From Controller User to Instrument Designer: Teaching NIME in a Contemporary Music Context

Akito van Troyer
avantroyer@berklee.edu
Berklee College of Music
Electronic Production and Design
Boston, Massachusetts, USA

Abstract

As commercial music technology and AI-assisted tools become increasingly accessible, more students enter higher education with substantial experience in digital music making. For NIME educators, this creates a new challenge: teaching physical prototyping to students who are already experienced users of digital music tools. While existing NIME pedagogy research primarily addresses novices, musicians familiar with traditional instruments, or engineering students, targeted approaches for students transitioning from proficiency with commercial tools to custom instrument design remain underexamined. This paper presents an undergraduate course specifically designed for this population, based on a full-cycle approach that integrates research, hardware prototyping, composition, public performance, and academic documentation. Analyzing outcomes from 16 students across 2 semesters, we observe that approximately half continued to iterate after the build phase, highlighting how interconnecting building, composing, and performing may foster thoughtful, extended engagement with instruments. The full-cycle structure was also within reach, with a strong majority of students completing all phases. We document the specific institutional conditions, student prerequisites, and resource dependencies under which this approach operates, thereby supporting educators in applying the proposed model in their own contexts.

Keywords

NIME education, digital musical instruments, pedagogy, physical computing, electronic music

1 Introduction

Digital musical instrument (DMI) design education faces a recurring challenge: bridging the gap between musical knowledge and technical implementation [25]. This challenge manifests differently across student populations. Engineering students may lack musical context for their technical work; art students may struggle with electronics and programming fundamentals; and students from mixed backgrounds require differentiated scaffolding [48].

A less-examined population comprises students who are already experienced users of commercial digital music tools, such as grid controllers, digital audio workstations (DAWs), and synthesizers, but lack the technical skills to design custom instruments. These students may possess sophisticated mental models of signal flow, sound design, and performance practice, yet cannot translate their musical ideas into physical interfaces. This paper addresses the research question: *How can NIME design pedagogy be structured for students who are already experienced users of commercial digital music tools but lack technical prototyping skills?*

Table 1: The emerging student profile: students enter higher education with digital music skills but no physical prototyping experience. The distinction between exposure (secondary) and operational fluency (higher education) reflects accumulated practice rather than novel skill acquisition.

Skill Domain	Secondary Education (Increasing Technology)	Higher Education (Paper's Context)	Skills Present
Digital Production	GarageBand, BandLab, Soundtrap	DAW Proficiency	Yes
Hardware Interfaces	Exposure to MIDI controllers and grid interfaces	Operational fluency with commercial controllers (e.gl, Push, Launchpad, and MPC)	Yes
Music Creation	Beat-making Composition	Sound Design Skills	Yes
Physical Computing	— (Not Typically Covered)	Physical Prototyping Skills	No

1.1 Why This Matters Now

The student population documented in this paper, experienced users of commercial digital music tools who lack prototyping skills, may represent a profile that becomes increasingly common in music technology education. While students taking NIME classes are musically curious and sometimes accomplished musicians, the change emerges from their familiarity with digital tools rather than traditional acoustic instruments. We identify three contextual trends that suggest this evolution. We acknowledge that these trends vary in strength and that current data cannot tell us how common this profile will become.

First, commercial music technology has become increasingly accessible at the secondary level. A systematic review spanning 1993-2025 documents that DAWs have become "prevalent" in secondary music classrooms, with platforms like GarageBand, Ableton Live, and Logic Pro appearing regularly in K-12 contexts [21, 45]. A scoping review of technology-enhanced creativity in K-12 music education similarly documents the expansion of technology integration, though it notes variation across contexts

NIME '26, June 23–26, 2026, London, UK

and implementation quality [39]. The COVID-19 pandemic may have accelerated adoption in some settings; one study found that teachers who had not previously used DAWs adopted them during the pandemic and intended to continue afterward [22]. Taken together, this evidence suggests a trajectory of increasing technology adoption, though the extent to which students enter higher education with substantial production experience likely varies by region, school resources, and individual interests.

Second, AI-assisted music tools may further lower barriers to digital music-making. Researchers and developers are actively integrating AI capabilities into music production workflows, including generative systems for composition [16] and AI-based plugins designed to democratize production techniques [61]. As these tools become embedded in DAWs and production environments, students interested in digital music-making may encounter them early, potentially accelerating the development of production skills.

Third, given the current trend toward the adoption of off-the-shelf music technology, educators may increasingly encounter students with proficiency in sound design and fluency with tools. For such students, the pedagogical challenge shifts from teaching music technology fundamentals to bridging the gap between digital fluency and physical prototyping.

Table 1 illustrates the student profile we address by situating skill development along a pipeline from secondary to higher education. Educators who encounter students with some of these characteristics may find specific elements of our approach adaptable, even if wholesale transfer is not feasible.

1.2 Contributions

We present a course, titled *Prototyping Electronic Digital Instruments*, at Berklee College of Music, designed specifically for this population [13]. Our contribution is threefold:

(1) **Documentation of a specific student population:** Our target population presents pedagogical challenges distinct from those faced by novices or engineering students.
(2) **Application of full-cycle pedagogy to this population:** While full-cycle approaches integrating building, composition, and performance exist in the literature [32, 62], these have not been applied to students transitioning from commercial tool expertise, nor have they reported on iteration patterns within the structure.
(3) **Preliminary empirical observations:** We report outcomes from 16 students across 2 semesters based on grading records and Git commits, including evidence that iteration occurs during the interconnected composition and performance phases.

2 Related Work

NIME pedagogy has been documented across diverse institutional contexts since the field's emergence. A systematic review on NIME education reveals that only one study explicitly addresses students who are "proficient sound programmers" with existing DAW experience [30]. Most NIME education research targets complete novices or students without electronic engineering backgrounds [3, 28, 62].

2.1 Existing Pedagogical Models

NIME pedagogy encompasses diverse approaches with different emphases. To situate our course, we identify three common pedagogical priorities, illustrated schematically in Figure 1, while acknowledging that many programs combine elements of both.

Design/Build approaches, common in engineering and interdisciplinary art-technology contexts, emphasize the acquisition of technical skills through instrument construction [40, 41]. Prototyping-focused NIME courses have been documented since the field's inception [69], with continued iterations at the same institution over two decades [42]. The primary outcome is a functional artifact; musical demonstrations, when included, typically serve to verify technical success rather than as extended practice in their own right.

Perform approaches integrate DMIs into existing performance curricula, emphasizing expression and musical practice over design and construction [55]. Students may work with existing instruments or interfaces to develop performance techniques and repertoire.

Full-cycle approaches that integrate building, composition, and performance exist in the literature. Jordà and Mealla [32] describe curricula that connect mapping and expressiveness to performance outcomes; Tomás [62] documents practice-based methods that link instrument construction to artistic exploration. Prior full-cycle courses have served cohorts ranging from non-musician art students [62] to mixed music-and-cognitive-systems graduate students [32].

Our course draws on these precedents while addressing a specific gap: none of the documented approaches report quantitative data on whether students iterate on hardware after initial construction, nor do they address students with pre-existing digital music expertise. The contribution of this paper is not the full-cycle structure itself, but its application to this emerging student population and the preliminary evidence we report on iteration patterns within that structure.

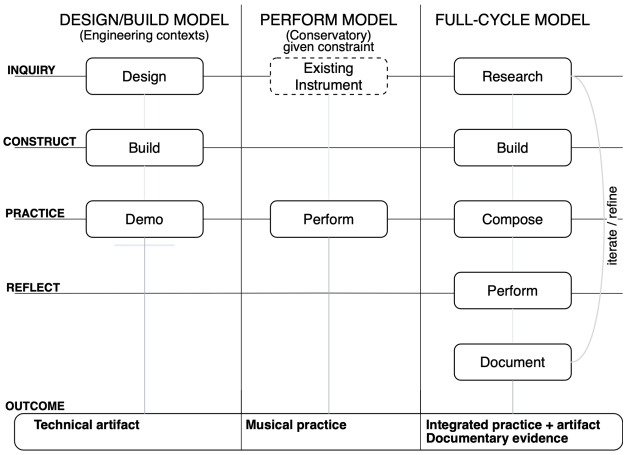

Figure 1: Schematic comparison illustrating different pedagogical emphases in NIME education. *Design/Build* approaches prioritize technical skill acquisition with demonstration as outcome; *Perform* approaches prioritize musical practice, often with existing instruments; *Full-Cycle* approaches integrate building, composition, and performance with explicit iteration. These are not mutually exclusive categories; many programs combine elements but represent different pedagogical priorities.

Several transferable strategies emerge from the broader literature: using visual programming environments that mirror signal flow concepts [46], providing working examples that students modify rather than build from scratch [3], implementing phased curricula with progressive difficulty [30], incorporating performance requirements that connect technical learning to musical goals [18, 30], and using restrictive tool sets to reduce cognitive overload [32].

These strategies inform the design of our course, though their application to students with existing digital fluency requires specific adaptations, which we detail in §5.

2.2 Evolving Student Profile

Research on secondary music education documents a clear temporal trajectory of increasing technology adoption. Early studies (1990s) examined composition programs as novel interventions; by the 2000s, technology integration was increasingly common but unevenly distributed; the 2010s saw technology described as prevalent and commonplace; and recent studies (2020-2025) document technology as foundational to music education transformation [21, 39, 49, 51, 56].

The COVID-19 pandemic accelerated this trajectory. Dockan et al. [22] found that teachers who had not previously used DAWs adopted them during the pandemic and intended to continue afterward, suggesting that external pressures can rapidly shift adoption patterns. The shift from desktop-based software to mobile and cloud platforms (BandLab, Soundtrap, GarageBand for iPad) represents a technological trajectory toward greater accessibility.

This evidence suggests that students entering higher education increasingly arrive with DAW experience, composition skills, and familiarity with commercial controllers: the profile that characterizes the targeted students in this paper. This analysis treats our institutional program as a specialized case and does not seek to generalize prevalence. Rather, the documented trends are used to identify structural conditions that may give rise to comparable populations in other music technology education settings as secondary-level technology adoption expands.

2.3 Gap in the Literature

The gap between musical knowledge and technical implementation is consistently identified as a central challenge [25, 38]. However, existing recommendations largely derive from studies of adjacent populations, such as novices, engineering students, or mixed cohorts. Targeted research on students transitioning from commercial digital music expertise to physical prototyping remains limited.

Our institution's context differs from most documented NIME programs. As a large, specialized institution dedicated to contemporary music education, it is neither a conservatory specializing in classical performance nor an engineering school. Students in our major arrive with substantial experience using commercial controllers (e.g., Ableton Push, Novation Launchpad, and Akai MPCs). Our course addresses their specific transition from controller *user* to instrument *designer*. While full-cycle approaches exist in NIME pedagogy, documented applications have not explicitly targeted this population or reported on iteration patterns within full-cycle structures for students with this profile.

3 Institutional Context

3.1 Student Profile and Prerequisites

The student profile described in this paper emerges from two institutional mechanisms: program admission requirements and prerequisite course completion.

Admission to the Electronic Production and Design (EPD) program at Berklee College of Music requires applicants to demonstrate existing proficiency with DAWs and music production tools, typically developed through introductory coursework [10]. Many students apply because they aspire to be electronic musicians and producers, bringing experience with commercial controllers. Once enrolled, students complete required coursework in modular signal flow, programming environments for interactive audio, and digital audio production before reaching our course. Prerequisites for both reported semesters included prior coursework in either visual programming for audio systems [12] or introductory text-based programming [9]. The course is typically taken by upper-level undergraduates who have completed the program's core curriculum.

Given this curricular pathway, the typical student in the program possesses proficiency with commercial DAWs, experience with commercial controllers, understanding of signal flow and MIDI routing, experience with visual programming (Max [20]) or text-based programming (Python [54] and JavaScript [23]), and performance experience in electronic music ensembles. Crucially, students lack fundamental electronics concepts (voltage, resistance, circuit construction), microcontroller programming, sensor integration, and physical prototyping skills. This profile, substantial digital fluency combined with absent hardware skills, defines the pedagogical challenge our course addresses.

As for prior NIME design exposure, students arrive with sustained experience in existing musical interfaces through prior coursework on modular synthesis [11], DAWs [14], and MIDI controllers [6]. Many also complete a Max programming course in which they build UI-based controllers and encounter mapping fundamentals: scaling sensor or MIDI ranges to synthesis parameters using objects such as *[scale]* and *[zmap]*. Students coming from a text-based programming course encounter mapping concepts via working with Processing [52] and Processing.py [53] via functions such as *map()* and *constrain()*. This prior exposure constitutes implicit design vocabulary: students arrive familiar with how existing systems map control to outcomes, even if they have not encountered NIME design as a named theoretical activity. The course builds on this baseline rather than introducing mapping concepts from first principles.

One institutional factor shaping the course's performance emphasis is Berklee's recognition of digital performance systems as legitimate instruments of study (titled *Electronic Digital Instrument*), equivalent to traditional instruments [8]. This recognition means public performance requirements align with existing program culture rather than introducing unfamiliar expectations.

3.2 Course Materials and Starter Kit

Every student in the course is provided with a standardized starter kit ($94) containing electronic components used throughout both laboratory exercises and project phases. The kit includes a Teensy 4.0 microcontroller [50], an accelerometer with a digital interface, an audio adaptor board, breadboards, resistors, LEDs, sensors (force-sensitive resistors, photocells, potentiometers), buttons,

and basic wiring components. A complete bill of materials, including model numbers, quantities, and sourcing information, is available in the course repository[1].

This standardized kit serves multiple purposes: it ensures all students have consistent materials for lab exercises, constrains initial project complexity to reduce cognitive overload, and establishes a baseline component vocabulary that the course materials support. Students typically supplement the starter kit with additional components (a recommended budget of $100) specific to their instrument designs during the building phase. All source codes and program files for the course are publicly available in the same repository.

Standardization of the kit underwrites the lab-phase toolchain constraint described in §5: every student works through identical hardware and software during the first five weeks, which keeps lab instruction efficient and enables peer troubleshooting across a cohort with no prior prototyping experience.

4 Course Design

EPD's program-level goals emphasize developing graduates who can both produce contemporary electronic music using existing tools and design new technical systems for music [5]. This course advances the latter by introducing physical prototyping as a complement to the digital production fluency students develop earlier in the curriculum. Intended learning outcomes are: (i) integrate sensors, microcontrollers, and audio synthesis to produce functional digital musical instruments; (ii) apply mapping strategies to translate gestural input into musical parameters; (iii) compose original musical material that exploits a custom instrument's affordances; (iv) deliver public performance with appropriate technical preparation; and (v) document technical and creative work to professional standards. Achievement is assessed through phase-specific rubrics described below.

The course carries three credit hours, against a minimum full-time undergraduate load of 12 credits per semester at the institution. The course meets weekly for a single 170-minute session combining lecture and laboratory work; lab exercises during weeks 1–5 are started during class time with instructor support, then extended, refined, and submitted as assignments via GitHub pull request by the following week. Project-phase weeks include intermediate submissions at proposal, review, and demonstration milestones (§ 4.1); each milestone receives formative feedback through pull-request review before contributing to summative assessment.

4.1 Five-Phase Structure

Our course implements a full-cycle structure, drawing on precedents in NIME pedagogy [32, 62] while adapting the approach for students with commercial digital music experience. Students complete five integrated phases across 15 weeks (see Figure 2):

Phase 1: Research (Weeks 1–6) Students analyze five contemporary DMIs, including at least two from NIME, documenting sensor technologies, processing platforms, interaction paradigms, and musical applications. A presentation synthesizes findings and proposes implementation approaches.

Phase 2: Instrument Building (Weeks 7–9) Students design and construct proof-of-concept prototypes of original DMIs. Using the starter kit, supplemented with project-specific components, students develop functional prototypes that demonstrate their core interaction concepts. Weekly milestones include proposal, materials review, and functional demonstration. During the project phase, students were encouraged to draw on NIME practices introduced in class, exploring directions such as augmented [37, 44], assistive [26, 34, 35], or educational instruments [60, 65], voice-based interfaces [33, 36, 58], participatory platforms [15, 64, 66], AI-driven systems [17, 68], and rhythm- or timbre-focused designs [57, 63, 67].

Phase 3: Composition (Weeks 10–11) Students compose original music for their specific instrument, exploring the reciprocal relationship between instrument affordances and compositional choices.

Phase 4: Performance (Weeks 12–14) Students prepare technical riders, conduct dress rehearsals, and perform publicly at an end-of-semester showcase.

Phase 5: Documentation (Concurrent, Week 15) Students maintain documentation on GitHub throughout the semester, culminating in a NIME-format paper. The NIME-format paper is scaffolded undergraduate technical writing rather than a peer-review-ready research paper. It documents the instrument, mapping decisions, compositional process, and performance outcomes using the NIME template, primarily to develop technical writing skills and produce portfolio material; students interested in publication are encouraged to develop the paper further outside the course.

4.2 Pedagogical Rationale

The course was designed to enable students to experience the full cycle of building a musical instrument as a NIME practitioner, an approach documented in prior NIME pedagogy [32, 62], but here adapted for students transitioning from off-the-shelf tool experience. Building a DMI requires interdisciplinary thinking that spans engineering and artistic expression. The course also ensures students produce substantial portfolio material for job and graduate school applications. All instrument-building, composition, and performance work is individual; each student designs, builds, performs with, and documents their own instrument. Peer feedback through in-class review is collaborative, but assessment is individual.

Requiring original composition and public performance together forces students to engage deeply with their instrument's affordances over extended periods. This extended engagement, in which students move among refining hardware, developing musical material, and preparing for performance, may help reveal interface limitations that are invisible during brief demonstrations. For example, when a gesture fails to reliably produce consistent sound, students revisit their sensor conditioning or code; when a mapping does not yield expected musical results, they refine their sound design and parameter ranges. The phases are intentionally intertwined rather than strictly sequential: preparing for performance necessitates revisiting both instrument and composition.

Requiring a public performance helps create authentic stakes that motivate technical reliability. Berklee students are accustomed to performance requirements in other coursework; the showcase aligns with the program culture while demanding a professional presentation. The public nature of performance increases pressure on students to produce high-quality work.

The choice of NIME paper formatting for the final documentation helps develop professional writing skills and creates portfolio

[1] https://github.com/EP-361/EP-361

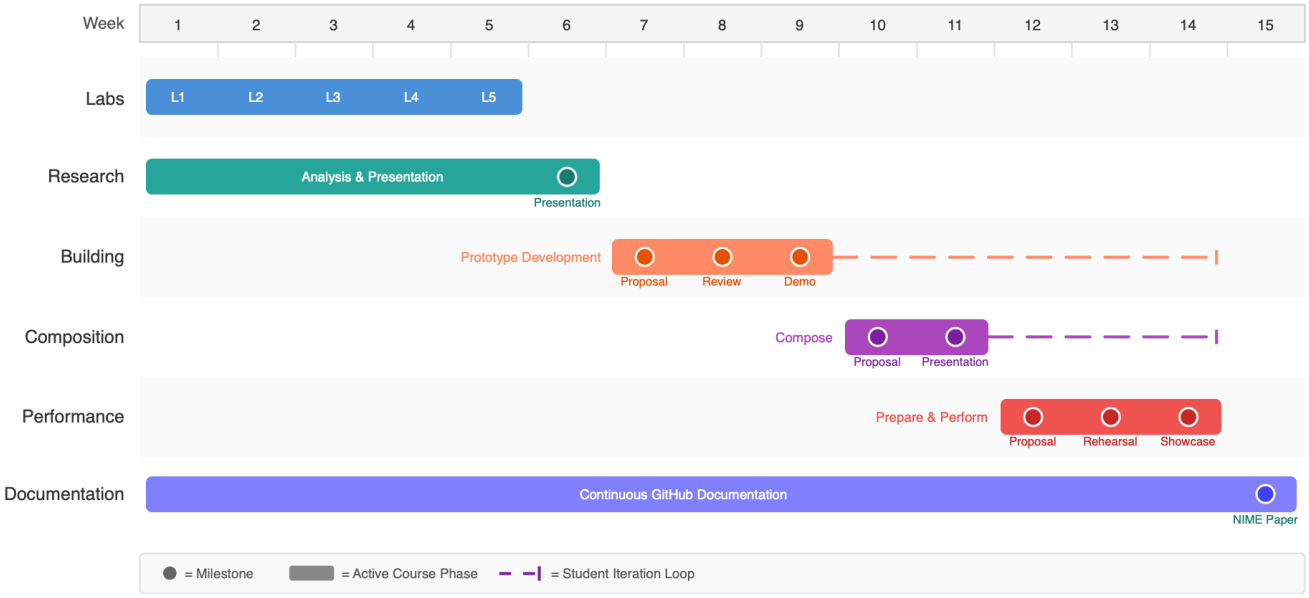

Figure 2: Semester timeline. Laboratory exercises (weeks 1-5) precede project phases. The dashed arrow indicates the student iteration loop. We observed that 47% of students modified hardware after the build phase concluded, during the interconnected composition and performance phases. Documentation runs concurrently via GitHub.

Table 2: Laboratory topics and associated skill development

Lab	Topic	Skills Developed
1	GitHub & NIME Analysis	Version control, technical writing
2	Electronics	Voltage dividers, sensor conditioning
3	Microcontroller	Digital/analog I/O, PWM, timing
4	MIDI	Protocol implementation, state management
5	Audio	Teensy Audio Library, synthesis

materials. For students interested in publishing, it provides a foundation that can be developed toward conference submission after the semester, which is valuable for graduate school applications. For example, students from the course have subsequently developed their documentation into conference submissions [31, 70].

4.3 Laboratory Exercises

Five laboratory exercises, conducted during weeks 1-5, scaffold technical skills before project phases begin (Table 2). These exercises systematically address the 'Physical Computing' skills identified as absent in Table 1. Labs use standardized starter kit components and provide working code templates that students modify rather than build from scratch. Example code and program files are shared through the class materials GitHub repository.

4.4 Design-Pedagogy Content

Design pedagogy in the course is concentrated rather than course-spanning. A dedicated lecture in Week 10 (Mapping and Scoring) covers mapping classifications: one-to-one, one-to-many, many-to-one, and many-to-many [29]; the relationship between mapping choice and expressive identity; response curves and noise conditioning; phrasing and the distinction between note-based and sound-object-based control; audience perception of gesture-sound relationships; and notation strategies including graphic scores and visual music. The lecture is deliberately timed to coincide with the start of the composition phase, where students directly confront the mapping decisions the lecture addresses. No standalone assignment accompanies the lecture; instead, mapping content is applied immediately in the composition project's iterative refinement of gesture-to-sound relationships.

This allocation differs from courses where design theory anchors the curriculum. Jordà & Mealla structure an entire term around mapping and expressiveness using a deliberately constrained technical platform [32], and Tomás dedicates weekly practice-based assignments to embodiment, conceptualization, and gestural design [62]. The narrower design-pedagogy footprint here reflects a tradeoff: students with strong production fluency but limited hardware experience often require more instructional time for prototyping scaffolding than peers with balanced or beginner skill profiles, leaving less time for explicit design theory. Student feedback after the first cohort identified mapping as a specific area for expanded coverage; the Week 10 lecture was correspondingly expanded for the second cohort (§6.4). The framing of design pedagogy as 'concentrated' therefore describes a trajectory rather than a fixed allocation (§7.4).

5 Bridging Strategies

Our course employs several bridging strategies informed by prior research on NIME pedagogy [3, 18, 30, 32, 46]. We do not isolate the effects of individual strategies; rather, we report them as design decisions whose collective contribution to observed outcomes cannot be separated in this study. These strategies may be relevant to other contexts where educators encounter students

with similar profiles: substantial digital music experience but limited physical prototyping skills.

Visual Programming Environments. Students leverage existing Max experience for synthesis engine development. Most students implement MIDI control from Teensy to Max patches or Max for Live devices within Ableton Live. Some students run synthesis engines directly on the microcontroller using the Teensy Audio Library for embedded instruments. The course remains open-ended regarding software choice in the project phase; students have also used SuperCollider [59] and Csound [19].

Working Examples and Templates. Laboratory exercises provide functional code templates that students modify incrementally. Example code is shared through a class materials GitHub repository organized by weekly topics. This approach parallels how students learn commercial tools: by modifying presets rather than building from scratch.

Phased Curricula with Progressive Difficulty. The five-phase structure implements progressive disclosure. Students encounter electronics and microcontroller programming (unfamiliar) before composition and performance (familiar), with familiar phases creating motivation and evaluation contexts. The emphasis is on the instrument-building phase, as students cannot compose or perform without functional instruments.

Performance Requirements. Public performance connects technical learning to musical goals that our students already value. Technical reliability becomes meaningful when the stakes are authentic. The public nature of performance increases pressure on students to produce high-quality work.

Constrained Tool Sets. The course operates a two-tier toolchain policy. During the laboratory phase (weeks 1–5), all students work with a single shared stack: Teensy 4.0 microcontrollers, the standardized starter kit, Fritzing [27] for circuit documentation, the Arduino IDE [1] for microcontroller code, and Max for audio synthesis. All instructional materials, such as lab handouts, code templates, circuit schematics, and example patches, are built around this stack. No deviation is permitted in the lab phase, which keeps cognitive load manageable for students with no prior prototyping experience and ensures peer troubleshooting is possible across the cohort.

During the project phases (weeks 6–14), scoped extensions are permitted where a project requires a capability the lab stack lacks, and the alternative platform preserves conceptual continuity with prior lab work. In practice, observed extensions include Bela [2] (which supports Arduino-style code structure compatible with Teensy programming patterns), ESP32 [24] (which supports Arduino code and adds wireless capability that the Teensy lacks out of the box), and SuperCollider and Csound (used by students who completed the program's prior SuperCollider [4] and Csound [7] courses and could translate Max concepts directly). Teensy was selected as the lab platform partly because of this transferability: students switching to Bela or ESP32 retain the Arduino programming model, and the Teensy Audio Library remains relevant when students choose to embed synthesis on the microcontroller. The constraint is therefore on initial learning, not on final implementation.

Reflective Documentation. Continuous documentation via GitHub serves multiple functions. Writing maintains critical thinking amid AI assistance: students are encouraged to use AI tools, but must articulate their intent through documentation. The shared GitHub repository facilitates peer learning, as students can view and contribute to one another's work. GitHub repositories store all project artifacts: README documentation, project proposals, Arduino sketches, synthesis patches, Fritzing-format circuit schematics, links to externally-hosted video and audio documentation, and reflection notes. The repository serves as both a working project space and a portfolio artifact at semester's end.

6 Results

6.1 Course Outcomes

We report course outcomes from 16 students across 2 semesters derived from two sources. First, completion rates, post-build modification, and project costs were collected through grading records for both semesters (n=16). Second, Git commit activity was analyzed only for the most recent semester (n=8) due to repository access constraints.

6.1.1 Grading Records

94% of students (15/16) completed all five project phases, including public performance and NIME paper submission. One student withdrew during the instrument-building phase (Week 7–9). We report completion as a feasibility indicator: the full-cycle structure is completable within a single semester, rather than as evidence of pedagogical effectiveness. Students who find the workload incompatible with their goals typically drop during the add/drop period; the completion rate among students who remain enrolled may reflect this self-selection rather than course design. We lack baseline data on completion rates for comparable courses at the institution that would enable meaningful comparison.

Among the 15 completed instruments, students employed diverse input modalities: pressure/force sensing (5 projects, 33%), gestural/accelerometer input (4 projects, 26%), capacitive touch (4 projects, 26%), proximity sensing (1 project, 7%), and hybrid approaches (1 project, 7%). See Figure 3 for examples of student projects completed in the class. The diversity of input modalities, with no single approach dominating, suggests that students leveraged starter kit components in varied ways rather than converging on a single design pattern. This may reflect the open-ended project structure or students' diverse musical backgrounds.

Regarding project costs, students submitted detailed bills of materials (BOMs) during the instrument-building proposal phase, documenting vendor part numbers, quantities, and unit costs. This requirement served dual purposes: teaching professional documentation practices for reproducibility, and enabling cost tracking. From these BOMs, the mean project cost was $113.54, and the median was $93.80. The recommended budget of approximately $100 proved realistic for most students.

Assessment is structured through phase-specific rubrics published in the course repository. Composition-project rubrics evaluate functional gesture-to-sound mapping refinements with technical justification, evidence of compositional material development through sketches, recordings, and patches, preliminary score concepts, and iterative refinement of mapping decisions across hardware, software, and musical material. Performance-project rubrics evaluate the development of a coherent performance concept, building on composition work, the integration of peer feedback with specific action items, evidence of the rehearsal process, and the delivery of polished performance-ready work. These rubrics target design and musical-expression criteria

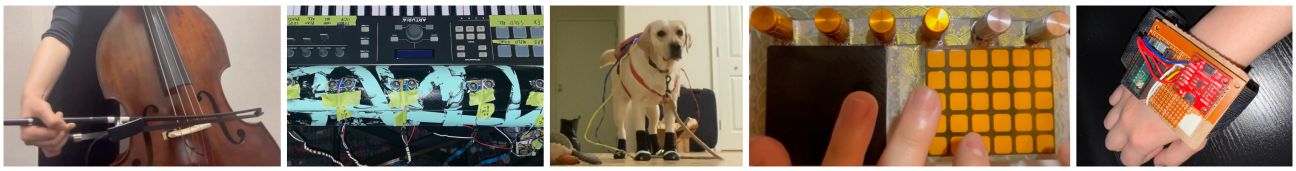

Figure 3: Student instruments from the class (2 semesters, n=16). Projects demonstrate diverse input modalities including pressure/force sensing, gestural/accelerometer control, capacitive touch, and proximity sensing. All instruments were performed publicly at the end-of-semester showcases.

rather than purely technical functionality; the 94% completion rate reported above reflects students meeting these criteria, not merely producing operational hardware.

6.1.2 Commit Activity

We report commit data for a single semester (n=8) because repository access and a consistent folder structure were available only for the most recent offering. While both semesters used GitHub for documentation, the earlier semester's repository was not accessible for analysis, and folder organization was less standardized. We therefore present commit patterns as illustrative of one offering rather than as representative of the full sample.

For the analyzed semester, students were required to organize submissions into phase-specific folders as part of the grading policy; this structure enabled the instructor to monitor progress and detect early signs of technical difficulties. We categorized commits by matching file paths against these folder patterns using regular expressions. Commits that modify files in multiple-phase folders were counted toward each relevant phase.

Total commits were 663 (mean 82.9 per student, median 88). Distribution across project phases: lab exercises (33.0%), performance (22.7%), composition (17.6%), research (14.8%), instrument building (11.9%). The relatively high proportion of performance-phase commits may indicate substantial file activity in that folder during the final weeks, though we cannot determine from commit data alone whether this reflects iteration on instrument design, performance preparation, documentation, or some combination of these.

6.2 Post-Build Hardware Modification

A key motivation behind the full-cycle design, informed by prior work on practice-based composition and instrument development [43, 47], is that extended musical engagement with an instrument, through composition and performance preparation, may reveal interface limitations that brief demonstrations do not expose. In our class, the composition phase (Weeks 10–11) and performance phase (Weeks 12–14) are intentionally intertwined: students preparing for public performance must revisit their instruments as they discover that certain gestures do not reliably produce consistent sounds, or that mappings do not yield expected musical results. This creates ongoing iteration between hardware, software, and musical material. We characterize this as hardware or code modifications to the instrument after the build phase concluded (Week 9).

47% of students who completed the build phase (7/15) modified their hardware design afterward, including circuit modifications, sensor additions, fabrication changes, or substantial code changes to mapping systems. The student who withdrew during the build phase is excluded from this calculation. Post-build modifications were identified through grading records, Git commits, and instructor observation of students' updated project documentation submitted during the composition or performance phases that differed from their Week 9 instrument-building deliverables.

This observation is consistent with the hypothesis that full-cycle pedagogy creates conditions for iteration that build-and-demo models may not provide. However, we cannot isolate which aspect of the full-cycle approach, compositional exploration, performance pressure, peer feedback, or their combination, drove specific modifications. The phases are deliberately interconnected; attributing iteration to a single cause may misrepresent the pedagogical design.

We present this finding as preliminary evidence warranting further investigation with comparison conditions.

6.3 Commercial Controller References

As supplementary evidence for the student profile established through prerequisite analysis (§3.1), we examined project proposals for references to commercial controllers.

63% of students (10/16) explicitly referenced commercial controllers, such as Ableton Push, Launchpad, and Akai products, in their proposals, either as inspiration, for comparison, or for design contrast. This figure includes all students who submitted proposals, including the one who subsequently withdrew during the build phase.

6.4 Student Feedback and Inter-Cohort Iteration

Anonymous end-of-semester surveys yielded six written responses across the two cohorts (response rate 38%). The volume is too small for systematic thematic analysis; we report it as informal triangulation. Three signals emerged from the first cohort. General reception was positive: respondents described the course as exceeding expectations and explicitly valued instructor support across heterogeneous skill levels. Workload was repeatedly salient: students described the course as demanding. One respondent identified mapping as an area where additional instructional support would have been welcome, particularly for students with weaker programming backgrounds.

This last comment directly informed the revision of the second cohort's curriculum: the Mapping and Scoring lecture (Week 10) was substantially expanded, with additional time allocated to mapping classifications, response curves, and worked examples that connect sensor input to synthesis parameters. Second-cohort feedback indicated continued positive reception and acknowledged workload demands, but did not flag mapping as a specific concern. We note that the two cohorts therefore received non-identical curricula; the outcomes reported in §6.1–§6.3 aggregate

across this difference, which the small sample size precludes us from disaggregating meaningfully.

7 Discussion

7.1 Lessons Learned

The observations below emerged from our specific institutional context. We offer them as potentially useful considerations for educators rather than as generalizable principles. Their relevance to other contexts will depend on local conditions, student populations, and available resources.

Sound design proficiency enables controller focus. Because students arrive with excellent sound design and music-making skills, instructional time can emphasize controller design, mapping strategies, and physical interface considerations rather than synthesis fundamentals. This represents a different allocation of instructional time than courses serving novice populations.

Peer discussion accelerates iteration. In-class "speed dating" activities in which students briefly paired to discuss problems and progress were highly effective. The shared GitHub repository enables code sharing and collaborative troubleshooting.

Composition vs. performance sequencing. Some students suggested reversing the composition and performance phases, arguing that performance practice should precede composition. The composition phase may need to explicitly state that students should explore their instrument through jamming before formal composition begins. Another approach to address this might be to incorporate 2 performance stages, as described in [32], the first occurring at the end of the instrument-building phase.

Stage plots and technical riders matter. Professional documentation requirements proved essential for the public showcase, where technical staff needed clear specifications. These artifacts also develop professional skills transferable to industry contexts.

Student feedback. Anonymous end-of-semester surveys had low response rates. Inter-cohort curricular changes (§6.4) introduce a confound that the small sample size precludes us from analyzing; future iterations should standardize curriculum across cohorts when reporting comparative outcomes, or formally document curricular evolution as part of the methodology.

7.2 Limitations

This work is limited by issues in commit categorization and context specificity. Commit categorization relied on file path matching against required folder structures. While this approach is objective and reproducible, it has the limitation of assuming that each commit is built equally. For example, commits that modify hardware or synthesis code during performance preparation may reflect iteration driven by compositional insight, deadline pressure, or peer feedback. Another instance is where commit data cannot capture physical hardware modifications that do not correspond to code changes.

This paper documents one pedagogical approach; whether alternative structures would yield different outcomes remains unexplored. Berklee's EPD program represents a specialized context with specific institutional affordances. The student profile is inferred from program admission requirements and prerequisite course completion rather than direct intake assessment. While this approach is standard in curriculum design, future work could incorporate diagnostic instruments to more precisely characterize incoming competencies. Although the number of target students appears to be increasing, there remains a wide range of profiles within the population that educators must consider when designing curriculum.

7.3 Resource Dependencies and Adaptation Considerations

Implementing a full-cycle prototyping course requires resources and institutional conditions that should be assessed early in the planning process. The course requires dedicated lab space with electronics workbenches and soldering equipment, a budget for starter kits and project components, and small class sizes enabling individualized feedback. Students are all expected to have prior knowledge in visual programming for audio or text-based programming, experience with commercial DAWs and controllers, and familiarity with performance culture and public showcases. The instructor must possess interdisciplinary competency spanning electronics, embedded programming, and music technology, as well as strong familiarity with NIME research and practice.

Educators lacking these conditions may consider adaptations: institutions without lab space might partner with engineering departments or makerspaces; programs without prerequisites in visual programming might extend laboratory exercises; contexts without a performance culture might substitute portfolio documentation for public showcases. The five-phase structure can be compressed or expanded depending on term length, though we have not tested alternative configurations.

7.4 Future Directions

Planned curriculum developments include formalizing fabrication instruction, continuing to expand mapping labs with machine learning tools (Wekinator, ML packages for Max), and requiring both visual programming for audio systems and introductory text-based programming as prerequisites. We also plan to establish tiered instrument-building milestones to accommodate variability in component delivery.

Most student instruments currently remain at the breadboard-and-housing stage with computer-dependent synthesis; formalizing fabrication instruction with soldering, enclosure design, and standalone instrument architecture is a planned curriculum development that would require either dedicated lab facilities or a partnership with the institution's existing makerspaces.

Future work should also examine how students' increasing familiarity with AI-assisted music tools affects their approach to NIME design: whether AI-mediated production fluency transfers productively to physical instrument design or creates expectations that require specific pedagogical intervention.

8 Conclusion

This paper documents the application of full-cycle pedagogy, integrating research, building, composition, performance, and documentation, to a specific underexamined student population: experienced users of commercial digital music tools who lack physical prototyping skills. Our preliminary observation that 47% of completing students modified hardware after the build phase concluded is consistent with the hypothesis that full-cycle pedagogy, where building, composing, and performing are interconnected, creates conditions for technical iteration.

In response to our research question, we identify several structural features that address this population's specific needs. Front-loading prototyping scaffolding through constrained-toolchain labs addresses the primary skill gap, while students' existing production fluency allows instructional time to shift toward controller design and physical interface considerations rather than synthesis fundamentals. Concentrating explicit design-theory instruction at the composition phase, where students directly confront mapping decisions, proved more effective for this cohort than introducing it abstractly early in the semester: a finding reinforced by inter-cohort curricular iteration (§6.4). Requiring composition and public performance created conditions under which approximately half of the students continued to modify their instruments after the nominal build phase, suggesting that extended musical engagement with custom instruments fosters iteration that demonstration-oriented structures may not provide.

More broadly, we suggest that the student profile documented here may become increasingly common in some music technology education contexts as commercial tools and AI-assisted systems continue to expand accessibility at the secondary level. For educators who encounter students with this profile, the pedagogical challenge shifts from motivating musical engagement to bridging digital fluency and physical prototyping, a different problem than that faced with novice or engineering populations.

We encourage NIME educators to consider how trends in secondary music technology adoption may shift the populations they encounter and to share their approaches, including adaptations to varying resource constraints, so that the community can develop a richer understanding of effective pedagogy across diverse institutional contexts.

9 Ethical Standards

This study reports on educational practices conducted in an established undergraduate course. The research involved analysis of pre-existing, de-identified course records and student artifacts produced as part of normal instructional activities. No new data were collected for research purposes, and no interaction with students occurred for research. Students were informed at the start and end of the course that de-identified, aggregated outcomes might be used for research publication, and each student individually provided verbal consent. The dataset was fully de-identified; no names, student IDs, or email addresses were accessed or retained at any stage of analysis. This project was reviewed by the Berklee College of Music Institutional Review Board, which determined that it involved no more than minimal risk to participants. This determination is maintained in IRB records.

Acknowledgments

Thanks to the students of *Prototyping Electronic Digital Instruments* for their creative work and feedback on course design. Thanks to the Electronic Production and Design Department at Berklee College of Music for supporting this curriculum.

References

[1] Arduino. n.d.. Arduino Software (IDE). https://www.arduino.cc/en/software/ Integrated development environment and tools for programming Arduino boards. Accessed: 2026-04-27.

[2] Bela Platform. n.d.. Bela. https://bela.io/ Open-source embedded platform for real-time audio and sensor processing. Accessed: 2026-04-27.

[3] Edgar Berdahl and Stephen D Beck. 2016. Electroacoustics laboratory assignments for computer music students. *Journal of the Acoustical Society of America* 139, 4_Supplement (2016), 2097–2097.

[4] Berklee College of Music. n.d.. Audio Programming in SuperCollider (MTEC-347). https://college.berklee.edu/courses/mtec-347 Accessed: 2026-04-27.

[5] Berklee College of Music. n.d.. Bachelor of Music in Electronic Production and Design. https://college.berklee.edu/electronic-production-and-design/bachelor-of-music-in-electronic-production-and-design Degree program description. Accessed: 2026-04-27.

[6] Berklee College of Music. n.d.. Control Systems in Electronic Production (EP-225). https://college.berklee.edu/courses/ep-225 Accessed: 2026-04-27.

[7] Berklee College of Music. n.d.. Csound: Sound and Plugin Design (EP-337). https://college.berklee.edu/courses/ep-337 Accessed: 2026-04-27.

[8] Berklee College of Music. n.d.. Electronic Digital Instrument Principal. https://college.berklee.edu/electronic-production-design/electronic-digital-instrument-principal Program description, Electronic Production and Design Department. Accessed: 2026-04-27.

[9] Berklee College of Music. n.d.. Introduction to Computer Programming (LMSC-261). https://college.berklee.edu/courses/lmsc-261 Accessed: 2026-04-27.

[10] Berklee College of Music. n.d.. Introduction to Music Technology (MTEC-111). https://college.berklee.edu/courses/mtec-111 Accessed: 2026-04-27.

[11] Berklee College of Music. n.d.. Modular Functions and Signal Flow (EP-223). https://college.berklee.edu/courses/ep-223 Accessed: 2026-04-27.

[12] Berklee College of Music. n.d.. Programming in Max (EP-341). https://college.berklee.edu/courses/ep-341 Accessed: 2026-04-27.

[13] Berklee College of Music. n.d.. Prototyping Electronic Digital Instruments (EP-361). https://college.berklee.edu/courses/ep-361 Accessed: 2026-04-27.

[14] Berklee College of Music. n.d.. Studio Technologies (EP-220). https://college.berklee.edu/courses/ep-220 Accessed: 2026-04-27.

[15] Tina Blaine and Sidney S. Fels. 2003. Contexts of Collaborative Musical Experiences. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (22-24 May, 2003). Montreal, Canada, 129–134. https://doi.org/10.5281/zenodo.1176490

[16] Renaud Bougueng Tchemeube, Jeffrey Ens, Cale Plut, Philippe Pasquier, Maryam Safi, Yvan Grabit, and Jean-Baptiste Rolland. 2023. Evaluating Human-AI Interaction via Usability, User Experience and Acceptance Measures for MMM-C: A Creative AI System for Music Composition. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, Edith Elkind (Ed.). International Joint Conferences on Artificial Intelligence Organization, 5769–5778. https://doi.org/10.24963/ijcai.2023/640 AI and Arts.

[17] Stephen Brade, Teng Ma, Lancelot Blanchard, Kimaya Lecamwasam, Carlos Mariano Salcedo, Suwan Kim, Perry Naseck, Andrew Li, Matthew R Michalek, Sebastian Franjou, and Anna Huang. 2026. Agents in Concert: A Case-Study of Bringing AI to the Stage in Practice. In *Proceedings of the 31st International Conference on Intelligent User Interfaces (IUI '26)*. Association for Computing Machinery, New York, NY, USA, 1340–1361. https://doi.org/10.1145/3742413.3789104

[18] Jennifer Butler. 2008. Creating Pedagogical Etudes for Interactive Instruments. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Genoa, Italy, 77–80. https://doi.org/10.5281/zenodo.1179503

[19] Csound Community. n.d.. Csound. https://csound.com/ Open-source sound and music computing system for synthesis and composition. Accessed: 2026-04-27.

[20] Cycling '74. n.d.. Max. https://cycling74.com/products/max Software product page. Accessed: 2026-04-27.

[21] Kirsty Devaney. 2019. 'Waiting for the wow factor': Perspectives on computer technology in classroom composing. *Journal of Music, Technology & Education* 12, 2 (2019), 121–139.

[22] David Dockan, David Knapp, Matthew Clauhs, and Bryan Powell. 2023. An exploratory study on the impact of the COVID-19 pandemic on music teacher engagement with Soundtrap. *Journal of Music, Technology & Education* 16, 1-2 (2023), 25–40.

[23] Ecma International. 2025. ECMA-262: ECMAScript Language Specification. https://ecma-international.org/publications-and-standards/standards/ecma-262/ 16th edition, June 2025. Accessed: 2026-04-27.

[24] Espressif Systems. n.d.. ESP32 Series: Wi-Fi & Bluetooth System-on-Chip (SoC). https://www.espressif.com/en/products/socs/esp32 Technical specification and product overview. Accessed: 2026-04-27.

[25] Sidney Fels and Michael Lyons. 2011. Advances in new interfaces for musical expression. In *ACM SIGGRAPH 2011 Courses*. 1–169.

[26] Emma Frid. 2018. Accessible Digital Musical Instruments - a Survey of Inclusive Instruments Presented at the NIME, SMC and ICMC Conferences. In *International Computer Music Conference*. https://quod.lib.umich.edu/i/icmc/bbp2372.2018.011/1

[27] Fritzing Project. n.d.. Fritzing. https://fritzing.org/ Open-source electronics design and prototyping platform. Accessed: 2026-04-27.

[28] Jiffer Harriman. 2015. Pd Poems and Teaching Tools. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 331–334. https://doi.org/10.5281/zenodo.1179074

[29] Andy D. Hunt, Marcelo M. Wanderley, and Matthew Paradis. 2002. The importance of Parameter Mapping in Electronic Instrument Design. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (24-26 May, 2002). Dublin, Ireland, 88–93. https://doi.org/10.5281/zenodo.1176424

[30] Alexander Refsum Jensenius. 2013. An action–sound approach to teaching interactive music. *Organised Sound* 18, 2 (2013), 178–189.

[31] Marko Jeremic and Akito van Troyer. 2026. e-baton: Recognising Conducting Gestures with Machine Learning. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. London, UK.

[32] Sergi Jordà and Sebastian Mealla. 2014. A Methodological Framework for Teaching, Evaluating and Informing NIME Design with a Focus on Mapping and Expressiveness. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, London, United Kingdom, 233–238. https://doi.org/10.5281/zenodo.1178824

[33] Rebecca Kleinberger. 2018. Vocal Musical Expression with a Tactile Resonating Device and its Psychophysiological Effects. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Thomas Martin Luke Dahl, Douglas Bowman (Ed.). Virginia Tech, Blacksburg, Virginia, USA, 92–95. https://doi.org/10.5281/zenodo.1302693

[34] Rebecca Kleinberger, Gershon Dublon, Joseph A. Paradiso, and Tod Machover. 2015. PHOX Ears: A Parabolic, Head-mounted, Orientable, eXtrasensory Listening Device. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Edgar Berdahl and Jesse Allison (Eds.). Louisiana State University, Baton Rouge, Louisiana, USA, 30–31. https://doi.org/10.5281/zenodo.1179106

[35] Rebecca Kleinberger, Alexandra Rieger, Janelle Sands, and Janet Baker. 2019. Supporting Elder Connectedness through Cognitively Sustainable Design Interactions with the Memory Music Box. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 355–369. https://doi.org/10.1145/3332165.3347877

[36] Rébecca Kleinberger, Nikhil Singh, Xiao Xiao, and Akito van Troyer. 2022. Voice at NIME: a Taxonomy of New Interfaces for Vocal Musical Expression. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Andrew McPherson and Emma Frid (Eds.). Auckland, New Zealand, Article 8. https://doi.org/10.21428/92fbeb44.4308fb94

[37] Rebecca Kleinberger and Akito van Troyer. 2016. Dooremi: a Doorway to Music. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Queensland Conservatorium Griffith University, Brisbane, Australia, 160–161. https://doi.org/10.5281/zenodo.1176052

[38] Shigeru Kobayashi, Takanori Endo, Katsuhiko Harada, and Shosei Oishi. 2006. GAINER: A Reconfigurable I/O Module and Software Libraries for Education. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Paris, France, 346–351. https://doi.org/10.5281/zenodo.1176945

[39] Chi Kai Lam. 2024. Technology-enhanced creativity in K-12 music education: A scoping review. *International Journal of Music Education* 42, 4 (2024), 691–703.

[40] Martin S Lawless, Melody Baglione, and George W Sidebotham. 2016. Developing and teaching an interdisciplinary musical instrument design course at the Cooper Union. *Journal of the Acoustical Society of America* 139, 4_Supplement (2016), 2096–2096.

[41] Paul D. Lehrman and Todd M. Ryan. 2005. Bridging the Gap Between Art and Science Education Through Teaching Electronic Musical Instrument Design. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Vancouver, BC, Canada, 136–139. https://doi.org/10.5281/zenodo.1176768

[42] Sasha Leitman. 2017. Current Iteration of a Course on Physical Interaction Design for Music. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Aalborg University Copenhagen, Copenhagen, Denmark, 127–132. https://doi.org/10.5281/zenodo.1176197

[43] Thor Magnusson. 2010. Designing constraints: Composing and performing with digital musical systems. *Computer music journal* 34, 4 (2010), 62–73.

[44] Andrew McPherson. 2012. Techniques and Circuits for Electromagnetic Instrument Actuation. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan, Ann Arbor, Michigan. https://doi.org/10.5281/zenodo.1180533

[45] Lance D Nielsen. 2013. Developing musical creativity: Student and teacher perceptions of a high school music technology curriculum. *Update: Applications of Research in Music Education* 31, 2 (2013), 54–62.

[46] Vesa Norilo and Alejandro Olarte. 2020. A Visual Programming Interface for Digital Luthiery: Implementing Circuits with Veneer. *Computer Music Journal* 44, 4 (2020), 8–25.

[47] Dan Overholt. 2009. The musical interface technology design space. *Organised Sound* 14, 2 (2009), 217–226.

[48] Margarida Pessoa, Cláudio Parauta, Pedro Luís, Isabela Corintha, and Gilberto Bernardes. 2020. Examining Temporal Trends and Design Goals of Digital Music Instruments for Education in NIME: A Proposed Taxonomy. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Romain Michon and Franziska Schroeder (Eds.). Birmingham City University, Birmingham, UK, 591–595. https://doi.org/10.5281/zenodo.4813210

[49] Adrian Pitts and Robert Mawuena Kwami. 2002. Raising students' performance in music composition through the use of information and communications technology (ICT): a survey of secondary schools in England. *British Journal of Music Education* 19, 1 (2002), 61–71.

[50] PJRC. 2019. Teensy 4.0 Development Board. https://www.pjrc.com/store/teensy40.html ARM Cortex-M7 microcontroller board (600 MHz). Accessed: 2026-04-27.

[51] Nina Marlene Prasso. 1997. *An examination of the effect of writing melodies, using a computer-based song-writing program, on high school students' individual learning of sight-singing skills.* Teachers College, Columbia University.

[52] Processing Foundation. n.d.. Processing. https://processing.org/ Open-source creative coding project. Accessed: 2026-04-27.

[53] Processing Foundation. n.d.. Python Mode for Processing (Processing.py). https://py.processing.org/ Documentation for Python mode extension to Processing. Accessed: 2026-04-27.

[54] Python Software Foundation. n.d.. Python Programming Language. https://www.python.org/ Official website. Accessed: 2026-04-27.

[55] Timothy Roth, Aiyun Huang, and Tyler Cunningham. 2021. On Parallel Performance Practices: Some Observations on Personalizing DMIs as Percussionists. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Shanghai, China, Article 76. https://doi.org/10.21428/92fbeb44.c61b9546

[56] Jonathan Savage. 2005. Working towards a theory for music technologies in the classroom: how pupils engage with and organise sounds with new technologies. *British Journal of Music Education* 22, 2 (2005), 167–180.

[57] Zefan Sramek, Arissa J. Sato, Zhongyi Zhou, Simo Hosio, and Koji Yatani. 2023. SoundTraveller: Exploring Abstraction and Entanglement in Timbre Creation Interfaces for Synthesizers. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (Pittsburgh, PA, USA) *(DIS '23)*. Association for Computing Machinery, New York, NY, USA, 95–114. https://doi.org/10.1145/3563657.3596089

[58] Dan Stowell. 2010. *Making music through real-time voice timbre analysis: machine learning and timbral control.* Ph. D. Dissertation. School of Electronic Engineering and Computer Science, Queen Mary University of London. https://qmro.qmul.ac.uk/xmlui/handle/123456789/412

[59] SuperCollider Community. n.d.. SuperCollider. https://supercollider.github.io/ Open-source environment and programming language for real-time audio synthesis and algorithmic composition. Accessed: 2026-04-27.

[60] Pierre-Valery Njenji Tchetgen. 2024. Can multimodal rhythmic interaction impact the literacy and socio-emotional development of children: the case of the African Talking Drums. In *Proceedings of the 19th International Audio Mostly Conference: Explorations in Sonic Cultures* (Milan, Italy) *(AM '24)*. Association for Computing Machinery, New York, NY, USA, 116–129. https://doi.org/10.1145/3678299.3678311

[61] Nao Tokui. 2020. Towards democratizing music production with AI-Design of Variational Autoencoder-based Rhythm Generator as a DAW plugin. arXiv:2004.01525 [eess.AS] https://arxiv.org/abs/2004.01525

[62] Enrique Tomás. 2020. A Playful Approach to Teaching NIME: Pedagogical Methods from a Practice-Based Perspective. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Romain Michon and Franziska Schroeder (Eds.). Birmingham City University, Birmingham, UK, 143–148. https://doi.org/10.5281/zenodo.4813280

[63] Akito van Troyer. 2012. DrumTop: Playing with Everyday Objects. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan, Ann Arbor, Michigan. https://doi.org/10.5281/zenodo.1178441

[64] Akito van Troyer. 2012. *Hyperaudience: designing performance systems for audience inclusion.* Master's thesis. Massachusetts Institute of Technology. https://dspace.mit.edu/handle/1721.1/77817

[65] Akito van Troyer. 2014. Composing Embodied Sonic Play Experiences: Towards Acoustic Feedback Ecology. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, London, United Kingdom, 118–121. https://doi.org/10.5281/zenodo.1178961

[66] Akito van Troyer. 2014. *Repertoire Remix in the Context of Festival City.* Springer International Publishing, 51–63. https://doi.org/10.1007/978-3-319-11152-0_3

[67] Akito van Troyer. 2017. MM-RT: A Tabletop Musical Instrument for Musical Wonderers. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Aalborg University Copenhagen, Copenhagen, Denmark, 186–191. https://doi.org/10.5281/zenodo.1176215

[68] Akito van Troyer and Rebecca Kleinberger. 2019. From Mondrian to Modular Synth: Rendering NIME using Generative Adversarial Networks. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Marcelo Queiroz and Anna Xambó Sedó (Eds.). UFRGS, Porto Alegre, Brazil, 272–277. https://doi.org/10.5281/zenodo.3672956

[69] Bill Verplank, Craig Sapp, and Max Mathews. 2001. A Course on Controllers. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (1-2 April, 2001). Seattle, WA, 7–10. https://doi.org/10.5281/zenodo.1176380

[70] Szymon Walendowski and Akito van Troyer. 2026. SonoCube: A Handheld Motion-Responsive Sound Object. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. London, UK.