunities.

> *"...now students come to see her after a disappointing exam and say, 'I have the illusion of knowing. How do I get better?'" — pg. 230* [5]

The book recommends we be transparent. That we are *deliberate* in what we do and we explain to the student *why* it is worth persisting through the difficulties for better learning in the end.

The second reason to do this is it helps us calibrate our own teaching. When our students understand their own learning better, they can help us calibrate our own judgment of whether an exercise is teaching them what we think it should be. That discourse is useful and included in many works surrounding teaching. We need to talk to our students to understand what they are and aren't getting our of an exercise, and the more they know about what we're trying to do the more they can help us.

Finally, the better our students are at learning and the better we are at teaching, the less we need to re-teach.

This has two benefits. Often times in engineering courses significant time is taken to re-teach material that was 'supposed to be taught' in pre-requisite classes that students 'didn't learn.' Equally often, the student gets blamed for not learning the content or not remembering it. We need to stop doing that, we need to stop blaming the student for not retaining information well when we haven't taught them to retain information well in the first place. If we taught them about learning, we'd spend less time teaching old content.

The second benefit is that we can teach *more*. *Make it Stick* notes that a key illusion of knowledge is that there is a limit. Science says there isn't one, our brains aren't like water buckets that can only 'hold so much at a time' [5]. This makes me think of the mind-palace idea from BBC's adaptation of Sir Arthur Conan Doyle's Sherlock Holmes books [6]. At any rate, if better teaching means we can teach more, maybe the idea of interdisciplinary education will become even more tractable.

Lastly, I would argue that learning to learn is a large portion of an undergraduate education. Therefor teaching our students about learning, and providing them with opportunities to engage in active learning fulfills one of the goals of an undergraduate education.

Future Work

There this no limit to the amount of future work that exists in education. We will always seek to teach, and we will always strive to teach it better.

In terms of general future work, I think a lot more could be done on connecting the science of learning to engineering education. Specifically in the realm of developing illustrative examples that show 'what works' and explaining *why* they work for various sub-fields of study. I think such a body of work in the form of a book, or a compiled online resource would greatly benefit teachers. I also think the extension of this past hardware, and physical exercises should be explored, and we should ask the question: what *simple* things can we do in an engineering classroom to improve retention that *don't* involve building things? We should share these techniques with each other in an effort to build better learning environments.

In terms of future work that exists specifically in relation to this thesis. It would be worth my time (or your time if you so choose) to see if we can integrate the exercises developed in this work in classes we are currently teaching (or new classes) and try them with students. It would be well-worth getting their direct feedback and analyzing the pitfalls and strengths of each exercise. And of course writing down what we learned so others may benefit.

In relation to the discussion on robotics and intelligent systems presented in this work, and in relation to the curricula presented in the appendices, I think the most useful thing to do with it would be to take it, and compile it into some form of visual guide for the student. The goal of such a guide would be to enable them to structure their own degree programs with an increased awareness of what engineering, specifically in the case of this thesis *hardware* engineering, looks like in the field. I think this could greatly solve a number of issues in relation to the accessibility of our degree programs.

Other things that are worth exploring that I didn't have time to delve into in this thesis is what place teaching history has in engineering education. MIT's Metals Lab [1] offers an interesting intersection of study between archeology and materials science. I wonder what this would look like for mechanical engineering, or electrical engineering. One idea I had for a class was *Every Motor Ever Made!* We'd start from the beginning and work our way to modern motors, and maybe build some of the key machines along the way. We could do the same thing for *Every Battery Ever Made!* Though it's possible building those would be a little less tractable.

On the topic of ethics, I believe this is the most critical area of future work especially in relation to teaching intelligent systems. Ethics in both robotics research, and teaching is generally included more as an after-thought rather than a central study. Primarily, we need to focus more on teaching students about the increasing accessibility of potentially dangerous technology that's being coupled with the increasing dis-regard for their potential affects and mis-use. We should not just be teaching utility. I include some books on the topic that are worth a read in the references below [7][8][9]. These are specifically in relation to AI, but I think more work needs to be done to think about how these concepts also translate to systems with physical components.

I end with this, to me teaching is far less about our ability to impart specific technical knowledge to a student than it is about *understanding* our students. Their hopes, dreams, and aspirations. And helping get them from where they are now, to where they want to be. Technical knowledge is *part* of that, but so is creativity, ethics, and understanding of history, and application of their work to making the world a better place. Teaching is about inspiring curiosity, and I think we can always find better ways to inspire curiosity our students.

References

1. Metals Classes, Merton C. Flemings Materials Processing Laboratory. See: https://metalslab.mit.edu

2. C. I. Cardozo-Aviles, "Many interesting things," MEng Thesis, Massachusetts Institute of Technology. Department of Electrical Engineering and Computer Science, 2018. Available: https://dspace.mit.edu/handle/1721.1/120386

3. A. R. Bielefeldt and N. E. Canney, "Changes in the Social Responsibility Attitudes of Engineering Students Over Time," Science and Engineering Ethics, vol. 22, no. 5, pp. 1535–1551, Sep. 2015, doi: https://doi.org/10.1007/s11948-015-9706-5.

4. A. Mehrotra and F. Moseley, Lecture 08—Hardware's Not Dead, The Art and Science of PCB Design, Massachusetts Institute of Technology, Jan. 2023. Accessinble: www.youtube.com/watch?v=TlRmg9Cv0sk

5. P. C. Brown, H. L. Roediger III, and M. A. McDaniel, Make it stick: the science of successful learning. Harvard University Press, 2014.

6. S. Zielinski, "The Secrets of Sherlock's Mind Palace," Smithsonian Magazine, Feb. 03, 2014. https://www.smithsonianmag.com/arts-culture/secrets-sherlocks-mind-palace-180949567/

7. S. U. Noble, "Algorithms of Oppression," NYU Press, Feb. 2018.

8. C. O'Neil, "Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy," Crown, Sept. 2016.

9. R. Benjamin, "Race After Technology: Abolitionist Tools for the New Jim Code," Polity, Jun. 2019.

Appendix

A — The Core Classes

This section presents the syllabi of the four 'core' classes of the curriculum. We assume these classes are taught over a 14-week semester and operate at 12 hours/week (equivalent to a standard 12-unit MIT course).

Introduction to Embedded Systems and Control

Course Description

Provides a hands-on and practical introduction to microcontrollers and the basic components of embedded systems. Students learn how to interface with sensors and control basic actuators, implement communication between distributed microcontrollers and between a microcontroller and computer system. Advanced topics include clock configuration and its effects on microcontroller operation, direct memory access, and flash memory. Concepts of latency and bandwidth are emphasized. Lab exercises also offer training on standard electrical lab equipment including oscilloscopes, function generators, logic analyzers, and more.

Hours: 4 lecture, 1 homework, 7 lab Units: 12

Learning Objectives

- Be able to implement a high-speed communications system between two microcontrollers, understand the operation of and select from different hardware communications protocols.
- Be able to setup the pin-mapping, and clock speed of an un-flashed microcontroller.
- Calculate the latency of a communication system given a baud rate and packet size. Explain the difference, advantages, and disadvantages between half-duplex and full-duplex communications systems.
- Be able to setup the microcontroller for and read from analog inputs. Be able to process these signals on the microcontroller to extract frequency content, or reduce noise.
- Be able to select an ADC / DAC and explain the effects of discretization on a signal in various applications.
- Be able to apply a transformation to a signal read into a microcontroller.
- Be able to implement a basic continuous-time feedback controller with an op-amp circuit.
- Be able to read from sensors (both digital and analog) and log that data to a (.csv) file on a paired computer from the microcontroller.
- Be able to design a simple high-impedance RC analog integrator network.
- Be able to design a shunt resistor circuit for current measurement. Be able to specify the power requirements for circuit components to avoid failure.
- Be able to perform standard position, velocity, and torque control on a DC motor with feed-forward compensation.

- Be familiar with the use of standard electrical lab equipment include oscilloscopes, function generators, and logic analyzers.
- Be able to read and understand an electrical data-sheet.

Assumed Pre-requisites

- Circuits / Introduction to Electronics
- Dynamics and Control
- Basic Programming Experience

Course Calendar

Week	Unit	Lecture	Lab
1	Unit 1: Microcontrollers	Lecture 1: Basic operation of microcontrollers. The Arithmetic Logic Unit (ALU). Lecture 2: Data Structures.	Introduction to Lab Equipment
2		Lecture 3: Instruction Execution. Memory. Registers. Lecture 4: Peripherals, Pins, Pin-types, and Setup. Analog/Digital Inputs and Outputs.	Lab 1: Introduction to Microcontrollers and Programming
3		Lecture 5: Communications I. UART and SPI. Lecture 6: Communications II. CAN and I2C.	
4		Lecture 7: ADCs, DAQs, Discretization. Lecture 8: Sampling. Aliasing. Nyqist.	Lab 2: E.1—A Simple Exercise in Communications
5		Lecture 9: State Machines. Interrupts. Lecture 10: Flash Memory.	
6	Unit 2: Sensors and Filters	Lecture 11: Direct-Memory-Access (DMA). Lecture 12: A review of sensors. Simple sensor models. Digital and Analog. Accuracy and Precision.	Lab 3: E.3—Magnetic Material BH Curve Measurement Device
7		Lecture 13: Digital Filtration. Moving Averages. Lecture 14: Analog/Hardware Filtration. Continuous Time Processing.	

8		Lecture 15: Fast-Fourier Transform (FFT). Superposition. Lecture 16: Comparators and Op-Amps. Hysteresis.	Lab 4: E.2—Electric Guitar Distortion Pedal
9		Lecture 17: Amplification. Signal Combination. Lecture 18: Sensor Calibration. Linearization. Non-linearities.	
10	Unit 3: Control	Lecture 19: Feedback Control I. Lecture 20: Feedback Control II.	Lab 5: E.4 DC Electronic Load
11		Lecture 21: Feedforward Compensation with Models I. Lecture 22: Feedforward Compensation with Models II.	
12		Lecture 23: Delays, Latency, Bandwidth, and Stability. Lecture 24: Continuous Time Control in Hardware I.	Lab 6: A.4—DC Motor Control
13		Lecture 25: Continuous Time Control in Hardware II. Lecture 26: Discrete Control I.	
14		Lecture 27: Discrete Control II.	

T.A.1—CALENDAR FOR THE INTRODUCTION TO EMBEDDED SYSTEMS COURSE.

Mechanical and Mechatronics Design

Course Description

Over the course of a semester, student design the electrical and mechanical system of a robot to compete in a game completing a set of pre-specified tasks for points. Emphasis is placed on using simple physics equations to size components and make design decisions. Robustness and manufacturability of the mechanical system and electrical system are emphasized. Multiple methods of actuation are explored as well as the topics of power conversion and simple thermal design. Lab time also offers hands-on experience with lathes, mills, laser cutters, water-jet machines, and other machine shop equipment.

Hours: 3 lecture, 2 homework, 7 lab Units: 12

Learning Objectives

- Be able to define hierarchical numerical functional requirements of a system.
- Be able to use basic physics equations to come up with initial component sizes of a system based on a set of functional requirements.
- Be able to use the design process to ideate and then select from concepts. Be able to justify the selection of design concepts using fundamental physics.
- Be able to use CAD to design parts and assemblies.
- Be able to use virtual work, and St. Venant's principal to make design decisions.
- Be able to model a motor's torque-speed curve and select a gear ratio to achieve a specific torque and speed combination.
- Be able to size a gear-train to operate at a motor's peak-power point.
- Be able to specify fasteners between connected parts. Be able to use bearings to provide appropriate constraints to parts with relative motion.
- Be able to select between hydraulic, pneumatic, and electromechanical actuation for various task descriptions.
- Be able to specify the voltage, capacity, and current requirements of a battery for a small system.
- Be able to specify off-the-shelf DC/DC power converters to supply power to a multi-component electrical system.
- Be able to specify a heatsink to cool an electrical component under a specified load.
- Be able to design a system to be machined on a standard 3-axis mill, water jet, lathe, and laser cutter.
- Gain familiarity with shop equipment, and be able to use it safely.
- Be able to perform electromechanical system integration including isolation of electronics from mechanical components.

Assumed Pre-requisites

- Mechanics and Materials
- Dynamics and Control
- Physics Mechanics
- Physics Electricity and Magnetism

Course Calendar

Week	Unit	Lecture
1	Unit 1: Fundamentals of Design	Lecture 1: Functional Requirements. Appropriate Analysis. Lecture 2: Ideation. Drawings. Prototyping
2		Lecture 3: CAD. Lecture 4: Power Transmission. Gears. Belts. Linkages. Screws.
3		Lecture 5: D'Alembert's Principle. Virtual Work. St. Venant's Principal. Lecture 6: Rotary to Linear Motion, CAMs, Interrupted Motion
4		Lecture 7: Constraints, Bearings, Relative Motion Lecture 8: Structures, Interfaces, Connections
5	Unit 2: Actuation	Lecture 9: Tolerances and Design for Manufacturing Lecture 10: Power, Efficiency, Transducers
6		Lecture 11: Electro-Mechanical Actuation: Motors, Voice-Coils, Piezos, Servos, Relays Lecture 12: Fluid-Based Actuation: Pneumatic and Hydraulic Systems
7		Lecture 13: Gravity-Compensation, Motion Compensation using Mechanical Methods Lecture 14: Motor Torque-Speed Curves, Power Curves, Efficiency Maps
8		Lecture 15: Optimizing Power Transmission for Peak Power, and Peak Efficiency
9	Unit 3: Power Sources and Conversion	Lecture 16: Power Converters and Distribution Lecture 17: Batteries, Battery Safety, and Management
10		Lecture 18: Energy Use Modeling, Continuous Power, Peak Power Design Lecture 19: Rectifiers and Inverters
11		Lecture 20: Basic Power Noise Filtration
12		Lecture 21: Introduction to Thermals and Temperature Control
13	Project Work Time	—
14		—

T.A.2—CALENDAR FOR THE INTRODUCTION TO MECHATRONICS COURSE.

Course Project

The course project for this class is outline in exercise A.3 presented in Appendix C.

Introduction to Modeling and Simulation

Course Description

Course provides an introduction to simulation and modeling. Emphasis is placed on designing experiments to build models of real systems and insert them into simulation. A term-long course project divided into a series of labs provides students hands-on experience in simulating systems. Accuracy of simulation compared to ground-truth data is emphasized. Students work in teams to complete the term project. Computational methods for modeling and parameter fitting are also explored. Homework exercises explore complex kino-dynamic system modeling and integration of models into the simulator while lab exercises focus on experimental measurement of models.

Hours: 3 lecture, 2 homework, 7 lab Units: 12

Learning Objectives

- Be able to develop a Lagrangian dynamics system model based on generalized coordinates for an arbitrary system.
- Be able to write the forward dynamics and kinematics for a generalized system.
- Be able to select a simulation time-step and solver based on desired accuracy of results, and simulation run-time.
- Be able to define constraints for simulation and optimization using energy, work, and equilibrium.
- Be able to model, and fit the parameters of the model, of a second-order mechanical system.
- Be able to model a thermal circuit and calculate its basic parameters.
- Be able to measure the torque-speed curve of a motor.
- Be able to measure the traction of a wheel.
- Be able to model a simple DCIR battery.
- Be able to use regression to fit model parameters using collected data.
- Be able to conduct an error analysis between ground truth data and simulation data.
- Be able to integrate multiple models in a hierarchy to model a complex system. Be able to define the interaction between these models using physics.

Assumed Pre-requisites

- Introductory Signal Processing
- Introduction to Numerical Methods and Numerical Simulation
- Dynamics and Control

Course Calendar

Week	Unit	Lecture	Lab
1	Unit 1: Review of Numerical Simulation Methods	Lecture 1: Numerical Integration of Differential Equations. Forward Euler, Initial Conditions, Constraints. Lecture 2: Solving Differential Equations Numerically. Ode45 and other MATLAB solvers.	Lab 1: M.5—Simulation Time-Stepping Exercise
2	Unit 2: Models	Lecture 3: Stability Analysis and Simulation Convergence. Reverse Euler, Implicit Euler, Runga-Kutta. Lecture 4: Review of Energy, Co-Energy, Work and Virtual Work. Equilibrium Equations.	
3		Lecture 5: Simple models Part I. Suspension systems and thermal circuits. Lecture 6: Simple models Part II. Motors and Batteries.	Lab 2: M.1—Suspension Model Measurement
4		Lecture 7: Simple models Part III. Models of sensors. Sampling. Lecture 8: Generalized Coordinates, State and Energy Variables. Parameters.	
5		Lecture 9: Lagrangian Dynamics. Lecture 10: Complex models Part I. Contact and Friction.	Lab 3: M.2—Motor Torque-Speed Measurement
6		Lecture 12: Complex Models Part II. Neural Networks as Models. Formulation and Fitting. Lecture 13: Hierarchical Modeling. Combining Models.	
7	Unit 3: Signal Processing and System Identification	Lecture 14: Experiment Design. Data Collection. Lecture 15: Extracting Data From Noise. Filters.	Lab 4: M.3—Wheel Traction Characterization
8		Lecture 16: Least Squares Regression. Lecture 17: Regression II.	

9		Lecture 18: Initial Guesses. Parameter Estimation. Lecture 19: Introduction to Spectroscopy, Bode Plots, Magnitude and Phase Response.	Lab 5: M.4—Battery Modeling
10	Unit 4: Model Verification	Lecture 20: Ground-truth data. Methods for acquiring ground-truth data. Lecture 21: Error Analysis.	—Project Work Time—
11		Lecture 22: Modifying Models based on experimental results. Lecture 23: When to stop modeling.	—Project Work Time—
12	Project Work Time	—	—Project Work Time—
13		—	—Project Work Time—
14		—	—Project Work Time—

T.A.3—CALENDAR FOR THE INTRODUCTION TO MODELING AND SIMULATION COURSE.

References

Embedded Systems Class

1. 6.115 Microcomputer Project Lab @ MIT MechE
2. Embedded Computing with the PIC32 Microcontroller (Kevin Lynch, Nicholas Marchuck, Matthew L. Elwin)— https://hades.mech.northwestern.edu/images/e/e3/EmbeddedComputingMechatronicsSampleChapters.pdf
3. 2.74 Bio-Inspired Robotics @ MIT MechE
4. 2.131 Advanced Measurement and Instrumentation @ MIT MechE
5. 6.101 (old number) Analog Electronics Lab @ MIT EECS
6. 6.111 (old number) Introductory Digital Systems Laboratory @ MIT EECS
7. 6.08 (old number) Introduction to EECS via Interconnected Embedded Systems @ MIT EECS

Modeling and Simulation Class

1. 2.74 Bio-Inspired Robotics @ MIT MechE
2. 6.141 Robotics Science and Systems @ MIT EECS + MIT AeroAstro
3. 2.12 Introduction to Robotics @ MIT MechE
4. Introduction to System Identification - FIRST Robotics (https://docs.wpilib.org/en/stable/docs/software/advanced-controls/system-identification/introduction.html)
5. ECE5560 System Identification @ University of Colorado Colorado Springs
6. 6.435 System Identification @ MIT EECS

7. 3.021J Introduction to Modeling and Simulation for Materials @ MIT DMSE

8. 2.141 Modeling and Simulation of Dynamic Systems @ MIT MechE

9. 6.336J SMA 5211 Introduction to Numerical Simulation Methods @ MIT EECS

10. A Brief, Informal Introduction to Motor Testing—https://evt.mit.edu/textbooks/Dynos_for_Dummies.pdf

Mechatronics Class

1. Embedded Computing with the PIC32 Microcontroller (Kevin Lynch, Nicholas Marchuck, Matthew L. Elwin)— https://hades.mech.northwestern.edu/images/e/e3/EmbeddedComputingMechatronicsSampleChapters.pdf

2. 2.74 Bio-Inspired Robotics @ MIT MechE

3. 2.737 Mechatronics @ MIT MechE

4. 2.007 Design and Manufacturing I @ MIT MechE

5. The Art + Science of PCB Design (pcb.mit.edu)

6. 6.131 (old number) Power Electronics Lab @ MIT EECS

7. University of Michigan, Short Course in Batteries—https://umbatterylab.engin.umich.edu/training-opportunities/

8. A Brief, Informal Introduction to Motor Testing—https://evt.mit.edu/textbooks/Dynos_for_Dummies.pdf

Additional References

1. Matthew Kelley, Canon Trajectory Optimization—http://www.matthewpeterkelly.com/tutorials/trajectoryOptimization/canon.html

B — Curriculum

This section describes the curriculum structure proposed for Intelligent Physical Systems. In this section we'll focus on building a curriculum specific to MIT, but the work we're doing here can be scaled to any school. We use MIT here as a case study.

Courseload

MIT students typically take 48 units of courses per semester across 8 semesters, totaling 384 units. Typically, full semester classes count for 12 units which represents '12 hours per week of student effort'. This includes all aspects of the course from attending lecture and lab through completing assignments. A number of 6-unit half-semester, or full-semester (equivalent work load-spread over a longer time) courses are also offered across the institute and remains to be a common format. The 9-unit format is rarer but allows additional flexibility over the 6-unit format depending on the nature of instruction (i.e. an additional three-hour lab component).

Assuming the student enters with no transfer credit, students are required to take (6x) classes as part of the institute science requirement including Chemistry, Physics (both Mechanics and Electricity-Magnetism), Mathematics (both Calculus I and II), and Biology. These classes total 72 units.

Students are also required to take (8x) classes from the Humanities Arts and Social Sciences (HASS) list which totals 96 units.

This leaves us around 216 units, or 18 full-semester classes for major and elective requirements to implement a curriculum with some freedom to replace 12 unit classes with 6 or 9 unit classes.

General Structure

1-2 Introductory Subjects allow students to explore the major. Generally these courses are structure to provide hands-on learning opportunities enabling students to gauge interested while learning practical skill before committing to a degree. These courses are often design to be fun in an effort to attract students to the major as well. Sometimes these courses are also replaced by first-year advising seminars design to build community while also providing an introduction to a specific subject area.

8-10 Foundational Subjects that provide both the pre-requisite knowledge for higher-level classes, but also build up a base that all students studying the major must have. These courses can include mathematical foundations such as differential equations, or introductory courses such as mechanics.

3-4 Core Classes is an idea that some curricula have and others do not. These classes build upon foundational subjects and provide domain-specific knowledge in the areas the student hopes to study. But they do not go into the level of depth that advanced classes do. Our concept for these courses is outlined in the Appendix A.

3-4 Advanced Subjects that allow students to dive deeper into sub-fields of their choice. Generally knowledge depth is created as these higher subjects of study. Many majors handle these as tracks.[10]

A Capstone Class that combines all the student has learned into a design project. Students engineer and build a real system. These classes should be structures as engineering simulators which require students to deal with the consequences of their engineering decisions.

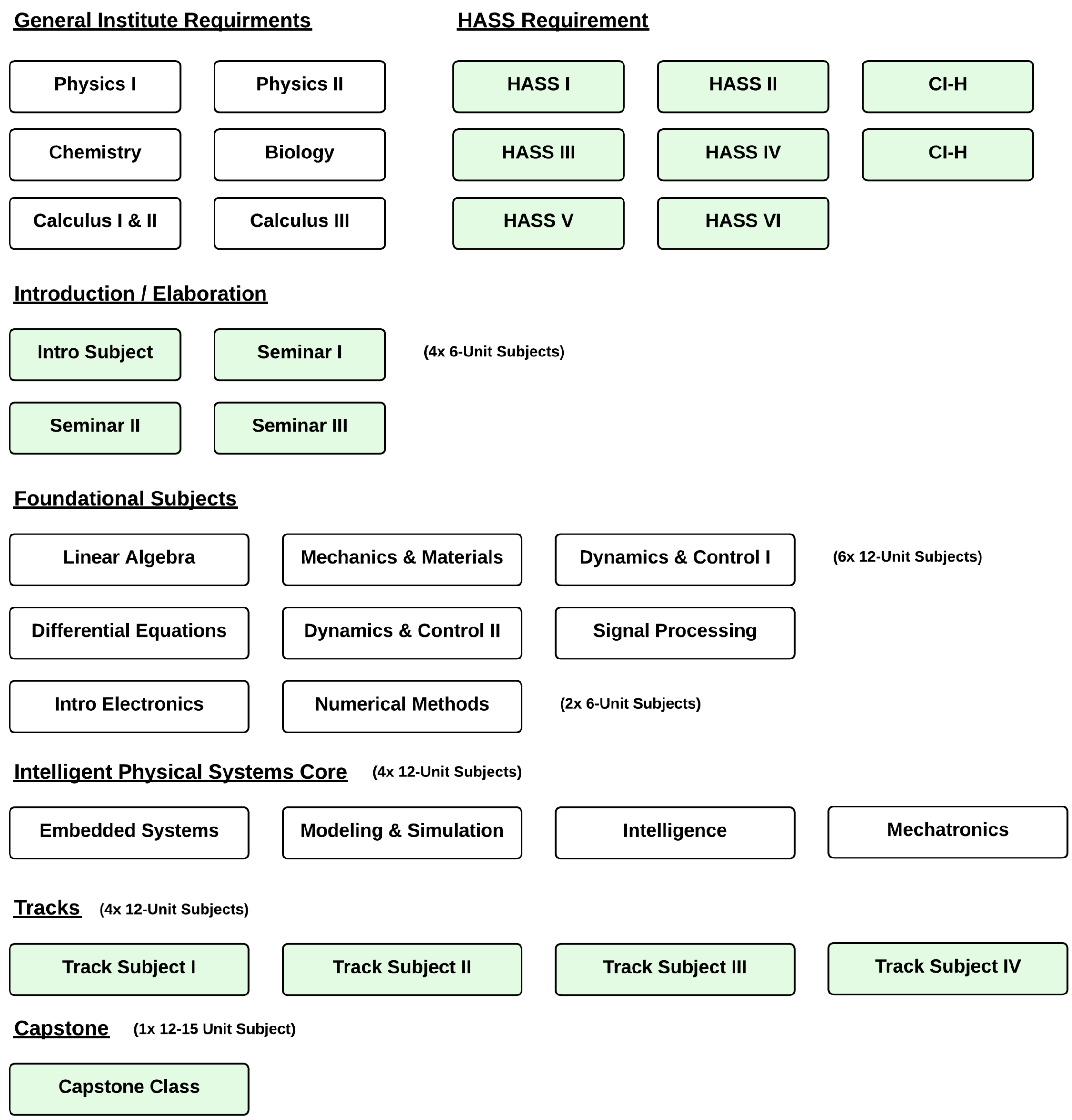
---

[10] See both the MIT 2A and 6-5 program

Elaboration Subjects

Our curriculum proposes the addition of (3x) 6-unit elaboration subjects or seminars. These subjects are designed to give students extra understanding of the history and context of their engineering work in their chosen field. They also offer the possibility to include \textit{interesting} department subjects in special areas in such a way that the student may exercise curiosity without dedicating themselves to a 12-unit subject

The Proposed Curriculum



F.B.1—THE CURRICULUM.

Notes on the Foundational Classes

- Mechanics and Materials would be similar to 2.001 Mechanics and Materials I @ MIT
- Dyanmics & Control I and II would be similar to 2.003 and 2.004 @ MIT
- Linear Algebra, Differential Equations, and Numerical Methods could all be combined into two classes such as Mathematics for Engineers I and II
- Introduction to Electronics would be similar to 2.678 @ MIT. It could also include content from 6.002 Circuits and Electronics @ MIT (old number)
- Introductory Subjects would be similar to those required for the MIT Course 2 degree: 2.00, 2.00C, 2.00A are example classes from this category
- One could think about including a course like 2.005 (Thermal Fluids Engineering I) into the foundational subjects

References

1. University of Sherbrooke, Bachelor's Degree in Robotics. https://www.usherbrooke.ca/admission/programme/259/baccalaureat-en-genie-robotique#structure/
2. ETH Zurich Degrees in Mechanical Engineering. https://ethz.ch/en/studies/bachelor/bachelors-degree-programmes/engineering-sciences/mechanical-engineering.html
3. University of Michigan, Undergraduate Program in Robotics. https://robotics.umich.edu/academics/undergraduate/program-requirements/

C — Additional Exercise Suggestions

The following section presents an exercises that we devised or brainstormed that are not presented in the main body of this thesis. We also note exercises that are presented in the main body of the thesis for reference. To save the trees (both digital and physical) they're not re-explained here.

Embedded Systems

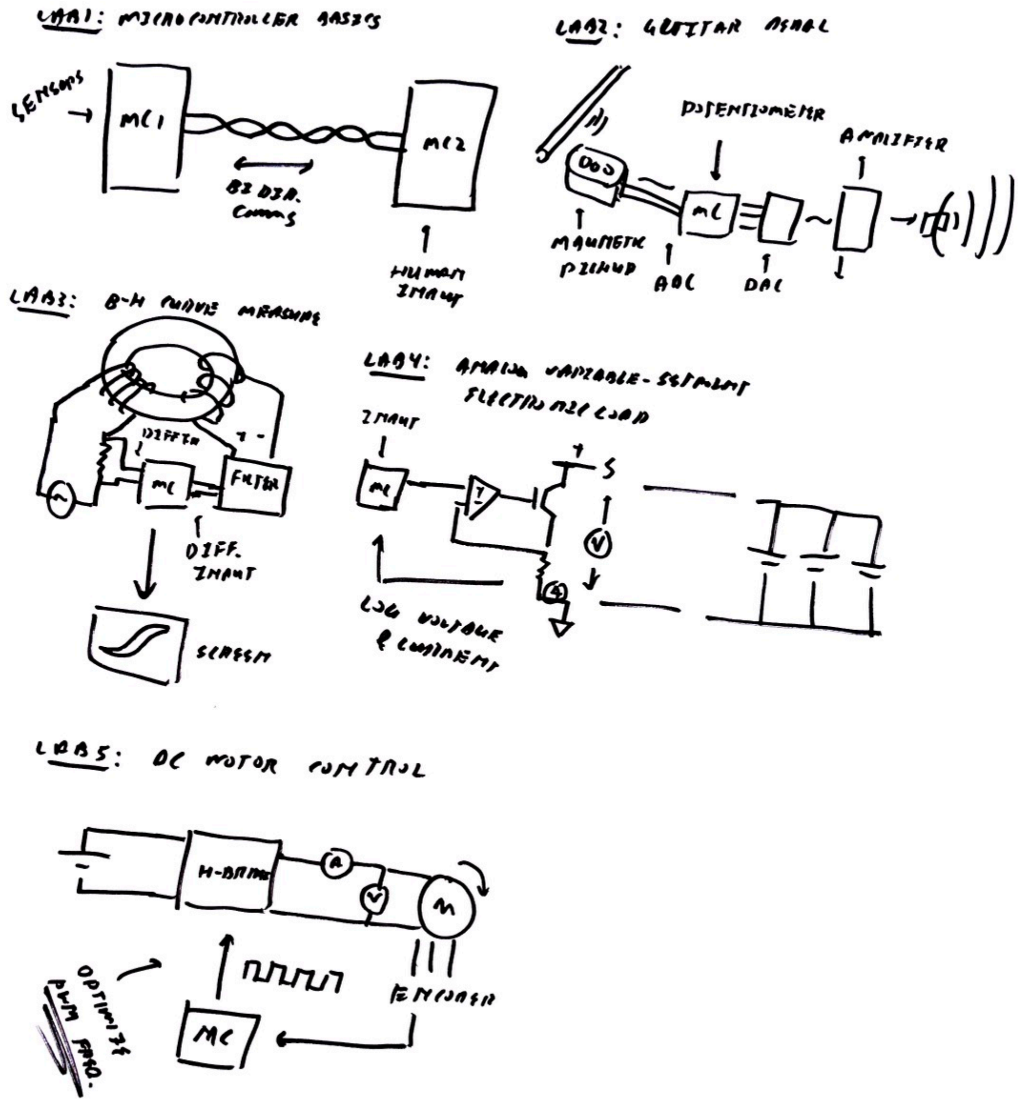
E.1—A Simple Exercise in Communications

E.2—Electric Guitar Distortion Pedal

E.3—Magnetic Material BH Curve Measurement Device

E.4—DC Electronic Load: Students are asked to build a closed-loop current controller for a DC electronic load with an op-amp. The set-point of the op-amp is determined by the output of a microcontroller, and the microcontroller logs voltage and current data to the computer. The microcontroller set-point may not be adjusted to control the current, the bandwidth and stability of the controller must be fully determined by the op-amp/BJT circuit.

E.5— Modeling and Accounting for Power Consumption: Students use the data sheet of the microcontroller and various lab equipment to measure or estimate the power consumption of the micro-controller. They use this information to specify a linear-regulator that could be use to power a single microcontroller and any active peripherals they choose to use.



F.C.1—SKETCHES OF THE EMBEDDED SYSTEMS PROJECTS.

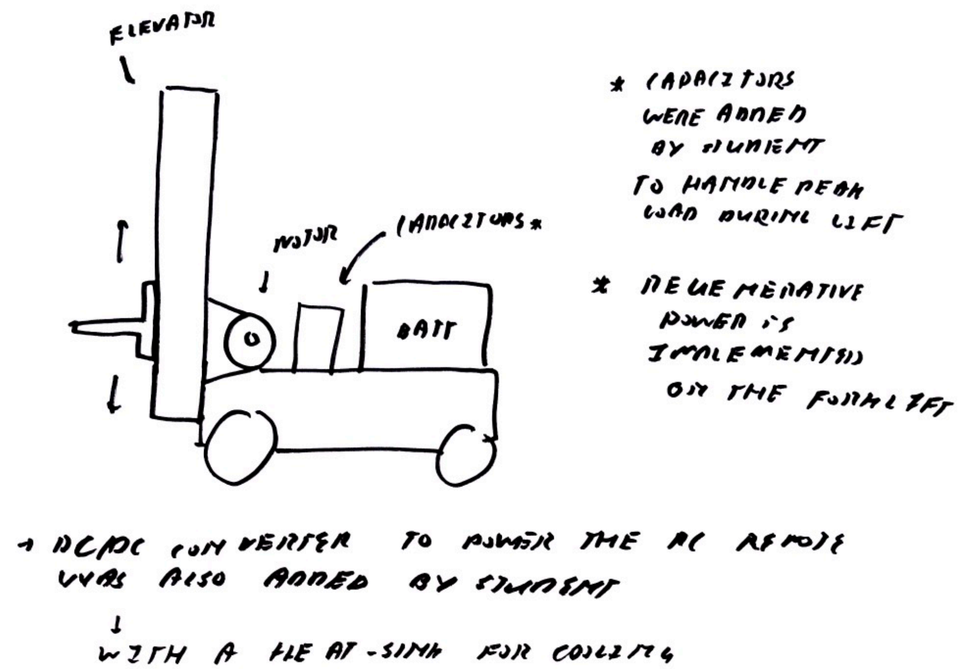
Mechatronics, Actuation, and Power Conversion

A.1—Feeling the Motor Limits

A.2—The Simplest Generator in the World

A.3—Design of a Robot for a Game: The project for this course will be similar to that of 2.007 Design and Manufacturing I at MIT. Students will be presented with a game or challenge with design requirements and are expected to translate these requirements into a mechatronics design. The addition in this course is the inclusion of extra information on electrical systems which students are expected to incorporate into their projects which include DC-DC converters, thermal management systems, power modeling for various stages of the challenge, and more. Additional actuators schemes such as basic fluid-actuators is also introduced. Previous years 2.007 competitions are available on the website (https://me-2007.mit.edu/).

A.4—DC Motor Control: Students are asked to implement closed-loop position, velocity, and torque control on a DC motor controlled by an H-Bridge with a power converter. Students are asked to measure the resistance, and inertia parameters of a basic DC motor, and implement the dynamics of the motor as a feed-forward compensator in the controller.



F.C.2—SKETCHES OF THE MECHATRONICS PROJECTS.

Intelligence and Decision Making

I.1—Conference Table Microphone: Students are asked to develop an algorithm that isolates the most likely direction of a sound source given a circular array of microphones on a table. Students are asked to then isolate the desired sound, and de-noise the signal within a certain time specification. The solution is evaluated on both solve time, and latency of transmission of the final audio signal.[11]

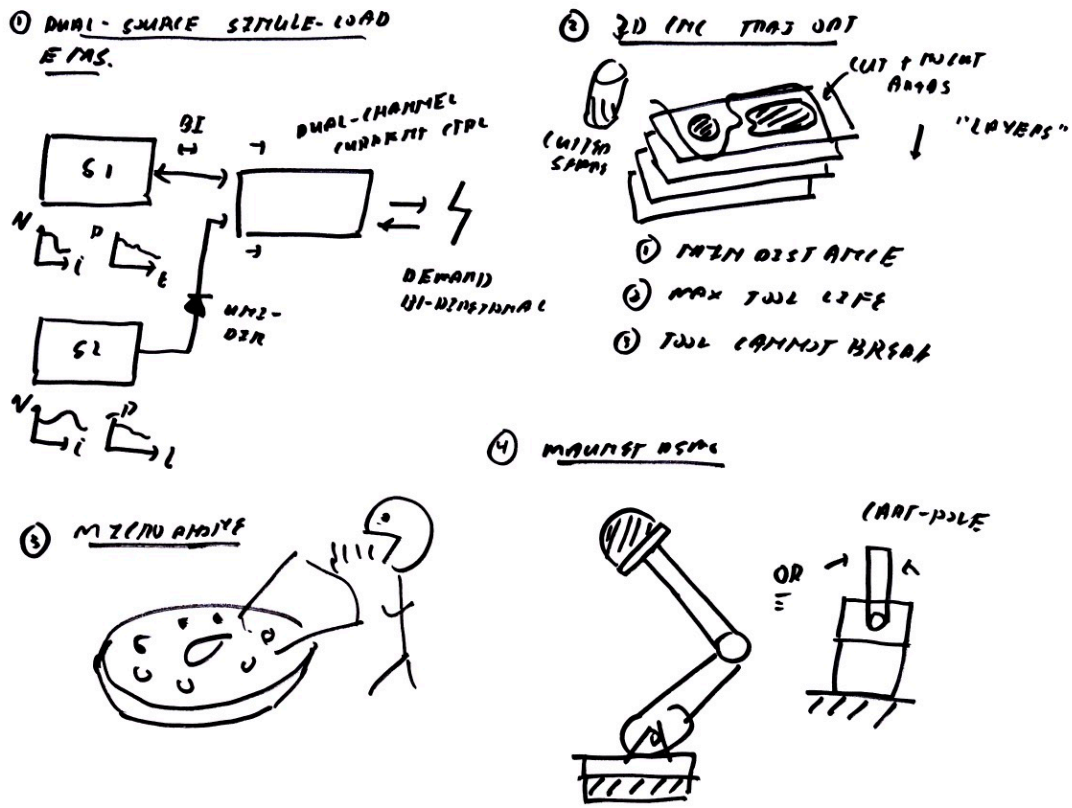
I.2—Dual-Source, Single-Load Energy Management: Students are presented with two possible power sources with different I-V and degradation characteristics. One source is bi-directional, the other source is uni-directional, the load is bi- directional. Students must write an algorithm that controls the current from each source to meet the power demand while minimizing the 'degradation' function of each source. Points are awarded for increased efficiency (i.e. operating closest to the peak efficiency point of each source).

I.3—3C CNC Trajectory Optimization: Students are provided a 3D map with each layer being a set of pixels with black shaded areas being 'no-cut' zones while white shaded areas are 'cut' zones. Given cutter geometry, and a known depth-of-cut, students write a minimum-distance trajectory optimization that accounts for tool loading constraints. Extra points are awarded for maximizing tool lifetime.

I.4—Robot Leg 'Magnet' Demo: A 3D force-sensor is attached to the end of a 3DoF robotic leg. Students are asked to develop a controller that enables the leg to apply the Newtonian force-couple of the applied force in real-time.

I.5—Reinforcement Learning for a Cart-Pole: Students will use reinforcement learning to design a swing-up controller for a basic cart- pole. The swing-up controller is designed using reinforcement learning, and students have the option to switch to LQR when the pole is within some minimum distance from the target that students define based on stability analysis.

[11] Credit for pointing me towards this system goes to Aashini Shah.



F.C.3—DIAGRAMS FOR THE INTELLIGENCE EXERCISES.

Modeling and Simulation

M.1—Suspension Model Measurement

M.2—Motor Torque-Speed Measurement

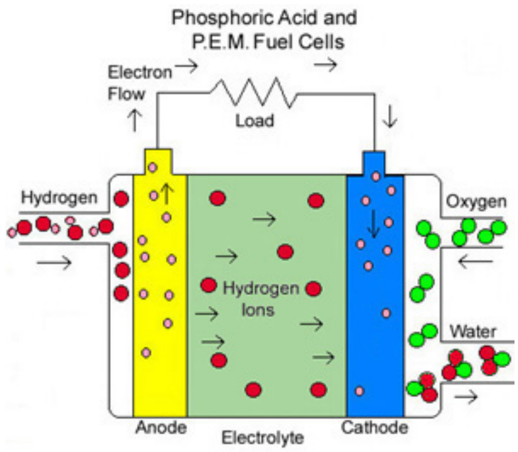
M.3—Wheel Traction Characterization

M.4—Battery Modeling

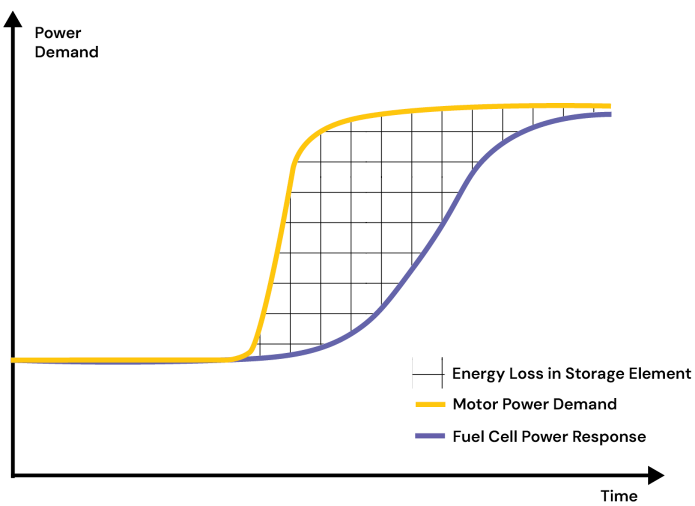
M.5—Simulation Time-stepping Example: The simulation time-stepping example was inspired by a problem I ran into simulating a power-converter in 6.334. Naturally, the simulation will not provide accurate results as the time-step approaches the time-constant of the dynamics of the system. This is a very simple concept but an illustrative example would be to provide the student with a dynamic simulation, and have them lower the time-step until their simulation converges to something accurate. They can also play with the trade-offs of solve-time and accuracy for the simulator.

D — Supplemental Material

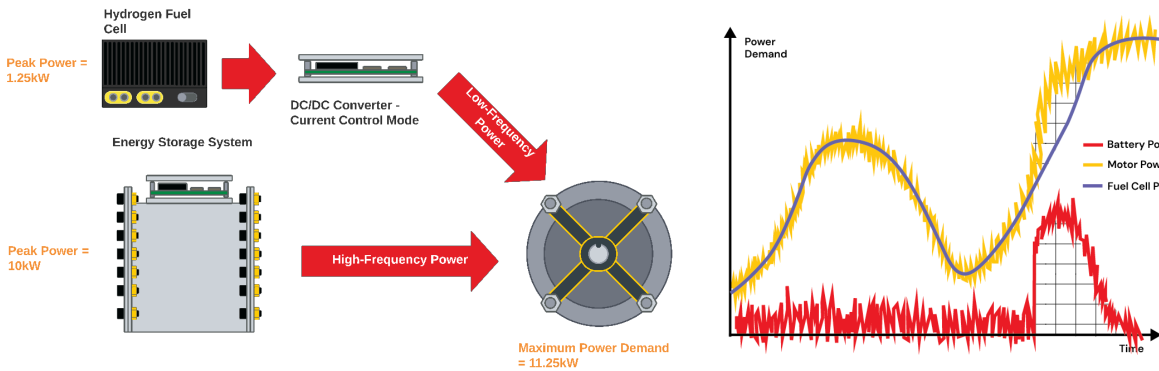
Additional Reference Figures



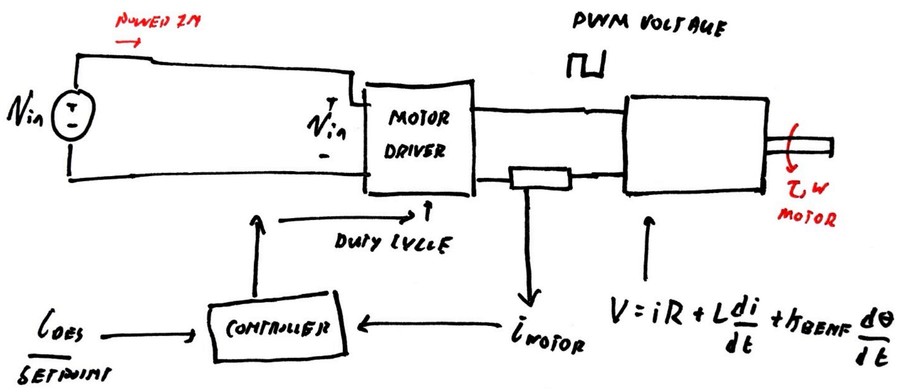
F.D.1—HOW A HYDROGEN FUEL CELL WORKS (HTTPS://AMERICANHISTORY.SI.EDU/FUELCELLS/BASICS.HTM).



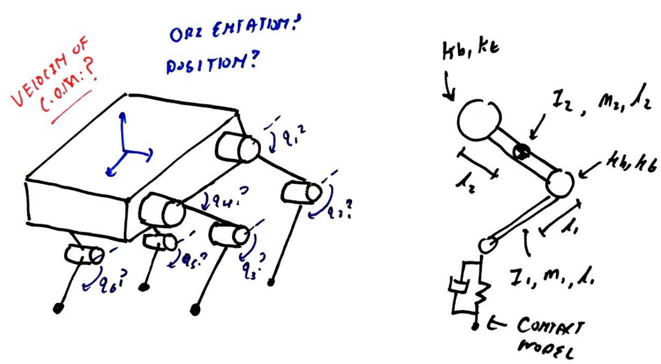
F.D.2—POWER DEMAND VS RESPONSE TIME OF A FUEL-CELL.



F.D.3—FREQUENCY SEPARATION OF POWER DRAW IN AN FCEV.

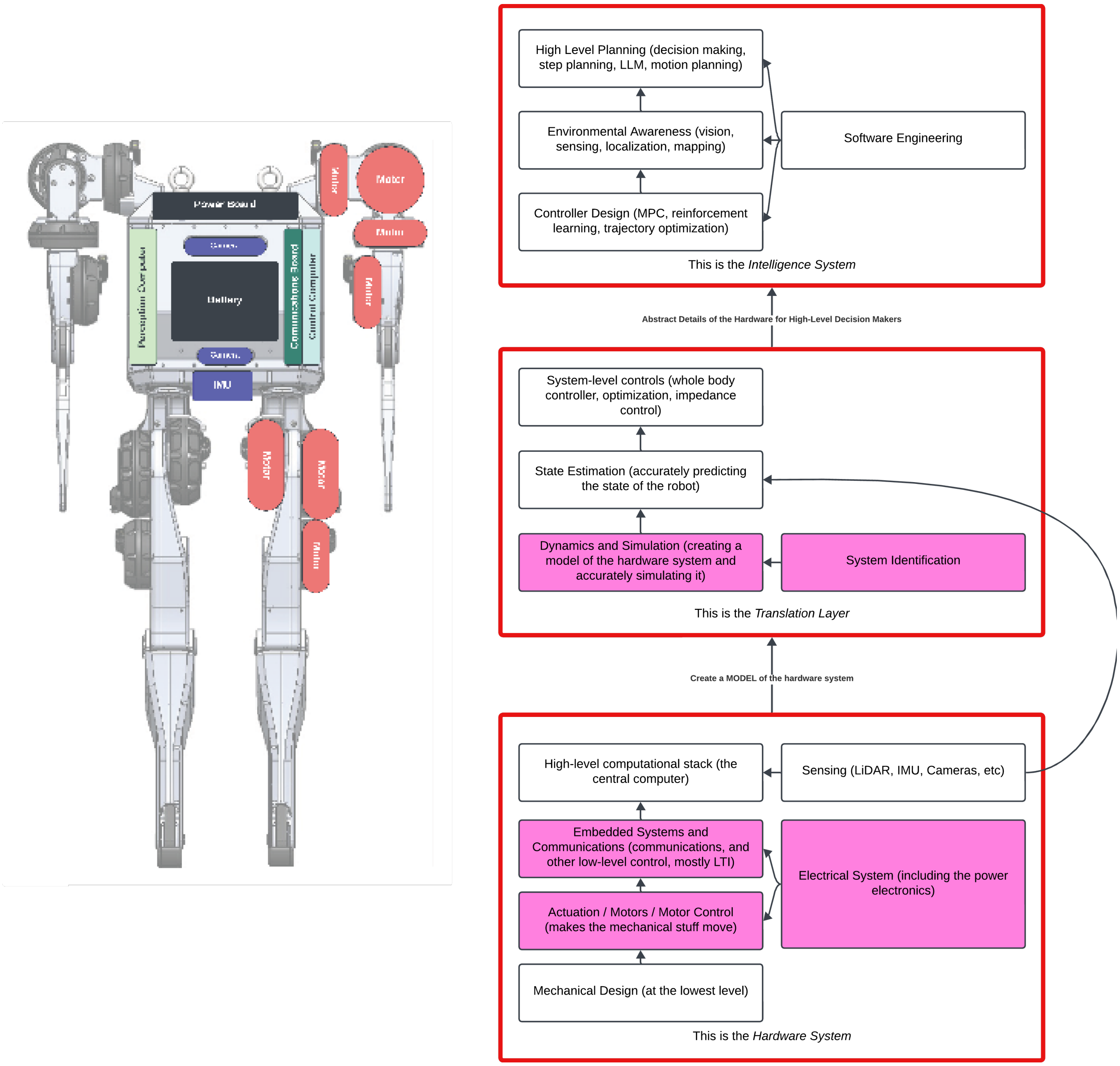


F.D.4—CLOSED-LOOP CURRENT CONTROL OF A DC MOTOR.

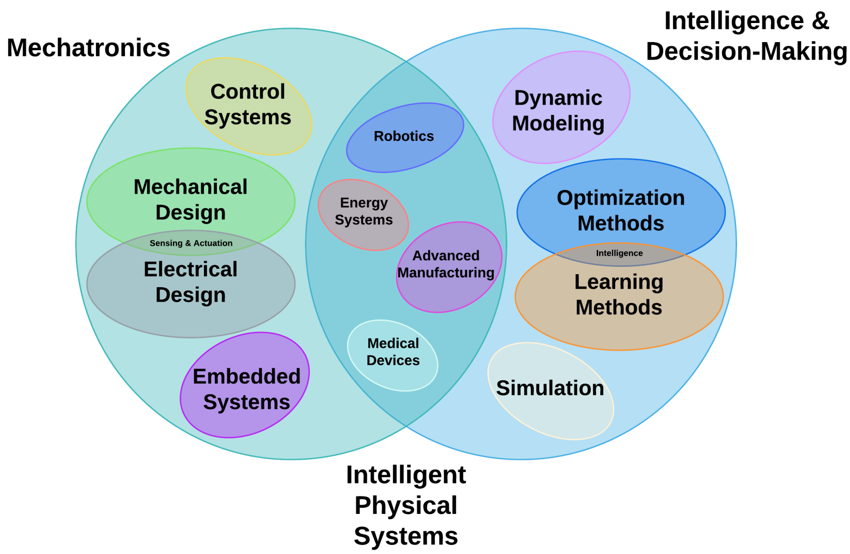


F.D.5—(A) STATE ESTIMATION INVOLVES ESTIMATING THE VALUES OF THE GENERALIZED COORDINATES OF THE SYSTEM INCLUDING VELOCITY, POSITION, AND ORIENTATION (B) SYSTEM-ID INVOLVES ESTIMATING THE MECHANICAL/ELECTRICAL PARAMETERS OF THE SYSTEM SUCH AS FRICTION AND INERTIA.

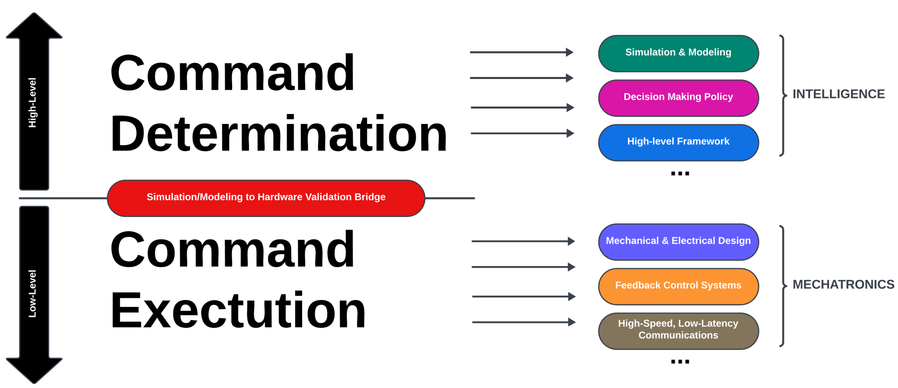
Diagrams and Descriptions of Intelligent Systems



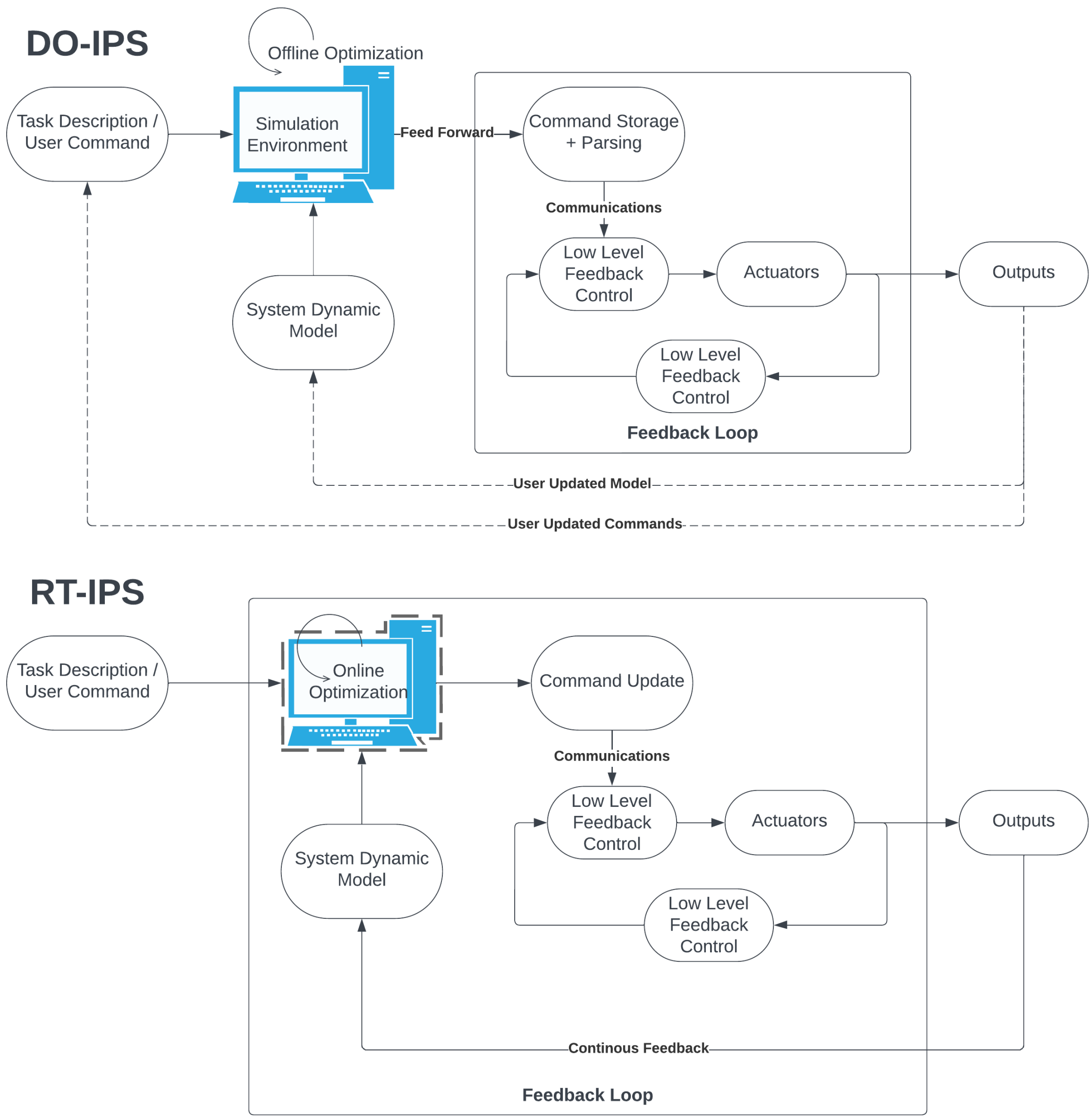
F.D.6—HUMANOID ROBOT AS A THREE-LAYER SYSTEM.



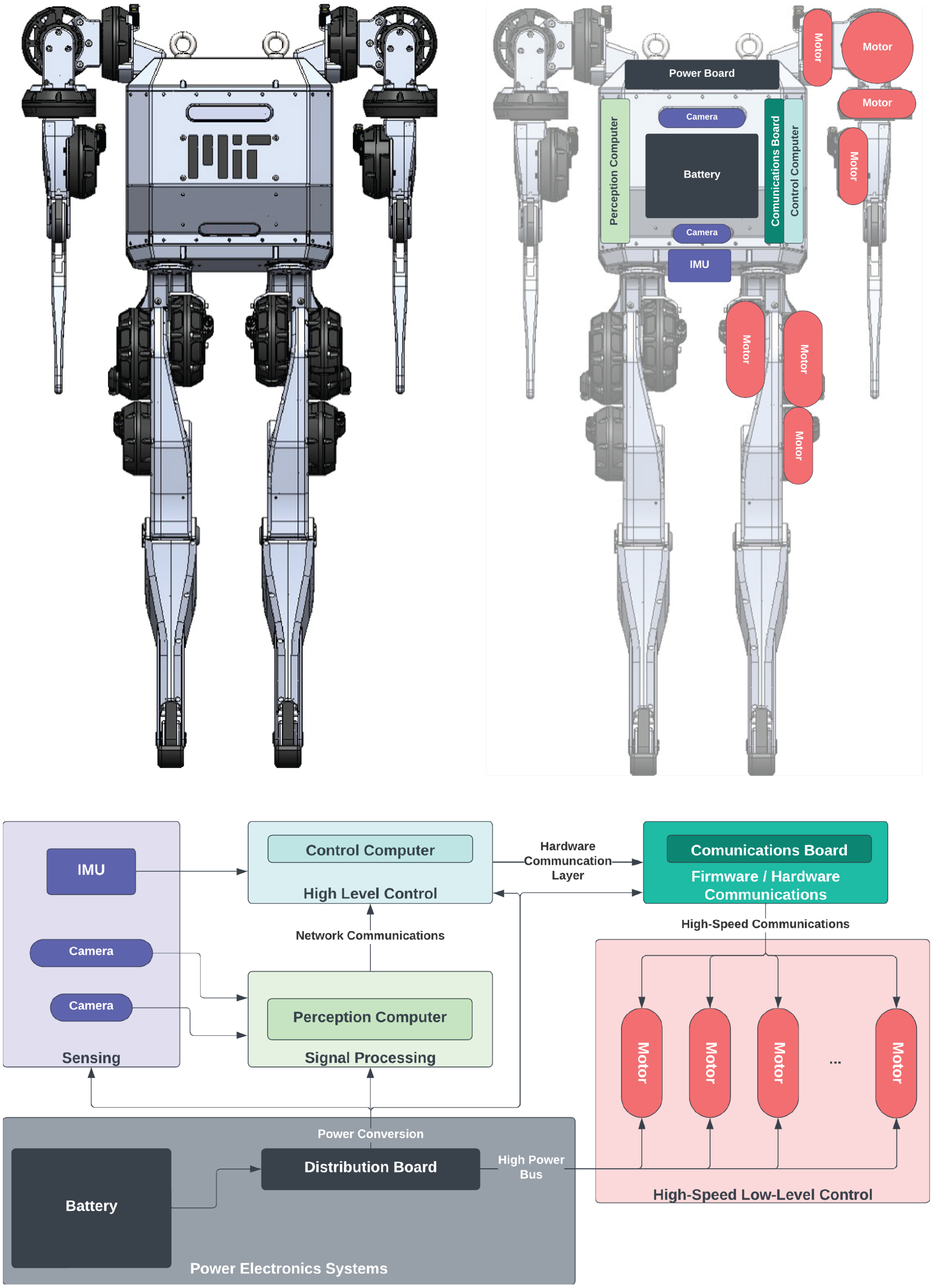
F.D.7—VENN DIAGRAM OF INTELLIGENT SYSTEMS.



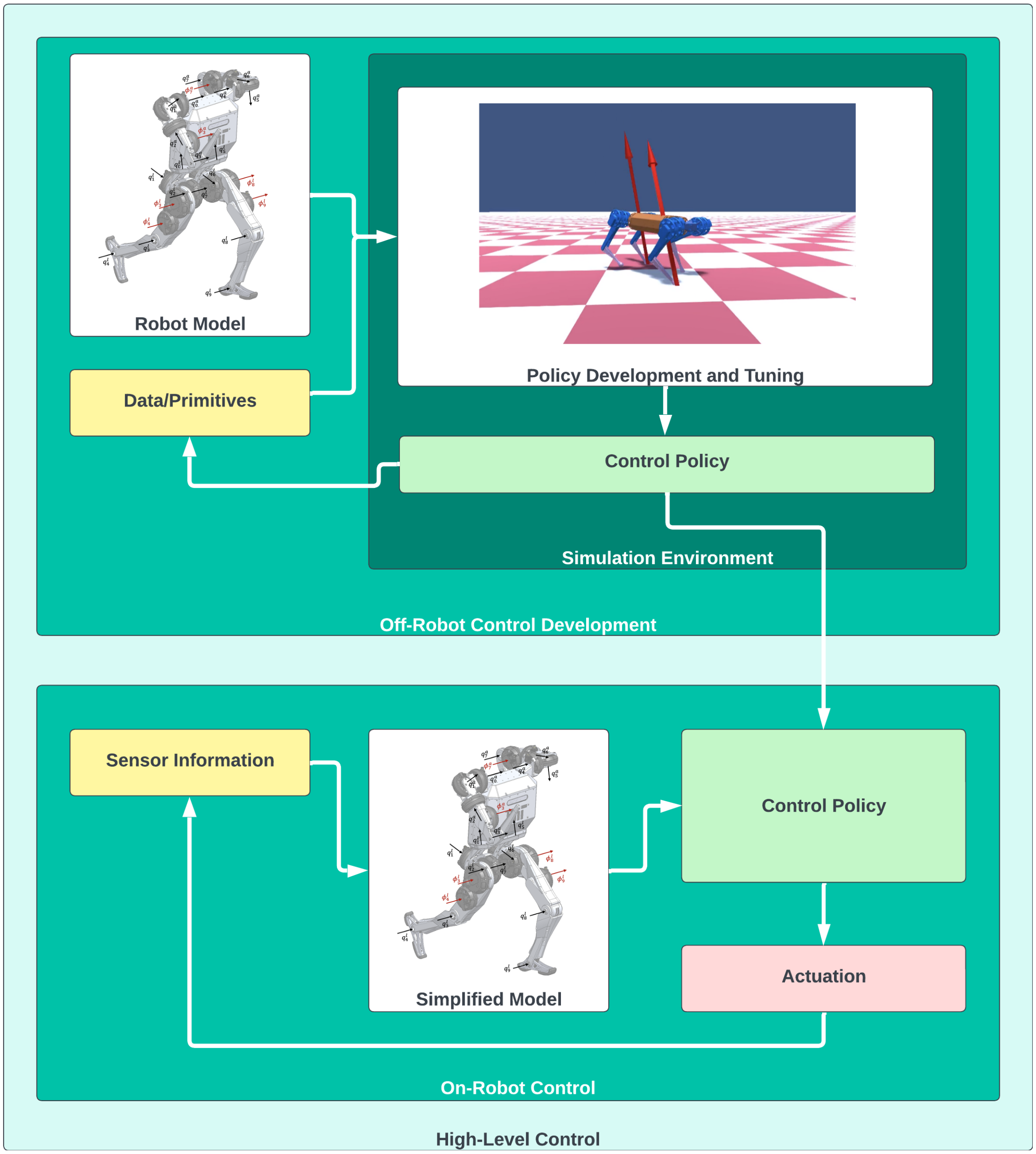
F.D.8—COMMAND DETERMINATION / EXECUTION STRUCTURE DIAGRAM.



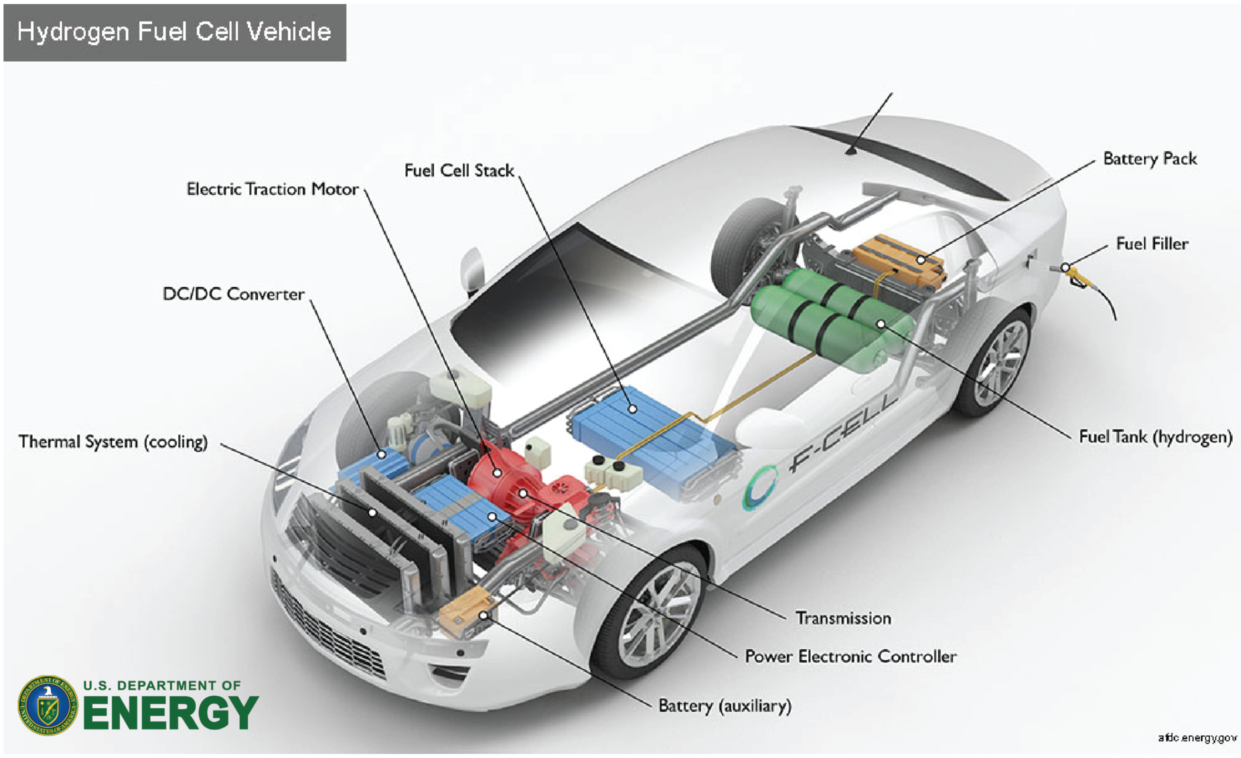
F.D.9—REAL-TIME VS. DECISION-OFFLINE INTELLIGENT SYSTEM.



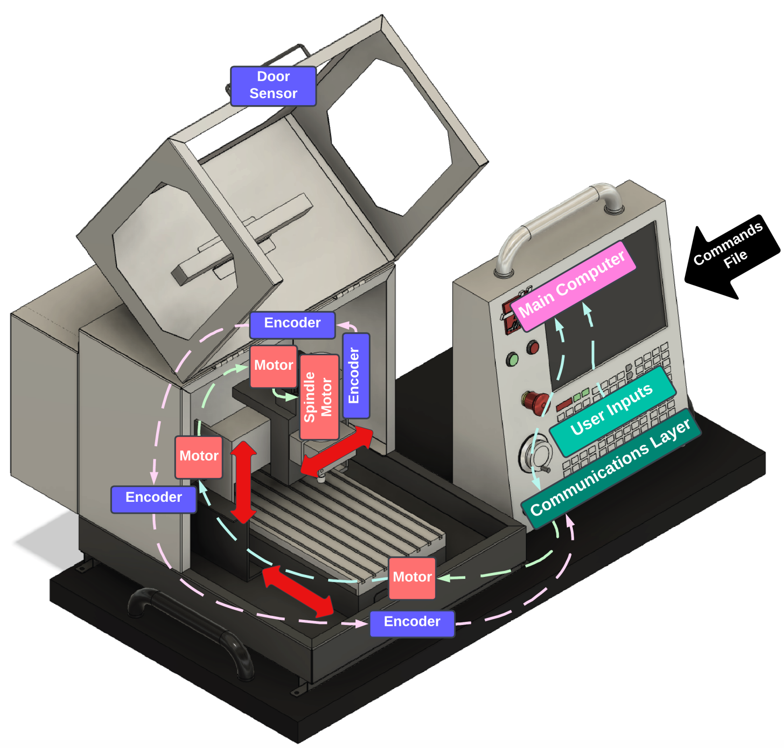
F.D.10—DIAGRAM OF COMPONENTS IN A HUMANOID ROBOT.



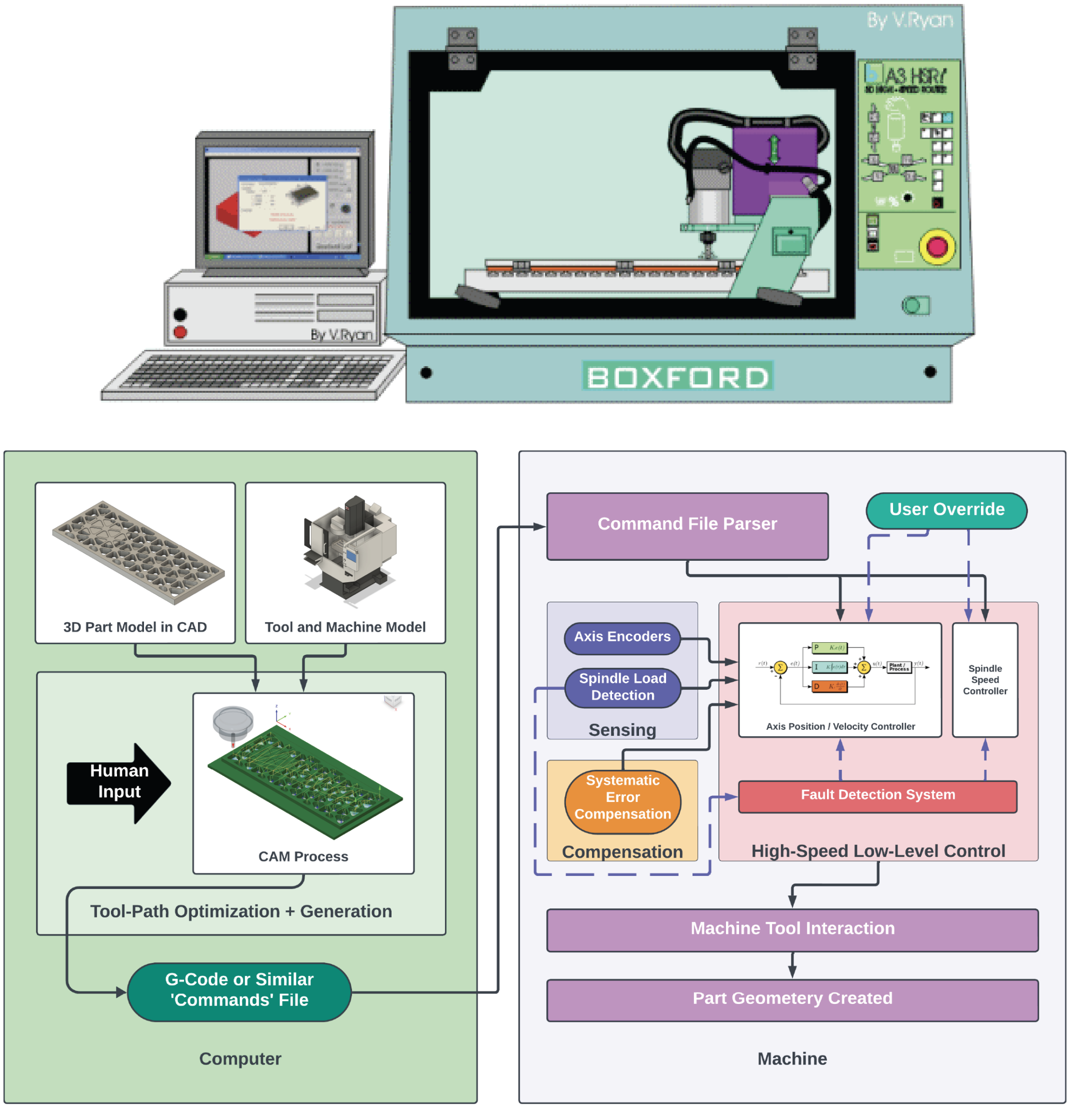
F.D.11—CONTROL ARCHITECTURE OF A HUMANOID ROBOT.



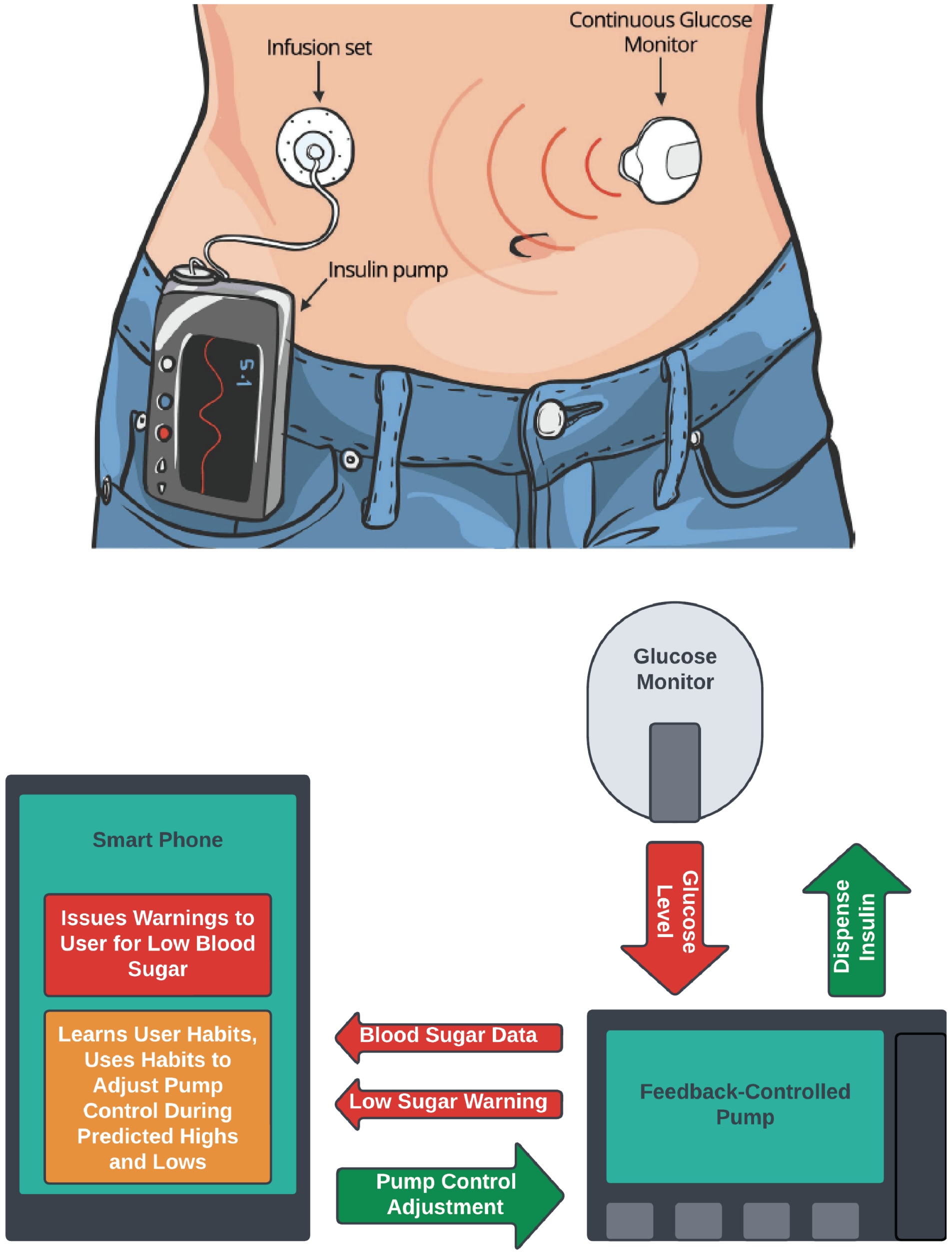
F.D.12—FUEL-CELL VEHICLE FROM (HTTPS://AFDC.ENERGY.GOV/VEHICLES/FUEL-CELL).



F.D.13—DIAGRAM OF A CNC MACHINE HARDWARE.

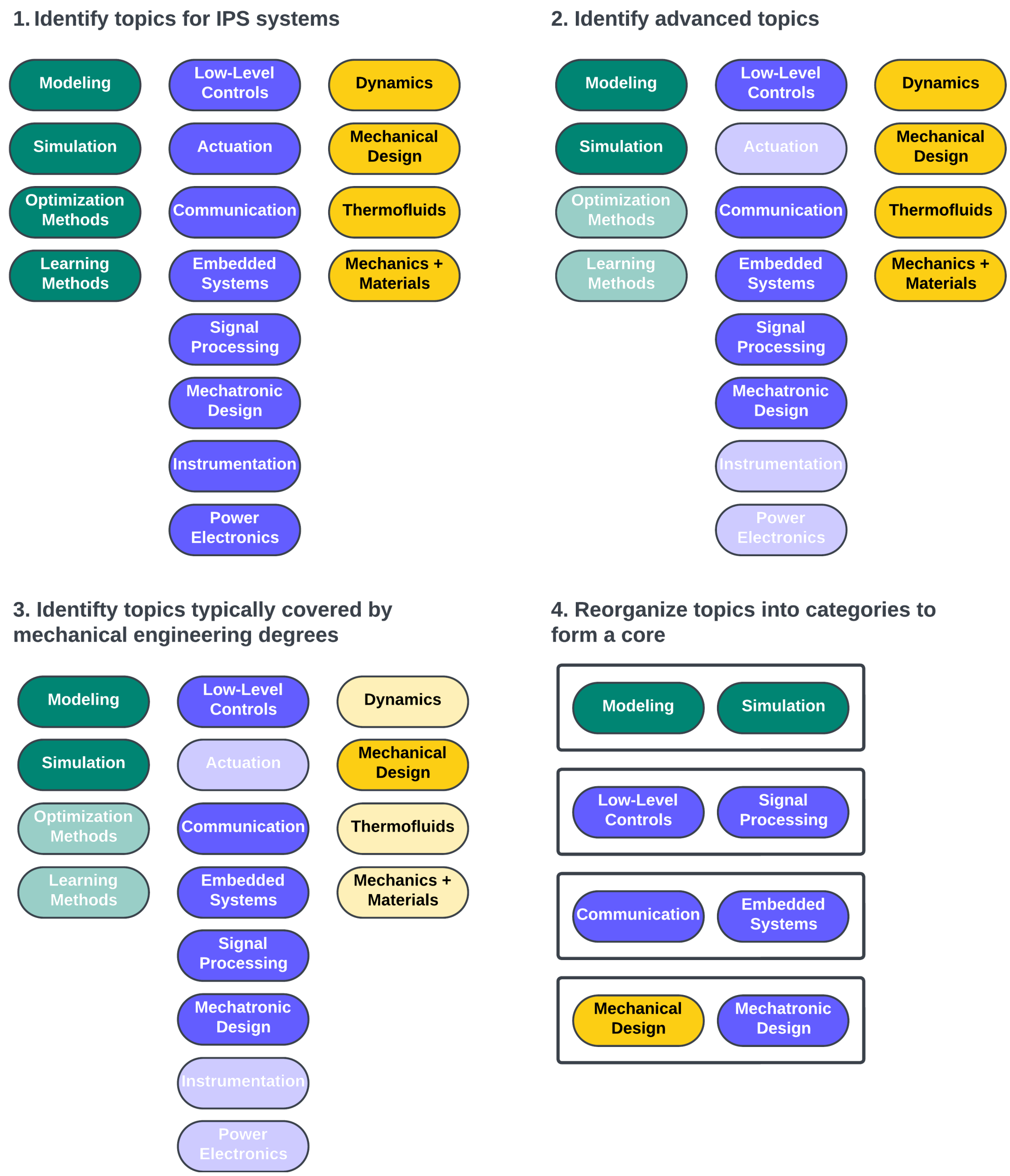


F.D.14—CONTROL STRUCTURE OF CNC MACHINE. INCLUDING GRAPHICS FROM V. RYAN, ARTURO URQUIZO, AND AUTODESK CAM.



F.D.15—DIAGRAM OF A CONTINUOUS GLUCOSE MONITOR. INCLUDING GRAPHICS FROM UMASS CHAN MEDICAL SCHOOL DIABETES CENTER.

Supplemental Figures on Curricula



F.D.16–IDENTIFICATION OF CORE TOPICS TO TEACH BASED ON THE ABOVE SYSTEMS (IN ADDITION TO ANALYSIS PRESENTED IN CHAPTER 1).