Bringing Harmony to Computational Science Pedagogy

Richard Roth[1][0000−0001−8622−8134] and William Pierce[2][0000−0002−1974−1779]

Hood College, Frederick MD 21701 USA
{rothr,pierce}@hood.edu

Abstract. Inherent in a musical composition are properties that are analogous to properties and laws that exist in mathematics, physics, and psychology. It follows then that computation using musical models can provide results that are analogous to results provided by mathematical, physical, and psychological models. The audible output of a carefully constructed musical model can demonstrate properties and relationships in a way that is immediately perceptible. For this reason, the study of musical computation is a worthwhile pursuit as a pedagogical tool for computational science. Proposed in this paper is a curriculum for implementing the study of musical computation within a larger computer science or computational science program at a college or university. A benefit of such a curriculum is that it provides a way to integrate artistic endeavors into a STREAM program, while maintaining the mathematical foundations of STEM. Furthermore, the study of musical computation aligns well with the arts-related components of Human-Centered Computing. The curriculum is built on the following two hypotheses: The first hypothesis is that the cognitive, creative, and structural processes involved in both musical composition and computer programming, are similar enough that skilled computer scientists, with or without musical backgrounds, can learn to use programming languages to compose interesting, expressive, and sophisticated musical works. The second hypothesis is that the links between music, mathematics, and several branches of science are strong enough that skilled computational scientists can create musical models that are able to be designed using vocabularies of mathematics and science. While the curriculum defined in this paper focuses on musical computation, the design principles behind it may be applied to other disciplines.

Keywords: Computational Science · Pedagogy · Curriculum Design · System Modeling · Music Composition · Algorithms · Sound Design · Digital Signal Processing · Psychoacoustics · Embedded Systems

1 Introduction

Outlined in this paper is a curriculum for implementing a computational music curriculum at a college or university. This curriculum is designed to be a sub-curriculum for a larger computer science or computational science program. In

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

this curriculum, music is studied using the vocabularies of mathematics and science. The purpose of this curriculum is to use musical models as pedagogical tools for computational science.

This paper begins with a discussion of music as it was studied in ancient Greece, as the study of numbers evolving over time. This discussion is included to provide a historical precedent for using music as a pedagogical tool for mathematics and the sciences. The paper then discusses the logic and usefulness of using shared vocabularies to study the relationships between diverse disciplines such as music, mathematics, and the sciences. Following this discussion, several examples are given to demonstrate how musical models can be effectively used in teaching multiple areas of modern mathematics and science.

The computational music curriculum is then discussed from a design perspective. This discussion is followed by a case study that provides details on how the computational music curriculum is implemented at Hood College. The paper concludes with observations and thoughts regarding the resources required to implement this curriculum, how the curriculum might be put in place at other institutions, and how computational music fits in with mathematics-based interdisciplinary trends such as Science, Technology, Engineering and Mathematics (STEM), as well as Human-Centered Computing (HCC).

2 The Laws of Musical Motion

Long before gravity and inertia were defined, the scientists, mathematicians, and philosophers of ancient Greece used the vocabulary of music to discuss the balance and beauty observed in the motion of the sun, moon, and stars. Pythagoras used the term *celestial harmony* to describe this motion. For Pythagoras, Plato, and others, an understanding of the universe was achieved through the study of the quadrivium: mathematics, geometry, music, and astronomy. The role of music in the quadrivium was the study of the evolution of numbers over time [2].

The Greek notion of celestial harmony, survived well into the seventeenth century, as evidenced by the publication of Johannes Kepler's "Harmonice Mundi" in 1619. In this text, Kepler uses his research and discoveries in astronomy, to support the theory that motions of musical voices, and the motions of the heavens, both abide by the same laws [5].

Later that century, composer, organist, and music theorist Johann Joseph Fux, defined in his text "Gradus ad Parnassum," the mathematical foundations of musical elements, and the laws governing musical counterpoint; which is the study of musical voices in motion [1]. Fux discussed musical counterpoint in a way that recalled the concepts of Harmonice Mundi and celestial motion by describing musical lines as *voices* that move through musical space, much like the way objects move through physical space.

Studying music as both an art and a science, has allowed intellectuals throughout history, with a framework, or pedagogical tool set, for studying and learning about elements and forces that may have been at their time, out of reach.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

3 Shared Vocabularies

Shared vocabularies allow the vocabulary from one field of study, to describe the properties of another field of study. This is what allowed Pythagoras and Plato to study astronomy and physics using the vocabulary of music theory. It also allows us today to discuss and understand much of music theory using the vocabularies of science.

The disciplines of computer programming and music composition demonstrate how instructors can use shared vocabularies as teaching tools. Melodies for example, are described by the Harvard Dictionary of Music as "successions of pitch-plus-duration values." This concept is easily described to a student of computer science as an array of pitch-value objects.

To continue this example, instructors might describe musical counterpoint as the execution of two or more well-constructed arrays of pitch-value objects that are cycled through on separate, but concurrent threads.

The object of drawing parallels between programming and music combination in this way is not to circumnavigate the study of traditional music theory. Instead, the object of drawing these parallels is to use musical concepts such as melodies and counterpoint to help computer science students gain a deeper understanding of computer programming concepts such as arrays, loops, structures and classes, and con-currency and multi-threading.

4 Pedagogical Areas

This section provides concrete examples of how musical components and constructs are used a pedagogical tools to approach concepts of computation, algorithm design, artificial intelligence, and kinetics.

4.1 Algorithm Design

Like most algorithmic compositions, the output of the program is different each time, but it is still recognizable from hearing to hearing. In the following code sample, complex musical phrases are created algorithmically from a single base pitch and an array of pitch changing values. The program derives rhythmic components by dividing a predetermined time interval (one second for example) into a variable number of parts. This code sample was written in the ChucK music programming language [6].

```
// Clarinet Solo
36 => int rootPitch;
[0,2,4,5,6,7,9,11] @=> int myPitches[];

void writeMusic(int beats, int pitches[])
// PRE: The number of measures in the composition is predefined
// FUNCTION: Determines notes and pitches for the current measure
```

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

```
// POST: The computer has composed and played one measure
{
    Math.random2(1,4) => int octaves; // Determine octave
    Math.random2f(-1.0,1.0) => s_pan.pan; // sound placement

    for( 0 => int j; j < beats; j++){ // generate music
        Std.mtof( (rootPitch + (12 * octaves))
        + pitches[j]) => s.freq;
        MAX / (ROUNDTRIP / beats) => float stepsize;
        // MAX 0.9, ROUNDTRIP 200

        for ( 0 => float i; i < MAX; stepsize +=> i ){
            i => s.gain;
            tempo => now;
        }
        for ( s.gain() => float i; i > 0; stepsize -=> i ){
            i => s.gain;
            tempo => now;
        }}}
```

4.2 Artificial Creativity

In this example, two arrays are used to store separate parts of a repeated drum pattern. The main function (not shown) uses concurrent threads, thus creating a complete pattern at run time. The program incorporates a simple algorithm that makes decisions on how to keep the pattern interesting over time.

```
// Drum Pattern
0.0625 => float VARIETY; // Variety Factor

// SETUP PATTERN:
// Measures are divided into four groups of 16th notes.
// 1 and 2 are drum beats, 0 is silence
[1,0,0,0, 2,0,0,2, 1,0,1,0, 2,0,0,0] @=> int kickSnareBeats[];
[2,0,1,0, 2,0,1,0, 2,0,1,0, 2,0,3,0] @=> int highHatBeats[];

// ADD VARIETY: Determine the likelihood array values will change
[0,1,0,2, 0,2,0,3, 0,3,0,4, 0,4,0,5] @=> int varietyLevel

for(0 => int b_ctr; b_ctr < bNum; b_ctr++){ //Play 1 measure
beats[b_ctr] => theBeat; // Get either a drum beat or silence
vBeat(b_ctr, theBeat) => theBeat; // Maybe change the beat setting
if (theBeat == 1)
0 => k1.pos;
}
```

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

4.3 Geometry in Motion

Figures one through four show two excerpts from "Gradus ad Parnassum," Johann Fux's 1725 pedagogical treatise on musical counterpoint. In these figures, pitches are shown first as notes in music notation, and second as frequency in Hz on a two-dimensional graph. It may be tempting to look at this set of pitches mathematically and conclude that there are two entities; one is a sound whose fundamental vibration frequency is x, and another sound whose fundamental vibration frequency is y. With this view one can observe that there is a distance in frequency space of y – x. This is shown in figure 1.

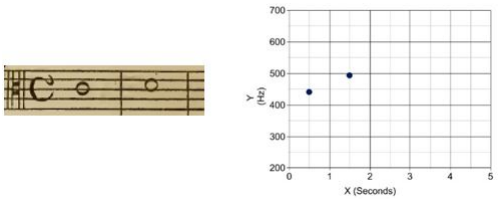

Fig. 1. The image on the left is an excerpt from Fux's Gradus ad Parnassum showing two pitches on a musical staff. The image on the right shows the fundamental frequency in Hz of the two pitches on a two-dimensional graph.

If however, an alternate view is taken, and one considers that a single sound source, or voice in musical terms, started at point x, and then moved over time, and in frequency space, to point y, then we can observe a direct parallel to kinetics; the study of geometry in motion. This is shown in figure 2.

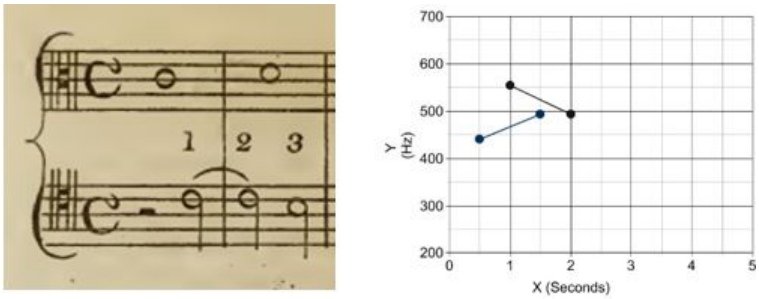

Fig. 2. The same two pitches from figure one, shown in Hz on a two-dimensional graph.

If we now consider a sound object consisting of several voices (v1, v2,. . . ,v6) we can map the movement of an object in two dimensional space, to music as shown in figure 3.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

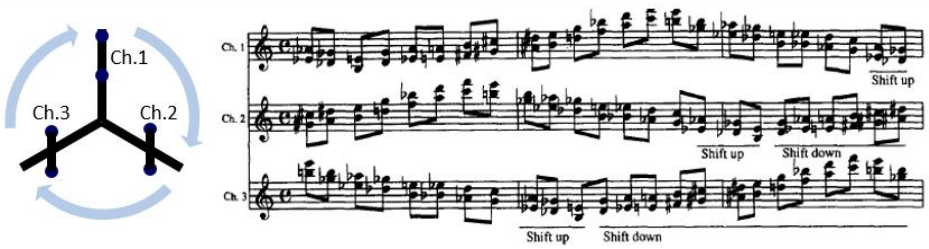

Fig. 3. In this example, each leg of the tripod on the left, is represented on the right by two voices in a musical mapping. A variation on Bresenham's circle drawing algorithm is used to rotate the tripod and to generate musical information.

Lastly if we add amplitude (loudness) as a third dimension as shown in figure 4, we can study properties of motion and force in a way that may by seen and heard.

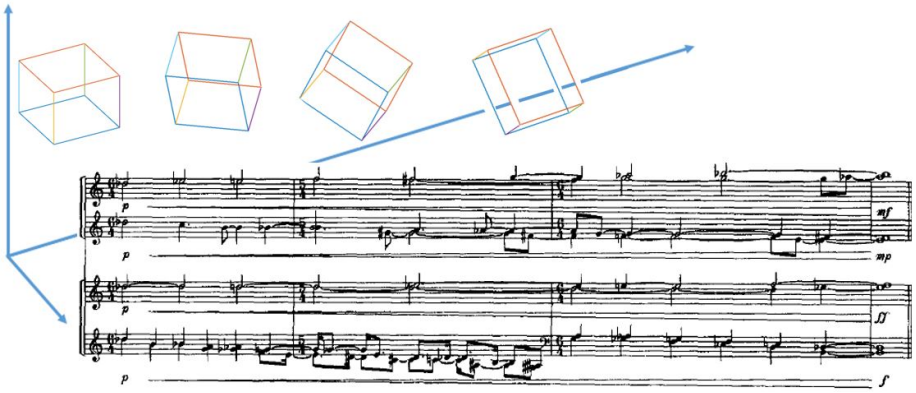

Fig. 4. In this example, the elegance of motion is translated to music using frequency (pitch), and amplitude (loudness) to create three dimensions for sound objects to travel through. A variety of three-dimensional motion algorithms drive the movement of the cube, and generate the musical content [7].

5 A Proposed Curriculum for Computational Music

To facilitate the discussion of music and computational science, the term *computational music* is introduced. This term is used in this paper to encompass the following:

– Discussions of musical properties such as harmony, melodic movement, consonance, and dissonance, in purely scientific terms.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

– Creation of musical models as approachable pedagogical tools for creating models related to science and mathematics.

5.1 Modifying Existing Programs

A problem many colleges and universities face, is the challenge of implementing new elements such as computational music courses into firmly established math and science programs.

One possible way to do this might be to design a single elective course in Computational Music where students complete projects that incorporate mathematical elements such as numeric patterns or evolving functions. Other projects might explore the creation of computer models that emulate musical instruments or natural sounds. This solution has been implemented successfully by many colleges and universities. A problem with this approach, however, is the possible novelty or survey course perception that has no path forward for interested students.

Another possible solution is to add topics such as music programming and music computation to graduate and advanced undergraduate courses in topics such as artificial intelligence or robotics. This approach however, provides no preparatory support in music programming, nor the scientific and mathematical properties of music. Students might therefore be limited in what they could accomplish in a semester, and the time spent teaching the musical material, might deter from the main objectives of the course.

The solution proposed in this paper is to offer a small but broad curriculum of computational music courses that allow students from different disciplines, and different stages of their education, to find courses that align with their objectives and interests.

5.2 Design Principles

This proposed curriculum is built on three design principles. Principle one: any course in the curriculum can serve as a possible entry point. Principle two: students may complete courses in an order that suits their interests and goals provided that prerequisites are met. Principle three: allowable prerequisites for advanced courses include other courses in the curriculum, and related courses from other disciplines. This flexible approach makes the curriculum available and accessible to the largest possible number of students.

Principle one allows for students to enter the computational music curriculum in a way that aligns with their academic year, or their major. First-year or second-year undergraduate students for example, may choose a survey course with no prerequisite as their entry point. Survey courses for this curriculum might include an introductory course in computer music, or a course on the scientific properties of music.

Third and fourth-year undergraduates, as well as graduate students, may choose a more advanced course that does have prerequisites for their entry point.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

More advanced courses in this curriculum might include a course in digital signal processing, or a course in music programming.

Principle two is concerned with the course or courses a student may take after completing their entry-point course. Students that started with a survey course now have the option of taking another survey course, or moving to a more advanced course. Students that started with a more advanced course may take another advanced course, a survey course, or move on to a specialty course.

Principle three allows students to work their way up to advanced or specialty courses within the curriculum, or to take these courses using prerequisites earned from other disciplines. Specialty courses in a computational music curriculum might have titles such as Music and Sound for x, where x might be an area in which the college or university already specializes, such as Robotics, Cognitive Psychology, or Algorithm Design.

An example of an implementation of these three design principles in a computational music curriculum follows in the next section.

6 A Case Study: Hood College, Frederick Maryland

6.1 Hypotheses

Computational music began in the Computer Science Department at Hood College, a small traditionally liberal arts college in Frederick Maryland, with a single summer-session course entitled *Musical Computing.* The starting hypothesis behind this course was that the cognitive, creative, and structural processes involved in both musical composition, and computer programming, are similar enough that skilled computer scientists, with or without musical backgrounds, can learn to use programming languages to compose interesting, expressive, and sophisticated musical works. The overall structure of the course is shown in figure 5.

This single course offering expanded into a six course curriculum designed to use musical models to assist in the pedagogy of computational science, and computer science. As this curriculum evolved, the original hypothesis expanded to the following: The links between music, mathematics, and several branches of science are strong enough that skilled computational scientists can create musical models that are able to be designed using vocabularies of mathematics and science.

Table 1 lists the course titles and prerequisites for the Hood College computational music curriculum. The following sections contain descriptions of each course. The descriptions include course overviews, special topics, and explanations of how the courses support computational science pedagogy.

6.2 The Science of Music

This is a course that approaches all aspects of music using the vocabularies of science, mathematics, and computational models. Specific topics include: the

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

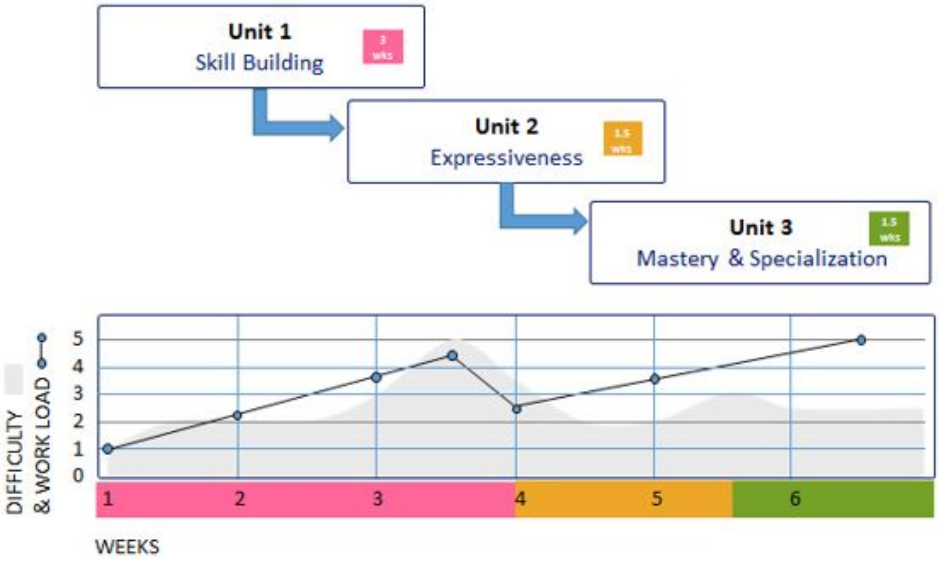

Fig. 5. A basic course structure in the Music Computation curriculum.

Table 1. Hood College Curriculum for Computational Music.

Course Title	Prerequisite
The Science of Music	None
Introduction to Computer Music	None
Digital Signal Processing	The Science of Music, Intro. to Computer Music, or an introductory course in electrical engineering.
Musical Computing	Intro. to Computer Music, or a previous course in a modern object-oriented programming language.
Algorithms and Music Composition	Musical Computing, or a Previous course in Data Structures and Algorithms
Music and Sound in Embedded Systems and Robotics	Musical Computing, Algorithms and Music, Composition, or a previous course in embedded systems programming, or robotics.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

physics of sound, auditory perception, derivation of intervals and scales, consonance and dissonance, and structures used in music composition. The value of this course in computational science pedagogy lies in its emphasis on physical properties. Furthermore, several lectures and class projects focus specifically on computational model creation. There are no prerequisites for this course.

6.3 Introduction to Computer Music

This course is an introduction to the use of computers for creating, recording and editing musical information. Specific topics include: music history leading to the advent of computer music, physical properties of sound, human perception of sound, and current trends in the musical applications of computers. Pedagogical objectives related to computational science include studies in computational thinking and algorithmic music. There are no prerequisites for this course.

6.4 Digital Signal Processing (DSP)

This course is concerned with the representation, transformation and manipulation of signals using computer technology. Specific topics include DSP theory, methods, and algorithms. This course supports computational science pedagogy in that the concepts, methods, and algorithms discussed in this class are directly related to digital representations of sound, speech, and music; and are used in the creation of any acoustic model. Allowable prerequisites for this course are The Science of Music, Introduction to Computer Music, or an introductory course in electrical engineering.

6.5 Musical Computing

This is a course in the musical applications of computer programming and computational science. This course in many ways resembles the Introduction to Computer Music course described earlier. Musical Computing however, assumes a higher level of skill and experience in programming and mathematics. In this course, students use the structural elements of computer programming such as arrays, loops, classes, objects, and multi-threading, to compose music and design sounds. Students enhance their coding skills by writing programs that produce sophisticated musical compositions. This course supports computational science pedagogy through programming projects that model and support human musical composition processes. Allowable prerequisites for this course are Introduction to Computer Music, or a previous course in a modern object-oriented programming language.

6.6 Algorithms and Music Composition

This is a course in composing music using algorithms and computational models. Specific topics include: algorithmically processing data for music creation,

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

creating algorithms that write music based on minimal input, and live algorithmic music performance. Computational science pedagogy is supported in this class through the study of creative algorithm design. Allowable prerequisites for this course are Musical Computing, or a previous course in data structures and algorithms.

6.7 Music and Sound for Embedded Systems

This is a course in the musical and sonic applications of embedded systems programming. In this course, students write programs that run on stand-alone microcomputers such as those found on Arduino and Raspberry Pi development boards. Specific topics include: efficient coding, multi-threading, use of actuators and sensors to generate and respond to sound, processing of sound information, and robotic music performance. This course supports computational science pedagogy through lectures and student projects relating to the processing of sound information. Allowable prerequisites for this course are Musical Computing, Algorithms and Music Composition, or a previous course in embedded systems programming or robotics.

7 Implementing a Computational Music Curriculum

7.1 Required Resources and Expenses

Realizing the pedagogical benefits of a computational music curriculum can start with little or no resources beyond what a typical college or university already has. These resources include computers, access to freely available software tools, and a standard classroom audio system.

As programs mature, institutions may focus on, or specialize in a specific area. At this point, additional expenses may be incurred. For example, consider a school wishing to create a psychoacoustic modeling lab. This type of modeling and experimentation uses meticulously created sound samples. Researchers accomplish this either by recording natural sounds, or by synthesizing them on a computer. The sound creation process, and subsequent playback both require that a sound's full frequency spectrum is preserved. This normally requires an acoustically treated environment that is equipped with sensitive and high quality microphones, amplifiers and speakers that can preserve a sound's full frequency spectrum. These expenses can be preventative in early stages, but may become possible as a program grows.

7.2 Textbooks

There are excellent texts on this subject, including Elaine Chew's text, "Mathematical and Computational Modeling of Tonality [3]." There is also an abundance of scholarly research in journals and conference proceedings that professors can incorporate into these courses. Nominal expenses may be incurred in subscribing to the societies and organizations that provide this information.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

7.3 Faculty

Teaching this curriculum requires faculty who can discuss sound and music using the following vocabularies: traditional music theory, particularly the principles of harmony, and music composition; the sciences, particularly mathematics, engineering, and physics; as well as computer science and auditory perception.

In the Hood College curriculum, the courses are taught by professors who have fluency in music and one or more of the scientific vocabularies listed. Cross-disciplinary approaches however, are also used at Hood College and might be an effective alternative for many institutions.

8 Conclusion

The rapid growth of the computational music curriculum in the Hood College case study are promising indicators that a curriculum such as the one outlined in this paper, may resonate with prospective students who view computation, science and mathematics as creative endeavors. Due to the success of the Hood College case study, the authors conclude that computational music, if implemented as a sub-curriculum in a larger program, is an effective and motivational tool for computational science pedagogy.

There are several trends in higher education that support this conclusion, including the growing number of Ph.D programs in Human-Centered Computing (HCC) such as those at Clemson University and Georgia Tech. Evidence that computational music can play a pivotal role in HCC, is provided by Nicu Sebe in his chapter on Human-Centered Computing in the "Handbook of Ambient Intelligence and Smart Environments" [8]. In this chapter, Professor Sebe states "Perhaps one of the most exciting application areas of HCC is art." To support this statement, Sebe points to an example where computing is used to translate human gestures to music.

Other trends where computational music can play a key role are STEM and STREAM. STREAM is an extension of STEM that includes *A* for the arts, and *R* which, depending on the context, can stand for reading and writing, or religion. The authors of this paper have concluded from the case study presented in this paper, that the study of computational music not only aligns with the objectives of STREAM initiatives, but maintains the scientific and mathematical foundations on which STEM is built.

In closing, perhaps the primary reasons why computational music brings harmony to computational science pedagogy, are the effectiveness with which one can define musical properties using the vocabularies of science and mathematics; and the fact that music has been a part of scientific and mathematical analysis since ancient times.

9 Acknowledgement

We thank the Hood College Graduate School and our corresponding departments for their support. We are indebted to our students who embraced this curriculum

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49

and worked with us along the way while we rethought, re-planned, and refined it. Sound files that accompany this paper, are available from the Hood College Computer Science Department.

References

1. Apel, Willi: Harvard Dictionary of Music, Harvard University Press, Cambridge Massachusetts, (1950)
2. Burkholder, J. Peter, Grout, Donald Jay, Palisca, Claude V: A History of Western Music. 9th edn. W.W. Norton Company, New York (1999)
3. Chew, E.: Mathematical and Computational Modeling of Tonality: Theory and Applications. Springer, New York (2014)
4. Chew, Elaine: Slicing It All Ways:Mathematical Models for Tonal Induction, Approximation, and Segmentation Using the Spiral Array. Informs Journal on Computing, Vol 18, No.3 (2006)
5. Gingras, B.: Johannes Kepler's Harmonice mundi: A "Scientific" version of the Harmony of the Spheres. Journal of the Royal Astronomical Society of Canada, Vol. 97, Issue 5, p.228 (2003)
6. Kapur, A., Cook, P., Salazar, S., Wang, G.: Programming for Musicians and Digital Artists: Creating music with ChucK. Manning Publications, New York (2014)
7. Roth, R.: Music and Animation Toolkit: Modules for Multimedia Composition. Computers and Mathematics with Applications, Vol. 32, No. 1, pp. 137-144 (1996)
8. Sebe N.: Human-centered Computing. In: Nakashima H., Aghajan H., Augusto J.C. (eds) Handbook of Ambient Intelligence and Smart Environments. Springer, Boston, MA (2010)

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50436-6_49