UNIVERSITY OF CALIFORNIA, SANTA BARBARA

CHON: From physical simulation to musical gesture

by

Rodney DuPlessis

Project documentation submitted in partial fulfillment for the degree of

Master of Science
in
Media Arts and Technology

Committee:
Professor Curtis Roads, Chair
Professor JoAnn Kuchera-Morin
Dr. Karl Yerkes

April 2021

CHON: From physical simulation to musical gesture

ACKNOWLEDGMENTS

I would like to thank my committee for their support and guidance over the course of my time in the Media Arts & Technology program. I also thank Andres Cabrera and the Allosphere Research Group for their work on Allolib, which provided a solid foundation for CHON. I am grateful to my cohort and colleagues for our countless stimulating conversations and for holding space for inspiration and creativity to flourish. I want to express my deep gratitude to my wife for supporting me through the long days and late nights of frenzied coding and experimentation. Finally, I thank Suki and Appa for their understated moral support.

ABSTRACT

CHON: From physical simulation to musical gesture

by

Rodney DuPlessis

Physical metaphor provides a visceral and universal logical framework for composing musical gestures. Physical simulations can aid the composer to create musical gestures based in complex physical metaphors. CHON (Coupled Harmonic Oscillator Network) is a cross-platform application for simulating mass-spring networks and sonifying the motion of individual particles. CHON is an interactive instrument that can provide complex, yet tangible and physically based, control data for synthesis, sound processing, and musical score generation. This system builds on the idea of the traditional LFO by coupling the movement of multiple control signals using physical principles.

Contents

Chapter 1

Introduction

1.1 Overview

Coupled Harmonic Oscillator Network (CHON) is a real-time, interactive application for generating sonic gestures and textures using a simulation of a physical dynamical system as a musical interface. The physical system is a network of particles connected by a spring-like force. The user sets the system into motion by displacing a particle, which causes a chain reaction governed by Newtonian mechanics. The user can intervene in the evolving system, dragging and moving particles at will. The system generates complex yet tangible control data that can be used to drive parameters for sound synthesis and other purposes.

CHON is part of a larger interest of mine in using physical metaphor in music. I have previously used intuitive and algorithmic methods to express physical metaphor in my music. CHON represents a hybrid approach, allowing intuitive interaction with the algorithmically generated simulation in real-time. The musical gestures that CHON can create are chaotic, but they seem nonetheless intelligible because of their connection to the physical world.

The coupled harmonic oscillator is a fundamental model in physics that has applications in many areas of research. It is an extension of the simple harmonic oscillator, which can be represented by a mass on a spring. The simple harmonic oscillator moves sinusoidally, but when coupled via a spring-like force to another harmonic oscillator (or many), complex superpositions of sinusoidal waves emerge. CHON goes further, allowing the masses in the coupled harmonic oscillator network to move in 3 dimensions and to be arranged in a

2-dimensional grid.

The application is written in C++ and uses the Allolib framework extensively. It is open source[1], under a GPLv3 license, and runs on Linux, MacOS, and Windows. It uses an explicit numerical method to solve the discretized equations of motion of the particles.

The visual interface is a 3D rendering of the particle system. The user interacts directly with the particles in the visual simulation using a computer mouse. A 2D graph can also be displayed which visualizes the displacement of each particle along a given axis. CHON has an internal synthesis engine that allows the user to sonify the movement of the particles. The instrument also generates a stream of OSC data from each particle, making it a versatile tool for generating up to 100 control signals that are linked by physical laws.

I have used CHON to generate sounds using the internal sound engine as well as using the OSC data it broadcasts to control external synthesis and notation software. For example, I used CHON to write a score for two pianists. The piece, *Pandæmonium*, was premiered by HOCKET in August 2020.

I am still exploring the musical possibilities afforded by CHON, and the possibilities of extending and improving CHON. I will discuss some of these future directions at the end of this paper.

1.2 Related work

Physical modeling is a massive interdisciplinary field of research spanning industrial, scientific, and artistic applications. In music, the vast majority of physical modeling is used for re-synthesizing believable facsimiles of real instruments[1][2]. The same technology that allows these recreations has also been applied to create fantastical virtual instruments of unreal proportions or configurations[3][4]. The first example of physical modeling synthesis of instruments was demonstrated in 1971 by Hiller and Ruiz[5]. Today, physical modeling synthesis, driven by increasingly powerful computers, can achieve very convincing results. On the other hand, physical modeling interfaces often suffer from problems of control and complexity; the high number of parameters in a given simulation may be overwhelming for performance purposes[6, p. 288][7].

There are many approaches to the design of systems for physical modeling synthesis including modal synthesis, waveguide schemes, source-filter models, and mass-spring

[1]The source code can be accessed at https://github.com/rodneydup/CHON

methods[8]. CHON falls into the realm of mass-spring methods. Mass-spring methods use Newtonian mechanics to simulate the dynamics of particles with mass and inertia connected by spring-like forces[6, p. 271]. This is the type of simulation that drives CHON.

The goal of most mass-spring applications is to simulate a physical body that can be described as a network of coupled masses. This, like every physical model, is a simplification of reality. Nevertheless, many types of physical bodies fit well into this model. The mass-spring model does a good job at representing the way that waves propagate through solid media such as strings and vibrating surfaces (like drums). The mass-spring model suffers from scalability issues, however. The cost of computation can increase quickly as more particles are used in a given simulation. Considering that physical modeling synthesis often needs to run at audio rate (updating 44,100 times per second) and that higher particle density generally results in a more realistic simulation, this is a difficult problem.

The focus of most mass-spring simulations is on the object that the system simulates. In other words, the aim is that the gestalt of the movements of the particles generates sound that resembles the sound of the object being modeled. This is where CHON differs from other mass-spring applications (and indeed the majority of physical modeling synthesis projects in general). Rather than create a sound object from an amalgam of particle interactions, CHON was designed with the goal of exploring interconnected particles as sound objects unto themselves. I am also interested in larger-scale gestures, phrases, and forms that can arise from the overlapping undulations of the coupled particles, but the foundation of CHON remains the individual sound-particles themselves.

Claude Cadoz and others at ACROE (Association pour la Création et la Recherche sur les Outils d'Expression) in Grenoble pioneered the use of the mass-spring model in physical modeling synthesis and musical creation. They generalized mass-spring models into what they call a *mass-interaction scheme*, where, in addition to more standard mass-spring situations, particles may be lumped into strange configurations and the forces connecting them may be non-linear[9]. ACROE's CORDIS-ANIMA system was one of the first digital synthesis applications for physical modeling, paving the way for future research. CORDIS-ANIMA is a sophisticated and technical modular system for simulation and sonification/visualization of mass-interaction models[9]. In CORDIS-ANIMA, the user can specify many physical parameters of each particle, including how and which particles are coupled to each other, their initial position and velocity, their mass, whether they are fixed or can move. Once this is configured, the user and then set the simulation into motion and they can watch and hear the results.

After decades of development, CORDIS-ANIMA is a massive software and one can pay to attend courses at ACROE to use and learn it. CHON is lean by comparison, but a user can download it and set it into motion and sound in less than two minutes. With a little more time, one can connect CHON to other audio software or hardware and begin exploring its full potential. My focus with CHON is to make it approachable. Even as I continue to add refinements and features, I will make sure the default use case of CHON is intuitive and immediately expressive.

Beyond sound synthesis, some researchers have recognized the potential for physical modeling in computer-assisted composition[10][11] [12]. Claude Cadoz wrote a piece called *pico..Tera* using the CORDIS-ANIMA and GENESIS systems[11]. In documenting the work, he spoke of the "instrument" and "instrumentalists" that were programmed in the system. Essentially there are virtual actors that excite an instrument in the CORDIS-ANIMA simulation. His description reveals an intricate sequence of events programmed in the system like a Rube Goldberg machine made from thousands of particles and dozens of larger-scale structures. Indeed, his stated purpose was to demonstrate the possibility of enacting an entire piece of music in a physically modeled simulation without intervention during the execution and without post-treatment.

Like Cadoz, I am interested in the possibilities of physical modeling beyond the synthesis of individual sounds, extending physical model methods to computer-aided composition[13]. CHON can be set into motion to create an infinite cascade of sound events. Certain configurations can produce quite complex results, but CHON surely can not hope to approach the diversity of programmed situations that Cadoz achieved with CORDIS-ANIMA in *pico..Tera*. CHON is a specialized case of highly interactive coupled mass systems in string-like and surface-like arrangements. Unlike the scripted system that defines *pico..Tera*, CHON is meant to be tampered with, interrupted, performed on, broken, energized, and generally manipulated by the user. Thus CHON benefits from the physical plausibility of connected sound events that Cadoz' methods provide, while also allowing for the rich and impulsive dimension of human intervention.

CHON is free, open-source, and easy to find and download on GitHub. This accessibility is at the heart of all of my projects. Unfortunately, I could not find any way to hear *pico..Tera*. Nor could I find any way to buy or download CORDIS-ANIMA. There are some projects that implement aspects of CORDIS-ANIMA in other platforms and programming languages. Mark Pearson's Tao system[14] and Cyrille Henry's PMPD pure data objects[15] are some early examples. Tao uses a scripting language as its interface and is non-real-time. PMPD allows for some real-time control and being in the pure data environment gives it significant flexibility. Mi-creative, an initiative founded by Jérôme Villeneuve and James Leonard, has some nice open-source tools inspired CORDIS-ANIMA[16]. These projects are flexible and powerful, but they all require a certain level of programming expertise to use. By comparison, CHON is more approachable for the non-specialist computer musician.

Chapter 2

Physical metaphor in music

The primary goal of CHON is to generate musical gestures based on the dynamics of a particular physical system – a coupled oscillator network. This is part of my broader investigation into physical metaphors in music. In this section I will discuss why and how I use physics to shape my music. I will then explain how this motivated the creation of CHON.

I am interested in physics and physical metaphor in composition because it can provide a visceral and universal logical framework for music. I will parse this statement in reverse order starting with the idea of a logical framework.

2.1 Logical Framework

I use the term logical framework to describe a set of rules that more or less constrain the possible states and transformations of a system. A logical framework is also a mental anchor that allows one to make inferences and predictions about the behavior of the system. The system in question is music and the logical framework constrains the structure and progression of music. This is closely related to what Fred Lerdahl calls *musical grammar* [17] and is comprised of a network of overlapping *causal logic* [18].

Composers apply logical frameworks to shape music at every level, to cull the infinite possibilities of arranging sound in time. Listeners employ logical frameworks as they discover and parse patterns and relations in music. The degree to which the frameworks of both parties align can vary greatly. These frameworks can also be active or passive, conscious or unconscious.

A logical framework suggests a set of causes and effects, which create expectations. A dominant chord in tonal music creates tension which seems to cause the release of tension via a tonic chord just like knocking a ball off of a table causes it to fall. A build-up in Electronic Dance Music seems to inevitably lead to a bass-drop the way over-inflating a balloon causes it to pop. These causal networks are illusory, but nonetheless crucial to musical reasoning. As Curtis Roads [19, p. 328] points out:

> An impression of causality implies predictability. If listeners are not able to correlate sonic events with any logic, the piece is an inscrutable cipher.

Music must rely on a logical framework to form a coherent narrative and carry meaning. The "meaning" I am referring to here is not necessarily tied to programmatic or referential elements. Narrative meaning can arise from the relationship between elements, regardless of the identities of the elements themselves. Byron Almén [20, p. 13] suggests that the abstract nature of musical narrative is one of its strengths (relative to literary or dramatic narrative, for example), allowing it to express a narrative without being bound to specific characters and settings.

A logical framework can emerge from preexisting stylistic conventions, extra-musical referential inheritance, and algorithmic design, among other things. A framework is often formed from a mixture of sources. This creates a tangled web of rules and relationships that allows room for multiple potentialities as they come into conflict. Conflicting rules create crisis, which opens up multiple possibilities for resolution. This in turn drives the narrative forward. This is an important ingredient for creating a compelling narrative as Almén [20, p. 52] notes:

> The psychologically compelling nature of the narrative arises from this perspective of progressive temporality: one cannot know the outcome of a narrative conflict in advance. There is always more than one possible resolution of a crisis, and to reduce out this uncertainty is to remove the essence of narrative.

Some logical frameworks are hopelessly complex and esoteric, such that the listener will almost certainly not be able to follow the logic of the piece's structure. On the other hand, a logical framework can be so simple that there is no uncertainty as to where the piece will go next. Complete chaos and complete predictability will bore the listener. It is up to the composer to craft a work that balances the two.

The composer's logical framework can be influenced by overt compositional strategies as well as more implicit biases. Some of these factors may be more or less unconscious, but the most effective composers are self-aware enough to balance them judiciously to shape their music into a compelling narrative. It is not guaranteed that a listener will hear the narrative or the structures that the composer intends. In fact, when one considers the multitude of possible influences on a listener's experience, from past experience to current mood, it seems highly unlikely. I imagine the relationship of composer's and listener's respective logical frameworks to be asymptotic: they can approach each other, but never completely align. This disconnect between the composer's intention and the listener's experience is part of the magic of music. Nevertheless, the power of shared experience is strong, and composers can utilize that to present persuasive musical proposals to the listener.

2.2 Universal

In some music, there is a logical framework which has been agreed upon (explicitly or implicitly) by composers and listeners. This is a cultural logic, it is not universal or innate to human experience. At one time, the cultural framework of tonal music provided a rapport between Western composers and listeners, which in turn allowed composers to play with expectations to convey musical narratives. Because it had been decided that a dominant chord should resolve to the tonic, it meant something when the dominant chord led to something else. It has been some time now since composers in the Western tradition began to radically break from this cultural agreement and explore other logical frameworks.

Musique concrète, initiated by Pierre Schaeffer in the late 1940s, was perhaps the furthest removed from traditional Western Classical music. Whereas some new directions changed the rules of dealing with musical material (melody, harmony, rhythm), musique concrète expanded the very category of what could be considered musical material. Western Classical music had no framework at all for dealing with recorded sounds such as paper ripping or doors creaking. This medium provided a great deal of freedom for the composer to explore the possibilities of sound organization. There were no rules of pitch or harmony, no rhythmic pulse in most cases, no restriction on the palette of sounds available to the composer.

This freedom came at a price: listeners and composers could no longer benefit from the established context of tonal music. The challenge now for composers was to write music that drew from other frameworks or that could establish its own context. In the case of musique concrète, the very sounds used were often recognizable and so the context was provided by everyday soundscapes. Acousmatic music plays with that very recognizability to create meaning. This is only one example. Hundreds of genres, styles, and practices emerged in the 20th century, each with their own logic.

In my own early work, I found myself inventing a new logic for almost every piece. Some were mathematical, others more vague. With these works, I needed to establish the grammar of the piece within the piece itself, or else rely on extensive program notes to explain the context of the piece (a sure way to alienate listeners). Establishing the logic of a piece within itself typically involves a great deal of repetition and consistency. Like Pavlov's bell, if a sound is consistently followed by another sound, they will come to seem related by some causal logic. This way, the process of the piece is made clear.

This approach of establishing new logic for each piece is way to bridge the gap between composer intention and listener experience, but it's an inefficient use of time and listeners only have so much patience. I wanted to find a common ground, something universally understood, to build the frameworks of my future pieces. I found this in physics and physical metaphor. Each piece has its own idiosyncrasies and can add something unique to this basis to build its logical framework, but physics provides the foundation.

Unlike cultural logical frameworks or invented systems, physical metaphor can be understood by anyone because we are all physical beings. We all know intuitively that a ball thrown into the air will fall back down, and that some objects can be moved if we push on them. Our imagination allows us to make inferences based on this knowledge. If we hear a loud crash outside, we understand that some massive objects probably collided. We even do this unconsciously. If I hear a sound getting rapidly louder from behind me, I will move out of the way reflexively because on some level I understand that something is approaching at a high velocity and I can further infer that it will collide with me and cause me harm if I do not move. This is not a cultural or idiosyncratic response. I suspect every person in possession of their faculties would react the same way.

These are just two examples of how sound can trigger our knowledge of physical interactions and even arouse images and narrative sequences in our imagination. Much more complicated and interesting sonic interactions can be built-up from this basis, but these basic examples illustrate not only the universality of physical logic, but also its unconscious or semi-conscious nature.

2.3 Visceral

All humans understand certain basic physical principles. Some of these physical principles affect our behavior on a fundamental level. We learn about the physical principles of the world as we grow up by observing and by doing. Research into mirror neurons has shown that observation and performance may not be entirely distinct. If seeing or hearing some kinds of events accesses deeper parts of our brains beyond our reasoning centers, then that presents a fertile ground for musical communication.

Mirror neurons are neurons that fire similarly when one performs an action and when observing that same action being performed [21]. It has been theorized that mirror neurons play a part in empathy, learning new skills, and understanding others' intentions and actions. Much research has been done regarding mirror neurons in humans and primates, mostly focused on visual stimuli [22]. However, it has also been shown that the sound of some actions being performed can activate mirror neurons as well [23]. Recent research has shown that mirror neurons also fire when listening to music, and that the communication of emotion in music arises at least partially from empathically feeling the motion of the music [24]. Music quite literally moves us.

Obviously, many kinds of music using varied logical frameworks can elicit this response. Music has always had the ability to move us. But if we feel music with the same mirror neurons that help us to understand the physical world, then it seems that music based on physical metaphor could access this level of cognition more directly than others. Music can take advantage of the fact that we all understand certain physical principles, and we understand them viscerally.

Of course we do not all understand physics completely or physicists would not need to attend university for years. Science communicators like to surprise audiences with experiments that demonstrate lesser known or poorly understood physical laws. This limit to our understanding provides a fertile ground for creating suspense, surprise, uncertainty, and intrigue in using physical metaphor in music.

2.4 Physical Metaphor in Music

What do I mean by physical metaphor in music, exactly? Metaphorically speaking, one can treat a sound as if it had physical properties such as mass, momentum, inertia, internal energy, heat, entropy, gravity, and others. Physical metaphor involves shaping the evolution of a sound, or a group of sounds, according to how real physical objects with these properties behave. For example, Newton's first law of motion states that an object at rest will remain at rest and an object in motion will remain in motion unless acted upon by an external force. This is the law of inertia. If we follow this law with sounds, then, for example, a sound at rest will remain at rest unless it is excited by some force, which could be another sound entering the field and colliding with it. The momentum of a sound and the metaphorical collision of sounds can take shape entirely in the imagination of the composer, and this is a rich domain for creativity.

Edgard Varèse was one of the first composers to appreciate this potential well of musical expression. He spoke of sound-masses, points of attraction and repulsion, and shifting planes, employing physical language extensively to describe his vision of the future of music [25]. Iannis Xenakis took up the mantle in the second half of the 20th century, employing computers to help him realize physical metaphors based especially in statistical mechanics [26]. Curtis Roads carried this approach into the 21st century, pioneering granular synthesis as a means of composing with clouds of sound, which, in his own compositions, evolve and interact in deeply physical ways [19].

So far my compositional approach to physical metaphor has been most fully realized in *De Rerum Natura*, which I first completed in 2019 and revised in 2020.

The piece begins with a metallic sound crashing onto the soundfield, throwing up a flurry of shrapnel that slowly settles onto a calm glassy surface. A new sound then swirls into the scene and kicks up the dust again. Emerging from the debris, a fluttering sound seems to strike a match that heats up another sound that builds up to an explosion. The fallout of the explosion is swept away unexpectedly by another clattering sound which sets up a domino effect leading to a door opening into a noisy soundscape. This is just one interpretation, but it illustrates how every sound in the piece has the illusion of being part of a physical chain of events.

In *Coacervate* (2020) for Violin and Electronics, different physical interactions are explored. The piece takes place inside an imaginary test tube or beaker filled with various liquid components. The sounds in this piece tend to float through the musical texture before suddenly and sometimes violently coalescing into droplets. Other times the sound field seems to get stirred up as a new component gets added into the mix. This all reflects the chemical process which gave the piece it's name.

While this kind of imagined field of interactions among sounds can be fruitful, and I will continue to compose in this way, I also have a desire to express more complex physical gestures in my music that require some computation. This is why I have created CHON.

Chapter 3

Physics

CHON is built on the idea of generating musical material from a computer simulation of a real physical system. The system in question is that of coupled harmonic oscillators. A system of this kind is formed by connecting (coupling) two or more oscillating nodes (harmonic oscillators) via a spring-like force such that they exert a force on one another. The physics governing this system are that of classical mechanics. It can be completely described by the simple laws of motion posited by Isaac Newton in 1687 [27], and by Hooke's Law [28].

I was inspired to create CHON when I saw two pendulums connected by springs oscillating in such a way that the two pendulums appeared to be, in a sense, passing an oscillation back and forth. see figure 3.1

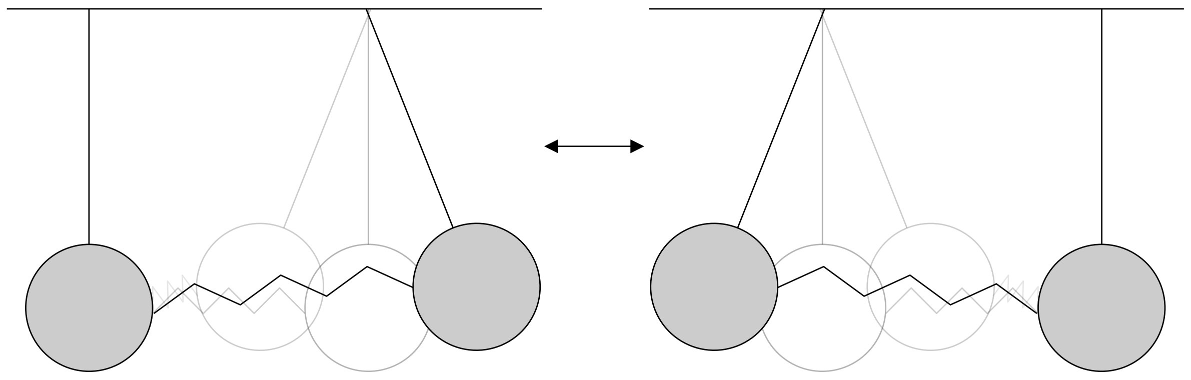


Figure 3.1: Coupled Pendulum - the amplitude of one pendulum's oscillation increases while the other decreases. Then, the process reverses.

This was in fact a superposition of the two modes of oscillation of the system, but the appearance of an exchange of energy back and forth was what captivated me. I imagined sounds forcing each other into activity and contrapuntal structures in which sounds, instruments, and textures passed musical energy from one to another in a cascade of antiphony. In order to experiment with this kind of musical structure, I began working on CHON.

Before looking at coupled harmonic oscillator systems or networks, I will describe the case of a simple harmonic oscillator. A simple harmonic oscillator can be most simply represented by a mass or particle on a spring. There is a point of equilibrium for this mass where it will remain unless it is perturbed. If the mass is moved away from equilibrium, it will exert a force proportional to the displacement. If it is then released, it will move back toward equilibrium, pass it, and continue to oscillate at a fixed sinusoidal frequency (determined by the spring stiffness and the mass).

The behavior of springs (within elastic limits) is well described by Hooke's Law. Hooke's law can be stated as: the restoring force of a spring is linearly proportional to to its displacement. Mathematically:

$$F = -kx$$

Where k is a constant "stiffness" of the spring, and x is the displacement of the spring away from equilibrium. If the spring is at equilibrium, then the spring will not move unless acted upon by an external force. To stretch or compress the spring away from equilibrium by x, one would need to exert a force equal to the product of k and x. The restoring force at that point x will then be equal and opposite to this force. If the spring is released, it will accelerate toward equilibrium. Furthermore, provided the spring is not over-damped, the spring will be set into *harmonic motion*, meaning it's displacement will follow a sinusoidal curve. If the spring is not damped at all, this harmonic motion will continue at the same amplitude indefinitely.

Newton's laws of motion can also be useful in describing the simple harmonic oscillator, and they come into play in the software implementation of CHON. These laws state:

1) A body's velocity remains constant unless acted upon by an external force.
2) The force exerted on a body is the product of mass times acceleration.
3) A body will exert an equal and opposite force to any force exerted upon it.

These laws describe the vast majority of macroscopic objects that we are familiar with.

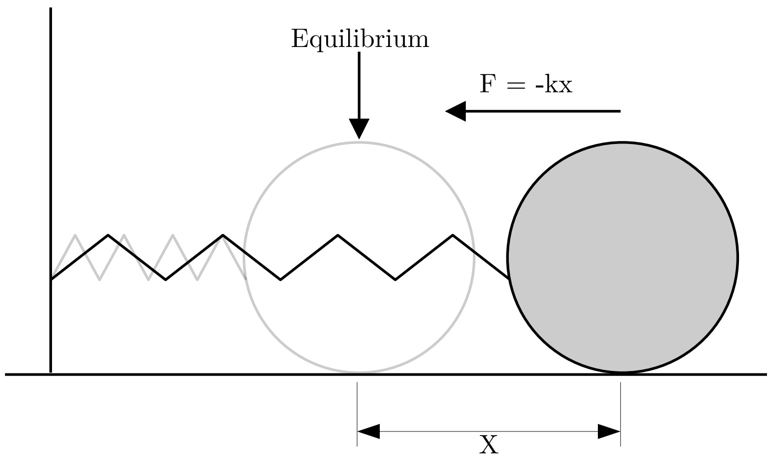


Figure 3.2: Simple Harmonic Oscillator - The restoring force is $F = -kx$. The motion is sinusoidal.

Using these laws, one can predict and model many kinds of physical interactions between objects.

The first law is sometimes referred to as the "law of inertia." Aside from the way it is stated above, another way of positing it is that an object at rest will stay at rest and an object in motion will stay in motion. In a simple harmonic oscillator, the oscillating particle will not move from equilibrium until it is displaced by an external force. Once it is in motion, it will continue to be in motion unless a damping force is exerted on it.

The second law gives us the mathematical means to calculate important properties of an object in motion. Stated mathematically, it is:

$$F = ma$$

This equation, when combined with Hooke's law, can give us more information about a harmonic oscillator, such as its acceleration (which is especially important in the implementation of CHON). This can be shown by combining Hooke's equation with Newton's:

$$F = ma = -kx$$

And re-arranging to find our desired property:

$$ma = -kx$$
$$a = \frac{-kx}{m}$$

The second law also allows us to model the mass-spring problem by incorporating a mass term into the equation. We will continue to treat the mass of the spring itself as negligible.

The third law tells us that a body exerting a force on another body will experience a force of equal magnitude and opposite direction. This gives us a shortcut to calculate the forces of coupled harmonic oscillators in CHON, as I explain in chapter 4.

The frequency in Hz of the harmonic motion of a simple harmonic oscillator is determined by the spring stiffness k and the mass of the object m. It is given by the formula:

$$f = \frac{1}{2\pi}\sqrt{\frac{k}{m}}$$

So, the higher the spring tension, or the lower the mass, the faster the oscillation. Conversely, lower spring tension or higher mass results in slower oscillations.

Up to this point, we have not accounted for damping forces that cause the energy of a harmonic oscillator to dissipate over time. Damping forces come from several sources in the real world including air resistance, friction, and gravity. We can approximate the effects of these damping forces with a constant b. The damping effect is also a function of the velocity of the object. This term can be introduced into our combined newton-hooke equation above like so:

$$ma = -kx - bv$$

This new equation describes a *damped* harmonic oscillator. This can be thought of as a simple harmonic oscillator that loses its energy over time. The oscillation of the particle is now constrained by an envelope that decreases its amplitude over time.

If we add another particle to this system and affix both particles to a springs connected to unmoving anchors, and also connect the particles to each other via a third spring, we now have a basic coupled harmonic oscillator system. Now, each particle can experience a force from the spring attached to the anchor as well as the spring attached to the other particle. Each of these forces is a vector with a direction and magnitude. The addition of these vectors will give us the net force on the particle.

$$\vec{F}_{net} = \vec{F}_1 + \vec{F}_2$$

Substituting for hooke's law we get:

$$\vec{F}_{net} = -kx_{anchor} + kx_{particle}$$

Previously, the displacement from equilibrium x of the particle was determined relative to a fixed anchor. Now, one of the anchors is in motion. With this, we now have to take into account the displacement from equilibrium of both particles(x_1 and x_2), the difference of which will determine our x.

$$x_{particle} = x_2 - x_1$$

And substituting into our previous equation:

$$\vec{F}_{net} = -kx_{anchor} + k(x_2 - x_1)$$

We can see that if the two particles move in sync with one another, there will be no force exerted between them, as the difference between their displacements will come out to zero.

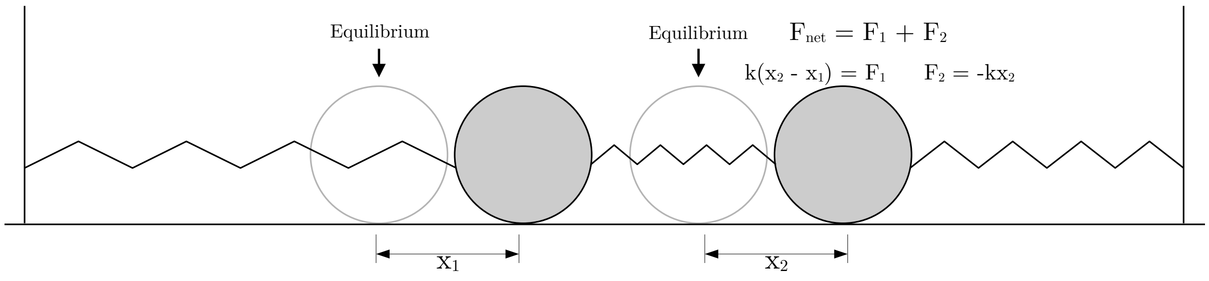


Figure 3.3: Coupled Harmonic Oscillator - The restoring force of each particle is a sum of two forces $\vec{F}_1$ and $\vec{F}_2$.

With this simple two-particle system, there are now two frequencies of oscillation that can occur. These are called the modes of the system. The first mode can be seen when the two particles are in phase, or moving in the same direction at the same time. The second mode can be seen when the two particles are 180 degrees out of phase, moving in opposition to one another. Of course many other patterns of motion can arise depending on the phase relationship of the particles in oscillation, but all of these are really a superposition of the two modes. It is in this state of superposition that we see the “exchange of energy” characteristic of the coupled pendulum system that inspired CHON. The effect is more pronounced if the outside springs are stiffer than the spring connecting the two particles. The number of modes in the system is equal to the number of particles in the system.

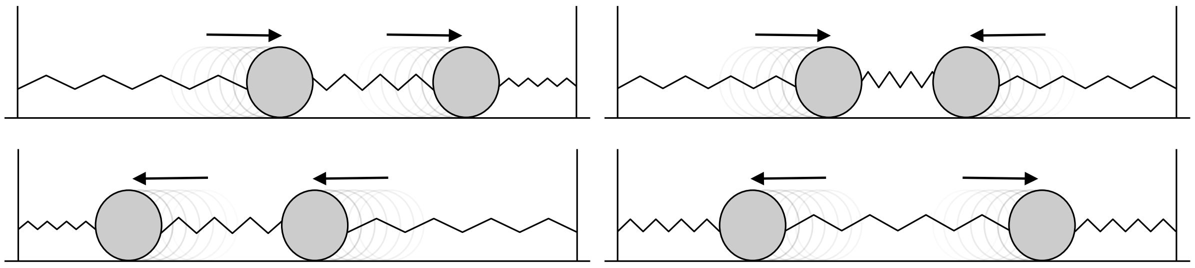


Figure 3.4: Modes - Left: Mode 1, Right: Mode 2

As particles are added to this system, it approaches a physical simulation of a string. This is the basis for the mass-spring method of string physical modeling (PM) synthesis. CHON bears a superficial resemblance to mass-spring PM synthesis, but their goals differ. PM synthesis utilizes the mass-spring method to simulate the displacement of a plucked string in order to generate a realistic string sound. In PM synthesis, each individual mass in the mass-spring system only matters inasmuch as it contributes to the overall string simulation. CHON uses a mass-spring system to simulate coupled particles and generate control data from each particle. The behavior of each individual particle is the central focus in CHON.

Another difference from a string simulation is that the particles of CHON can move in three dimensions. In a mass-spring string simulation, each mass moves only transversely (perpendicular to the length of the string). In CHON, the user can select the degrees of freedom. By default, the particles of CHON are constrained to move only longitudinally (in the x-axis). But the user can choose to allow the particles to move in the y-axis and z-axis as well, or in any combination of the three axes. The only change in the physics of this situation is that we now have to calculate $k(y_2 - y_1)$ and $k(z_2 - z_1)$ as well in our calculation of the net force on the particle.

In addition to laying out coupled harmonic oscillators in 1-dimension like a string, CHON can arrange particles in a 2-dimensional grid. This produces more complicated modal behavior and superpositions. As the number of particles increases in this grid, the simulation begins to resemble a membrane like a drumhead. In this configuration, each particle experiences forces from four neighboring particles.

Chapter 4

Software implementation

CHON is coded completely in C++. Its foundation is built on the Allolib multimedia library, developed by the Allosphere Research Group at UCSB and runs on Linux, MacOS, and Windows. CHON can simulate the physics of networks of up to 100 coupled harmonic oscillators moving in up to 3 dimensions. CHON can sonify the oscillator network with its internal audio engine, but it is designed to be part of a modular workflow, providing control data to external applications.

4.1 Libraries

Much of CHON's low-level functionality is provided by the Allolib library[29], which in turn is powered by various other C++ libraries. These libraries wrap many functions that provide access to the system CHON is running on. These libraries are cross-platform, which allows CHON to be built on Linux, MacOS, or Windows, without needing to account for idiosyncrasies in those operating systems in the code of CHON itself.

Allolib is a C++ library developed by the Allosphere Research Group at UCSB. It provides a framework for cross-platform development of interactive multimedia applications and tools. The features of Allolib itself that CHON primarily makes use of is its structure for initializing an application and running audio and video threads (allowing for communication between threads), its management of parameters, and its convenient wrapping of low-level functions and libraries.

The audio input and output streams are handled by the RtAudio library[30]. RtAudio provides an API for cross-platform, real-time audio input and output. Window creation and graphics API are provided by OpenGL and GLFW[31]. Interaction with the physical simulation (i.e. the ability to drag the particles on screen with the mouse) was provided by Tim Woods' "pickable" classes in Allolib. The GUI is provided by the ImGui (Dear ImGui) library by Omar Cornut[32]. Some synthesis classes such as basic sine oscillator and reverb were borrowed from the Gamma library.

4.2 Physics computation

The heart of CHON is a real-time and interactive simulation of a mass-spring system. There are many solutions to the mass-spring system suitable for different purposes, but for CHON to be both real-time and interactive, I needed to use a numerical method for solving the differential equation of motion for each particle at each call of a synchronous thread.

The physics simulation of CHON employs a forward Euler method. This method was chosen for its simplicity and relatively low computational cost. The forward Euler method solves numerical integration using an explicit method. This means that the state of the system at every time step of the simulation is calculated from the state at the previous time step. Essentially, the forward Euler method is an efficient and simple way to quantize the physics involved in CHON so that it can be calculated at every step in the simulation.

The equation of motion for each particle is:

$$v_{now} = \frac{v_{prev} + \frac{\vec{F}_{net}}{m} - bv_{prev}}{h}$$

Where v_{now} is the velocity or amount to move the particle this frame, v_{prev} is the velocity of the particle in the previous frame, $\vec{F}_{net}$ is the net force on the particle from neighboring particles, m is the mass of the particle, b is the damping constant, and h is the frames per second of the simulation.

The net force on a particle $\vec{F}_{net}$ is calculated every frame based on the difference in the displacement between the particle and each of its neighbors. This difference in displacement is equivalent to the amount the spring is stretched or compressed relative to its equilibrium state. The equation for $\vec{F}_{net}$ on a particle looks like:

$$\vec{F}_{net[x]} = k(x_2 - x_1)$$

$$\vec{F}_{net[y]} = k(y_2 - y_1)$$

$$\vec{F}_{net[z]} = k(z_2 - z_1)$$

For efficiency, the opposite of this force is immediately added to the neighboring particle. This effectively halves the number of times this $\vec{F}_{net}$ value needs to be calculated because each particle only needs to calculate the force from the next particle, not the previous one.

I made a *Particle* class to keep track of parameters relating to each individual particle such as 3D arrays for velocity, acceleration, equilibrium position, and current displacement from equilibrium. When a particle is created, the equilibrium position is set according to how many particles there are in the CHON system. All other physical parameters are initialized to zero.

The physics simulation of CHON takes place in the visual thread. Since the visual thread runs at 60 frames per second, this means that the step size of the simulation is 1/60th of a second (16.7ms). Moving the simulation to a higher frequency thread, such as the audio thread, would increase the accuracy of the simulation by reducing the step size, but the computation cost would increase proportionally. This would limit the feasible complexity of the CHON system so that only a few particles could be simulated consistently in real-time. Conversely, increasing the step size would allow for greater complexity in the CHON system (more particles), but reduce the simulation accuracy. It is absolutely essential that CHON be real-time, so the balancing factors are the number of particles CHON can handle vs. the accuracy of the simulation. The 16.7ms update rate of the visual thread was deemed an appropriate compromise, allowing for smoothly simulating up to 100 particles on my laptop.

The fact that the audio thread, which updates at a much higher rate, uses this data, caused some audible quantization noise in early experiments. For this reason, I implemented a basic smoothing class called *SmoothValue*, which acts like a low-pass filter. This allows the audio thread to smoothly change the synthesis control data from the visual thread between visual thread calls.

4.3 Sound engine

CHON's internal sound engine serves to quickly and easily sonify the physical simulation. Though more powerful and flexible sound design is possible by pairing CHON with an external audio application, the internal sound engine in CHON does create some fairly compelling sonic results. There are two main paradigms under which the sound engine operates: *Continuous* and *Trigger*.

The continuous paradigm is represented by the “additive synth” in the sound engine. When the particle count is 1, this synth generates a sine tone at the user selected fundamental frequency. The synth generates an additional sine tone at integer multiples of that fundamental for each particle beyond 1, so each particle has a sine tone associated with it. By default, this synth simply plays a harmonic spectrum. If the user couples the FM or AM engine of the additive synth to one of the 3 axes (x, y, or z), then the sound becomes more interesting. The AM coupling ties the amplitude of the sine tone of each particle to the displacement of that particle (in whatever axis is chosen). The FM coupling ties the width of the FM effect to the displacement of the particle.

Both of these couplings work the same in the way they couple to the particles movement. Each particle’s displacement is stored as a public variable in the Particle class. This is updated on each visual frame, and is read by the audio thread if the AM or FM coupling is active. If the particle moves in an axis that is not coupled, it will not affect the sound. For example, if AM is coupled to the y axis, but the particle is moving in the x axis, then there will be no audible effect. On the other hand, if AM is coupled to the x axis, and the particle moves in the x axis, then the amplitude will gradually increase and decrease with the particle’s displacement. This is the reason for calling it the “continuous” paradigm.

Each particle has a sine wave for their own FM synthesis, the modulator wave. The value of this wave is added to the particle’s main oscillator’s frequency (the carrier wave) at every sample. The modulator wave’s frequency is equal to the particle’s carrier frequency multiplied by a user-defined ratio. The amplitude of the modulator wave is determined by a user-defined value, and is additionally modulated by the movement of the particle. This way, the user controls the intensity of the effect of the particle’s movement on the sound. If the user-defined FM width is set to zero, then the frequency of the particle will rise and fall with the particle’s displacement, rather than at the FM frequency.

The AM synth is much simpler. The amplitude of the particle’s main oscillator is multiplied by the displacement of the particle. If the particle is at rest, then the amplitude is zero and there is no sound. When the particle is moved from equilibrium, the amplitude increases. In this way, the activity of the synth is tied to the activity of the physical simulation.

The trigger paradigm is represented by the bell synth. The bell synth is a sine tone tuned according to user-defined settings, with an amplitude envelope governed by a 1-second ADSR. The ADSR envelope is triggered whenever the particle crosses its equilibrium point in the selected axis. This allows cascading effects where each particle triggers its bell sound and pushes the next particle to trigger its bell sound and so on.

4.4 OSC

OSC (Open Sound Control) can be used to send control data between audio applications[33]. In CHON, OSC allows for the displacement of each particle to be sent out to control parameters on another audio application. In actuality, the data can be used for any purpose.

The OSC output of CHON can be turned on or off by the user. OSC messages are broadcast in the format:

/dispX/NUM x

Where NUM is the number of the particle and x is the displacement in the x axis. The displacement in all 3 axes can be sent out simultaneously. For the displacement in the y and z axes, the format is "/dispY/NUM y" and "/dispZ/NUM z", respectively.

The address and port over which CHON broadcasts the OSC data can be changed by the user at will. When these settings are changed, the OSC server is restarted.

Some examples of using CHON to control external audio applications are discussed in Chapter 6.

Chapter 5

Interface

CHON was designed to allow the user to interact with the particle system as directly and intuitively as possible. The visual interface consists of the particle system itself, a graph readout of all the particle displacements, and a GUI (Graphic User Interface) for changing various parameters of the system. The auditory interface of CHON consists of various optional synthesis algorithms. CHON accepts user input via mouse and keyboard.

5.1 Visual interface

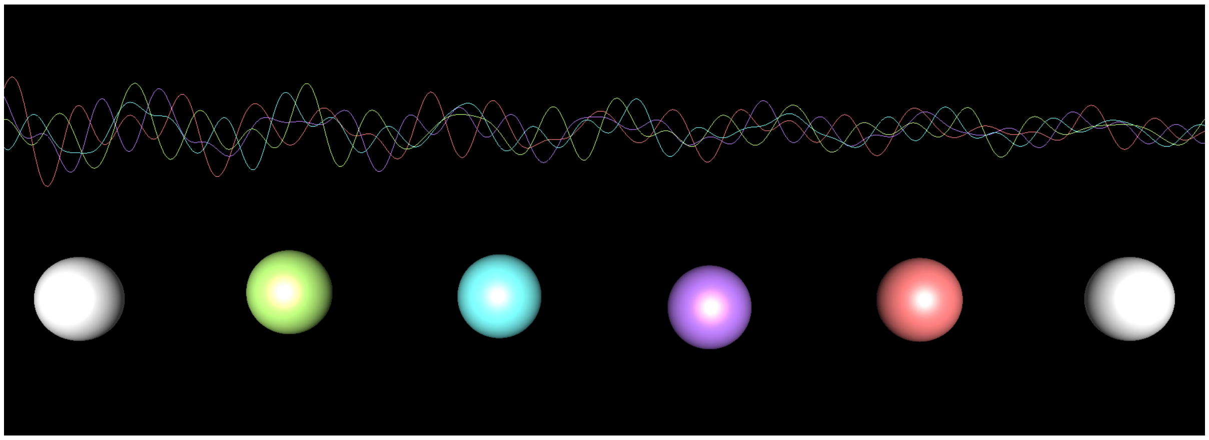


Figure 5.1: The visual particle interface (bottom) and the displacement graph (above).

CHON's visual interface is dominated by the particle simulation in the center of the screen. It is roughly skeuomorphic — the kind of particle system it simulates is mimicked in the visual representation on screen. The spheres on screen are analogous to the masses in a mass-spring system, and if you displace the system, it behaves as you would expect a real mass-spring system to behave. However, it is an idealized simulation without gravity and (optionally) without air resistance. The major difference between CHON and a real mass-spring system is that the springs are not illustrated in CHON. This was a conscious omission as it seemed that visual connections between the particles would clutter the visual display and would provide no useful information to the user. The boundaries or anchors at the edges of the system may be visualized or not at the user's discretion. They are represented by white, immobile particles. The particles where given different colors to differentiate them and to allow them to be mapped to the 2D displacement graph.

The displacement graph is located above the particle simulation on the screen. In the graph, there is a line for each particle that is mapped to the same color as the particle it represents. The graph shows the displacement of each particle from its equilibrium position in a particular axis. The user may choose which axis is graphed using the GUI described below. Each line on the graph moves up or down when its corresponding particle moves right or left (if x-axis is selected), up or down (if y-axis), or forward or back (if z-axis). Time is represented on the graph horizontally as a continuously scrolling timeline with the present values being added on the right, pushing older values to the left. The graph serves as an additional readout for the user that allows them to understand the dynamics of the system better. It also provides a way to visualize the displacement data that can be broadcast over OSC.

The view of the particle system is set to a sensible default, directly facing the string of particles so they are spread horizontally on the screen, with the y axis up and down relative to the camera, and the z axis forward and back. The user can change the view manually if desired using the arrow keys on their keyboard. This is necessary when the user wants to move the particles in the z axis or when they simply wish to view the system from another angle for example. The up arrow moves the camera up, the down arrow moves the camera down, and the left and right arrows rotate the camera around the particle system. The user can press the "v" key on their keyboard at any time to reset the view.

When particles are added in the y-axis, CHON enters grid mode. In grid mode, the camera defaults to a position and angle where the user can see the entire two-dimensional grid of particles and manipulate them easily.

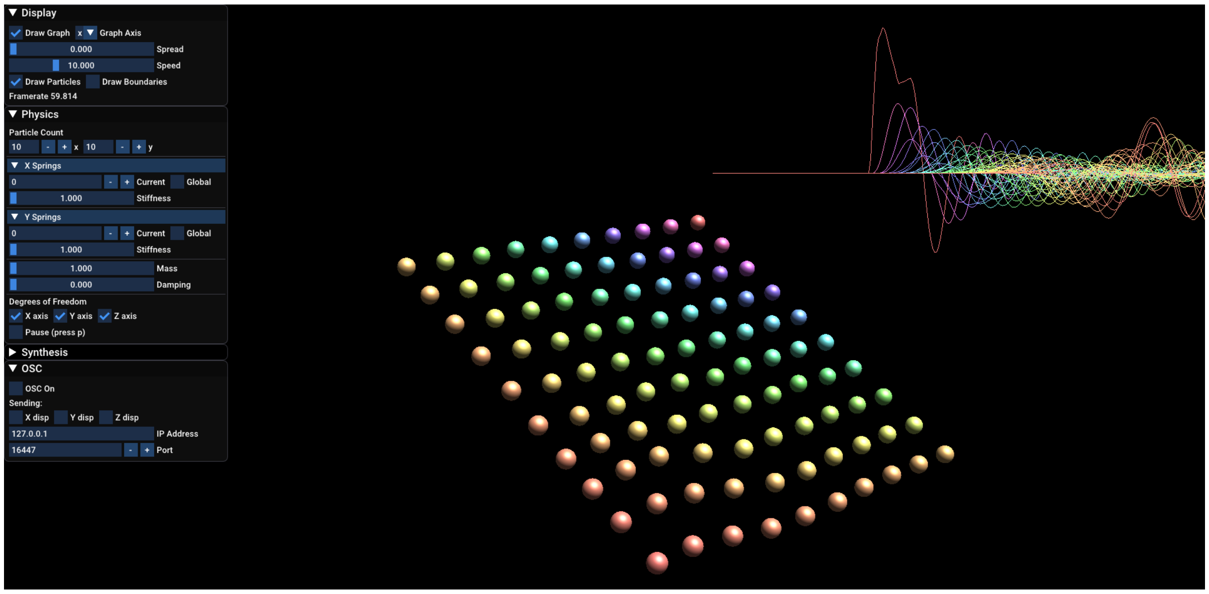

▼ Display
✓ Draw Graph x ▼ Graph Axis
0.000 Spread
10.000 Speed
✓ Draw Particles ☐ Draw Boundaries
Framerate 59.814
▼ Physics
Particle Count
10 - + x 10 - + y
▼ X Springs
0 - + Current ☐ Global
1.000 Stiffness
▼ Y Springs
0 - + Current ☐ Global
1.000 Stiffness
1.000 Mass
0.000 Damping
Degrees of Freedom
✓ X axis ✓ Y axis ✓ Z axis
☐ Pause (press p)
▶ Synthesis
▼ OSC
☐ OSC On
Sending:
☐ X disp ☐ Y disp ☐ Z disp
127.0.0.1 IP Address
16447 - + Port

Figure 5.2: Grid mode

5.2 GUI

The GUI of CHON presents further options to control the program. It is divided in to four windows: *Display*, *Physics*, *Synthesis*, and *OSC*. The windows are arranged on the left side of the screen, and each one can be collapsed or expanded using the triangle in the top-left of the window. Pressing "g" on the keyboard shows/hides the GUI.

The GUI controls consist of sliders, checkboxes, number-boxes, and drop-down menus. Sliders are operated by either left-clicking and dragging, or holding control and clicking on the slider and typing the desired value. Checkboxes are boolean "on or off" boxes that are toggled by left-clicking on them. Number-boxes can accept a typed number or they can be incremented/decremented via the "+" and "-" buttons beside them. Drop-down menus reveal their options when they are left-clicked, and a choice can then be selected by left-clicking.

The Display window controls what is visible and some other aspects of visual display. The user can show or hide the graph, particles, and boundary particles, using the checkboxes in this window. The *Graph Axis* drop-down menu can be used to change which axis of movement of the particles is mapped to the graph. The *Spread* slider allows the user to separate the graph lines by spreading them out vertically. The *Speed* slider controls how fast the graph moves. At the bottom of the Display window there is also a readout showing the frame-rate of the program.

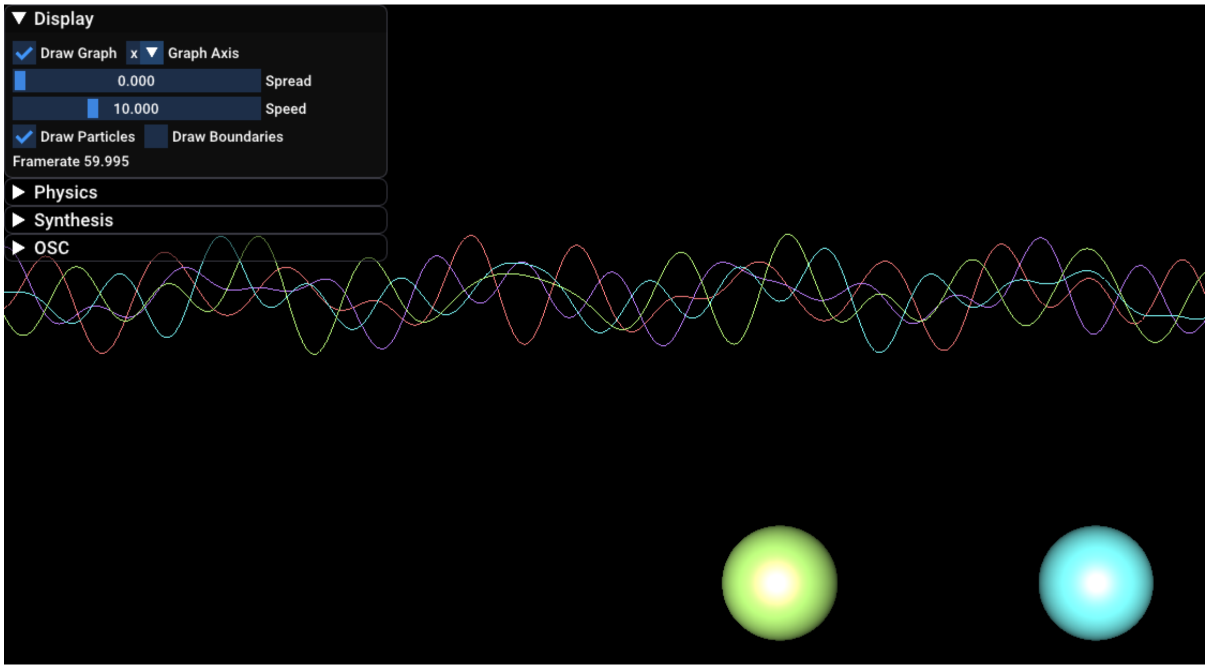
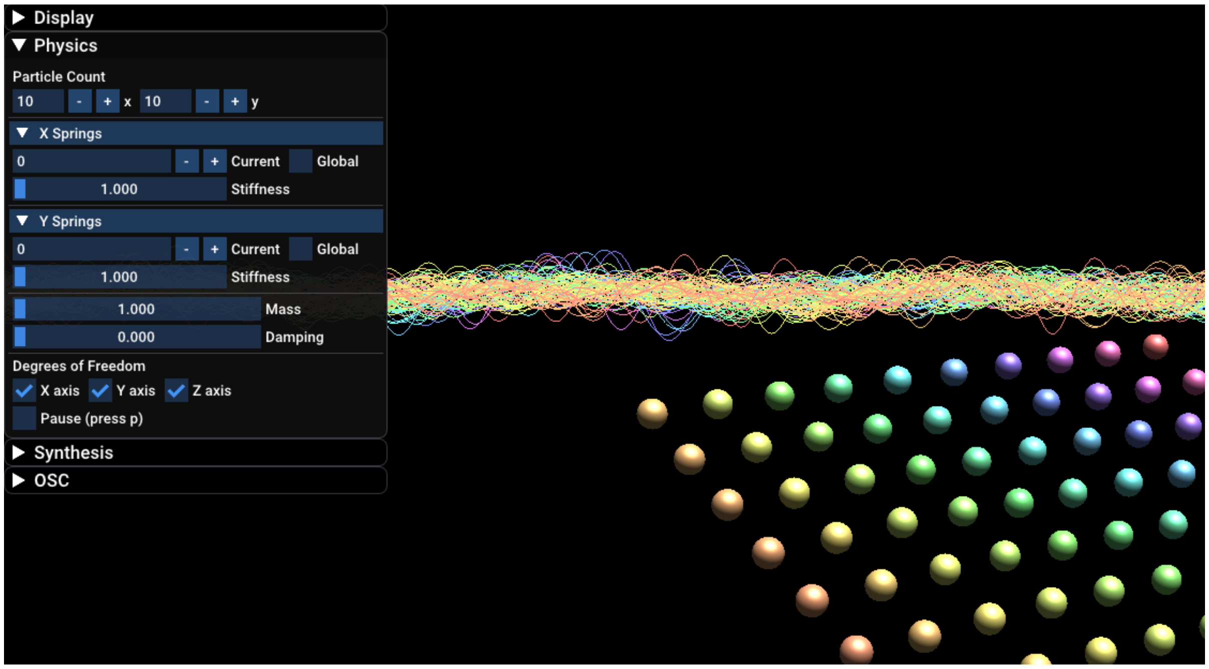


Figure 5.3: The Display window (left)

The Physics window allows the user to alter the physical properties of CHON. The Particle Count section controls how many particles are present in the system. There are two number-boxes, controlling the number of particles in X (left-right) and Y (up-down) directions of a 2D plane of particles, respectively. For example, if the the x number-box is set to 5, and the y number-box is set to 4, then the particles will be arranged in a 5 by 4 grid. By default, Y is set to 1, and the system is in "1D" mode. If the user sets Y to a value greater than 1, CHON automatically enters "2D" mode. CHON will re-enter 1D mode if the user sets Y back to 1. The maximum number of total particles in the system is limited to 100.

The *X Springs* section of the window provides control over spring stiffness in the X direction. If *Global* is selected, the *stiffness* slider controls the stiffness of all springs oriented in the X direction. If *Global* is unselected, then a number-box appears with increment/decrement buttons. The number in the number-box indicates a particular spring in the system, with "0" representing the spring between the left-most particle and the left boundary, and 1 is the spring between that particle and it's neighbor on the right, and so on. Moving the *Stiffness* slider with *Global* unselected changes the stiffness of only the selected spring.

If CHON is in 2D mode, then the *Y Springs* section appears, allowing similar control over the springs in the Y direction. In the Y direction, springs are numbered from bottom to



Particle Count
10 - + x 10 - + y

X Springs
0 - + Current Global
1.000 Stiffness

Y Springs
0 - + Current Global
1.000 Stiffness

1.000 Mass
0.000 Damping

Degrees of Freedom
✓ X axis ✓ Y axis ✓ Z axis
Pause (press p)

Figure 5.4: The Physics window

top. So, for example, if the stiffness of *X Spring* 0 and *Y Spring* 0 are increased to 50, then the bottom row of springs and the left-most column of springs will have a stiffness constant of 50.

The *Mass* slider controls the mass of all the particles. There is no way to change the mass of individual particles because that would be redundant (raising spring stiffness is equivalent to lowering mass). The *Damping* slider controls the global damping of the system. A damping value of 0 will allow CHON to continue oscillating indefinitely. There are three *Degrees of Freedom* checkboxes that allow the user to constrain the dimensions that the particles may move in. The *Pause* button allows the user to freeze CHON in place. When CHON is frozen, the positions and momenta of the particles are preserved, but the positions may be altered by the user by clicking and dragging particles. Deselecting *Pause* causes CHON to resume simulation.

The Synthesis window provides control over the internal sound engine of CHON. The sound engine is described in Chapter 4. The Bell Synth, Additive Synth, and Reverb, are hidden by default and can be turned on and off with their respective checkboxes. By default, all of these checkboxes are unselected, and the synthesis controls are hidden. Pressing the triangle on one of the headers will reveal the controls for the respective section.

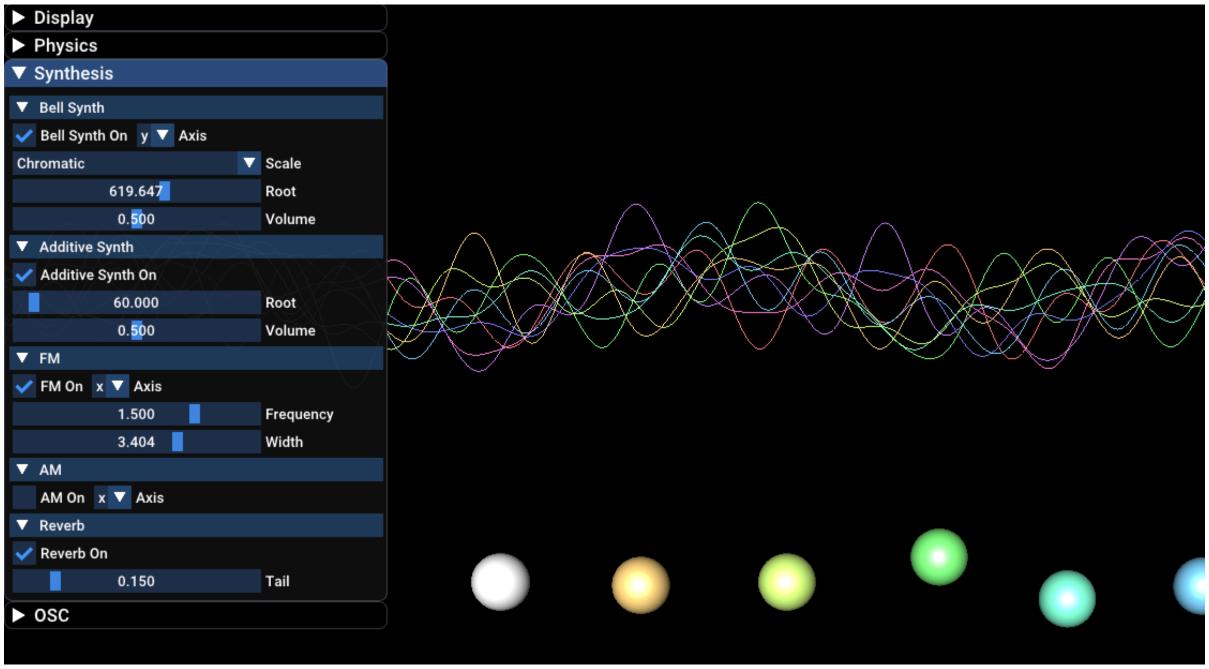

▶ Display
▶ Physics
▼ Synthesis
▼ Bell Synth
✓ Bell Synth On y ▼ Axis
Chromatic ▼ Scale
619.647 Root
0.500 Volume
▼ Additive Synth
✓ Additive Synth On
60.000 Root
0.500 Volume
▼ FM
✓ FM On x ▼ Axis
1.500 Frequency
3.404 Width
▼ AM
AM On x ▼ Axis
▼ Reverb
✓ Reverb On
0.150 Tail
▶ OSC

Figure 5.5: The Synthesis window

The *Bell Synth* section contains the *Bell Synth On* checkbox, the **Axis* drop-down, the *Scale* drop-down, as well as the *Root* and *Volume* sliders. *Bell Synth On* toggles the bell synth on and off. The *Axis* drop-down menu controls whether the bell sound of each particle is triggered when it crosses it's x-axis, y-axis, or z-axis equilibrium point. The *Scale* drop-down provides a few options for tuning the particles to scales. The *Root* slider determines the frequency of the bell of the left-most particle (bottom-left if in 2D mode), and the frequency of the bells of the rest of the particles are determined relative to this root frequency according to the selected scale. The *Volume* slider determines the volume of the bell synth.

The *Additive Synth* section contains the *Additive Synth On* checkbox, the *Root* and *Volume* sliders, as well as the FM and AM subsections. The *Additive Synth On* checkbox toggles the Additive synth on and off. The *Root* slider, like the one in the bell synth section, determines the frequency of the sine oscillator of the left-most particle (bottom-left in 2D mode). The *Volume* slider controls the volume of the additive synth. There is no scale drop-down here like there is in the bell synth. The frequencies of the particles are tuned to harmonic overtones of the root frequency. By default, the particle system has no effect on the additive synth. The user must activate the FM and/or AM functionalities to tie the additive synth to the state of the particle system.

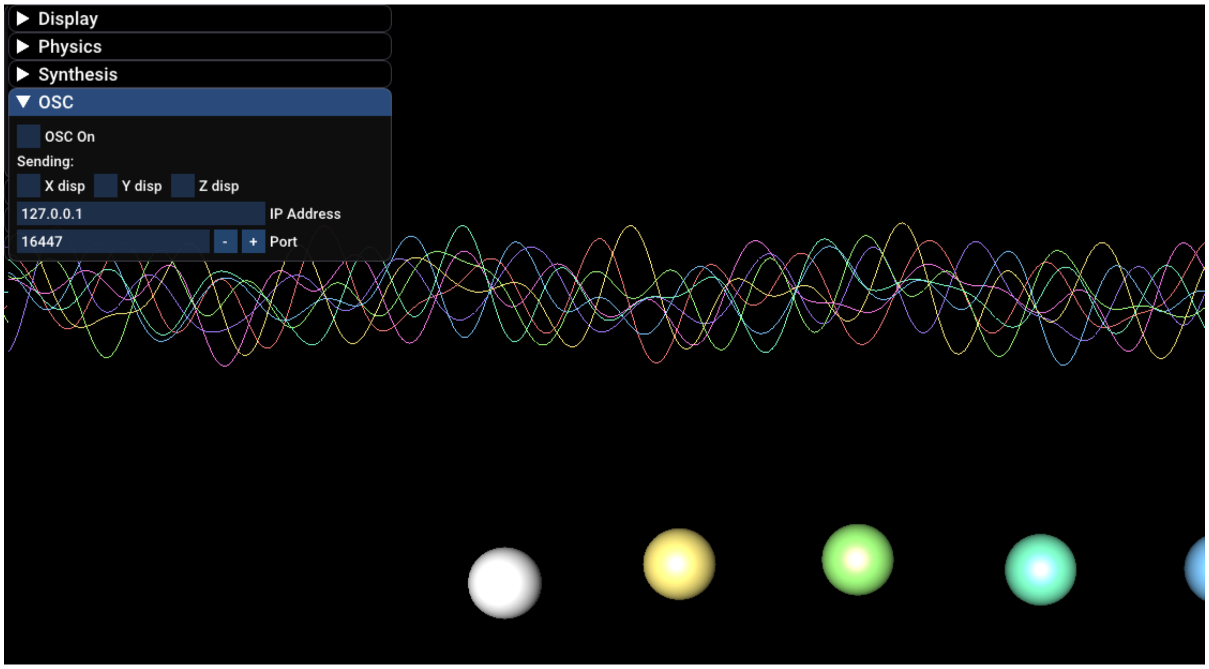
The *FM* subsection contains an on/off checkbox, an *Axis* drop-down, and *Frequency* and

Width sliders. Selecting the *FM On* checkbox allows the oscillators of the additive synth to be frequency modulated by the displacement of particles. The *Axis* drop-down determines which displacement axis is mapped to the FM synth. As described in Chapter 4, the displacement of the particles controls the width of the modulating signal that is acting on that particle's carrier signal. The modulating signal's frequency is determined by the *Frequency* slider. The number in the frequency slider corresponds to a ratio relative to the carrier signals. If the *Frequency* is set to 1, then the modulating signal is the same frequency as the carrier signal. If it is 2, then the modulating frequency is double that of the carrier frequency. A setting of zero on this slider is a special case in which the carrier signal's frequency is modulated exactly by the displacement of the particle. The *Width* slider allows the user to amplify and attenuate the FM effect on the additive synth.

The *AM* subsection contains only an on/off checkbox and an *Axis* drop-down. If the *AM On* checkbox is selected, then the amplitude of each oscillator of the additive synth is modulated by the displacement in the *Axis* direction of its corresponding particle. With AM turned on, the additive synth will make no sound when all particles are resting at equilibrium.

The *Reverb* section allows the user to apply a global reverb to the sound engine. This section contains an on/off checkbox and the *Tail* slider. If the *Reverb On* checkbox is selected, then a reverb is applied to both the bell synth and the additive synth. The length of the decay of this reverb can be increased and decreased via the *Tail* slider. The user should be cautious. If the *Tail* slider is set to 1, it will produce an infinite reverb that can become quite loud.

The OSC window provides control over the OSC broadcasting functionality of CHON. It contains the *OSC On* checkbox, three checkboxes for controlling what data is broadcast, a text box for inputting the IP address of the OSC broadcast, and a number box to change the port to send the OSC data through. The OSC server is off by default and can be turned on with the *OSC On* checkbox. With OSC turned on, it still will not broadcast any data unless at least one of *X disp*, *Y disp*, or *Z disp* are selected. For example, if only *Y disp* is selected, then only the displacement of the particles in the Y direction will be broadcast, and if all three are selected, then the displacements in all three directions will be broadcast. The user can change the IP address by typing in the text box and pressing enter to confirm. The Port can be changed by typing in the *Port* number box and pressing enter, or by incrementing/decrementing the value with the "+" and "-" buttons. The OSC client will automatically reset when these values are updated.



127.0.0.1 IP Address
16447 Port

Figure 5.6: The OSC window

5.3 Interaction

The interface for CHON is centered around directly manipulating the particles with a mouse. The user can left-click and hold on any particle and drag the particle from it's equilibrium position. The whole system will react in real time to this action, as other particles that are coupled to the selected particle are pulled along as well. If the user releases the mouse button, then the particles will be freed and set into oscillatory motion. The user can click and hold on any particle while it is in motion to grab it. It is also possible to drive the system manually by moving the mouse back and forth while holding a particle.

This direct interaction with the particle system in CHON is vital to the mission of CHON to allow the user to directly experiment and play with the mass-spring simulation. Because of this design, the user can easily influence the system and get immediate feedback, encouraging an iterative feedback loop conducive to creative exploration. The user can set CHON into motion and it will produce a chain reaction on its own from there. However, the user can also intervene to interrupt a moving particle, break symmetry, and cause new ripples, waves, and explosions. CHON then provides visual and auditory feedback so that the user can learn and understand the effects of their actions.

Aside from using the mouse to interact with the particle system and the GUI, there are also a limited number of keyboard shortcuts. The “p” key toggles the *Pause* checkbox in the Physics window, allowing the user to pause and unpause the simulation without accessing the GUI. The “g” key toggles the GUI’s visibility on and off. The “r” key adds a random value to the velocity all of the particles.

Chapter 6

Applications and Limitations

6.1 Applications

6.1.1 Continuous control data

CHON simulates the continuous movement of particles, so the most obvious mapping of this system is to continuous control data. A continuous control signal is a constant stream of values that are relatively fine-grained. This is the default situation in CHON. The OSC data is sent out in a continuous stream at 60 frames per second, reporting the displacements of each particle in the x, y, and z directions. The additive synthesis engine in CHON also uses this data to modulate its FM and AM parameters.

When CHON is set to only have one particle, the movement of that particle is strictly sinusoidal. In this configuration, the signal that CHON generates resembles a very common control signal in electronic music: the LFO. An LFO (low frequency oscillator) is a continuous signal that modulates a parameter of sound synthesis or sound processing. LFOs come in various wave shapes, but the basic case is a sinusoidal LFO. Composers of electronic music often have dozens of independent LFOs controlling various parameters in their digital audio workstation. One way to think of CHON is to imagine a group of LFOs that are not independent. Instead, they push and pull on one another, creating a physical connection between modulating parameters.

Mapping the continuous displacement of particles in CHON presents many possibilities. Some basic synthesis options are included in the internal audio engine of CHON, as described in chapter 4. Pairing the OSC output of CHON with a digital audio workstation, one could map the displacement of each particle to the panning or volume of various tracks. CHON could also control audio processing parameters on each track such as filter frequency, reverb wet/dry signal, pitch shift, distortion/saturation level, and others.

CHON can also be treated as a module on a modular synthesis rack. A physical module that translates OSC data into control voltage such as 2.4SINK by Instrument of Things, or the WiFiMIDI from SDS Digital can enable CHON to gain control over an entire modular synthesis patch. Of course a simpler, and much less expensive, version of this is to use the software modular synthesis application VCVRack. One capable virtual module for this purpose in VCVRack is TrowaSoft's cvOSCcv. Using this module, and its extensions the user can send dozens of control signals from CHON all over a virtual patch. The possibilities are endless.

6.1.2 Trigger control data

In some cases, it does not make sense to use continuous control data. There are various ways to take the continuous data from CHON and extract discrete trigger data. Unlike continuous data, trigger control data only causes a change in the target parameter at a moment in time, when a condition "triggers". This is a boolean logic. The trigger signal sends a constant value of 0 until the condition is met, at which time it sends a single 1, which can trigger an event.

This is how the Bell synth in the internal synthesis engine of CHON works. When a particle crosses its equilibrium point, it triggers the bell sound, which decays on its own according to its ADSR envelope. Each particle has 3 equilibrium points, since it can move in up to 3-dimensions. CHON can cause triggers when a particle crosses its x, y, and z equilibrium points, respectively.

6.1.3 Generating musical scores

CHON can also be used to write instrumental music. In Summer of 2020, Hocket, an LA-based piano duo, put out a call for very short compositions of 1 minute or less. I took the opportunity to make a first attempt at composing a piece of music using CHON. The result is the 45-second miniature piece called *Pandæmonium*.

I set up a 4-particle system in CHON and sent the OSC output to an algorithmic music composition tool called SCAMP created by Marc Evanstein[34]. I configured SCAMP to interpret the OSC data as the contour of a unique arpeggio for each hand. Each of the 4 particles dictated the movement of one hand of the performers, so that the movement of the performers hands to the right and left of the keyboard would reflect the movement of the particles from right to left in CHON. The displacement data of each particle was quantized to a different scale.

I decided that the entire piece would be one single gesture. I set CHON into motion and configured the damping effect so that the particles would come to rest after about one minute. The result is like a sonic explosion that gradually dissipates while each arpeggio pushes and pulls on one another.

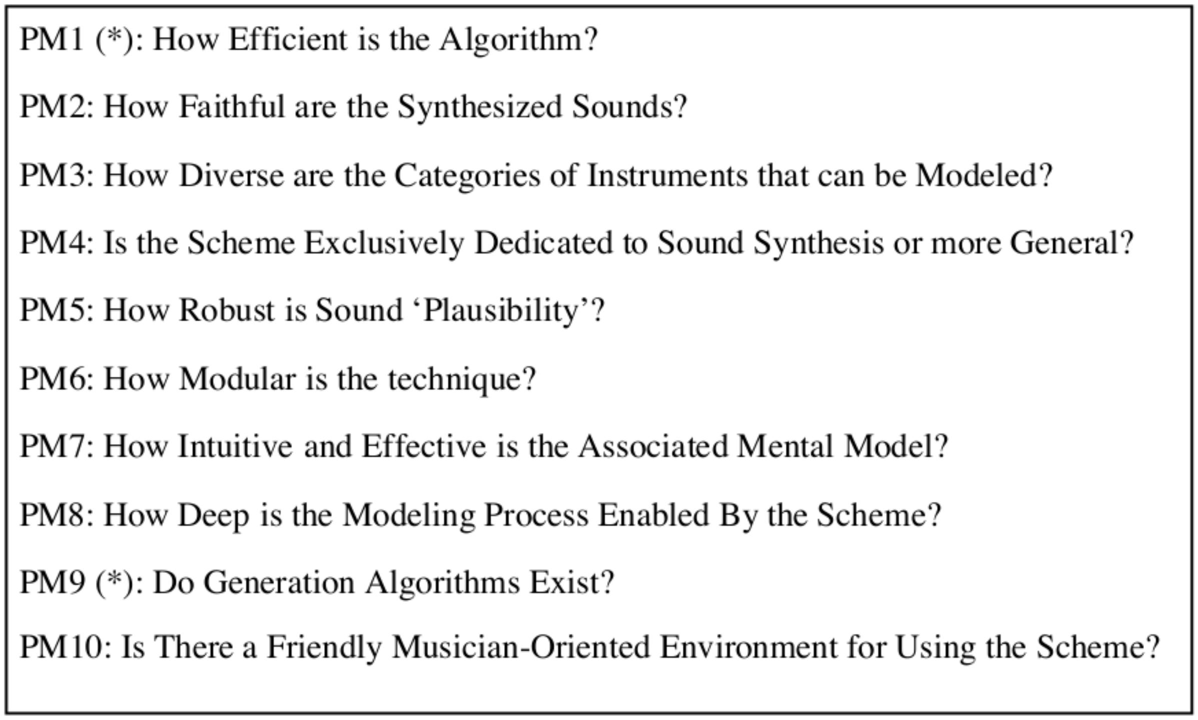
6.2 Limitations

CHON is designed to simulate many configurations of one particular kind of physical system: a network of coupled harmonic oscillators. While this is a fundamental model in theoretical physics, it is not well-suited to every application.

CHON is not meant to be a scheme for physical modeling synthesis, but rather a control signal scheme for shaping musical parameters using physically plausible interactions. Rather than an innovation in physical modeling synthesis, I consider CHON to be an innovation on the traditional LFO (Low Frequency Oscillator) model. Nonetheless, CHON does employ physical modeling for musical purposes, and so we can look to prior research for evaluating these types of systems.

Castagné and Cadoz have conceptualized a convenient framework to evaluate Physical Modeling schemes for musical applications[13]. Under this framework, CHON scores well in criteria *PM1* (efficiency – CHON is efficient enough to achieve its goal of real-time interactive simulation of up to 100 particles), *PM4* (extra-sonic applications – CHON can provide control signals for any medium), *PM5* (robustness of "plausibility" – a complete novice can make plausible musical gestures with CHON), *PM6* (modularity – CHON is highly modular in its applications thanks to its OSC functionality), *PM7* (intuitiveness of mental model – CHON's behavior is predictable even to a non-physicist, with room for surprise and discovery), *PM8* (deepness of model – CHON simulates a fundamental physical system), *PM10* (user-friendly interface - CHON is highly accessible to non-specialists).

PM2 (faithfulness of reproduced sound), *PM3* (diversity of simulated instruments), and *PM9* (existence of generation algorithms) are mostly irrelevant to the goals of CHON. However, the Euler Method, which is used in CHON to calculate the movement of the particles, is known to have a margin of error proportional to the framerate of the



PM1 (*): How Efficient is the Algorithm?

PM2: How Faithful are the Synthesized Sounds?

PM3: How Diverse are the Categories of Instruments that can be Modeled?

PM4: Is the Scheme Exclusively Dedicated to Sound Synthesis or more General?

PM5: How Robust is Sound 'Plausibility'?

PM6: How Modular is the technique?

PM7: How Intuitive and Effective is the Associated Mental Model?

PM8: How Deep is the Modeling Process Enabled By the Scheme?

PM9 (*): Do Generation Algorithms Exist?

PM10: Is There a Friendly Musician-Oriented Environment for Using the Scheme?

Figure 6.1: Criteria for evaluating physical modeling schemes[13].

simulation. Therefore if *PM2* is taken to mean more generally "how well does the scheme fit to physical reality," then the scheme that CHON employs is not the most precise when compared to others.

CHON can simulate a complex system with many moving parts. The motion of an individual particle in CHON may be difficult to predict, but it is not random. When set in motion and left alone, CHON is deterministic (non-linearities can be introduced by user-interaction). CHON is not suited for random signal generation or chance music. The movement of a particle in CHON may appear complicated, but it also has an intuitive tangibility thanks to its physical basis.

Some systems, such as the CORDIS-ANIMA software[9], go far beyond CHON in terms of flexibility, allowing the user to construct systems of thousands of particles and define parameters for each one. CHON is limited to 100 particles in a 1D or 2D configuration. This limitation in physical configuration is a deliberate choice to make CHON easier to use. I determined that 100 parameter control signals is sufficient for the vast majority of use-cases.

The problem of parameter mapping is an important and sometimes difficult one in algorithmic composition. CHON's internal sound engine provides three examples of mapping its control signals to synthesis parameters. However, when broadcasting OSC signals from CHON to an external application, two problems become apparent. The first thing one will likely notice is the challenge of deciding to which parameter the control signals should be attached. The possibilities are practically limitless and it may take some serious experimentation and imagination to find the most compelling configurations. The second challenge is that setting up a high number of OSC connections can become quite tedious. This isn't exactly a problem with CHON in and of itself; CHON automatically sends out OSC data with the click of a button. Nonetheless, the daunting task of manually configuring 50 or 100 parameters in your DAW to accept the OSC data from CHON may present a barrier to some of the more elaborate setups.

Chapter 7

Future directions

CHON presents new possibilities for musical expression, and I have only scratched the surface. I would like to extend the software to make it easier to use and more accessible so that other composers might make use of it. But I am still experimenting with the ways CHON can already generate sonic gestures in its current form.

In *Pandæmonium*, I decided to align the notes to an eighth-note grid to make the music more feasible to play by the two performers. I think this diminishes the physical metaphor somewhat because it obscures the varying speed of the particles. However, removing the strictly quantized grid in a future piece will require some other solution for keeping performers in sync, such as a click-track. I also think it might be fruitful to explore mapping other parameters of the particles to a score. For example, the velocity of each particle averaged over a short time or the total potential energy of the system might be used to create interesting musical gestures.

In addition to instrumental music, I plan to write an electronic piece using CHON. It will likely use some synthesized sounds like those CHON can produce from its internal sound engine, but I am also experimenting with having CHON manipulate sound clips. CHON could trigger sounds and control playback rate, transposition, filtering, panning, and other effects. While this composition will not present the problem of generating a playable score for performers, grappling with the sheer amount of possibilities is an obstacle in its own right.

I have also imagined musical works for live performer and CHON where the sound of the performer drives CHON in sympathetic resonance. This could lead to many performance network topologies such as a feedback loop where CHON generates control data which generates sound that drives CHON, or a duet where two performers drive CHON with the sound they generate. This will require further development and UI decisions to make sure it is simple and reliable to use.

CHON has been under continual development for more than a year and will likely continue to grow and improve. I plan to add several features in the near future. I want to add a preset function that allows the configuration of CHON to be saved and recalled at a later time. I also want to add more physical parameters to the OSC stream. As stated before, this could include velocity, potential energy, collisions, and other physical attributes of the particles. Other aspects of CHON will continue to be developed and refined to make CHON more user-friendly and flexible.

Another area that CHON could be extended is in the way the user can interact with it. The gestural basis of CHON's philosophy suggests that it would be well-suited to gestural control. CHON could benefit from gestural interfaces like the LeapMotion[35] or a simple webcam paired with a gesture recognition software such as Wekinator[36]. Admittedly, a mouse is not the most intuitive controller for CHON, but it is the most ubiquitous. Therefore, I chose to focus on mouse control for the first version of CHON, leaving the option open for other control methods in the future.

Finally, one of my active areas of research is to extend the classical mechanics already represented in my music into the realm of quantum mechanics. While classical mechanics fulfills the role of a universally intrinsic logic for my music to follow, quantum mechanics is far less intuitive to most people. However, my goal in using quantum mechanics in music not to rely on an established intuitive logic for my music. Rather, I aim to render intangible quantum interactions more tangible through sonification. Whereas through classical mechanics I aim to make my music tangible, I hope to make quantum mechanics tangible through my music. Thus, I plan to extend CHON or a version of CHON to represent "quantum harmonic oscillators". This will add a probabilistic element to the particles and their interactions.

In developing CHON and experimenting with it, I have already begun to realize my vision of energetic sonic objects pushing and pulling on one another, but I have only begun to explore the musical possibilities that CHON affords. The physics of coupled harmonic oscillators seems to have a wealth of potential in creating physically coupled sound events and transformations. I will continue to explore this new musical territory and to develop CHON, and I hope that making CHON open source and freely available will encourage others to do the same.

Bibliography

[1] Modartt, “Pianoteq,” *Madartt: Virtual instruments, physically modelled.* Accessed: Mar. 26, 2021. [Online]. Available: https://www.modartt.com/.

[2] Arturia, “V-Collection.” Accessed: Mar. 28, 2021. [Online]. Available: https://www.arturia.com/products/analog-classics/v-collection/overview.

[3] Eckel, Iovino, and Caussé, “Sound synthesis by physical modelling with modalys,” *Proceedings of the International Symposium on Musical Acoustics*, pp. 479–482, 1995.

[4] AAS, “AAS—Instruments, synthesizer, and effect plug-ins based on physical modeling.” Accessed: Mar. 28, 2021. [Online]. Available: https://www.applied-acoustics.com/.

[5] L. Hiller and P. Ruiz, “Synthesizing Musical Sounds by Solving the Wave Equation for Vibrating Objects: Part 2,” *Journal of the Audio Engineering Society*, vol. 19, no. 7, pp. 542–551, Jul. 1971, Accessed: Mar. 26, 2021. [Online]. Available: https://www.aes.org/e-lib/browse.cfm?elib=2156.

[6] C. Roads, *The Computer Music Tutorial.* MIT Press, 1996.

[7] J. O. Smith III, “Physical Modeling Synthesis Update,” *Computer Music Journal*, vol. 20, no. 2, pp. 44–56, 1996.

[8] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, “Discrete-time modelling of musical instruments,” *Rep. Prog. Phys*, vol. 69, pp. 1–78, Jan. 2006, doi: 10.1088/0034-4885/69/1/R01.

[9] C. Cadoz, A. Luciani, and J.-L. Florens, “CORDIS-ANIMA: A Modeling and Simulation System for Sound and Image Synthesis - The General Formalism,” *Computer Music Journal*, vol. 17, nos. 1, Spring 1993, pp. 19–29, 1993, Accessed: Mar. 28, 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01234319.

[10] A. Vinjar, “Bending Common Music with Physical models,” Mar. 2021.

[11] C. Cadoz, “The Physical Model as Metaphor for Musical Creation: ”Pico..TERA”, a Piece Entirely Generated by Physical Model,” *nternational Computer Music Conference,* pp. 305–312, 2002.

[12] C. Chafe, “Case Studies of Physical Models in Music Composition,” 2011.

[13] N. Castagné and C. Cadoz, “10 criteria for evaluating physical modelling schemes for music creation,” *Proc. of the 6th Int. Conference on Digital Audio Effects*, 2003.

[14] M. Pearson, “TAO: A physical modelling system and related issues,” *Organised Sound*, vol. 1, no. 1, pp. 43–50, Apr. 1996, doi: 10.1017/S1355771896000167.

[15] C. Henry, “Pmpd : Physical modelling for Pure Data,” Jan. 2004.

[16] J. Villeneuve and J. Leonard, “Mi-Creative.” Accessed: Mar. 28, 2021. [Online]. Available: http://www.mi-creative.eu/about.html.

[17] F. Lerdahl, “Cognitive constraints on compositional systems,” *Contemporary Music Review*, vol. 6, no. 2, pp. 97–121, Jan. 1992, doi: 10.1080/07494469200640161.

[18] L.-G. Bodin, “Music — an artform without borders?” Unpublished, 2004.

[19] C. Roads, *Composing Electronic Music: A New Aesthetic*, 1st edition. Oxford ; New York: Oxford University Press, 2015.

[20] B. ALMÉN, *A Theory of Musical Narrative.* Indiana University Press, 2008.

[21] C. Keysers, “Mirror neurons,” *Current Biology*, vol. 19, no. 21, pp. R971–R973, Nov. 2009, doi: 10.1016/j.cub.2009.08.026.

[22] P. F. Ferrari and G. Rizzolatti, “Mirror neuron research: The past and the future,” *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 369, no. 1644, Jun. 2014, doi: 10.1098/rstb.2013.0169.

[23] V. Gazzola, L. Aziz-Zadeh, and C. Keysers, “Empathy and the Somatotopic Auditory Mirror System in Humans,” *Current Biology*, vol. 16, no. 18, pp. 1824–1829, Sep. 2006, doi: 10.1016/j.cub.2006.07.072.

[24] H. Chapin, K. Jantzen, J. A. Scott Kelso, F. Steinberg, and E. Large, “Dynamic Emotional and Neural Responses to Music Depend on Performance Expression and Listener Experience,” *PLoS ONE*, vol. 5, no. 12, p. e13812, Dec. 2010, doi: 10.1371/journal.pone.0013812.

[25] E. Varese and C. Wen-chung, “The Liberation of Sound,” *Perspectives of New Music*, vol. 5, no. 1, p. 11, 1966, doi: 10.2307/832385.

[26] I. Xenakis, *Formalized Music: Thought and Mathematics in Composition.* Pendragon Press, 1992.

[27] I. Newton 1642-1727, *Newton’s Principia : The mathematical principles of natural philosophy.* First American edition, carefully revised; corrected / with a life of the author, by N. W. Chittenden. New-York : Daniel Adee, 1846., 1846.

[28] R. Hooke, *De potentia restitutiva.* London: Printed for J. Martyn, 1678.

[29] A. R. Group, “Allolib.” Allosphere Research Group, Mar. 2021, Accessed: Mar. 26, 2021. [Online]. Available: https://github.com/AlloSphere-Research-Group/allolib.

[30] G. Scavone, “RtAudio.” Accessed: Mar. 26, 2021. [Online]. Available: https://www.music.mcgill.ca/~gary/rtaudio/.

[31] “GLFW,” *GLFW.* Accessed: Mar. 26, 2021. [Online]. Available: https://www.glfw.org/.

[32] O. Cornut, “ImGui.” Mar. 2021, Accessed: Mar. 30, 2021. [Online]. Available: https://github.com/ocornut/imgui.

[33] M. Wright Matt, “OpenSoundControl CNMAT,” *opensoundcontrol.org.* Accessed: Mar. 26, 2021. [Online]. Available: https://cnmat.berkeley.edu/opensoundcontrol.

[34] M. Evanstein, “SCAMP:Suite for Computer-Assisted Music in Python.” 2019, Accessed: Mar. 26, 2021. [Online]. Available: http://marcevanstein.com/Writings/Evanstein_MAT_Thesis_SCAMP.pdf.

[35] Ultraleap, “Leap Motion Controller.” Accessed: Mar. 26, 2021. [Online]. Available: https://www.ultraleap.com/product/leap-motion-controller/.

[36] R. Fiebrink, “Wekinator Software for real-time, interactive machine learning.” Accessed: Mar. 26, 2021. [Online]. Available: http://www.wekinator.org/.

[37] C. Roads, “Sound Composition with Pulsars,” *Journal of the Audio Engineering Society*, vol. 49, no. 3, pp. 134–147, Mar. 2001, Accessed: Mar. 26, 2021. [Online]. Available: https://www.aes.org/e-lib/browse.cfm?elib=10198.

Appendix A: Timeline

What follows is a timeline of the most significant milestones I achieved while pursuing my Masters of Science in Media arts and Technology.

Fall 2017

- Took first course in MAT (240C – Karl Yerkes)
- Appointed CREATE Technical Coordinator

Winter 2018

- Finished *PISCES* (Pisces Interactive Spectral Compression Engine & Synthesizer) as final project for 240B. This was my first software implementation of what I would come to call "spectral dilation" effects. (https://github.com/rodneydup/Pisces)
- Premiered *Quinto Suono* - a string quartet composed using an algorithm I created to generate difference tones. (http://rodneyduplessis.com/musicPages/quinto%20suono.html)

Spring 2018

- Premiered *Disconnect* - a piece for saxophone and live electronics (performed by Henrique Portovedo)(http://rodneyduplessis.com/musicPages/Disconnect.html)

Summer 2018

- Presented *BachFlip* at ICMC - BachFlip is my first piece composed using spectral dilation and I used PISCES to realize it.
- Directed the UCSB Summer Music Festival, invited MAT students to present installations.
- Premiered *Sisyphe Heureux* - a percussion quartet composed by algorithm to create waves of sonic intensity, a conceptual precursor to CHON (http://rodneyduplessis.com/musicPages/Sisyphe%20Heureux.html)

Fall 2018

- Appointed Teaching Assistant in MUS 109 (Instructor of record: Curtis Roads)
- Created *Monopus* soft musical interface with Aaron Anderson as final project for MAT 594 (instructor: Yon Visell) (https://nextcloud.carterduplessis.ca/index.php/s/FxEqMfJffy3nSp7)
- Presented guest lecture in ART 22 at UCSB: *Coding as compositional process*

Winter 2019

- Premiered *Mysterium Cosmographicum* for Oscilloscope and Stereo Playback - realized during the Vector Hack workshop led by MAT guest lecturer Derek Holzer (http://rodneyduplessis.com/videos/Mysterium%20Cosmographicum.webm)

Spring 2019

- Created the first version of *pulsar~* - An implementation of Pulsar Synthesis[37] in Pure Data (https://github.com/rodneydup/pd-pulsar)
- Created *xieve* - An implementation of Xenakis' sieve technique[26] in Pure Data (https://github.com/rodneydup/xieve)
- Won 1st prize Corwin Award (Percussion category) for *Sisyphe Heureux*
- Won 2nd prize Corwin Award (Electroacoustic category) for *Glossopoeia*
- Laptop duo performance at MAT End of Year Show with Aaron Anderson
- Presented *Mysterium Cosmographicum* at MAT End of Year Show
- Network music performance with Ken Fields and others at Sound + Science Symposium

Summer 2019

- Attended field recording masterclass at the Bogong Center for Sound Culture in Alpine National Park, Australia.
- Attended live electronics masterclass at the Chigiana Academy in Siena, Italy.
- Attended Summer Academy at Musiques & Recherches in Brussels, Belgium.
- Co-directed UCSB Summer Music Festival, invited MAT students to present installations.

Fall 2019

- Appointed Teaching Assistant in MUS 109 (Instructor of record: Curtis Roads)
- Network music performance with Sudo Ensemble at NowNet Arts Conference 2019
- Premiered *De Rerum Natura* - An acousmatic piece composed using physical

metaphorical intuitive techniques described in this text.
(http://rodneyduplessis.com/musicPages/De%20Rerum%20Natura.html)

- Created a beta version of *CHON* (https://github.com/rodneydup/CHON)

Winter 2020

- Created *HeatWaves* data visualization for the Allosphere with Kramer Elwell and Raphael Radna. Not presented in the Allosphere due to the COVID-19 shutdowns. (https://nextcloud.carterduplessis.ca/index.php/s/TGHPDmfctf2i66T)
- Led workshop at Womxn/Hacks 2.0 Conference at UCSB: *Intro to computer music programming in Pure Data*
- Co-Led (with Dr. Elizabeth Hambleton) workshop at Alliance of Women in Media Arts & Sciences at UCSB: *Intro to computer music programming in Pure Data*

Spring 2020

- Finalist in ASCAP/SEAMUS Award for *De Rerum Natura* (under the name "*Dimensionless*")
- Presented *De Rerum Natura* (under the name "*Dimensionless*") at the SEAMUS 2020 National Conference
- Awarded Graduate Division Dissertation Fellowship for Fall 2020

Summer 2020

- Created *Alloscope* - A software made in allolib for visualizing stereo sound as an oscilloscope in x-y configuration (https://github.com/rodneydup/Alloscope)
- Network music performance with Sudo Ensemble at Earth Day Art Model conference (online)
- Left CREATE Technical Coordinator position
- Premiered *Pandæmonium* for piano four hands - the first piece composed using CHON (http://rodneyduplessis.com/musicPages/Pandaemonium.html)

Fall 2020

- Awarded MAT Merit Supplemental Support Research Stipend for academic excellence and service to the program.
- Completed and released *EmissionControl2* with Curtis Roads and Jack Kilgore (https://github.com/EmissionControl2/EmissionControl2)
- Presented guest lecture at Composition Forum UCSB: *Controlling EmissionControl2*
- Installed *Oscilla*, an interactive audiovisual installation, at The Museum of Sensory and Movement Experiences with Xindi Kang

(http://rodneyduplessis.com/musicPages/Coacervate.html)

- Began working on *PRISM* (ongoing) a suite of VST plugins for spectral manipulation of sound including spectral dilation.
- Premiered *Coacervate* for violin and electronics - composed using physical metaphorical intuitive techniques and spectral dilation using *PRISM* (http://rodneyduplessis.com/musicPages/Coacervate.html)
- Honorable mention in Destellos International Electroacoustic Competition for *De Rerum Natura*
- Finalist in SIMEC Electroacoustic Music Competition for *De Rerum Natura*
- Finalist in Musica Nova International Electroacoustic Music Competition for *De Rerum Natura*

Winter 2021

- Won 1st prize Corwin Award (Solo/Chamber category) for *Coacervate*
- Presented guest lecture at Composition LAB (Estonian Academy of Music and Theatre): *CHON: from physics to musical gesture*
- Presented guest lecture in the Sonic Art class at Colby College: *Algorithmic Composition*

Spring 2021

- Appointed Teaching Associate for MUS 8/88 (as instructor of record)
- Presented *Coacervate* at SEAMUS 2021 National Conference
- Completed *CHON* version 1.0
- Completed Masters of Science Degree in Media Arts & Technology