Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025 ASEE
Paper ID #48753

Considerations from Co-design with Special Education Teachers: Integrating Computational Thinking with Executive Functioning Skills for Autistic Middle School Students (RTP, Diversity)

Dr. Robert Hayes, Tufts Center for Engineering Education and Outreach
Dr. Jennifer Cross, Tufts Center for Engineering Education and Outreach

Dr. Jennifer Cross is a Research Assistant Professor at the Tufts University Center for Engineering Education and Outreach where her primary research interests include human-robot interaction with a focus on the educational applications of robotics and diversity in engineering education.

Elissa Milto, Tufts Center for Engineering Education and Outreach

Elissa Milto is Director of Outreach at the CEEO. She holds two masters degrees in education allowing her to focus on special education and engineering. Currently, she leads Novel Engineering, an interdisciplinary engineering literacy project. Her work fo

Considerations from Co-design with Special Education Teachers: Integrating Computational Thinking with Executive Functioning Skills for Autistic Middle School Students (RTP, Diversity)

Abstract

We report on the co-design of technologies for Opportunities for Robotics, Building, and Innovative Technology (ORBIT), an educational robotics program for autistic middle school students designed to integrate learning computational thinking (CT) practices with executive functioning (EF) skills. We are developing this program through a research-practice partnership between researchers at a private northeastern university and practitioners at a local public school with a sub-separate, special education program designed for autistic students.

Our program comprises a sequence of CT and robotics activities and student-facing scaffolds, co-developed with teachers through an interactive design and feedback process. The robotics program was designed around the LEGO® SPIKE™ Prime platform. We initially set out to develop a digital coding environment tailored to the needs of autistic middle school students—guided by universal design for learning (UDL) principles—and accompanying activities aimed at supporting students' CT and EF skills, but soon identified the need for student-facing scaffolds that aided students making connections between classroom learning and ORBIT. We report on the first three of six planned co-design workshops with teachers, focused on exploring design goals, learning goals, and needs; testing iterative prototypes of digital coding environments; and developing an instructional sequence for practicing CT.

In this paper, we explore the following question: What design considerations inform structuring the ORBIT program technology to support students developing independence in computational thinking alongside executive functioning skills? We collected audio and video data from all co-development workshops, along with artifacts generated during workshops. We analyzed data through an iterative process of coding for themes related to the design of supports and tools, as well as rationales relating to CT and IEP goals. We then checked and refined themes with teachers.

We identified themes motivating teachers' design feedback, which informed three design considerations: 1) ORBIT curriculum and technology should support a teacher-mediated, dynamic trajectory of scaffolds towards students' independent participation in robotics and CT practices, 2) task structures should include feedback mechanisms and routines to support students' continued independent participation by attending to executive functioning needs, and 3) ORBIT resources should include multiple means of representation to enhance the bidirectional transfer between the ORBIT program and other classroom resources that are already part of the students' routine.

We exemplify how to implement these three design considerations through three design embodiments that emerged in our co-design workshops: a physical planning board, introductory robotics coding missions, and a poster-style glossary. The design embodiments are intended to work together to provide support to students at all phases of using the ORBIT program with the intention of building students' independence to engage in CT and robotics practices.

Introduction

Often autistic students do not get the same access to STEM as their typically developing peers, specifically, as discussed in this paper, access to computational thinking and robotics. We report on the co-design of technologies for Opportunities for Robotics, Building, and Innovative Technology (ORBIT), an educational robotics program for autistic middle school students designed to integrate learning computational thinking (CT) practices with executive functioning (EF) skills. The program includes a computer coding component and several student-facing scaffolds. We are developing this program through a research-practice partnership between researchers at a private northeastern university and practitioners at a local public school within a sub-separate, special education program designed for autistic students. Our program comprises a sequence of CT activities and supporting technologies, co-developed with teachers through an interactive design and feedback process. We use identity-first language (i.e. autistic student) rather than person-first language (student with autism) because that is generally preferred by individuals in the autism community [1].

Computational Thinking and Executive Functioning

Computational Thinking (CT), a term coined by Wing, is described as the skills and practices involved in "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science," [2]. CT is further described as "a way humans solve problems" and "requires thinking at multiple levels of abstraction" which Wing [2] presents as a fundamental skill for everyone in modern society, separate from knowing how to write computer code. We have chosen to conceptualize students' robotics learning using a computational thinking approach known as PRADA [3] because its core practice components align with executive functioning skills, and can build from learning contexts already familiar to autistic students.

Executive functioning (EF) refers to skills that are used to plan and carry out goals. Diamond [4] describes executive functions as a set of cognitive processes essential for goal-directed behavior, including working memory, cognitive flexibility, and inhibitory control. These skills support planning, problem-solving, and regulating thoughts and emotions. Computational thinking and robotics activities require executive functioning throughout the process in order to define and then meet the goals of a robotics activity.

There is a need for CT learning technology designed with autistic students in mind

Autistic students are often underrepresented in STEM, academically and vocationally. This disparity is exacerbated when students receive special education services outside of mainstream classrooms, and compounded even more when students are part of a sub-separate program where they spend the majority of the school day in a special education setting. Robotics and CT are often offered to students outside of their core classes (math, English, social studies, and science) which is when students are usually pulled out for special education services. Although special education teachers are experts in the pedagogy related to helping their students access the curriculum, they usually do not have a background in STEM. Therefore, there is a diminished chance of autistic students spending time during the school day engaging in robotics and CT

learning. With little experience in robotics and CT, the pathway to jobs related to these fields are also diminished.

Although there is technology geared toward neurodivergent users, they are often not STEM learning tools. Robotics and programming applications exist that target skills such as social, and speech and language, but not to our knowledge as a medium to teach CT and robotics specifically to autistic students. STEM technology is, for the most part, not designed to reflect needs for autistic users or to scaffold how they begin using the technology in a way that supports CT and IEP skills.

It is not enough to give autistic students access to CT and robotics. Just as curricula for core academics and other subjects should be designed to reflect the needs of autistic students, programs that teach CT robotics should also take into account these considerations. Interactions should be scaffolded so that they build on their strengths, address their needs, and lead them toward independence. Since educators have first-hand experience working with their students, it is important that they be part of the development of materials that are for autistic students. Alignment with Universal Design for Learning helps ensure that there are multiple means of representation, expression, and action to broaden accessibility, but learning environments, particularly those for autistic users, should reflect the needs of individual users as well. It is our intention for the ORBIT program to provide such a learning environment.

Design considerations for designing robotics technology for autistic middle school students

As part of the ORBIT program, researchers are working with classroom teachers who are autism specialists to co-develop a flexible, comprehensive CT and robotics curriculum and associated scaffolds, including customized learning technologies. Through the co-design process, we have identified three design considerations that we believe central to integrating CT practices and EF skills development.

> Consideration 1. ORBIT curriculum and technology should support a teacher-mediated, dynamic trajectory of scaffolds towards students' independent participation in robotics and CT practices.
>
> Consideration 2. Task structures should include feedback mechanisms and routines to support students' continued independent participation by attending to executive functioning needs.
>
> Consideration 3. ORBIT resources should include multiple means of representation to enhance the bidirectional transfer between the ORBIT program and other classroom resources that are already part of the students' routine.

The design considerations are not new concepts, but are salient for curriculum and technology designers as they think about CT and robotics with autism students. In this paper, we will explore why these design considerations are important and how to incorporate them into CT and robotics activities. We exemplify how to use these considerations for design through three design embodiments that emerged through our own co-design process. These are

Embodiment 1. A **planning board** that introduces programming icons and step-by-step algorithms before students program on a screen.

Embodiment 2. A series of structured, online skill-building **coding missions** that introduce core icons and behaviors when students start programming online.

Embodiment 3. A **glossary** poster of relevant terminology and programming icons that leverages existing strategies for teaching vocabulary with autistic students.

It is our hope that designers of CT technologies and curricula will find these three design considerations, coupled with examples, a productive starting point for developing more specialized resources to create opportunities for autistic students to access robotics and computational thinking.

Conceptual Framework

A primary goal of the ORBIT program is to design technologies that support autistic students developing computational thinking practices alongside executive functioning skills. Here we unpack how we conceptualize CT and EF.

Interdisciplinary Computational Thinking and PRADA Framework

While there is no one definition of Computational Thinking and its components that is generally accepted across all disciplines and contexts [5], it is generally agreed that CT practices are valuable not only for those interested in careers related to computing but also for everyone engaging with the ubiquitous computing of modern society. The K-12 Computing Science Framework [6], a joint work by computer science education organizations including ACM, Code.org, and CSTA, highlights that while computer science offers "unique opportunities for developing computational thinking," CT practices are also "explicitly [...] and implicity" referenced in the standards frameworks of other disciplines, such as math and science. Simultaneously, various education interventions have explored approaches to enhancing interdisciplinary integration of CT practices within non-CS disciplines [7], [8], [9], [10]. In our conceptual framework for this work, we adopt the CT definition from the PRADA framework [3], which focuses on the practices of pattern recognition, abstraction, decomposition, and algorithms, as listed in Table 1.

Table 1: PRADA Computational Thinking Framework Definitions

Component	Definition
Pattern Recognition	Observing and identifying patterns, trends, and regularities in data, processes, or problems
Abstraction	Identifying the general principles and properties that are important and relevant to the problem
Decomposition	Breaking down data, processes, or problems into meaningful smaller, manageable parts
Algorithms	Developing step-by-step instructions for solving a problem and similar problems

The PRADA framework has multiple advantages for this project. It comprises "four elements [that] are detailed enough to capture the essence of CT defined in previous frameworks" which are generalized to be suitable for interdisciplinary integration "not bounded by content area or tools," [3]. This aids us in lowering barriers for special education teachers with limited prior knowledge of these practices, by distilling the breadth of CT practice into these four clearly distinguished components and a memorable mnemonic. Likewise, focusing our CT definition on widely applicable practices supports our design in offering opportunities for students to engage in skills that are also applicable in other disciplines and domains that appear as part of IEP goals.

Executive Functioning

Miyake et al. [11] define executive functioning as a collection of higher-order cognitive processes, including inhibition, working memory, cognitive flexibility, planning, and problem-solving, which enable goal-directed behavior and adaptive responses to complex or novel situations." It involves the interrelation of working memory, inhibitory control, and cognitive flexibility (see Table 2). It is contextual and fluid which means that for the same person, the ability to access executive functioning may be different from day to day and activity to activity.

Table 2: Executive Functioning Components (adapted from [14]).

Component	Definition
Working Memory	Ability to hold information "on-line" and manipulate it, differentiated according to whether the information is verbal or spatial in nature
Inhibitory Control	Capacity to hold a rule in mind, responding according to this rule, and resist prepotent response
Set-Shifting (Cognitive Flexibility)	Ability to shift flexibly one's attentional focus

Demetriou et al. [12] found that autistic children exhibit differences in core components of executive functioning, however these present uniquely for each student. Improvement of executive functioning skills is often included as a goal on autistic students' Individualized Education Plans (IEPs) [13], with schools providing scaffolds to help students improve their executive functioning skills for educational and personal reasons. Autistic people have differences in their executive functioning from neurotypical people and we need to understand each person's individual skills and needs in order to figure out appropriate support.

ORBIT

The ORBIT Program is being developed as part of a collaborative design process with educators who work with autistic students. By using a human-centered, participatory design approach we have moved from an initial concept to prototype design and refinement [15], [16], [17]. During meetings and co-design workshops, educators identified skills that are often addressed as part of students' IEP goals that could overlap with CT goals. Their input as users is crucial to developing a program that fits the values, needs, and cultures of their classrooms while also providing knowledge about special education pedagogy and experience with technologies in

their classrooms. For CT/robotics to be beneficial to students in their classrooms, teachers must understand the technology. The benefits of robotics activities can only happen when "the technology is used skillfully by the teachers, aligning the tools with the students' educational needs" [18].

The initial ORBIT Program design objectives were to:

1. Co-design and develop a flexible curriculum framework based on students' interests and needs.
2. Co-design and develop tools/technologies with customizable interfaces for high levels of individualization.

Hardware

The educational robotics hardware basis for ORBIT was the LEGO® SPIKE™ Prime education kit. The LEGO® SPIKE™ Prime was chosen because the students have some experience building and programming with the platform, which had already been used in these classrooms as part of outreach programs. Although the initial cost of the kits is expensive, the LEGO® bricks, microprocessor (hub), sensors, and motors are reusable. Additionally, the functionality of the buttons on the programmable hub and the attachment of the inputs and outputs are relatively simple to understand and operate.

Digital Coding Interface

The educational robotics hardware basis of the digital programming interface builds on educational software tools such as Scratch [19] and ScratchJr [20] as well as educational robot coding tools including the CREATE Lab Visual Programmer [21] and Flutter [22]. Our values as designers and educational technology developers therefore are similarly inspired by the desire to provide "low floor" and "wide walls" [23].

Several prototype iterations were presented for teacher feedback through a series of co-design workshops, starting with a paper prototype, moving to Figma, a screen-based prototype, and then an app with limited functionality.

Process of program development

As the ORBIT program is an ongoing iterative project, we will present program development up to the first viable pilot of the coding application. During the sixteen-month period leading up to classroom implementation, the middle school students continued to do engineering once a week with undergraduate engineering students to give them experience with the engineering design process. Additionally, researchers and practitioners met informally several times as the first step in the co-design process. Formal codevelopment work was done as part of Co-development Workshop—three reported in this paper (see Table 3), one since writing the paper, and two scheduled for the spring semester.

Table 3: Codesign workshops' description and goals

Workshop & Duration	Description	Goals
1: Storytelling (3 Hours)	Activities using storytelling to learn about participants' teaching experiences	To understand participants' professional values and goals, and barriers faced with educational technologies in the classroom
2: Paper Prototyping & Design (2.5 Hours)	Using paper prototyping, a low-fidelity, hands-on method of visualizing and testing user interfaces	To quickly iterate design concepts, gather user feedback, and identify usability issues before committing to more detailed and costly development
3: Digital Prototyping & Design (2.5 Hours)	Digital prototyping with Figma is a hands-on method of visualizing and testing user interfaces through preliminary digital images and graphics	To iterate design concepts, gather user feedback, and identify usability issues before committing to more detailed and costly development

Methods

Design Team & Partners

The participatory design work was conducted as a partnership between university researchers and teachers from local, urban middle school classrooms in a public school district outside of Boston. The involved researchers have a background of education, engineering education, engineering, and special education. They share a belief in students' creative learning and making capacity—grounded in a constructivist, dynamic systems epistemology—which informs a commitment to developing technologies and learning resources that support students' agentive learning.

The two partner classroom teachers are autism specialists who have worked with the university through engineering education outreach programs for three years prior to this project. They lead classrooms that are part of a sub-separate program for autistic students who require more specialized instruction than is possible in a general education classroom and a smaller setting. Each classroom has a lead teacher, two to three paraprofessionals, and between six to eight students. The school librarian, who had extensive experience with LEGO® SPIKE™ Prime kits, also participated in design team activities.

Design Research & Co-Design Process

Co-development work was done during information meetings and workshops as outlined above. The team of researchers provided the structure for the interactions and was responsible for development of the application. The teachers provided input on the proposed meeting structure and in all phases of the design process for ORBIT.

Two of the original proposal objectives centered on 1. Researching the skills, tools, and resources necessary for IEP teams to integrate computational technology into their disciplinary instruction, and 2. Understanding necessary curricular components to support students'

individualized robotics learning while meeting IEP-goals. As we began to pursue these objectives through co-design workshops we developed the research question for this paper. What design considerations inform structuring the ORBIT program **technology** to support students developing independence in computational thinking alongside executive functioning skills?

Notably, this first stage of program development was centered on the development of technology. In our initial conception technology referred primarily to the development of a software application for coding with LEGO® SPIKE™ Prime robotics hardware; with input from teachers, our conception of technology quickly expanded to include scaffolds to support students as they learned to use the coding interface.

We use iterative thematic coding to analyze the data collected at the meetings and workshops [24]. The workshops and meetings were facilitated by the research team, but conducted collaboratively in order to ensure representation of everyones' ideas. Data includes audio and video recordings, and notes of the sessions. After each workshop, the audio recordings were transcribed. The transcripts along with notes served as the primary sources of data analysis. Coding results were shared with the teachers to allow them to confirm or challenge the themes and design embodiments.

Findings

Coding and thematic analysis process

In each of the three workshops, we identified between 65 to 119 design ideas proposed by either teachers or researchers (Table 4) – around 80% of ideas coded were from teachers, and the remaining 20% from researchers. Through a first round of thematic analysis, we grouped these ideas into between 11 to 18 categories, with the greatest idea diversity occurring during the brainstorming workshop (#1) and the greatest idea specificity occurring during the prototyping sessions (#2,3) (Table 4).

Table 4. Summary of coded design ideas from three co-design workshops

Workshop Number and Activity Description	# of Coded Ideas	# of Themes
1: Brainstorming discussion between researchers and teachers	102	17
2: Paper prototype exploration of activity sequence and app features	119	11
3: Computer prototype exploration of activity sequence and app features	65	11

Themes included both practical design considerations—such as providing multiple representations of text, e.g. audible speech, or affirmations—as well as student outcomes motivating the design—such as student independence and skills development, e.g. reading and executive functioning. Some themes were unique to a single workshop—for instance, support for group tasks only showed up in brainstorming, but did not emerge in the prototyping. Many themes were consistent across all three workshops, for instance that the teacher should have the ability to customize the program for individual students.

Through a second-round analysis, three researchers examined the themes that appeared in more than one workshop to try to understand what motivated specific design ideas. Across three coding discussions, each between one to two hours long, we settled on three design considerations that appeared to underpin the themes we had coded previously, which we summarize and elaborate on in detail below.

Design Consideration 1—Students navigate a teacher-mediated trajectory of structured tasks, moving toward independent participation in CT practices

The first design consideration has to do with how teachers dynamically structure the space in which students are engaging in computational thinking practices, based on student needs. The driving motivation for design ideas in this category is that each student has their own unique needs for scaffolding executive functioning. Those needs will change over time as students gain familiarity with use of the technology, as well as day-to-day. At a zoomed-out perspective, teachers want students to achieve (as) independent proficiency (as possible) with the technology, and want to be able to construct a flexible, teacher-mediated trajectory of executive functioning scaffolds to support each student in meeting this goal.

Zooming in to the level of design ideas, teachers repeatedly suggest the ability to limit options within the software interface, particularly as students first use it, to reduce opportunities for distraction (addressing Inhibitory Control and Working Memory) and to facilitate coordination between planning and executing code to make a robot do whatever the student is aiming to accomplish. Teachers proposed many ways for them to adjust individual elements of the interface on a per-student basis, ranging from constraining the number of specific kinds of coding blocks available (e.g. sound blocks), to adjusting the way elements are represented (we elaborate on this point in Design Consideration 3 below). The culmination of this line of thinking was the notion of teacher-defined student profiles to ease in structuring the environment, and changing how that structure evolves over time as students gain proficiency in navigating the software and employing computational thinking practices.

One final design idea motivated by this consideration of a trajectory towards student independence is a set of initial *robotics coding missions* to familiarize students with the software interface and CT practices in a controlled, incremental fashion. *Introductory explorations* to support executive functioning and interactions with the software and hardware would occur before the students' first interaction with the software and hardware. Additionally, these exploratory activities would be customizable for each student, with some students skipping the first few if they had mastery of learning goals of individual activities. We explore coding missions further in the Design Embodiments section of the Discussion, and examine how they address this design consideration (students navigating a teacher-mediated trajectory of structured tasks towards independent participation in CT practices) along with the other two design considerations.

Design Consideration 2—Task structures include feedback mechanisms and routines to support students' continued independent participation by attending to executive functioning needs

The second design consideration has to do with how the task structures themselves support executive functioning to foster students' emergent and ongoing independent participation in

computational thinking practices and robotics activities. The types of design ideas informed by this consideration can be grouped into two large categories: providing feedback mechanisms, and establishing routines.

Teachers frequently suggest variations on positive feedback across all the workshops, and especially in engaging with the software interface. The most common request is immediate audible affirmation, such as a ding or boop noise, when correctly performing a coding task—this serves both to scaffold executive function, such as Task Switching, while simultaneously attending to student affect (providing regular, positive reinforcement). In addition to audible affirmations, teachers suggest complementary visual cues (such as using color to indicate right or wrong). Extending beyond immediate feedback is the idea of an incentive system such as stars for completing coding tasks. Communicating when students get something right is motivated not just by positive reinforcement but also by a desire to minimize confusion and unnecessary back-tracking. To that end many ideas emerge around elements of the interface which are unintuitive and might prompt a student to undo something they had done correctly.

In addition to feedback, many ideas emerge around establishing familiar routines within the coding tasks and within how students engage with the coding interface. One ubiquitous routine is that of beginning, middle, and end, which the teachers connect directly to their ongoing classroom experiences with their students. Related is a suggestion to incorporate checklists into the task structure and into the interface itself. Possibly the most impactful instantiation of a new routine to the ORBIT program development is the creation of a planning board to scaffold Planning, Task Switching, and Working Memory for students as they work on robotics coding challenges. We expand on the Planning Board in the Design Embodiments section of the Discussion.

Design Consideration 3—Resources incorporate multiple means of representation to enhance bidirectional transfer between the designed program and external resources

The majority of specific design ideas center on the third design consideration, around how to use representation to enhance bidirectional transfer between the ORBIT program and external resources. This consideration can be broken down into three broad suggestions: resources should use multiple representations, consistent representations, and unambiguous representations.

When workshopping the software interface teachers regularly suggest ways that it can use multiple complementary ways of communicating the function of things in the coding environment. One motivation for this is accessibility, as embodied in the suggestions to incorporate audible speech and icons alongside written text, as well as for multilingual support. This idea is especially salient with this population which may include students who are non-readers or need reading support.

Complementing multiple representations was a strong signal from teachers that representations should be consistent: within the program, between the program and the physical components used for robotics, and with prior classroom experience. Representational consistency within the program referred primarily to using the same icons for different representational modalities—for example, a paper planning board should use the same icons for representing components as does the software interface. In addition to being consistent across program modalities, icons that represent physical components or actions should be recognizable as such: so an icon that

represents a motor action should look enough like the motor component to be identifiable, and one that represents an action such as turning should as much as possible match the physical action of turning. This last design idea is abstract, and deciding upon a representation is aided by the suggestion to make representations consistent with prior experience. This could include familiar icons such as a caret to represent a drop-down box in other software environments; it could include icons from similar environments such as Scratch (which for instance uses a curved arrow to represent turning); or it could be codified representations such as sight words and Mayer-Johnson symbols (which represents the word turn using two people and an arrow, in the sense of "whose turn is it").

Finally, teachers stress a need to minimize ambiguity in work or icon meanings. For example, the word "wait" can mean *pause for some time*, or it can mean *stop doing what you're doing*, and so the meaning of a "wait block" might be ambiguous to students. In instances of ambiguity, teachers suggest erring on the side of concrete language—in the case of a "wait block" maybe a "pause for seconds block" is a more concrete wording. All three of these suggestions—providing multiple, consistent, unambiguous representations—serve to support students meaning making while engaging with computational thinking within the ORBIT program **and** to enhance their ability to take ideas learned in the program to other contexts.

Discussion

During co-design, three main design embodiments emerged which exemplify how to implement the three design considerations: a physical **planning board**, introductory **robotics coding missions**, and a poster-style **glossary**. The design embodiments are intended to work together to provide support to students at all phases of using the ORBIT program with the intention of building students' independence to engage in CT and robotics practices.

Design embodiments

Planning board

The first major design embodiment to emerge from our co-design workshops is the Planning Board, a laminated paper table which students use to plan code sequences before programming them in the software interface (see Figure 1). The planning board is modeled after one that students use as executive functioning support for other classroom tasks. It structures sequencing and decomposition of coding tasks by explicitly creating an ordered space to keep track of step number, a description of the task to be completed, a description of the desired robot action (at a more meso level, such as move the robot forward some amount), and a representation of the code which might accomplish that action (at a more micro level, such as turn the motors forward for ten rotations).

The planning board serves as a teacher-mediated tool which helps students develop fundamental computational thinking skills before engaging with the software interface. Furthermore, the planning board scaffolds students' executive functioning, particularly around planning and working memory, in a way that teachers can lean on or remove depending on their assessment of student needs, serving as a way to transition from supported to independent programming.

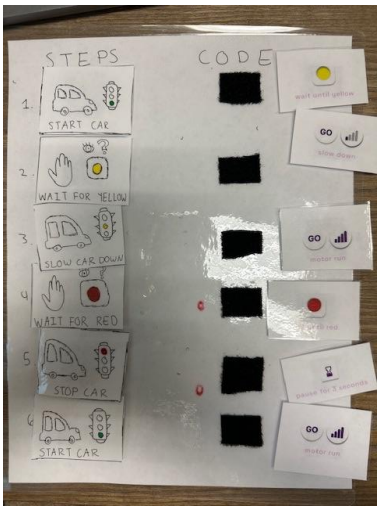

Figure 1. Early iteration of laminated paper planning board, which scaffolds coding sequences with descriptions of actions to be completed by a robot.

Possibly most helpfully the planning board establishes a routine for engaging in computational thinking practices. It reinforces two important habits of going through a coding sequence step-by-step, and of decomposing coding tasks from task to desired action to code. The tactile interface also provides a space for teachers to leverage other familiar routines such as working through a sequence using discrete trial training (DTT), a strategy already employed in the classroom. According to Smith [25], discrete trial training is "a method for individualizing and simplifying instruction to enhance children's learning" and is helpful to teach autistic children advanced skills. Classroom teachers use a prompting hierarchy (e.g. pointing, naming, moving components) as they model, prompt, and reinforce functional communication skills. In this case, teachers would use function communication strategies as students learn the name, graphics, and functions of programming icons. The physical modality of the planning board not only complements the digital modality of the software interface, it also reinforces the iconography, vocabulary, and spatial representation of coding sequences (top to bottom) through the principle of consistency.

An additional affordance of the planning board is how it supports the development of a clean, user-friendly software interface. Because the planning board scaffolds working memory by tracking the coding actions AND their sequence, this information does not need to be emphasized as strongly in the software. This in turn opens up screen real estate for large text and iconography, clearly separated into functionally distinct zones. Where the coding sequence information *is* tracked it is consistent with the planning board direction and order.

Robotics Coding Missions

While co-designing the software application (Figure 2), an idea emerged to develop Robotics Coding Missions—a scaffolded sequence of software activities in which interface options are limited to only necessary components, that students can engage with before using the full interface (similar to a tutorial), hereafter referred to as Missions. Missions help to show students how to use the software interface (which elements are interactable, how code is organized, where to find commands) while also introducing them to coding concepts in a highly constrained

environment. So, for example, in the first mission students simply make a single motor turn with more than one rotation. They learn where the motor commands are, how to add a motor command to the sequence, and how to run their code—this structure has the added benefit of allowing teacher facilitators to debug potential hardware issues in a very constrained environment, a feature which is informed by researchers' experiences teaching robotics.

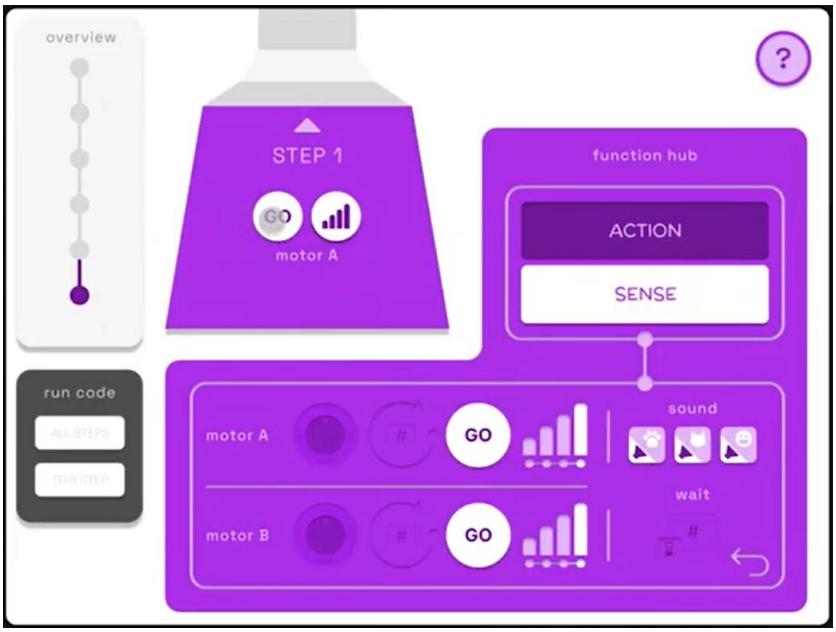

Figure 2. Software application interface tested by the teachers in Workshop 3, broken into three distinct zones: left is the coding sequence overview; top middle is the coding steps; right is the function hub.

Missions provide a software-mediated trajectory for developing independence in using ORBIT program technologies. They scaffold both executive functioning and computational thinking early on to acclimate students to the coding environment, and set a path for students to develop both EF & CT together. In addition to supporting this trajectory, Missions can serve as useful references later when students may feel stuck on more open-ended tasks.

Since Missions are more structured, it is easier to program feedback mechanisms into the software itself. This includes clear affirmations for accomplishing the mission tasks, supporting student engagement in the early stages of coding, as well as prompts to help students make progress when stuck, which are more difficult to implement for open-ended coding tasks. In leveraging prompts for help, students can develop routines for help-seeking behavior in more independent work.

By introducing software iconography in a concrete, bounded context, Missions help to mitigate ambiguity in representations used in the ORBIT software interface, reinforcing coding vocabulary. Furthermore, Missions introduce students to accessibility features, such as audio, which can aid their independence in later coding work.

Glossary (poster)

Another early design embodiment to emerge in workshop discussions is the coding Glossary. Drawing inspiration from a Communication Board in ASD classrooms, the Glossary is a poster which connects words and icons used across the ORBIT program (e.g. with the planning board and software interface) to images of physical objects represented. The Glossary is available to students from the start of their coding journey, and is kept nearby for reference as needed.

The Glossary scaffolds working memory, which is important early in the trajectory of learning new materials and skills, including learning to recognize new words/symbols unique to coding work. As with the structure of the Planning Board, the Glossary can be used in a functional communication sequence, which can help bridge early work in the physical and digital coding spaces.

Students can use the Glossary for immediate feedback in interpreting iconography and vocabulary. Furthermore it reinforces common language in the classroom, which can facilitate communication and help students to share their ideas, supporting important social learning dynamics which have to this point been backgrounded in our design considerations. Lastly, the Glossary can help students to coordinate across multiple representations (word, icon, image), and supports consistency across modalities (software, planning board, physical materials).

Contributions

In this paper we've identified three design considerations that emerged in a technology co-design for teaching robotics and computational thinking practices alongside executive functioning skills with autistic middle school students. We've summarized these as:

1. Technology supports students to navigate a teacher-mediated trajectory of structured tasks, moving toward independent participation in CT practices
2. Task structures include feedback mechanisms and routines to support students' continued independent participation by attending to executive functioning needs
3. Resources incorporate multiple means of representation to enhance bidirectional transfer between the designed program and external resources

Taken in the abstract these considerations are not new ideas, and could be considered typical of most curriculum development. For example, the consideration of a teacher-mediated trajectory informed the idea of Missions, which take inspiration from tutorials and could be grouped under a large umbrella of competency-building lesson sequences that might be found in any number of technology units.

Instead we contend that it is unpacking why and how these three design considerations are important and useful for teaching robotics with autistic middle school students which may serve as a productive starting point for curriculum designers. The idea of a teacher-mediated trajectory normally informs lesson sequences or tutorials, as named above, but in this context it is especially important to provide teachers with the ability to provide more dynamic, micro-level adjustments based on individual student needs, particularly related to executive functioning skill scaffolding and development. Feedback and routine are important to any educational technology,

but do not typically explicitly address Task Switching, Working Memory, or Inhibitory Control, and would not have likely led to the suggestion of a Planning Board in the process of developing a software interface. Multiple means of representation is a standard Universal Design for Learning principle, but taken in conjunction with the other two design considerations, along with explicit attention to executive functioning, ideas around consistency gain prominence in the development of a Glossary that can both scaffold Working Memory and support routines typical in this learning environment.

Furthermore, we think it especially important that any emerging work developing computational thinking learning resources for autistic students avoid deficit framings. Throughout our work we instead emphasize the considerations and strategies that practitioners bring to their classrooms, such as an attention to executive functioning skills and means of providing individual student support. This last point cannot be over-emphasized—many of our design ideas align with principles of Universal Design for Learning, which serves a great starting point for developing accessible resources, but resources for students with IEPs must be flexible and provide options for teacher adjustment to individual student needs.

Limitations and Future Work

The design considerations described in this paper come from the initial stages developing the ORBIT program. Most notably, the materials that emerged from these considerations have not yet been used *with students*. Furthermore, our emphasis to this point has been to develop the technology features of our program. In our most recent (fourth) workshop (not reported on here) we have discussed how these ideas can aid in the development of activity sequences organized around skill-building, most notably including social and life skills that were not as foregrounded in discussions centered on the development of the coding application. This activity sequence will serve as a final design step before working directly with middle-school aged autistic students in the program. During instruction with students we will use classroom observations, artifacts such as screen capture of students' code, and teacher feedback on students' experiences, to contextualize, refine, and expand these design considerations. A limitation of this work is that we are not able to directly include autistic students in the design process in this phase of the project; as we develop our partnership with local schools and community we plan to include the voices of autistic students in design.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. DRL-2318191.

References

[1] A. Taboas, K. Doepke, and C. Zimmerman, “Preferences for identity-first versus person-first language in a US sample of autism stakeholders,” *Autism*, vol. 27, no. 2, 565-570, 2023.

[2] J. M. Wing, “Computational thinking,” *Communications of the ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[3] Y. Dong, V. Catete, R. Jocius, N. Lytle, T. Barnes, J. Albert, and A. Andrews, "PRADA: A practical model for integrating computational thinking in K-12 education," *Journal Name*, vol. X, no. Y, pp. 906–912, Feb. 2019.

[4] A. Diamond, “Executive functions,” *Annual Review of Psychology*, vol. 64, pp. 135–168, 2013.

[5] V. J. Shute, C. Sun, and J. Asbell-Clarke, “Demystifying computational thinking,” *Educational Research Review*, vol. 22, pp. 142–158, 2017.

[6] K–12 Computer Science Framework, 2016. [Online]. Available: http://www.k12cs.org.

[7] S. Yeni, N. Grgurina, M. Saeli, F. Hermans, J. Tolboom, and E. Barendsen, “Interdisciplinary integration of computational thinking in K-12 education: A systematic review,” *Informatics in Education*, vol. 23, no. 1, pp. 223–278, 2024.

[8] D. Weintrop, E. Beheshti, M. Horn, K. Orton, K. Jona, L. Trouille, and U. Wilensky, “Defining computational thinking for mathematics and science classrooms,” *Journal of Science Education and Technology*, vol. 25, pp. 127–147, 2016.

[9] D. Bernstein, G. Puttick, K. Wendell, *et al.*, “Designing biomimetic robots: Iterative development of an integrated technology design curriculum,” *Educational Technology Research and Development*, vol. 70, pp. 119–147, 2022. DOI: https://doi.org/10.1007/s11423-021-10061-0.

[10] L. Cabrera, D. J. Ketelhut, K. Mills, H. Killen, M. Coenraad, V. L. Byrne, and J. D. Plane, “Designing a framework for teachers' integration of computational thinking into elementary science,” *Journal of Research in Science Teaching*, vol. 61, no. 6, pp. 1326–1361, 2024.

[11] A. Miyake, N. P. Friedman, M. J. Emerson, A. H. Witzki, A. Howerter, and T. D. Wager, “The unity and diversity of executive functions and their contributions to complex 'frontal lobe' tasks: A latent variable analysis,” *Cognitive Psychology*, vol. 41, no. 1, pp. 49–100, 2000. DOI: https://doi.org/10.1006/cogp.1999.0734.

[12] E. Demetriou, A. Lampit, D. Quintana, *et al.*, “Autism spectrum disorders: A meta-analysis of executive function,” *Molecular Psychiatry*, vol. 23, pp. 1198–1204, 2018. DOI: https://doi.org/10.1038/mp.2017.75.

[13] J. M. Langberg, J. N. Epstein, S. P. Becker, E. Girio-Herrera, and A. J. Vaughn, “Evaluation of the Homework, Organization, and Planning Skills (HOPS) Intervention for Middle School Students with ADHD as implemented by school mental health providers,” *School Psychology Review*, vol. 40, no. 3, pp. 423–443, 2011.

[14] E. Pellicano, "The development of executive function in autism," *Autism Research and Treatment*, vol. 2012, pp. 1–8, 2012. DOI: 10.1155/2012/146132.

[15] M. J. Muller and S. Kuhn, "Participatory design," *Communications of the ACM*, vol. 36, no. 6, pp. 24–28, Jun. 1993.

[16] E. B. N. Sanders and P. J. Stappers, "Co-creation and the new landscapes of design," *Co-design*, vol. 4, no. 1, pp. 5–18, 2008.

[17] S. Bødker, C. Dindler, O. S. Iversen, and R. C. Smith, *Participatory Design*. Springer Nature, 2022.

[18] S. Thomaz, A. Aglaé, C. Fernandes, R. Pitta, S. Azevedo, A. Burlamaqui, A. Silva, & L.M. Gonçalves, RoboEduc: a pedagogical tool to support educational robotics. *In 2009 39th IEEE Frontiers in Education Conference*, San Antonio, TX, USA, pp. 1-6, 2009. DOI: 10.1109/FIE.2009.5350439.

[19] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, and Y. Kafai, "Scratch: programming for all," *Communications of the ACM*, vol. 52, no. 11, pp. 60–67, 2009.

[20] L. P. Flannery, B. Silverman, E. R. Kazakoff, M. U. Bers, P. Bontá, and M. Resnick, "Designing ScratchJr: Support for early childhood learning through computer programming," in *Proceedings of the 12th International Conference on Interaction Design and Children*, New York, NY, USA, Jun. 2013, pp. 1–10.

[21] J. Cross, C. Bartley, E. Hamner, and I. Nourbakhsh, "A Visual Robot-Programming Environment for Multidisciplinary Education," in *Proceedings of the 2013 IEEE Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013, pp. 445–452.

[22] E. Hamner, L. Zito, J. L. Cross, M. Tasota, P. Dille, S. Fulton, M. Johnson, I. Nourbakhsh, and J. Schapiro, "Development and results from user testing of a novel robotics kit supporting systems engineering for elementary-aged students," in *2017 IEEE Frontiers in Education Conference (FIE)*, Oct. 2017.

[23] M. Resnick and B. Silverman, "Some reflections on designing construction kits for kids," in *Proceedings of the 2005 Conference on Interaction Design and Children*, Boulder, CO, USA, Jun. 2005, pp. 117–122.

[24] C. Corbin and A. L. Strauss, *Basics of qualitative research : techniques and procedures for developing grounded theory* (3rd ed.). Sage Publications, 2008.

[25] T. Smith, “Discrete Trial Training in the Treatment of Autism,” *Focus on Autism and Other Developmental Disabilities,* vol. 16, no. 2, pp. 86-92, 2001.