Computational Thinking through Modular Sounds Synthesis

Andrew M. Olney

2022-08-25

Contents

Welcome

This is the official website for “Computational Thinking through Modular Sound Synthesis”. This book will teach you computational thinking through modular sound synthesis (hereafter *modular*). You’ll learn how to trigger sounds, create sounds, and modify sounds to solve specific sound design problems and create compositions. Along the way, you’ll learn computational thinking practices that transcend modular and can be applied to a variety of problem-solving domains, but which are particularly relevant to information processing domains like computing.

If you’re wondering whether this is a book about computational thinking, or a book about modular, the answer is both: on the surface, most content is about modular, but computational thinking is a style of thinking reflected in the presentation of the material and gives it additional coherence. As you work through the book, you’ll become more proficient in computational thinking practices like decomposition, algorithmic design, evaluation of solutions, pattern recognition, and abstraction.

This book is *interactive*, which is why it is an e-book rather than a paper book. Throughout you will encounter examples, simulations, and exercises that run in your browser to demonstrate and reinforce key concepts. Don’t skip the interactive activities!

Chapter 1

Introduction

1.1 Why this book?

Let's start with why I'm writing it – that may help justify why you might want to read it. I got into electronic music in the 90's when I lived in London but never transitioned from DJing to making music, though several of my friends did. A few years ago, they started talking about modular, and in talking to them and trying to find out more about it, I realized a few things:

- The best books (to me) were from the 70's and 80's[1]
- Modular synthesis is really well aligned with *computational thinking*

If you've never heard of computational thinking and don't know much about modular, that last point won't make a lot of sense, so let's break it down.

Modular sound synthesis (modular) creates sound by connecting modules that each perform some function on sound. Different sounds are created by combining modules in different ways.

Computational thinking creates runnable models to solve a problem or answer a question. Models can be simulation models (e.g. meteorology), data models (e.g. statistics/data science), computation models (computer science), and perhaps other kinds of models.

How are they connected? Modular involves computational thinking when we:

[1]Old books that I like are Crombie (1982) and Strange (1983). Newer books of note are Bjørn and Meyer (2018), which gives a great overview of module hardware and history, Eliraz (2022), which gives a broader overview of issues related to musical equipment and production, and Dusha et al. (2020), which gives a modern but briefer introduction to modular than the older books. There are also some online courses (paid), but since I haven't taken them, I'm not listing them here.

- Simulate an instrument by reverse engineering its sound
- Create new sounds based on models of signal processing

So this book is about modular, but it approaches modular in a way that highlights computational thinking. I believe this deeper approach to modular will help you do more with it and also help you with other synthesizers or studio production tools. The computational thinking approach should help accelerate your learning of computational-thinking domains in the future.

Since computational thinking involves problem solving, this book is full of interactive activities that will let you hone your modular skills.

1.2 Computational thinking

Tedre and Denning (2016) present a nice overview of the history of computational thinking. Here's a brief summary.

When the field of computing was taking off in the 1950s, there was interest and discussion about how it was different from other fields (e.g. math). One argument was that computing involved *algorithmic thinking*, which is designing algorithms to solve problems (cf. programming), and this kind of thinking was unique to computing. Some even thought that this kind of thinking could improve thinking generally.

Computational thinking appears to have been coined in the 1980's by Seymour Papert and popularized in his book *Mindstorms* (Papert, 1980). Papert was a mathematician by training, and his approach was much broader than the algorithmic thinking folks that came before. Instead, his approach was empirical and embraced model building, with an emphasis on simulated microworlds containing robots (LEGO Mindstorms takes its name from this work).

Unfortunately today it's very hard to get agreement on what computational thinking is, so definitions tend to be squishy. Some want to reduce it to computer literacy, others to basic programming, and yet others to discovery learning with computers, etc.

I take a more unified view based on model building and problem solving. I define computational thinking as building a *runnable* model to solve a problem:

- For an algorithmic problem, this is a model of computation (the original computer science view)
- For data science/statistics, this is a model of data
- For general scientific fields, this is a model of a phenomenon or process

The model doesn't need to run on a computer, but to be a runnable model, it needs to be mechanistic. If you think it's crazy to talk about computational thinking without computers, consider that teaching computer science without computers has been part of the model curriculum for almost 20 years (Tucker, 2003; Bell, 2021).

So how do we learn to build runnable models to solve problems (i.e., how do we learn computational thinking)? Well, models are made of interacting elements, so we need to learn those elements, and we need to learn how the elements interact.

Once we know those things, we can customize general problem solving, which has the same basic steps (Polya, 2004):

- Understand the problem
- Make a plan
- Implement the plan
- Evaluate the solution

For any new domain, the big things to learn are the elements, how they interact, and the "make a plan" step of problem solving.

1.3 Modular synthesis

While we can pinpoint the invention of modular synthesis with some precision, it is useful to consider it in a broader context. It seems man has long been interested in instruments that incorporate automation or somehow play themselves. Wind chimes, which play a series of notes when disturbed by wind, can be found in the historical record thousands of years ago. Even before the complete electrification of instruments (synthesizers are electric by definition), there were numerous attempts to partially automate or model sounds using mechanical means, such as barrel organs, player pianos, or speech synthesis using bellows (Dudley and Tarnoczy, 1950) as shown in Figure 1.1.

Consider the difference between wind chimes or a player piano and this speaking machine. Neither of the former is a model of the sound but rather uses mechanical means to trigger the sound (later we will refer to this as sequencing). In contrast, the speaking machine is a well-considered model of the human speech mechanism.

Synthesizers using electricity appeared in the late 19th century. Patents were awarded just a few years apart to Elisha Gray, whose synthesizer comprised simple single note oscillators and transmitted over the telegraph, and Thaddeus Cahill (1879), whose larger Telharmonium could sound like an organ or various wood instruments but weighed 210 tons!

The modular synthesizer was developed by Harald Bode from 1959-1960 (Bode, 1984), and this innovation quickly spread to others like Moog and Buchla. The

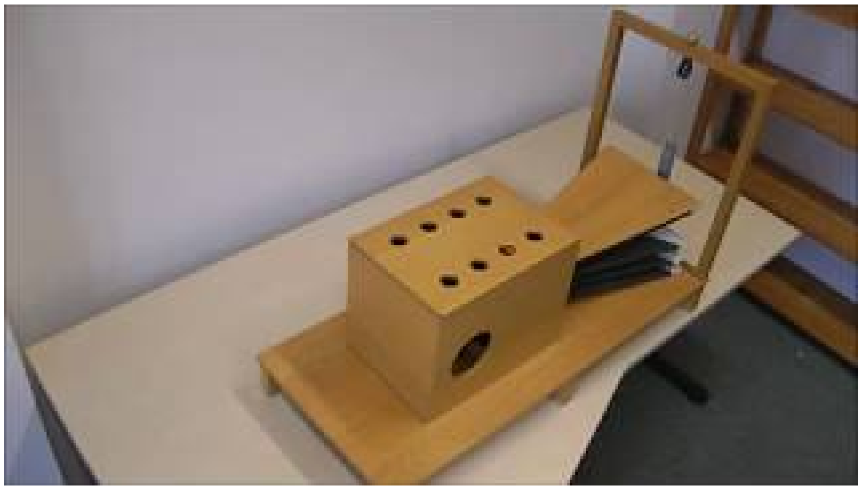

Figure 1.1: YouTube video of Wolfgang von Kempelen's speaking machine circa 1780.

key idea of modular is flexibility. This is achieved by refactoring aspects of synthesis (i.e. functions on sound) into a collection of modules. These modules may then be combined to create a certain sound by patching them together and adjusting module parameters (e.g. by turning knobs or adjusting sliders).

Tcherepnin); modules combined with patch cables to make sounds; a Moog modular then in today's cost is about \$96K. 1970s – Semi-modular synthesizers developed; can be patched like modular but are invisibly pre-patched, which patching overrides; a Minimoog then in today's cost is about \$10K.

1980s – Digital makes low-cost devices powerful; a Yamaha DX7 then in today's cost is about \$6K. 1990s – Computers become powerful enough that making/recording music is possible, reducing costs further; a Gateway computer with Cubase then in today's cost is about \$8K. 2010s – Modular synthesizers resurge with lower costs connected to widespread electronics manufacturing, smartphones, and the open-source movement; an ALM System Coupe modular system costs \$2.4K and VCVRack (virtual modular) is free.

Table here showing cost reductions over time

link to this somewhere https://120years.net/wordpress/

Reducing cost for wider adoption

Simplifying use for wider adoption Semi-modular Special purpose modules vs. general purpose modules

Diversification of uses A complete composition A single voice An effects box

Part I

Sound

Chapter 2

Physics and Perception

Chapter 3

Harmonic Sounds

Chapter 4

Inharmonic Sounds

Part II

Fundamental Modules

Chapter 5

Basic Concepts

Chapter 6

Trigger

Chapter 7

Create

Chapter 8

Modify

Part III

Sound Design 1

Chapter 9

Kick & Cymbal

Chapter 10

Lead & Bass

Part IV

Complex Modules

Chapter 11

Trigger

Chapter 12

Create

Chapter 13

Modify

Part V

Sound Design 2

Chapter 14

Minimoog & 303

Bibliography

Bell, T. (2021). CS unplugged or coding classes? *Communications of the ACM*, 64(5):25–27.

Bjørn, K. and Meyer, C. (2018). *Patch & Tweak: Exploring Modular Synthesis*. Bjooks. Google-Books-ID: xmrVvQEACAAJ.

Bode, H. (1984). History of electronic sound modification. 32(10):730–739.

Crombie, D. (1982). *The Complete Synthesizer: A Comprehensive Guide*. Omnipress Books.

Dudley, H. and Tarnoczy, T. H. (1950). The speaking machine of Wolfgang von Kempelen. 22(2):151–166. Publisher: Acoustical Society of America.

Dusha, T., Martonova, A., and Johnston, P. (2020). *Modular Sound Synthesis On the Moon*. Modular Moon. Google-Books-ID: llA4zgEACAAJ.

Eliraz, Z. (2022). Loopop's In-Complete Book of Electronic Music.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, New York.

Polya, G. (2004). *How to solve it: A new aspect of mathematical method*. Princeton University Press.

Strange, A. (1983). *Electronic Music: Systems, Techniques, and Controls*. William C Brown Publishers, 2 edition.

Tedre, M. and Denning, P. J. (2016). The long quest for computational thinking. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, Koli Calling '16, page 120–129, New York, NY, USA. Association for Computing Machinery.

Tucker, A. (2003). A model curriculum for K–12 computer science: Final report of the ACM K–12 task force curriculum committee. Technical report, New York, NY, USA.