Supporting Computational Music Remixing with a Co-Creative Learning Companion

Erin J.K. Truesdell[1], Jason Brent Smith[2], Sarah Mathew[1],

Gloria Ashiya Katuka[4], Amanda Griffith[4], Tom McKlin[3],

Brian Magerko[1], Jason Freeman[2], Kristy Elizabeth Boyer[4]

[1]Expressive Machinery Lab, Georgia Institute of Technology, Atlanta, GA 30308, USA
[2]Center For Music Technology, Georgia Institute of Technology, Atlanta, GA 30308, USA
[3]The Findings Group, Decatur, GA 30030, USA
[4]Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611, USA

{erinjktruesdell, jsmith775, smathew64}@gatech.edu

Abstract

Intelligent learning environments have demonstrated effectiveness for providing individualized instruction to students of computer science (CS). However, the great potential of intelligent agents has not yet been explored within expressive environments, which are increasingly common for supporting and motivating K-12 students. This paper presents the prototype design and implementation of a novel Co-creative Artificial Intelligence (CAI) integrated within EarSketch, an online environment for learning introductory computing concepts through code-driven, sample-based music remixing. CAI is intended to scaffold student learning from EarSketch's expressive computing curriculum by co-creating algorithmic music alongside a human learner. This paper presents an initial version of CAI, which engages with EarSketch users by offering menu-based dialogue and suggestions based on the state of a project. We report a pilot study in classrooms, showing promising results in students' satisfaction with the system's capabilities. The findings of this pilot study suggest the ability of a co-creative agent to support users in learning and creative objectives, and should inspire research into combined computational and creative user models.

1 Introduction

Many computer science education environments for K-12 students focus on creative expression as a means of increasing student engagement in programming (Resnick et al. 2009; Grover, Basu, and Schank 2018). This shift raises the unique challenge of offering adaptive support to learners working on open-ended creative tasks. To achieve educational objectives within creative platforms, the field needs to move toward intelligent learning support within creative environments.

Recent years have seen computational creativity researc oriented towards systems that can collaborate with humans (Davis et al. 2019; Cheatley et al. 2020; Zacharakis et al. 2021; Guzdial et al. 2019).

Systems such as these are typically designed to enhance the user's experience and produce valuable artifacts, but have not been designed to support learning. EarSketch, a platform that uses the expressive medium of sample-based music creation to engage students with programming (Magerko et al. 2016), provides an ideal environment to investigate a co-creative agent for education.

EarSketch is an expressive programming environment in which students create sample-based music (primarily in the hip-hop and electronic genres). The EarSketch interface is shown in Figure 1. EarSketch users write Python or JavaScript code to place professionally-produced sound samples and audio effects on a multi-track timeline. The accompanying curriculum covers topics relevant to programming and algorithmic music production.

EarSketch is designed for students with little to no prior experience in making music. Consequently, music theory and reading skills are not required for learners to create songs on the platform. Previous EarSketch studies have indicated strong gains in student attitudes and intent to persist in computing, especially in populations that are underrepresented in the field of computer science (Magerko et al. 2016; Engelman et al. 2017; McKlin et al. 2018).

CAI (Co-creative Artificial Intelligence) is a learning companion aimed at scaffolding learning by co-creating algorithmic music alongside a human student. Our aim with the development of CAI is to create an agent that supports the student the way a slightly more knowledgeable peer would. The current version of CAI offers this support by suggesting additions to students' code and music, and through additional support of students' sound selection and debugging activities in EarSketch. As students interact with CAI, the agent offers suggestions to further their creative goals for their EarSketch projects while promoting the use of more advanced coding and musical techniques. The agent supports beginner EarSketch users as well as those who have experience with the platform. CAI interacts with the EarSketch user in a non-intrusive manner, providing chat-based suggestions and assistance as students work on their

Proceedings of the 12th International Conference on Computational Creativity (ICCC '21)
ISBN: 978-989-54160-3-5

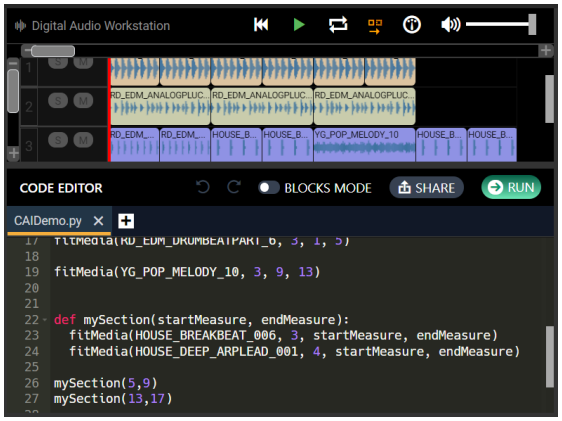

Figure 1: The EarSketch application's code editor (bottom) and digital audio workstation (DAW) (top).

EarSketch projects. It is our goal that students who work with CAI demonstrate growth in core computing and musical learning objectives, and indicate in self-reported measures that CAI provides valuable input as they develop both aspects of their EarSketch projects.

EarSketch users select dialogue options from a menu-based interface to interact with CAI. Rather than adding code or sounds into a project, CAI offers suggestions via chat-based dialogue. This gives greater control of the project to the student and encourages them to develop projects themselves rather than relying on the agent to do so. CAI's suggestions are made using analysis tools we developed to inform the agent about the code complexity and musical structure of the project (Smith et al. 2020). CAI presents suggestions related to advancing the project's code and music, and aids students in sound sample selection and debugging.

We deployed a version of CAI within a pilot study in high school classrooms during the winter of 2021. Students interacted with the system, and we collected data on their interactions as well as survey data about their perceptions of CAI. The results of the survey indicate that students found CAI's suggestions valuable and that students who interacted with more of the CAI system perceived CAI as having helped them make a better program and a better song. The remainder of this paper discusses a survey of related work; a description of CAI's project analysis capabilities, co-creative move generation, and dialogue tree; results of a pilot study of CAI; and future areas for development and study. The results of this work suggest that co-creative AI is a promising means of applying computational co-creativity to support learners.

2 Related Work

Recent work in co-creative AI and learning companions demonstrate ongoing interest in the development of both computational creative partners as well as agents to support learning across a variety of subject areas. Co-creative systems in the domains of drawing, design, and music display multiple levels of initiative (Davis et al. 2019; Cheatley et al. 2020; Zacharakis et al. 2021; Guzdial et al. 2019). Their authors distinguish which audiences may find which levels of initiative most useful in the creative process. While intelligent learning environments have been developed for creative domains such as music composition and performance, it is not typical for these systems to be co-creative or for a co-creative system to be a pedagogical tool. CAI aims to bridge this gap by presenting pedagogy in an explicitly co-creative context, engaging EarSketch students to build competency in both musical composition and programming. This aligns with EarSketch's goal to build confidence and interest in computing by offering an authentic expressive computing environment (Engelman et al. 2017).

2.1 Human-Computer Co-Creativity

Recent publications in computational creativity have described systems that work with humans on artistic pursuits including sketching (Davis et al. 2019), songwriting (Cheatley et al. 2020), music harmonization (Zacharakis et al. 2021), and game level design (Guzdial et al. 2019). Some of these systems, including the ALYSIA songwriting system (Cheatley et al. 2020) and the CHAMELEON harmonization tool (Zacharakis et al. 2021), were especially useful for supporting domain novices through the creative process. This dynamic is of interest for the development of CAI, which is intended to support students as they develop mastery in the EarSketch environment. Systems aimed at a broader range of mastery levels, such as Morai Maker (Guzdial et al. 2019), offer unexpected ideas and make moves in line with users' demonstrated design style. Results of these projects have indicated that more advanced users value co-creative systems that serve in an adaptive "follower" role, while novice users typically benefit from systems that provide support using levels of knowledge which the users themselves do not have. In musical applications, co-creative artificial intelligence agents are widely used in algorithmic music composition (Lopez-Rincon, Starostenko, and Ayala-San Martín 2018). In live settings, AIs can function as "tools" for musicians or as "actors" that perform collaboratively with them (Caramiaux and Donnarumma 2020). CAI reflects these paradigms of human-AI musical collaboration by presenting users with suggestions for compositional tools such as sound samples and by evaluating a project as a student iterates on it in order to provide collaborative feedback.

Co-creative AI projects typically aim to create a satisfying interaction for the user and/or a valuable end product. Algorithm selection, interaction design, and initiative mechanisms are selected to maximize systems' performance towards these goals. CAI extends the work in this domain: in addition to providing an enjoyable user experience and supporting the collaborative creation of valuable artifacts, CAI includes pedagogical strategies as part of the interaction between system and user.

2.2 Intelligent Learning Environments

The field of intelligent tutoring systems (ITS) was inspired by the goal of providing individualized instruction to learners (Sleeman and Brown 1982). From that line of research, *learning companions* arose: their goal is to support students

on their learning trajectories while leveraging the benefits of social context (Chan and Baskin 1990). Both ITS and learning companions provide personalized instruction and feedback for students. An ITS does so in a tutor or authority figure manner (Ma et al. 2014), while learning companions are more often modeled as peers. The architecture of these systems often includes four components: domain expertise, a model of the student, a model of a tutor or learning companion, and an interface. The student model considers how users learn and make mistakes, and the tutor model contains intervention strategies to use. This brief literature review discusses ITS and learning companion systems that support computer science and music, as the present work lies at the intersection of these two domains. All systems discussed here are designed for novice learners in that domain, usually for an introductory course on the subject.

Computer Science Intelligent Learning Environments Intelligent learning environments for computer science generally support learning two kinds of knowledge: a particular programming language's syntax or general computer science concepts. One of the earliest ITSs for programming is the LISP tutor (Anderson and Skwarecki 1986), which offers feedback and questions for the student specific to that programming language. Since then, many ITSs for CS have been built and investigated (Crow, Luxton-Reilly, and Wuensche 2018; Nesbit et al. 2014). CS has also seen learning companion research: learning companions have been built that support online education for Scratch (Ocaña et al. 2020) and Java (Faraco, Rosatelli, and Gauthier 2004), a robot offers feedback for tasks in the LEGO Mindstorms environment (Ahmed, Lubold, and Walker 2018), and one companion both teaches and learns about algorithms (Petry and Rosatelli 2006). These systems are task-based and do not afford open-ended or artistic coding projects. Like these systems, CAI supports learners in coding in a specific programming language (Python or JavaScript), but unlike prior systems, CAI is designed to act as a co-creative companion, supporting both aesthetic and technical decisions.

Intelligent Learning Environments for Music Intelligent learning systems for music have been developed to focus on topics such as music theory, harmony, or playing instruments. Computer-assisted musical instrument tutoring systems (Percival, Wang, and Tzanetakis 2007) come in two varieties: specific-goal oriented projects (such as learning chords), and general-instruction systems. Piano Tutor (Dannenberg et al. 1990) and pianoFORTE (Smoliar, Waterworth, and Kellock 1995) are two systems that specialize in teaching piano to beginner students. Piano Tutor's expertise is in reading sheet music. The system interjects if the student makes mistakes while practicing a piece. pianoFORTE's focus is more advanced. It visualizes how keys are pressed, or more generally, how a piano should be performed.

More recent social learning systems have been companions whose expertise is on how instruments should be physically played. Pianobot (Ritschel, Seiderer, and Andre 2020) and instruMentor (Bagga et al. 2019) are robotic tutors that react and offer advice on piano musical performance and recorder physical performance, respectively. XR (extended reality) systems can track how students play the piano by overlaying virtual notes and feedback over keys. These include piARno (Rigby, Wünsche, and Shaw 2020), and Mixed Reality Piano Tutor (Molloy, Huang, and Wünsche 2019). The MRLS learning companion system teaches users about rhythm and allows users to create music collaboratively with one another (Wang and Lai 2011). While these systems can detect how instruments are physically played, and offer lesson plans, they do not support introducing these concepts for student-created music projects.

CAI's scaffolding and student modeling processes are guided by practices from ITS design: the system evaluates coding concepts such as loops and modularity, and music concepts such as repetition and contrast. From other learning companions, we integrate social learning aspects and regular feedback for the changes students make to their projects. CAI incorporates these principles into its co-creative actions. By integrating modeling of students' code and music within a co-creative system, CAI enables novice and experienced students to build domain knowledge while providing them with tools to create personally meaningful and culturally relevant artifacts (Magerko et al. 2016).

3 System Description

3.1 Analysis Module

CAI includes an Analysis Module that comprises a suite of code and music analysis tools. The Analysis Module provides CAI with snapshots of a student's project that update while the student works and are used to help CAI select suggestions to make. These snapshots provide the system with scores describing the complexity of the student's code, as well as a model of the musical structures and sound characteristics in the music produced by the student (Smith et al. 2020).

Code Complexity CAI's Code Complexity Calculator evaluates the levels at which students use 15 concepts covered in the EarSketch curriculum. The concepts are: *integers, floating-point numbers, strings, Boolean values, lists, variables, mathematical operators, string operators, list operators, comparisons, Boolean logic, user-defined functions, console input, for loops,* and *conditional statements.* Other works have adapted Bloom's Taxonomy for the purposes of evaluating students in computer science (Starr, Manaris, and Stalvey 2008; Thompson et al. 2008). Similarly, CAI uses a knowledge taxonomy that draws from Krathwohl and Anderson's Flattened Bloom's Taxonomy (2009) and is tailored specifically to EarSketch learning targets. Our use of this taxonomy as a basis allows the system to model demonstrated understanding of each curriculum topic using distinct hierarchical levels.

CAI's taxonomy enumerates 3-5 levels of complexity for each listed curriculum topic. The first complexity level for any concept is usage of it in the student's code. The second level is usage that is not identical to sample code from the EarSketch curriculum. Subsequent levels for a concept are specific to each and outline increasingly complex uses. Table 1 describes the complexity levels for the "String" concept as an example. When the student runs a project, the

Complexity Calculator generates an object that includes representations of the student's demonstrated complexity for each topic. The system also compares this output with the complexity object from the last time the student ran their code. The agent uses this information to select appropriate suggestions to present.

Level	Description
0	Does not use a string
1	Uses a string copied directly from sample code
2	Uses a string not copied directly from sample code
3	Uses a string not copied from sample code as a function argument or in a comparison
4	Iterates over or indexes from a string not copied from sample code in a for loop, function argument, comparison, etc.

Table 1: Knowledge levels for the "String" concept.

Music Analysis The CAI Analysis Module also includes tools to analyze the musical output of a student's EarSketch project (Smith et al. 2020). Students use EarSketch to create algorithmic music by selecting a combination of sounds and audio effects. CAI's analysis tools combine representations of a student's code and musical choices to represent the project as a whole. We have designed CAI to include discussion of the creative elements of EarSketch, which have led to increased positive attitudes towards computing, especially in underrepresented populations (Magerko et al. 2016). As such, the music analysis is used to gather information about students' preferences and musical choices so that CAI can make suggestions, rather than measuring student progress in discrete levels as with the code complexity rubric.

The CAI Analysis Module creates a hierarchical representation of a student's project in order to represent and understand its musical form or structure. Music in an EarSketch project is represented symbolically instead of as raw audio. As a script runs, sounds and effects used in the project are listed by track number to be displayed on the Digital Audio Workstation (DAW) view. The Analysis Module converts this symbolic representation from a track-based listing to a timeline view. It compares the timeline to a series of measure-to-measure distance thresholds to identify transitions between sections (large changes in instrumentation or effects that divide a song into distinct parts) and subsections (smaller changes that indicate transitional phases of a musical section). The representation of musical form created by the Analysis Module is stored in a dictionary called the Sound Profile, in which each section contains a nested structure of subsections. Every section and subsection includes the sounds and effects used at each measure within it, the parameter values for those effects, and which lines of code were used to add them to the output.

In order to identify how a student is using coding concepts to enact specific musical ideas, CAI must identify items in a student's code and music that are used in combination. To do this, the Analysis Module allows CAI to index the Sound Profile by specifying an input and output type from the choices of *section, measure, line, sound,* and *effect*, along with an input value. The Sound Profile indexing function returns the corresponding values for anything of the specified output type that coincides with the input. For example, selecting *input:measure, output:sound, inputValue:14* will return all sounds that play during measure 14 of the project. Selecting *input:section, output:line, inputValue:"A"* will return all lines of code that manipulate the sounds and effects in section *A*. This system allows CAI to observe music and code in tandem, in order to suggest actionable code changes that affect the musical output and further student understanding of both domains.

3.2 Co-Creative Moves

We observed the conversational flow between student collaborators to discover the dialogue patterns that emerged and to gain insight about creative collaboration in EarSketch to inform the development of CAI. We performed a pilot study in which we logged student-to-student communication through a chat application within EarSketch as they completed a co-creative learning task (Griffith et al. 2021).

Based upon our observations, and on our intention to make CAI an agent that promotes learning in the computational domain, we selected three primary co-creative moves as the initial set available for CAI's interaction with students: code and music concept suggestions, sound sample recommendations, and debugging assistance. These moves were selected because they represent core parts of the experience in creating EarSketch projects: selecting samples for use in a piece of music, using code to place and manipulate those samples, and debugging code when errors arise. Furthermore, we identified these moves as key areas in which CAI aims to support students.

Other co-creative moves and dialogue tags that were found in the student-to-student studies were socially-oriented. While these are important in rapport-building and human conversation, our priority for the first version of CAI was to include the pedagogical tools necessary for the system to function. Sample selection and code suggestions work in tandem to scaffold students' learning in both areas represented in the EarSketch platform. Debugging assistance from CAI provides context for errors and is intended to reduce student frustration. We anticipate developing additional moves for CAI in future iterations of the system.

Sound Sample Selection CAI's ability to suggest sounds based on a student's project supports student expression in a primary aspect of EarSketch project creation. Messages between students from our study of student-to-student co-creative dialogue included a number of exchanges where users discussed the aesthetic properties of their projects. Suggestions for changes to a project's samples were also found in observation of collaborative live coding using EarSketch (Xambó et al. 2018). Our inclusion of sample-focused interactions supports CAI in providing an experience similar to that of collaborating with a partner, and seeks to enhance students' experience in the expressive environment that EarSketch provides.

Additionally, CAI's recommendation system can help students navigate the EarSketch sample library. The library includes nearly 4000 professionally-produced sounds spanning multiple popular genres of music. It includes samples from sound engineer Young Guru and EDM artist Richard Devine, as well as samples from songs by popular artists such as Ciara, Common, and Pharrell Williams. While such size benefits EarSketch users by providing a variety of sound options, the size of the library can become overwhelming to users (Smith et al. b). CAI aims to encourage students to experiment with new and potentially unfamiliar or unexpected sounds in their projects.

CAI uses its Sound Profile (see section 3.1) to create sound recommendations for specific sections of a song through an adaptation of EarSketch's existing hybrid recommendation system (Smith et al. a). CAI's sample recommendation nodes present 1-3 samples from the recommendation engine. When these dialogue nodes are presented to the student, recommendations are generated based upon the Sound Profile created by the Analysis Module from the most recently run version of the student's project (and whether or not the student requested recommendations for a particular section). The sample recommendations are included in CAI's message to the student.

Code And Music Concept Suggestions A primary goal of CAI is the development of students' programming skills. One of the chief capabilities of the current version of CAI is the proposal of additions to students' projects. The CAI agent includes a mechanism for selecting and presenting one of several dozen authored suggestions based upon the state of the user's project. Included in the suggestion selection mechanism is a decision tree for determining which suggestion is presented, based on characteristics of the project and on what, if anything, has changed in the project's code complexity since the student last ran their code.

Several suggestion options are targeted specifically towards changes in the student's code complexity score. For example, if the student's complexity score for User-Defined Functions goes from 0, "Does Not Use," to 2, "Uses Uniquely," between two snapshots, CAI will prompt the student to call the function they have just made. Many suggestions include additional messages for explanations and example code, which may be accessed by the student through menu options after the suggestion has been made.

Debugging Assistance We included debugging assistance as a core capability for CAI. While CAI offers more help with error debugging than a student partner might, we elected to include this functionality to reduce student frustration and provide additional support as users work through problems common to beginner programmers. The system stores a dictionary of all possible error types in EarSketch and a corresponding explanation of the error for each. CAI presents the explanation when prompted for error help by the student. Post-surveys of study participants indicated that students valued CAI's assistance with error debugging.

3.3 Dialogue with Student

CAI interacts with EarSketch users through a menu-based chat system (see Figure 2). The chat system contains a dialogue tree that outlines the utterances CAI can make, as well as follow-up dialogue options to present to the student. A section of the CAI dialogue tree is depicted in Figure 3. Many of the nodes in the dialogue tree contain strings that signal the system to perform actions in addition to sending an output utterance to the student. For example, `[sound_rec]`, shown in several nodes in Figure 3, signals to the system that that text should be replaced by a sound recommendation for the project. Similarly, `[SUGGESTION]` indicates that the text in the node should be replaced by a suggestion provided by the suggestion selection mechanism. This flexibility in dialogue allows CAI to respond to a wide variety of project states while reducing the burden of dialogue node authorship.

The pane in which students chat with CAI is presented on the right-hand side of the browser window (See Figure 2). The interface is integrated into the EarSketch site, allowing CAI to respond to changes in the Code Editor in real time. The interface replaces the curriculum panel when active, and users can toggle freely between the two. Students interact with CAI by selecting dialogue options from a menu presented in the CAI interface. Menu options allow users to prompt CAI for suggestions or ask for help debugging code. CAI responds to the user in the same interface, and menu options update with each interaction as the user converses with the agent. This user interface allows CAI to continuously present information without distracting from the user's view of the Code Editor, and the user is able to reinforce CAI's suggestions by switching between CAI and the curriculum.

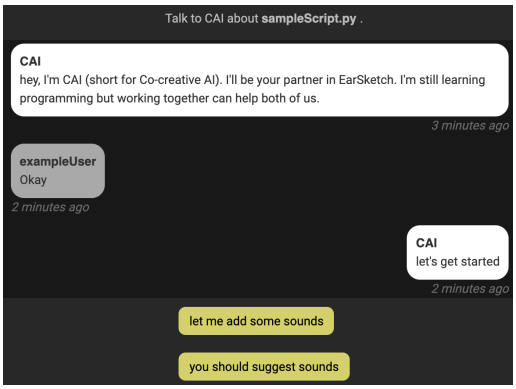

Figure 2: CAI chat interface. Users can communicate with CAI by selecting dialogue options (bottom).

Dialogue Tree In our initial classroom studies, we collected 3402 textual utterances from 68 pairs of students co-creating in EarSketch. We observed that, within the context of co-creative interactions between human collaborators, collaborators often conversed about coding challenges, sound choices, or both. The dialogue exchanges reflected both social and task-based interactions such as greetings, brainstorming about sound choice, sharing ideas, establish-

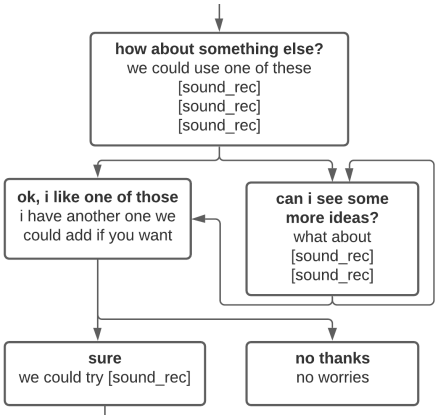

Figure 3: An excerpt from the CAI Dialogue Tree. In these nodes, the student is able to query CAI for sound recommendations for the project.

ing common ground on how to approach code challenges, and negotiating sounds to add while helping one another to resolve code errors. When all requirements of the task were met, both partners agreed to end and submit the project.

The interaction trajectories we observed in our foundational studies informed the design of CAI's dialogue tree. Our goal is to emulate the patterns and elements of dialogue exchanges between human collaborators to portray CAI as a partner (André and Pelachaud 2010). To capture the social and task-based exchanges prevalent in student-to-student conversations, CAI begins with a greeting introducing itself as a "co-creative agent" that will be a partner to the student. We chose to include "I'm still learning programming" in CAI's introduction to reinforce the idea that it is a partner and not an authoritative helper agent. CAI continues the dialogue by working with the student to select sounds for the project. If the student faces an error in the code at any point during the session, CAI's dialogue buttons populate to provide the students with the option to ask CAI for help.

Once this initial interaction is complete, the student has multiple options for continuing their conversation with CAI as they continue to develop their projects. The student can select one of three dialogue options to re-initiate interaction, or CAI can re-initiate dialogue by presenting a suggestion when the student runs their code. If CAI has a relevant code suggestion from the code suggestion module when the student runs their code, or if the student prompts CAI for a general suggestion, CAI presents a recommendation to the student for a code and/or musical addition to the project. CAI then offers a menu option where students can request explanations for its suggestions and examples of how to use relevant code concepts, as a slightly more knowledgeable partner would, without providing the exact solution. These suggestions are based on the student model, which tracks the student's knowledge level, and suggestion selection system. A menu item where students can query CAI for sound sample ideas for a section or the whole song is also available, and CAI's menu populates with an error query option every time the student runs code with an error.

4 Classroom Study

We conducted a pilot study of the CAI system in three public and charter high school classrooms in districts in the Southeastern United States. The goals of this study were to determine how well the existing CAI system allows students to improve their music and code and to identify areas for improvement. Overall, 56 students consented to participate in the study, and we have pre-survey, post-survey, and project data from 42 students. The race/ethnicity makeup of the students is: White = 55%, Asian = 18%, Black = 18%, Hispanic/Latino = 7%. Further, 82% are male and 18% are female, and they range in age from 14 to 17 with a majority (64%) in 9th grade, 20% in 10th, 14% in 11th, and 2% in 12th. Students had all worked with EarSketch as part of their class curricula before the study took place. Due to the COVID-19 pandemic, students accessed CAI from both in-person classroom environments and virtual learning environments. During the class period used for the study, students were instructed to complete or continue a task in EarSketch assigned by their teacher. Students accessed the version of EarSketch that included CAI by using a special version of the EarSketch URL. Students who participated in the study completed pre- and post-surveys about their experience with coding and music and their perceptions of CAI. Data was collected about students' interactions with CAI during the session using a series of data collection tools added to CAI to aid in system evaluation and supply interaction context for the pre- and post-survey results.

4.1 Instruments

We adapted CAI's student modeling tools to track students' interactions with CAI during our pilot studies. The version of CAI used for the study stores history for each student and each project directly to the EarSketch database. Tracked data include the dialogue and suggestion nodes visited by the student, as well as when the student ran the script (along with whether or not the script ran successfully, and a code complexity score if the code did run), and visits to pages in the EarSketch curriculum.

The system also notes when the student uses sound or code suggestions from the CAI agent. Code suggestion use is measured by comparing the project's code complexity score with the expected complexity score if the student were to implement CAI's suggestion. If the two scores match, the system considers the suggestion to have been implemented. Using this stored data, we utilized a variety of metrics to explore student-to-CAI dialogues and investigate their effects on survey outcomes. In addition, we prepared a summary for each student that included information on which nodes the student visited in which order, and information on how many of CAI's suggestions the student implemented.

This data was collected in tandem with pre- and post-survey instruments designed to collect information on students' perceptions of CAI. Our post-survey instrument includes questions on CAI's cognitive support, interaction quality, concentration/flow, psychosocial support, and overall satisfaction with the system.

Questions included four-point Likert-style ratings about various aspects of the CAI system, including the agent's

technical competency and its timing of suggestions. We use a four-point scale to encourage participants to reflect and determine whether they agree or disagree with the prompt.

In addition to the Likert-style questions, the post-survey includes free-response questions for students about why they responded they would like to work with CAI again (if they responded that they would) and about the most valuable suggestion CAI made.

4.2 Results and Discussion

Based on log file data, we compiled the number of CAI dialogue and suggestion nodes each student accessed, and correlated those numbers with student ratings of CAI on the survey items "CAI helped me make a better program" and "CAI helped me make a better song." Students that visited a higher percentage of the nodes, and thus viewed a greater portion of the dialogue tree, were significantly more likely to agree or strongly agree with the two statements. Of the participants, 42 responded to both items, each with a mean of 2.86 on a 4-point likert scale (1 = Strongly Disagree, 2 = Disagree, 3 = Agree, 4 = Strongly Agree). We calculated correlation between response to the survey item and the percentage of tree nodes visited using Spearman's rho: $r(44) = .30$, $p = .044$. Figure 4 summarizes these responses. The correlation between "CAI helped me make a better song" and the percentage of suggestion nodes viewed using Spearman's rho yielded $r(37) = .37$, $p = .024$. Additionally, we analyzed open-ended survey responses from students participating in this study. As we discuss below, the results indicate that CAI's core co-creative moves were valued by student co-creators. Students' responses frequently mentioned CAI's suggestion of sounds and forms, coding support, and debugging.

Sound Suggestions - Survey Responses Students made an average of 2.4 sound sample requests per project and 1.1 code/musical structure requests per project. This indicates that students are seeking sound support more frequently than code support in the current version. Frequency of sound requests may be partially caused by students having no way to ask CAI for samples in particular genres or using specific instruments, and instead resorting to making multiple requests until CAI suggested a sample for the desired characteristic. Future versions of CAI will implement options for students to ask for sample suggestions within a specific genre. The lack of frequency of code/music structure requests may be a result of the limited number of code/music structure suggestions made available to the learner. Future iterations on the CAI suggestion system will offer a larger number and variety of suggestions. Additionally, future studies with CAI may investigate the relationship between students' request frequencies and their previous experience with coding and making music.

The following looks at students' open-ended responses for both sound suggestions and coding support. In response to the question, "What was the most valuable suggestion that CAI made?," students offered that CAI's sound suggestions were helpful: "It suggested a very good starting rhythm/sound, then suggested sounds that worked well with it" and "The beats/sounds it suggested were great." One participant noted CAI's help in traversing the large sound library: "[CAI was] giving me good sounds to add to my music so I don't have to scroll through thousands of sounds to find the right one with my indecisive self." Another student appreciated that CAI was not only making general sound suggestions, but also suggestions tailored to specific parts of the student's project: "it suggested I use specific sounds in specific measures."

Code Suggestions and Debugging Assistance - Survey Responses While students made fewer code requests of CAI, many of the students found the coding support helpful. Students reacted to the prompt "What was the most valuable suggestion that CAI made?" with a number of code-related support items: "It helped me fix code" and "How to do a function." We also asked students, "Why would you like to work with CAI again?" Students specifically commented on CAI's support with debugging: "I would like to work with CAI because it helped me pick out sounds to add to my music and debug my program" and "Because he helped me when I had an error.[1]"

Taken together, the results of the pilot study indicate that the initial co-creative moves implemented in CAI were successful and that CAI helped students with their EarSketch projects. Students appreciated both the aesthetic and technical moves that CAI provided. The current pilot study has several limitations. First, we did not measure student learning (for example, with a pre/post test). This study design featured self-report data commensurate with usability studies, but future studies should also measure learning. Additionally, the present study was conducted in only three classrooms in the southeastern United States, and it will be important to pilot future versions of CAI with a broader and more diverse set of users.

The combination of open-ended survey responses along with the significant, positive correlation between the number of nodes visited and student ratings of CAI suggest that students perceive CAI to be beneficial. However, the results do not establish whether more interaction with the decision tree *causes* greater agreement that CAI helped make a better program. It could be that a positive attitude toward CAI contributes to greater interaction with the decision tree. We will investigate this relationship in future work that analyzes whether students continue to show a positive correlation in new iterations of CAI, if the magnitude of that correlation increases, and which students tend to show positive correlations (by computing or music-making confidence, gender, or race/ethnicity).

5 Conclusion and Future Work

This paper has presented the first co-creative intelligent agent designed to support CS learning in the context of computational music remixing. We have presented CAI's modeling tools, which analyze the musical structure and code complexity of student projects, and we have described the

[1]Though we introduce CAI as a non-gendered agent, some students referred to CAI as "he" or "she."

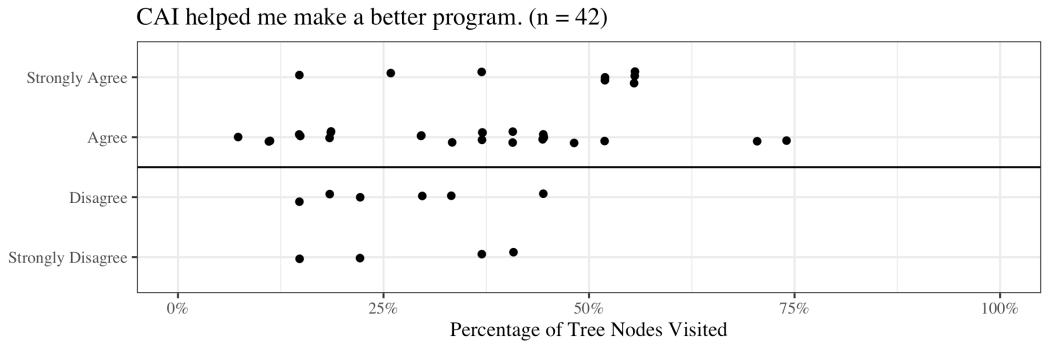

Figure 4: Survey responses to "CAI helped me to make a better program" compared with percentage of the dialogue tree visited.

menu-based dialogues CAI engages in with users. The results of the pilot study suggest that high school students valued CAI's suggestions, and that there was a positive correlation between students who interacted with more of CAI's suggestions and dialogue tree and those who felt that the agent helped them to improve their song and their project, respectively. These results suggest that co-creative moves from an intelligent agent supporting CS learning objectives have the potential to improve student attitudes in both CS and a creative domain.

The findings presented here point to several important directions for future work. With regard to CAI, we will expand its functionality to include a project model that tracks goals for EarSketch projects, and augments CAI's dialogue capabilities accordingly. More broadly, the results point to the great promise of co-creative intelligent agents for supporting learning. Future research should examine co-creativity in other educational contexts and domains, including within human-human and human-computer partnerships. Future work should also examine phenomena including turn-taking and the effectiveness of various co-creative strategies. By moving toward intelligent agents that can co-create with learners, we can provide highly effective, adaptive instruction within expressive domains.

6 Acknowledgments

This material is based upon work supported by the National Science Foundation Award No. 1814083. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. EarSketch is available online at https://earsketch.gatech.edu.

References

Ahmed, I.; Lubold, N.; and Walker, E. 2018. ROBIN: Using a programmable robot to provide feedback and encouragement on programming tasks. In *Artificial Intelligence in Education*, 9–13.

Anderson, J. R., and Skwarecki, E. 1986. The automated tutoring of introductory computer programming. *Communications of the Association for Computing Machinery* 29(9):842–849.

André, E., and Pelachaud, C. 2010. Interacting with embodied conversational agents. In *Speech Technology*, 123–149.

Bagga, S.; Maurer, B.; Miller, T.; Quinlan, L.; Silvestri, L.; Wells, D.; Winqvist, R.; Zolotas, M.; and Demiris, Y. 2019. instrumentor: An interactive robot for musical instrument tutoring. In *Towards Autonomous Robotic Systems*, 303–315.

Caramiaux, B., and Donnarumma, M. 2020. Artificial intelligence in music and performance: A subjective art-research inquiry. *Handbook of Artificial Intelligence for Music: Foundations, Advanced Approaches, and Developments for Creativity*.

Chan, T.-W., and Baskin, A. B. 1990. Learning companion systems. *Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education* 1:6–33.

Cheatley, L.; Ackerman, M.; Pease, A.; and Moncur, W. 2020. Co-creative songwriting for bereavement support. In *Proceedings of the Eleventh International Conference on Computational Creativity*, 33–40.

Crow, T.; Luxton-Reilly, A.; and Wuensche, B. 2018. Intelligent tutoring systems for programming education: A systematic review. In *Proceedings of the 20th Australasian Computing Education Conference*, 53–62.

Dannenberg, R. B.; Sanchez, M.; Joseph, A.; Capell, P.; Joseph, R.; and Saul, R. 1990. A computer-based multimedia tutor for beginning piano students. *Interface* 19(2-3):155–173.

Davis, N.; Siddiqui, S.; Karimi, P.; Maher, M. L.; and Grace, K. 2019. Creative sketching partner: A co-creative sketching tool to inspire design creativity. In *Proceedings of the Tenth International Conference on Computational Creativity*, 358–359.

Engelman, S.; Magerko, B.; McKlin, T.; Miller, M.; Edwards, D.; and Freeman, J. 2017. Creativity in authentic steam education with earsketch. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 183–188. New York, NY, USA: Association for Computing Machinery.

Faraco, R. A.; Rosatelli, M.; and Gauthier, F. O. 2004. Adaptivity in a learning companion system. *IEEE Interna-*

tional Conference on Advanced Learning Technologies 151–155.

Griffith, A. E.; Katuka, G. A.; Wiggins, J. B.; Boyer, K. E.; Freeman, J.; Magerko, B.; and McKlin, T. 2021. Discovering co-creative dialogue states during collaborative learning. In Roll, I.; McNamara, D.; Sosnovsky, S.; Luckin, R.; and Dimitrova, V., eds., *Artificial Intelligence in Education*, 165–177. Springer International Publishing.

Grover, S.; Basu, S.; and Schank, P. 2018. What we can learn about student learning from open-ended programming projects in middle school computer science. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 999–1004.

Guzdial, M.; Liao, N.; Chen, J.; Chen, S.-Y.; Shah, S.; Shah, V.; Reno, J.; Smith, G.; and Riedl, M. O. 2019. Friend, collaborator, student, manager: How design of an AI-driven game level editor affects creators. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–13.

Lopez-Rincon, O.; Starostenko, O.; and Ayala-San Martín, G. 2018. Algoritmic music composition based on artificial intelligence: A survey. In *2018 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, 187–193.

Ma, W.; Adesope, O. O.; Nesbit, J.; and Liu, Q. 2014. Intelligent tutoring systems and learning outcomes: A meta-analysis. *Journal of Educational Psychology* 106:901–918.

Magerko, B.; Freeman, J.; Mcklin, T.; Reilly, M.; Livingston, E.; Mccoid, S.; and Crews-Brown, A. 2016. Earsketch: A steam-based approach for underrepresented populations in high school computer science education. *ACM Transactions on Computing Education*.

McKlin, T.; Magerko, B.; Lee, T.; Wanzer, D.; Edwards, D.; and Freeman, J. 2018. Authenticity and personal creativity: How earsketch affects student persistence. In *Proc. of the 49th ACM Technical Symposium on Computer Science Education*, 987–992.

Molloy, W.; Huang, E.; and Wünsche, B. 2019. Mixed reality piano tutor: A gamified piano practice environment. *2019 International Conference on Electronics, Information, and Communication (ICEIC)* 1–7.

Nesbit, J. C.; Adesope, O. O.; Liu, Q.; and Ma, W. 2014. How effective are intelligent tutoring systems in computer science education? In *2014 IEEE 14th International Conference on Advanced Learning Technologies*, 99–103.

Ocaña, J. M.; Morales-Urrutia, E. K.; Pérez-Marín, D.; and Pizarro, C. 2020. Can a learning companion be used to continue teaching programming to children even during the COVID-19 pandemic? *IEEE Access* 8:157840–157861.

Percival, G.; Wang, Y.; and Tzanetakis, G. 2007. Effective use of multimedia for computer-assisted musical instrument tutoring. In *Proceedings of the International Workshop on Educational Multimedia and Multimedia Education*, 67–76.

Petry, P. G., and Rosatelli, M. C. 2006. Algolc: A learning companion system for teaching and learning algorithms. In *Intelligent Tutoring Systems*, 775–777.

Resnick, M.; Maloney, J.; Monroy-Hernández, A.; Rusk, N.; Eastmond, E.; Brennan, K.; Millner, A.; Rosenbaum, E.; Silver, J.; Silverman, B.; and Kafai, Y. 2009. Scratch: Programming for all. *Communications of the Association for Computing Machinery* 52(11):60–67.

Rigby, L.; Wünsche, B.; and Shaw, A. 2020. piARno - an augmented reality piano tutor. *32nd Australian Conference on Human-Computer Interaction* 481–491.

Ritschel, H.; Seiderer, A.; and Andre, E. 2020. Pianobot: An adaptive robotic piano tutor. In *HRI 2020 Workshop on Exploring Creative Content in Social Robotics*.

Sleeman, D., and Brown, J. S. 1982. *Intelligent Tutoring Systems*.

Smith, J.; Jacob, M.; Freeman, J.; Magerko, B.; and Mcklin, T. Combining collaborative and content filtering in a recommendation system for a web-based daw. In *Proceedings of the International Web Audio Conference*.

Smith, J.; Weeks, D.; Jacob, M.; Freeman, J.; and Magerko, B. Towards a hybrid recommendation system for a sound library. In *Joint Proceedings of the ACM IUI 2019 Workshops*.

Smith, J.; Truesdell, E. J. K.; Freeman, J.; Magerko, B.; and Boyer, K. E. 2020. Modeling music and code knowledge to support a co-creative ai agent for education. In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, 134–141.

Smoliar, S.; Waterworth, J.; and Kellock, P. R. 1995. pianoFORTE: a system for piano education beyond notation literacy. In *MULTIMEDIA '95*, 457–465.

Starr, C. W.; Manaris, B.; and Stalvey, R. H. 2008. Bloom's taxonomy revisited: Specifying assessable learning objectives in computer science. *ACM SIGCSE Bulletin* 40(1):261–265.

Thompson, E.; Luxton-Reilly, A.; Whalley, J. L.; Hu, M.; and Robbins, P. 2008. Bloom's taxonomy for CS assessment. In *Proc. of the Tenth Conference on Australasian Computing Education-Volume 78*, 155–161.

Wang, C.-Y., and Lai, A.-F. 2011. Development of a mobile rhythm learning system based on digital game-based learning companion. In *Edutainment Technologies. Educational Games and Virtual Reality/Augmented Reality Applications*, 92–100.

Xambó, A.; Roma, G.; Shah, P.; Tsuchiya, T.; Freeman, J.; and Magerko, B. 2018. turn-taking and online chatting in remote and co-located collaborative music live coding.

Zacharakis, A.; Kaliakatsos-Papakostas, M.; Kalaitzidou, S.; and Cambouropoulos, E. 2021. Evaluating human-computer co-creative processes in music: A case study on the CHAMELEON melodic harmonizer. *Frontiers in Psychology* 12:322.