Interest-Driven Creative Programming for Youth

STEFANIA DRUGA

This paper provides an overview of how we might work toward interest-driven creative programming for youth by using design-based research. I synthesize and discuss different conceptions of what it means to design creativity supports for youth, surfacing open questions and critiquing each perspective. I identify ways in which cognitivist and constructionist approaches to youth creative programming present challenges and opportunities and distinguish promising theoretical frameworks for my research. Finally, I propose designing a system that allows children to fully and fluently express themselves and their vision via creative programming in a diverse set of microworlds.

1 CREATIVE THINKING WITH AND FOR YOUTH: PAST, PRESENT AND FUTURE

For over ten years now, I have been intrigued and inspired by the following question: "How do we inspire children to build a better tomorrow and give them the tools to do it?" I tried to tackle this question in multiple ways by playing the role of educator, community organizer, researcher, tool designer, and developer. This question could be seen as contributing to a broader global effort to prepare youth with skills for the 21st century [96]. These efforts are even more critical now after more than one year of pandemic lock-down. How do we prepare our youth for the unknown? How do we escape the confines of our educational background confines when doing so? [77] How best to critically make use of ever-evolving technologies that transform every aspect of our lived experiences without falling into techno-progressive traps?[14] In this context, I see a unique and significant role for youth's creative thinking [130] as a driver for constant adaptation, life-long learning, problem-solving, inclusion, and openness. Suppose the altruistic and meaningful future argument is not sufficient. In that case, the need for creativity education for future generations is ever-present now with advances in automation and AI. It represents a key differential factor in the competitive global economy [126].

Human creativity moves hand in hand with technology and geo-political context. Guilford started the first wave of creativity research in 1958 [69]. During World War I and II, he researched intellectual abilities for the Air Force personnel. His project continued to receive generous financial support from the US Navy and Air Force. It allowed Guilford to expand the research on intelligence he conducted with youth[1]) and propose his Structure of Intellect Theory, giving birth to an entire generation of studies on creativity and learning. The second wave in creativity research was inspired by the new design guidelines for creative support tools proposed by Shneiderman in 2007 amid digital technology adoption [143]. We are now seeing the third wave of research on creativity support tools, primarily motivated by new technological affordances in the space of machine learning and AI technologies [62].

For youth, creativity is driven by imagination and play and is both practical and conceptual. What youth create today depends on the tools and materials that are accessible to them. From Froebel gifts [3] to LEGO Mindstorms [5] and creative learning tools such as Scratch [112], notable efforts have been made in creative learning and creative coding for youth. These initiatives represent a counter-culture response to the dominant instructional education instituted during the industrial revolution [64]. Creative learning ambitions flourished primarily outside of traditional school classrooms. The constructionist vision focused on two critical aspects of creativity for children: allowing them to tinker, construct, debug, and modify ideas [137] and encouraging them to collaborate in communities of practice [26, 80].

[1]these were Stanford-Binet Intelligence tests [147]

Author's address: Stefania Druga.

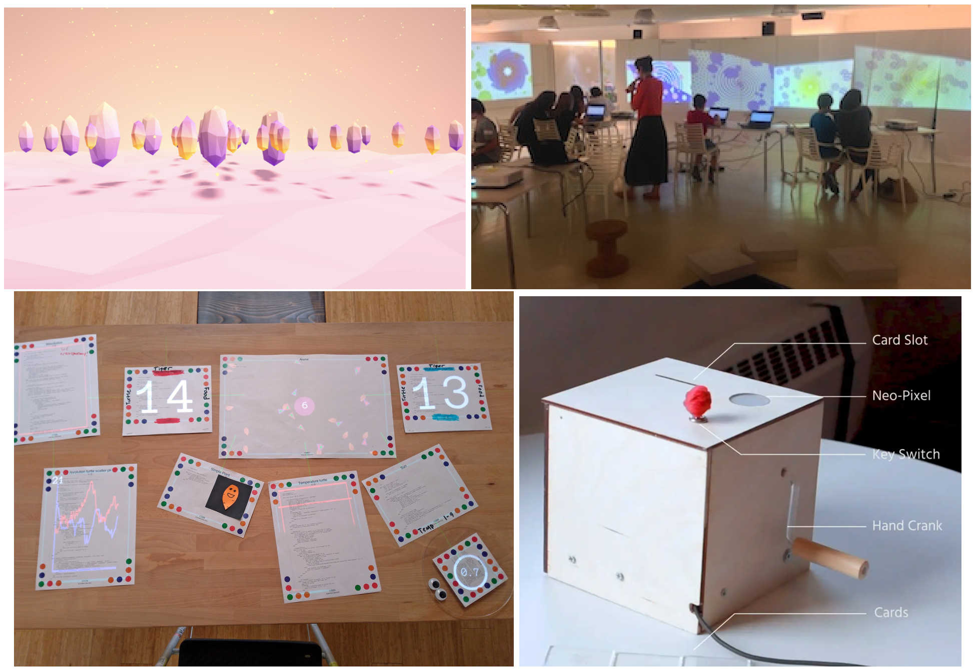


Fig. 1. Examples of creative coding in the wild: 1.Legends installation [37], 2.Museomix [7], 3.Dynamicland [2], 4.Humming Box [38]

With time the success of these projects also drove change in the way creative thinking and creative coding are taught in schools, with more initiatives focused on project-based learning and coding in schools [131, 136]. However, questions remain as to how best to balance structure vs. agency in programming for youth [32]. As youth are now growing up with computers, apps, and AI, our framing, understanding, and development of intelligence and creative thinking are again under scrutiny. Recognizing that every child is born with immense natural talents [63] and innate creative potential [153], how do we design new learning opportunities and tools for creative thinking that allow children's creativity to flourish in an era of constant technological change and consumption? Based on research I conducted in the past four years on AI education and literacy for youth [52] , I see a unique opportunity for designing new forms of creative coding for youth that involve authentic [61], personalized, and dynamic creative collaboration with virtual agents. With this current research proposal, I aim to frame creative coding for youth from a stance of epistemological pluralism [152], recognizing "the validity of multiple ways of knowing and thinking ." I believe that future programming tutorials should balance introducing children to new computing topics ("level up") while also leveraging children's creative engagement as a way to enable self-expression and to re-imagine computing norms.

Engaging in creative expression with code is a social phenomenon where learning does not happen in isolation. Children are working in a social context even if they create individual coding projects. In this context, I believe it is essential to take an ecological approach when developing my research contributions and consider my work's short and long-term implications via an ecological systems lens [33]. To this end, I pose the following research question: *How might learners engage in self-directed creative learning coding activities when collaborating with an adaptive system?*

The first step in articulating and this research direction is to understand different creative processes children use when creating programs with a limited dynamic vocabulary based on their interests, working both on open-ended [150] vs. closed-ended projects [100]. I wish my research contribute rich and "thick" [65] descriptions of multiple new ways in which youth now collaborate with AI for creative coding and propose new pathways to engage diverse learners in creative thinking and coding (see fig.1. I contemplate the future of creative coding for youth as a place where creativity is a language on its own, just like Romanian, Python, or Math. It creates a vocabulary of heuristics, objects to think with [151] and processes that inherently allow children to express their creative ideas with code freely. Imagine a collection of dynamic code grammars and interfaces adapting to children's skills, wants, and content interests. In this work, I aim to leverage children's voices and creative processes through co-design at every step of the process.

We now recognize that knowledge is advanced rather than accumulated; it is dynamic, taking in new and discarding outdated "epistemic artifacts."[139] Children are uniquely positioned to help advance to move from an information society to a creative society [133]. To help inform the design of this study, I consider below creativity supports, cognitive vs. constructionist theories of programming and present a series of design guidelines for interest-based creativity supports for youth .

2 BEYOND CREATIVITY SUPPORT: EXPRESSION FLUENCY

Creativity Support Tools (CSTs) were first introduced by Ben Shneiderman in 2002, building on creative problem-solving methodologies [40] and social creativity [41]. He defined CSTs as tools that enable people to collect, relate, create and donate digital works. Shneiderman noted the gap in creativity research to include digital artifacts and encouraged the designers of programming interfaces, interactive tools, and rich social environments to enable more people to be more creative. The diversity of existing definitions of creativity is a testament that we only understand a partial approximation of the complex cognitive and behavioral system at play when people engage in creative acts [142, 143]. As a result, the collection of CSTs currently being developed in the Human-computer interaction (HCI) community is highly diverse, and so is the type of creative support they provide.

A recent systematic literature review study sheds light on the diversity of approaches to designing CSTs [62]. The study classification of CSTs found six significant categories of support in the creative process organized around: pre-ideation, idea generation, evaluation, implementation, iteration, and reflection (see table. 1). In their systematic review, Frich et al.showed that current CSTs are by enlarging driven by what we can measure or what we can build, and fail to become widely used or long-lived, with only 4% of the tools they analyzed is still available to the public. Their analysis of 143 ACM papers between 1999 and 2018 revealed that a majority of CSTs are digital (92%), low-fidelity is more recent (since 2009) and represent only 17% of tools, most of the tools focus on ideation (45%) and implementation (41%), and evaluation (18%), almost half of the tools are low complexity (48%). From the sampled CSTs, 38% did consider the level of creative experience or expertise of their target audience, and works supporting iteration and reflection are only marginally present.

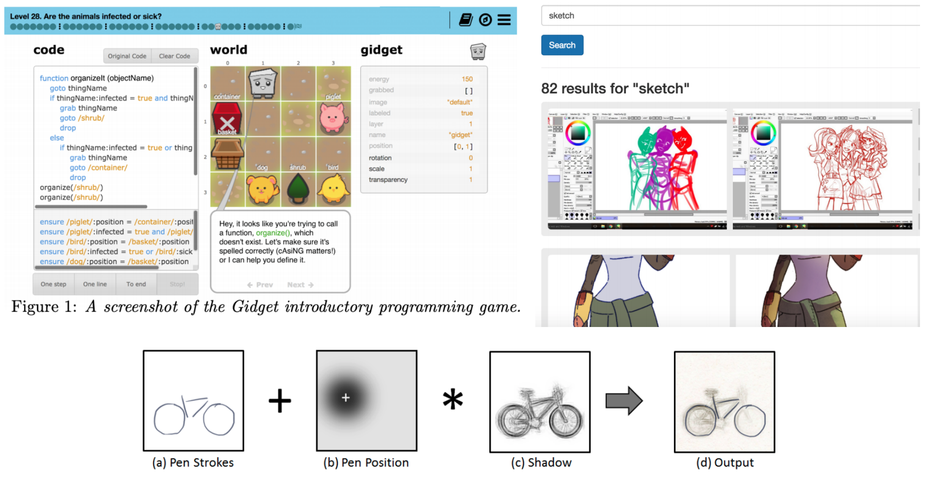
To define conceptions of "support" in programming for youth, I draw on the previous systematic literature review and present a series of recent CSTs for youth creative learning (see table. 1). Based on this review, I describe several cross-cutting open questions and future directions below.

2.1 Cross-cutting Open Questions and Future Directions

2.1.1 Personalization of Creative Process Support. Barbot et al. proposed an "optimal-fit" view of creativity. The creativity of a product depends on the fit between a creative task characteristic and a person's profile of resources

Category	Level	Definition	Example Projects
Complexity	*Low*	One or two features	Let's chance[46]
	Medium	Semi-complex system	danceON[125]
	High	Entire system	Mosaic[88]
Maturity	*Lo-fi mock-ups*	Paper/WoOz	Yolo[12]
	Prototype	Working prototype	Codeopticon[70] StoryDrawer[155]
	Public release	Final product	Scratch[8]
Creative process	*Pre-ideation*	Background research	Idea Wiki[4]
	Idea generation	Ideation or generation	Yolo[12], Let's chance[46]
	Evaluation	Critique of outcomes	Scratch[8]
	Implementation	Support building	Shadow Draw[103]
	Iteration	Support iterating	Mosaic[88], Scratch[8]
	Reflection	Support reflection or documentation	ScratchEncore [60] Idea Wiki[4]
User Group	*Novice*	New learners	Scratch Microworlds[150]
	Casual user	Medium experience	Yolo[12]
	Expert	Advanced experience	3Buddy[110]
	Unspecified	Not defined in the tool	Mosaic[88]
Collaboration Support	*Individual*	Focus on one person	AgentCubes[6] Drawing apprentice[43]
	Collaborative with people	Support social collaboration	Scratch[8], Dynamicland[2] MOOSE crossing[34]
	Collaborative with technology	Co-creative with system/agent	Creative Sketching[84] 3Buddy[110]

Table 1. Classification of HCI Creativity Support Tools (CSTs) for youth building on review from Frich et al. [62]



```
function organizeIt (objectName)
  goto thingName
  if thingName:infected = true and thingN
    grab thingName
    goto /shrub/
    drop
  else
    if thingName:infected = true or thing
      grab thingName
      goto /container/
      drop
organize(/shrub/)
organize(/shrub/)
```

```
ensure /piglet/:position = /container/:posit
ensure /piglet/:infected = true and /piglet/
ensure /bird/:position = /basket/:position
ensure /bird/:infected = true or /bird/:sick
ensure /dog/:position = /basket/:position
```

Hey, it looks like you're trying to call a function, organize(), which doesn't exist. Let's make sure it's spelled correctly (cAsiNG matters!) or I can help you define it.

Figure 1: *A screenshot of the Gidget introductory programming game.*

Fig. 2. CST Optimal Fit Process Examples: Gidget [100] (left), Mosaic [88] (right) and ShadowDraw [104] (bottom)

underlying creativity in that task [18]. I present three different CSTs for youth that tackle this question of optimal fit differently: Gidget, Mosaic, and ShadowDraw.

Gidget: Personalization of CST feedback. Michael Lee built and evaluated a videogame (Gidget) that teaches programming. One of his findings is that this game could improve student retention by making the programming language "personable." He created an avatar for the computer, which looked sad when encountering an error, and happy when the program worked correctly. His findings showed that learners engaged in the game more when using the game version with the personable avatar [100].

Mosaic: Curator vs. Creator Mosaic is an online creative community where sharing process, rather than showcasing finished projects, is the primary method of sharing creative work. Illustrators are encouraged to document and share work-in-progress snapshots [88]. This design direction builds on the work of Dow, who shows that parallel prototyping in creative work leads to better design results and increased self-efficacy [51].

ShadowDraw: In-Situ Adaptive Guidance ShadowDraw automatically blends relevant images from an extensive database to construct the shadows. Based on this dataset, the system dynamically adapts to the user's drawings in real-time and produces suggestions accordingly. When authors tested their approach in user studies, they found that creators who used their system produced more realistically and proportioned line drawings and wished to continue using the tool after the study [104].

From these examples, we see the designers tackled the issue of creative process optimal fit in different ways, by either trying to personalize the support across affective, documentation, and curation or skill dimensions. These examples point to new possibilities in terms of technical affordances for CSTs' personalization and providing creators with the choice of the type of creative support they might want to receive, like in the Mosaic example.

However, personalization CSTs could come at a cost; the pitfalls in personalizing mobile apps for children serve as a cautionary tale. The highly personalized app identity for youth is multifaceted, outward-facing, and constrained primarily by the programming decisions of the app designer [63]. Such approaches risk locking in children's creative potential by framing creative expression through the lens of the designer's creative values [95]. In order to avoid such pitfalls, future CSTs that want to support a diverse community of creators should design and decide *with* them how best to support a wide variety of creative practices.

2.1.2 *Interest-Based Programming CSTs.*

As pointed out by Frich et al., much of CSTs are disconnected from the creators' daily practices [62]. In the context of computing education, youth want their programming learning to be authentic [141]. Authenticity in creative coding could involve providing the proper media supports like in the case of the danceON project [125] and the opportunity to work on microworlds with curated programming activities, such as fashion or music [150] (see fig.3).

Lave and Wenger point out that we compare ourselves to experts in the community of practice that we hope to join [97]. In this context, there is an opportunity to design CSTs that support a wide diversity of programming interests and create, as Dhariwal Shruti and Manuj describe it: "clay-like computational material that children can dynamically shape and tinker within real-time to imagine and construct countless creative, generative, artistic, mathematical, playful, and serious possibilities of probability in their minds and their creations."[46] By creating new programming building blocks, we will build expertise around creative coding. This would also encourage the growth of programming communities which are authentic, inviting, and supportive for a diverse group of learners and creators [54].

2.1.3 *Many Ways of Asking & Receiving Support.*

In the field of Human-Robot Interaction(HRI), several studies have shown that having embodied agents, such as robots or connected toys, could lead to increased engagement and learning

(b) Black Lives Matter

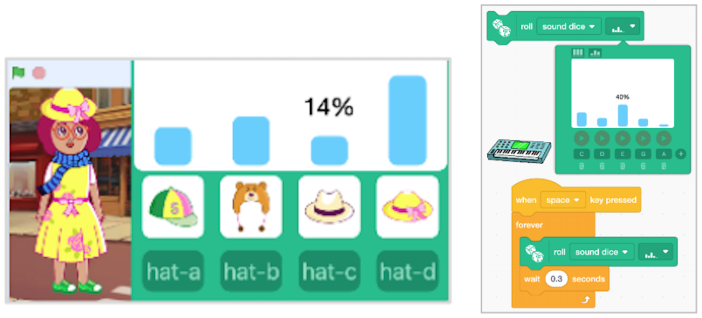


Fig. 3. Interest-Based Programming CSTs: danceON BLM project [125] (top), Let's Chance playful probabilistic blocks [46] (bottom)

Fig. 4. Many Ways of Asking & Receiving Support: StoryDrawer [155] (top), Tega robot (bottom) [123]

effects for youth [11, 123]. Incorporating physical components in CSTs, such as StoryDrawer providing an idea button [155], could make it easier for children to ask or receive support from the system (see fig. 4).

In digital sketching [84] and game design [110], Karimi and Lucas show different ways for creators to ask for help when interacting with AI-powered CSTs. They provide slider controls where creators can vary the degree to which

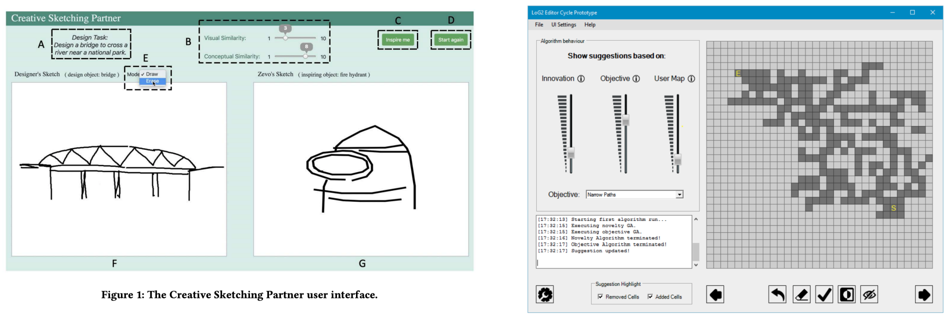

Creative Sketching Partner
A
Design Task: Design a bridge to cross a river near a national park.
B
Visual Similarity: 1 3 10
Conceptual Similarity: 1 8 10
C
Inspire me
D
Start again
E
Mode ✓ Draw
Erase
Designer's Sketch (design object: bridge)
Zevo's Sketch (inspiring object: fire hydrant)
F
G
LoG2 Editor Cycle Prototype
File
UI Settings
Help
Algorithm behaviour
Show suggestions based on:
Innovation
Objective
User Map
E
S
Suggestion Highlight
Removed Cells
Added Cells

Figure 1: The Creative Sketching Partner user interface.

Objective: Narrow Paths

```
[17:32:13] Starting first algorithm run...
[17:32:15] Executing novelty GA.
[17:32:15] Executing objective GA.
[17:32:16] Novelty Algorithm terminated!
[17:32:17] Objective Algorithm terminated!
[17:32:17] Suggestion updated!
```

Fig. 5. Many Ways of Asking & Receiving Support: Creative Sketching Partner [84] (left), 3Buddy Game Design [110] (right)

they want the system to support them (i.e., visual similarity 10%). These two CSTs also control the various aspects of the creative process support (i.e., visually vs. conceptually) (see fig. 5). These examples suggest different ways in which children might work together with their CSTs and engage in co-creativity with intelligent systems [83].

When peaking into the world of CSTs for youth, we recognize several tensions around providing support without locking in creators. I identified opportunities around designing CSTs that provide an optimal fit to creators' skills and needs, integrate creators' interests and media preferences, and make it easy for creators to ask and receive help from the system.

2.2 Looking forward: a story about CSTs in the wild

To better understand how CSTs apply to the phenomenon I am aiming to study, let us imagine the following situation: Marie, a 7.6 years old girl, interacts with Ott, her digital creative coding partner:

- Ott: "What is on your mind, Marie? What would you like to make today?"
- Marie: "I was wondering why tigers fly. Do you know ?" [2]

Depending on our definition of creativity, we might choose to interpret this situation in many different ways and design a system that would respond accordingly to our inherited views of the creative process and Marie's intentions and mental process.

Based on Guilford's structure of intellect theory and definitions of creativity (p. 82), learning (p. 54), and information (p. 55), we would try to assess if Marie's question represents a change in her behavior based on recent information she discovered. She might have processed this information consciously or subconsciously. We would design Ott to ask further questions to determine if her question's semantic content is a unit, transformation, or implication resulting in divergent production, convergent production, memory retrieval, cognitive processing, or evaluation. Ott would ask different questions of figural, symbolic, semantic, and behavioral nature over time to triangulate via factorial analysis Marie's creative personality (p. 77) and decided how best to respond [69]. Well, all of that is not very easy, so maybe Ott would ask Marie to use circles as a prompt for different drawings [3], draw for 2 minutes and then evaluate her drawings based on fluency, flexibility, originality, and elaboration. Marie would probably say she does not care about circles, and she wants to figure out why tigers fly.

[2]This question is inspired by many other similar questions children expressed in my youth workshops.
[3]This is an actual creativity measure from Torrance tests [149]

The goal of using this anecdote is to make a significant point. Our current definitions of creativity, either as divergent production, lateral thinking, contextual composition, possibility thinking, fail to consider how sometimes we are not able to assess if children are in the process of developing their creative voice. Marie might be speaking out loud some of her imaginative p-prims [50] or try to engage in pretend-play. When our theoretical frameworks establish measures of novelty and originality determined by adults, childrens' bids for play and explorations are ignored or suppressed.

A valuable inspiration for how we might approach this phenomenon differently comes from the philosophy and pedagogy of Reggio Emilia, emphasizing listening, documentation, and critical reflection [59]. The following principles guide their community:

- Creative values are strength & power of kids, pedagogy of listening.
- Creative relationships are attentive and respectful.
- Creative environments are physical and emotional.
- Behavior and dispositions matter, support holistic learning and creative thinking.

Current efforts on creative learning for youth are driven by what we can measure, like the prompt of drawing circles from the above example. Our current creativity support designs fail to support creative expression in the wild. There are still many valuable lessons we can learn from existing efforts in the field of creativity support. However, I believe it is essential to acknowledge how little we know and the limitations of our current understanding of creativity.

A future agenda on creativity with and for youth will need to actively and genuinely respect and listen to children's voices in the process of co-designing new forms of creative coding in digital, physical, and mixed mediums. Beyond the metaphor of "support," we have an opportunity to embrace the metaphor of "collaboration" that positions both child and agent on equal footing in the act of creative expression (with code and beyond) [45].

3 COGNITIVIST VS CONSTRUCTIONIST APPROACHES TO YOUTH PROGRAMMING

Cognitivism is a learning theory that focuses on the processes involved in learning and knowledge development rather than studying behavior. Cognitivism focuses more on the internal processes of learning and the connections in people's minds during learning. Cognitivism aims to open and understand "the black box" of the mind by focusing on comprehension, abstraction, analysis, synthesis, generalization, evaluation, decision-making, problem-solving and creative thinking. The learner is viewed as an agent that processes information and develops new knowledge in a continuous process. Knowledge in cognitivism is formalized as schema or symbolic mental constructions, and learning is defined as a change in a learner's schemata.

Ulric Neisser was the "father of cognitive psychology" and an advocate for ecological approaches to cognitive research with the publication of Cognitive Psychology (1967), where he presents research concerning perception, problem-solving, attention, remembering, and pattern recognition [116]. Jean Piaget proposed the constructivism theory, which defined *learning* as a process of association between new information and stored knowledge. Jean Piaget first identified two processes by which these associations enter memory, assimilation and accommodation [128]. He conducted pioneering research with youth and put forth his four stages of cognitive development (sensorimotor, preoperational, concrete operational, and formal operational), which is still being used and referenced when learning is discussed and researched.

In K-12 education, one of the theories of cognitivism that is widely used is Bloom's taxonomies of learning objectives [27]. Bloom and his colleagues claimed that there are three critical domains of learning: cognitive(thinking), affective (feeling) psycho-motor(doing). Each of these domains is related to developing different learning skills or ways of learning for youth. Cognitive approaches to learning in k-12 programming education encourage focusing on teaching learners how to learn, developing more robust or new mental processes for future learning, and developing a more profound and constantly changing understanding of concepts and ideas.

3.1 Papert's constructionism v.s. Piaget's constructivism

> "We understand "constructionism" as including, but going beyond, what Piaget would call "constructivism". The word with the v expresses the theory that knowledge is built by the learner, not supplied by the teacher. The word with the n expresses the further idea that this happens especially felicitously when the learner is engaged in the construction of something external or at least shareable, a sand castle, a machine, a computer program, a book. This leads us to a model of using a cycle of internalization of what is outside, then externalization of what is inside." – Seymour Papert, 1991 [122].

In his seminal 1981 book, Mindstorms, Papert proposes using computing to create different micro-worlds for programming learning based on children's interests (i.e., "Mathland"). Papert suggested that when learners make a mistake (a bug) in their Logo programs, it would likely reflect a misconception in their thinking about the underlying mathematics. He proposed that when learners debug their programs, they also correct their thinking *bugs* [120]. His vision and Logo, the first programming language for kids he invented [121] inspired generations of constructionist designs and projects.

3.2 Cognitivist examples

Cognitivism defines *programming* as the act of assembling a set of symbols representing computational actions. Learners can express their intentions to the machine using these symbols [72]. In this context, a cognitivist approach to programming for youth would mean focusing on supporting learners to develop new or more robust mental processes for future learning and developing a deeper understanding of computing concepts [50, 73].

Many cognitivist efforts in computing education research are designed around the hypothesis that a primary barrier in learning to program lies in the mechanics of writing programs and propose frameworks for programming understanding [92, 106, 118], tracing [117], debugging [66, 89].

Letovsky proposed the basis of program understanding and described how it involves processes at different levels of temporal resolution [106]. In order to support learners in overcoming their barriers in programming [92], cognitivism created interactive representations of program behavior (i.e., [47, 91]).Benedict du Boulay also contributes the idea of presenting learners with a notional machine (see fig.6), defined as a mental model of what a computer can do [28]. Notional machines as abstract execution models, have risen to the point of gaining acceptance as a useful devices in computing education [47].

Recent studies showed that developing a notional machine requires the ability to both read and write programs. Raymond Lister suggested that reading programs is a necessary developmental stage to writing programs and predicting the computer's behavior for a given program [107].

A growing body of work is showing the effectiveness of well-designed educational games to support student learning in computing. The first of these to show significant learning benefits was Wu's Castle which taught iteration [56]. The results from Michael Lee and Amy Ko's work on Gidget are promising [100]. There is strong evidence that asking students to play an educational video game is more likely to lead to learning than doing a programming assignment [102]. More recently, Lee has also shown that auto-generated game levels increase novice programmers' engagement [98] and that providing animated hints help novices complete more levels in an educational programming game [101].

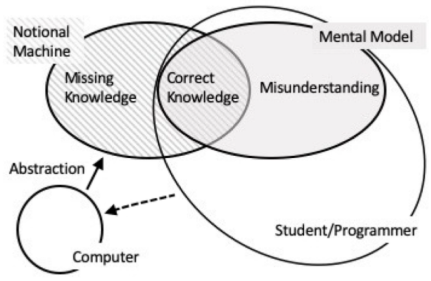


Fig. 6. Notional machine: an abstract model of an execution environment. Source image Dickson et al 2020 [47]

3.3 Constructionist examples

Constructionists like Papert, Resnick, and Eisenberg describe *programming* as a process where the learner's interaction with her program could transform how she thought about the program's domain [121, 134]. The process they describe is much like what Bereiter and Scardamalia had also seen working when a writer interacted with her text [139]. Resnick also emphasizes the importance of *fluency* in creative coding, defined as: "the ability to express one's-self ideas with code in a similar way we do it with the written text" (p. 48 [136]).

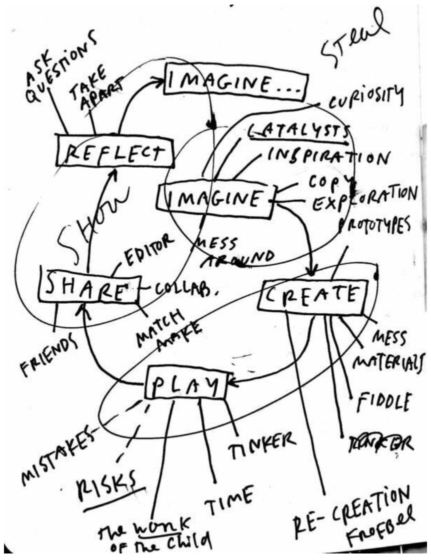


Fig. 7. Creative Learning Spiral presented by Mitchel Resnick in 2007 [132] remixed online [9]

Constructionist approaches to programming learning have resulted in the development of widely used tools such as Scratch [112], which is the largest platform for youth coding with more than 68 million users and is being used in formal and informal learning settings. Resnick invites designers, educators, and parents to consider creative learning with programming and calls for using coding to sow the seeds of a more creative society [133]. He envisions creative coding as an iterative cycle where learners imagine, create, plays, share, reflect (see fig.7) [132]. His approach to programming for children is inspired by how children naturally learn in kindergarten and Froebel's approach to learning [129].

Constructionism also inspired and contributed to the birth and development of the maker movement [24, 26], from initial LEGO Mindstorm construction kits to the a myriad of physical computing prototypes [25]. Yasmin Kafai was one of the pioneers who expanded the constructionist model into programming for science learning [81]. Over the years, Kafai has had children build programs to learn and create in various media for a variety of learning purposes. She had students construct video games [80], craftwork [82], and e-textiles [78]. More recently she called for moving from focusing primarily on computational thinking, and consider computational participation in K-12 education [79].

3.4 Best of both worlds?

Both cognitivist and constructionists recognize programming for youth as a form of fluency where learners can express their ideas and achieve their goals with code but take different approaches to achieve this [72, 131]. The difference between the focus on the logic model in cognitivism and the focus on tinkering, craft, and creative expression in constructionism often lead to very different approaches in designing tools and learning activities for computing learning. Karen Brennan studied the issues of balancing structure and agency in computational creation, in and out of school [1, 32], recognizing existing challenges in defining computational thinking in a way where learner's agency is respected when including creative coding in classrooms [30].

The Scratch success is a testimony for the importance of emphasizing and designing for children's agency. It showed that collaborative online communities are powerful engines for children learning. However, it also showed that sometimes collaboration comes at the cost of originality. Most importantly, constructionist approaches to programming learning recognize the importance of shifting power from teachers to learners and providing multiple pathways to encourage computational fluency and personal expression. The importance of putting the learner's interest at the center is starting to trickle in the cognitivist world as well. However, there is much work to be done to change the culture of learning of the settings in which such work is carried. An authentic computing pedagogy focused on learners' interests and passions would have to suspend the current disbelief in constructionist approaches, often expressed as "Looks like they are having fun but are they learning?"[127]

Opportunities to engage both social and cognitive aspects of children's forays into the computing world are ever needed and crucial now that computing permeates all aspects of our lives. The current tensions between cognitivist and constructionist approaches to programming learning are reminiscent of an older and more significant divide between humanities and science. C.P. Snow wrote the Two Cultures book based on his experience as a British science advisor during World War II. In his book, he describes the split he saw between the humanities and science views of the world [144].

He argued that scientists needed to understand more of the moral and cultural issues that the humanities perspective offered. He critiqued humanities scholars for not appreciating the power and insights of science. Snow saw a similar split because too few people learn to compute. In order to expand and increase participation in computing for youth, their families, and communities, we need to question the existing norms of computing learning for youth. This would allow our community of scholars to address equity issues, expand participation, promote a critical understanding of new automated technologies. To this end, both personally relevant explorations and high-quality, engaging learner-centered instruction are necessary and should be elegantly balanced.

4 DESIGN GUIDELINES

Based on the synthesis of existing CSTs work, with a primarily constructionist lens, I build on the design principles for tools to support creative thinking proposed by Mitchel Resnick, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike Eisenberg and propose the following designs guidelines for future CSTs for youth programming:

(1) **Support Authentic learning**: encourage children to work on relevant topics and engage in authentic creative practices with their family and community. The tool should encourage children to work on projects they are passionate about and provide ways to share their work with their community [42, 48, 49, 58, 60, 150**?**].

(2) **Support Experimentation**: support children to start by exploring, experimenting, and thinking with digital and physical materials. The CST's interface and activities should be designed to encourage rapid experimentation and prototyping cycles [94, 113, 137].

(3) **Low floors, Wide Walls, and High Ceilings**: include elements and features that are easy for kids to understand(low floors), but general enough to support diverse uses (wide walls), and additional ways for children to customize and define new features (high ceilings) [120, 135].

(4) **Design for Collaborative Scaffolding**: allow children to do both individual and collaborative activities (i.e., remix a prior curated project, take turns with tool support system)) [31, 68, 80, 138].

(5) **Design for Playful Debugging**: children should be able to learn to break and fix things on the platform, provide ways for them to do playful debugging (e.g., fix a prior broken project, trick the tool support system) [74, 90, 99, 100].

(6) **Choose Black Boxes Carefully**: one of the most critical decisions for CST design is designing the"primitive elements" that users will manipulate. This design will determine what ideas users can explore with the tool – and what ideas are abstracted from view. One promising direction in the programming language design for children is to create dynamic primitives that can adapt based on child needs, learning, and project domain [71, 75, 93].

(7) **Explainable system**: allow children to see how the agent (or the computer) sees the world or makes decisions in the tool, so they can better understand how it works and set up their expectations accordingly [13, 19, 20, 109, 146, 154].

(8) **Multimodality**: allow children to express in different modalities (e.g., drawing, speaking, building something)how they imagine the future of these technologies and build features of the platform to match or challenge their expectations [39, 76, 111].

(9) **CSTs an "Object to think with"**: encourage reflection on the artifact and process and enable children to document various stages of their creative process and also discover other work in progress [10, 105, 140, 145].

(10) **Design for designers**: enable others to design, create, and invent things using the tool. Support sketching-like interactions with the tool, encourage re-purpose and interface modifications, support creative self-efficacy [16, 17, 21, 22, 85, 115, 119].

5 PROPOSED STUDY DESIGN: PWOZ WITH CODE PLAYGROUND

Code Playground [4] is an interest-driven creative programming platform that aims to enable children (ages 7-12 years old) to engage in self-directed creative coding activities. With the iterative and participatory design of this platform,

[4]working title

I aim to answer the following research question: *How might learners engage in self-directed creative learning coding activities when collaborating with an adaptive system?*

5.1 Theoretical framing: Creative self-efficacy & Theory of Mind

Recent CSTs in the field of equitable computational tools for artists [87] or embodied AI-art collaboration [15] are building on the cognitive theory of creativity support [44]. This theory aims to understand novices and their creative barriers: fear of failure, time commitment, and lack of skill. To date, this theory has been used for the design of more than 18 CSTs and it may present opportunities to understand children's creative process better. However, I believe it approaches children's creative expression from a deficit perspective and is more suitable for close-ended CSTs

> "Innovativeness requires an unshakable sense of efficacy to persist in creative endeavors when they demand prolonged investment of time and effort, progress is discouragingly slow, the outcome is highly uncertain, and creations are socially devalued when they are too incongruent with pre-existing ways."
> (Bandura, 1997, p. 239, [17])

In turn, the creative self-efficacy theory [148], derived from Bandura's more general concept of self-efficacy [17], allows CST designers to focus on children's beliefs that he or she can successfully perform in a specific creative process. This approach also builds on the "mini-c" creativity from the Four-C model of creativity. The "mini-c" describes the level of creative production at which children are generating new knowledge for themselves [86]. My choice for self-efficacy theory also builds on my reflection of the future of computing learning for youth and the importance of considering both social and cognitive aspects of learning presented in the previous section.

We also know children's collaboration with other children or adults is very much supported by their theory of mind and ability to understand and assume what the other person might be thinking [35]. A recent study proposed a mutual theory of mind framing when designing a natural long-term interaction (12 weeks) between a virtual teaching assistant and a class of 376 students from an online master of computer science program at Georgia Tech. The study found that students' perception of voice assistant anthropomorphism and intelligence changed significantly over time and emphasized the need for adaptability and the importance of understanding human-human social interactions as a way to improve human-AI interaction [154]. There is also prior work done with social robots that point in the same direction [67].

In our recent research, we analyzed longitudinal interaction with various intelligent technologies. We found that programmability plays an essential role in shifting the agency from intelligent technology to children. Children's hypotheses about AI agents the ways they test them were highly diverse [53].

Building on our prior work on creative coding and AI literacy for kids, as part of future research, I aim to extend the development of the Cognimates platform I created and developed since 2018. The future goal is to include various possibilities to collaborate with autonomous agents (called Cognimates) in the context of curated micro-worlds that will be designed to match the diverse children's interests, such as intelligent games or drawing with code. The first step in this direction is to co-design a series of prototypes of the platform interface and coding activities with children. Moreover, I see a unique opportunity in engaging children themselves to play the role of the Cognimate for other children and therefore emulate and embody the creative and learning supports they would like to receive(if any). I plan to build a simplified version of the Cognimates platform for this study design, which is titled Code Playground in the context of this paper.

5.2 Code Playground Platform

In this section, I describe the Code Playground platform proposal and address the design guidelines I identified in this paper, which will be referenced as D1-D10.

5.2.1 Interaction flow. Imagine Jay, an eight-year-old girl, visiting the Code Playground platform for the first time. When landing on the welcome page (see fig.8) where she will be able to explore and discover a curated collection of coding projects grouped in the following categories: choose your adventure, smart games, and interactive storytelling (D1, D2, D3). Jay would be able to interact with any of the projects ("Modify a project" option D4, D5) before deciding if she wants to modify them, or she would be able to start her project either by getting an idea prompt ("Give me a prompt" option D8, D9) or not ("DIY" option D10). Before Jay starts coding, she will be prompted to select an agent she could collaborate with. Jay would be able to pick different avatars showing a robot, a pet, or a child or decide to create her design for the agent avatar (see the right image in fig 8, D10). Once Jay will select her agent, she will see the Code Playground programming interface (see fig 10). The agent would also be present on this interface and will quickly tell her with text on the screen; it is there if she wants to collaborate, pointing her to the options at the bottom of the page (see box E in fig 10, D7). Jay would slide the robot button on the page to ask for ideas, hints, help, wild card effects, or code examples from other children while coding (D1, D4, D9). She will also choose not to have any interaction with the agent and place the robot button on the stop option (D10). After Jay finishes her project or when she decides to stop, she would be prompted to provide quick feedback on her project and her agent (see fig.11, D10). She would also record reflections at any steps of her programming process or the end (see fig. 12, D10.

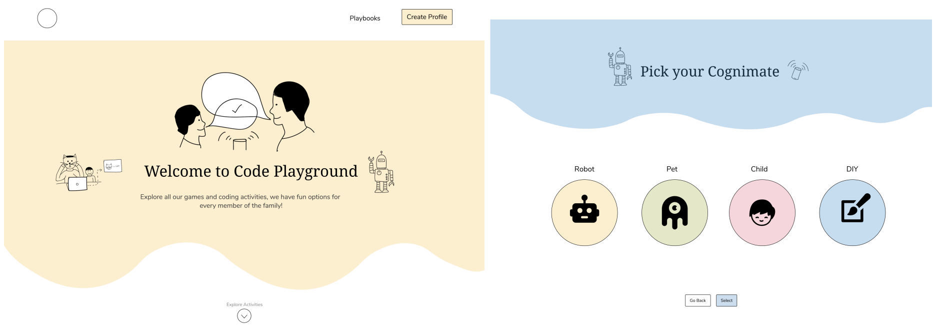

Playbooks
Create Profile
Welcome to Code Playground
Explore all our games and coding activities, we have fun options for every member of the family!
Explore Activities
Pick your Cognimate
Robot
Pet
Child
DIY
Go Back
Select

Fig. 8. Code Playground landing page(left) and agent selection page (right)

5.3 Proposed Study Procedure

In the following, I describe my proposed study procedure by describing the selection and participation of children, methodology, study materials, data collection, and analysis.

5.3.1 Selection and Participation of Children. For my user study, I plan to recruit a diverse population of children from the Seattle Area. In total, I aim to recruit 20 children of ages 7-12 years old who will work together in 10 pairs as wizards and creators. Each child will complete an intake questionnaire (Demographics, NARS, PSS) and be asked to describe her or his programming experience and will be randomly assigned to a wizard or creator role. The study will take place on the University of Washington campus.

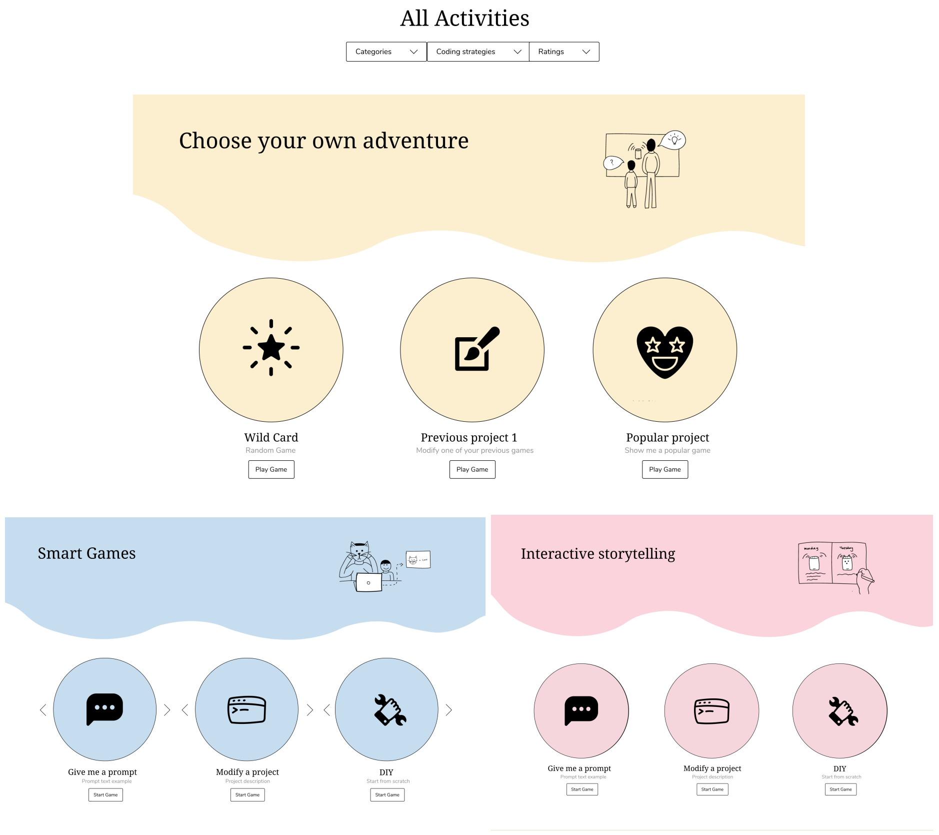
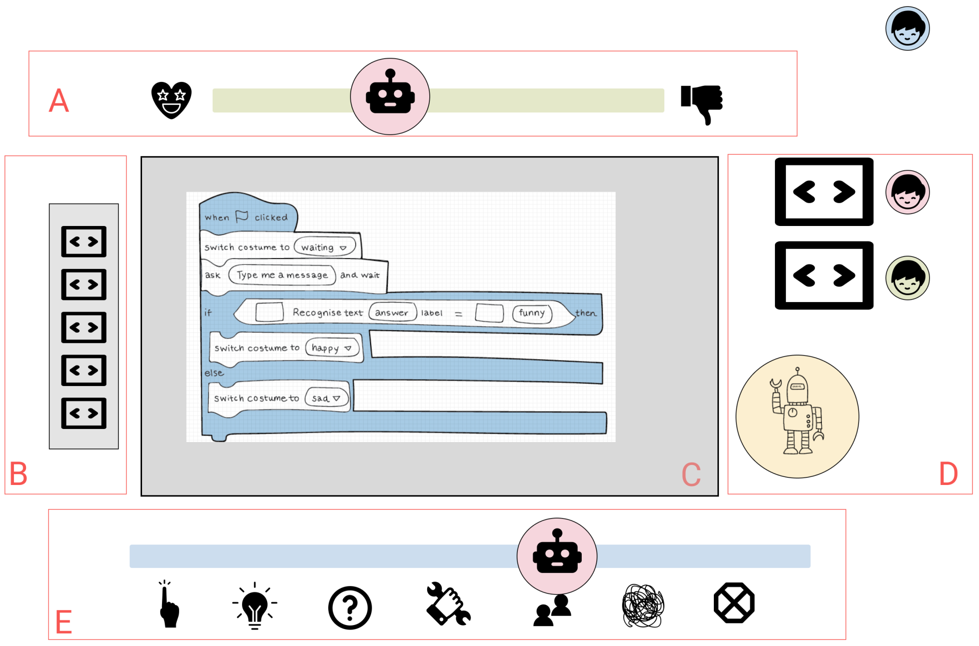


Fig. 9. Activities pages

5.3.2 Methodology. Stemming from a human-centered approach, I will use a Participatory Wizard of Oz (PWoz) interaction method that engaged children as wizards or creators in a uniquely transparent interaction (see fig.15). Björling et al. recently proposed the PWoz method, where the typical Wizard of Oz (WoZ) method where the researcher operates a system without the participant's awareness. Similar to the the suggested framework of Druin [55] where child participants were placed in multiple roles, e.g., user, tester, informant Borling proposes and user study design where the participants themselves could either play the role of the wizard or user [23].

5.3.3 Study Materials: Creative Micro-worlds, Creative Prompts, Self-Perception Game. In this section, I present the creative activities I will use in my study and the perception game we used to measure children's shift in perceptions.

Creative Micro-worlds To help children imagine possibilities without fixing a starter program, I propose to curate themed *micro-worlds*, such as "Interactive storytelling", "Smart Game", or "Choose your own adventure" 9. A micro-world comprises just a few blocks carefully chosen to express programs that are sufficiently varied to engage kids' creativity. For example, Figure 13 shows a "Conversation" micro-world consisting of commands that prompt a user for input, switch a character's costume, speak something, and check input for a condition. This small set of commands can be assembled into a diversity of programs, such as the example in Figure 14, which also invokes a text classifier using blocks



```
when [flag] clicked
switch costume to (waiting ▽)
ask (Type me a message) and wait
if < Recognise text (answer) label = (funny) > then
    switch costume to (happy ▽)
else
    switch costume to (sad ▽)
```

Fig. 10. The Code Playground programming interface is composed of 5 main components: A. Agent Evaluation, B. Coding Blocks Library, C.Coding Area, D.Agent Response Area, E. Agent Actions Menu.

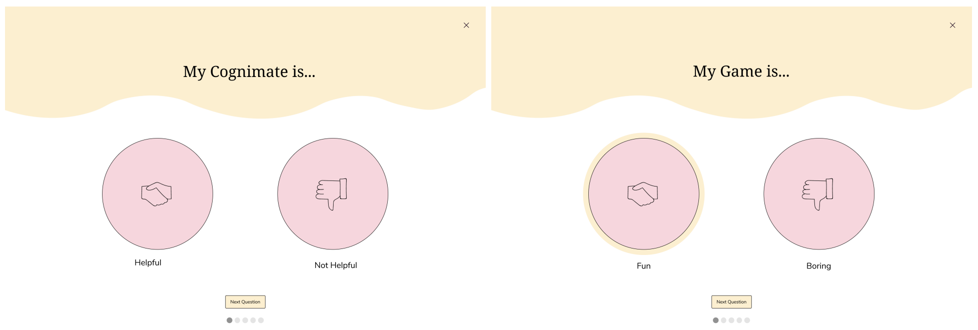


Fig. 11. Feedback and self-evaluation pages

added to the micro-world when the child is ready. In our preliminary work, when I have given such micro-worlds to children instead of the full platform, they had an easier time constructing valid programs, imagining potential behaviors for characters, and motivating interest in training the character using Code Playground's AI features to detect more nuanced phrases such as back-handed compliments or ironic remarks [52]. *Creative Prompts* Suppose children do not want to start by modifying an existing project but do not have a specific idea. In that case, we will present them with a series of creative prompts inspired by free and creative writing methodologies [57]. Prompts could be obstructions (i.e., "This whole project is about cars. What would a house for cars look like?"), incomplete stories (i.e., "Imagine you are in

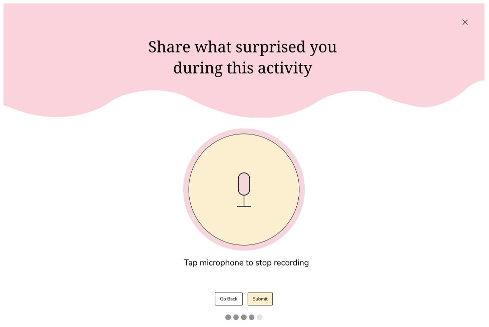

×
Share what surprised you during this activity
Tap microphone to stop recording
Go Back
Submit

Fig. 12. Reflection page.

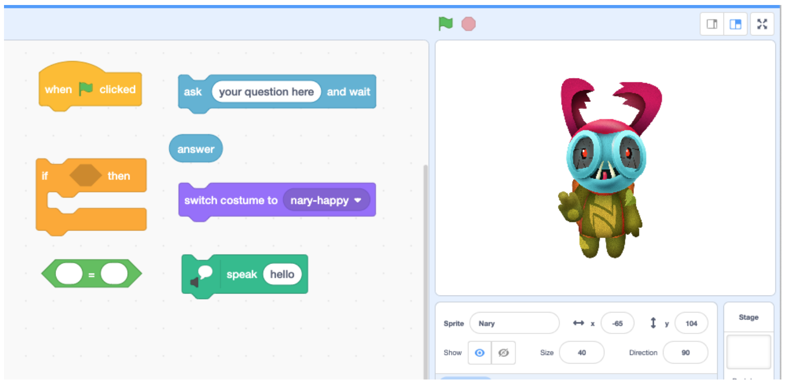
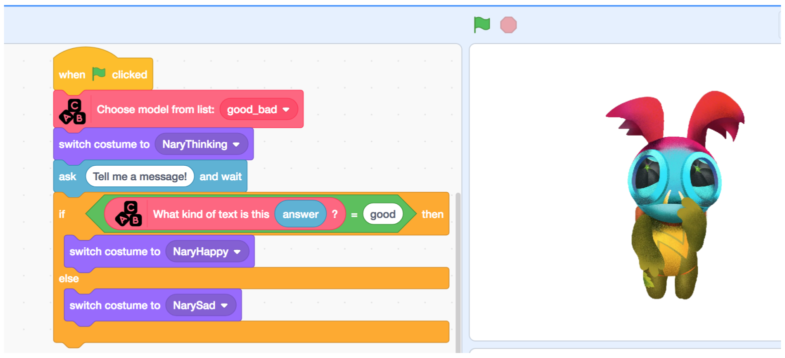


Fig. 13. **A "Conversation" micro-world.** Available blocks (left) and the agent controlled by the program (right).



```
when clicked
Choose model from list: good_bad
switch costume to NaryThinking
ask Tell me a message! and wait
if What kind of text is this answer ? = good then
    switch costume to NaryHappy
else
    switch costume to NarySad
```

Fig. 14. **An example program** assembled with the “Conversation” micro-world in Figure 13, as well as additional blocks added later. This program classifies the sentiment of the text entered and changes the characters’ "costume" based on whether it is positive.

a forest. What do you notice around you?”), an environment inspired (i.e., “Make a game you can play with the first object you see in your room”), meta-prompts (i.e., “Describe your creative autobiography”).

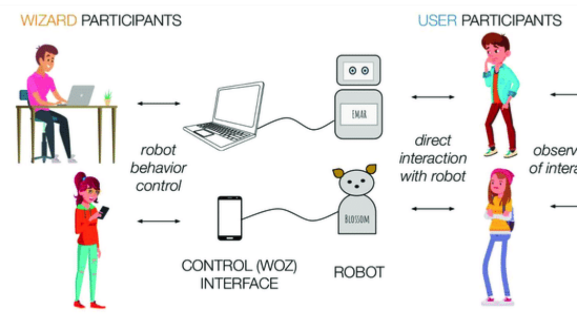


Fig. 15. Description of the PWoz method. Source Bjorling 2020 [23]

Self-Perception Game I plan to measure children's creative self-efficacy (pre and post-programming activity) with the three-item questionnaire developed by Tierney and Farmer [148], which included statements about perceived efficacy in producing ideas, solving problems, and elaborating or improving upon others' ideas. In order to make the questionnaire child-friendly, I will present the questions in the form of a tangible game similar to the monster game used in this prior study [54].

5.4 Data Collection

I will collect pre-and post- self-efficacy data as well as video recordings of all sessions. Children will also be enabled to record activity feedback and in-situ [108] as part of the slider provided on the platform, and their screen activity will be recorded, logged, and time-stamped. We will also record logs on all the support provided by wizards. Both creators and wizards will be encouraged to speak aloud [36] during the activity. After the activity, the creators will meet the wizards, and they will discuss the interaction. I will prompt them with reflection questions based on observed moments in the interaction and ask them to pick three favorite moments and three frustrating moments by showing them the screen recording of their actions.

5.5 Qualitative analysis

For the qualitative analyses, I will transcribe the videos and note comments on children's body language and non-verbal interactions. Once all the transcriptions are finished, I will review all the text, looking for ways of explaining the different phases of the study. In this process, we will analyze each transcript using a combination of etic codes developed from the CSE theoretical framework and emic codes that will emerge from the children's reflections themselves [114, 124]. After I develop a final coding frame, I will code all the transcripts. If new codes emerge, I will document discrepancies in the analyses until the codes are stable. I will then use this coding process to develop categories, which will be then conceptualized into broad themes [29].

5.6 Feasibility and challenges

I estimate the most challenging part of implementing this study design will likely be accommodating for Covid requirements. Ideally, I would like this study to occur in person, but if that is not possible, I will have to modify the PWoZ study methodology to for an online setup.

5.7 Validity and challenges

Conducting a PWoZ study offers benefits for children. First, it provides a level of ecological validity by having the children develop the scripts and operate the agent for the study. Their actions and choices provide the primary direction and data for the study. Further, making both the constraints and limitations of the agent make the creative support technology more transparent. The low fidelity nature of the Code Playground platform will be revealed to the participants in the study providing a more realistic impression of the technological capability of the *intelligent* agents, which is relatively modest. This study design provides a more authentic interaction between children and agents.

I want to conduct all of the study sessions on campus in order to maintain ecological validity. However, this might also mean that some factors will be out of my control, such as potential changes in Covid response guidelines. In addition, our sample is at risk of being reasonably homogeneous given the geographic location of our campus . Thus, it is essential to consider similar studies in other locations, e.g., rural. There may also be cultural aspects (family culture and ethnic cultures) that will influence my data and subsequent data analysis; aiming to diversify my participant sample and collaborate with a diverse research team is essential. Finally, the children might have varying degrees of experience with programming, which is very likely to influence the data and needs to be considered.

6 CONCLUSION

By developing and leveraging children's creative fluency and imagination, we would allow them to not only be prepared for the 21'st century but to inspire and advance our use of computational tools in unique and unforeseeable ways. As our world moves, so does our art and our creativity. As researchers and designers, we need to decide how much we would want to include intelligent technologies in our creative and learning tools and for what purposes. Engaging children as design partners in our creative coding and future tools design will ensure a future worth building up through.

REFERENCES

[1] [n.d.]. Ph.D. Dissertation.

[2] [n.d.]. Dynamicland. https://dynamicland.org/. Accessed: 2021-5-21.

[3] [n.d.]. Froebel Gifts Kindergarten Presechool Curriculum Educational Toys. https://www.froebelgifts.com/gifts.htm. Accessed: 2021-5-22.

[4] [n.d.]. Idea Wiki. https://ideas.fandom.com/wiki/Ideas_Wiki. Accessed: 2021-5-21.

[5] [n.d.]. LEGO Mindstorms. https://www.lego.com/en-us/mindstorms.

[6] [n.d.]. Programming For Kids. https://agentsheets.com/. Accessed: 2021-5-21.

[7] [n.d.]. Programming in a museum? Creative coding for art mediation. http://ils.sont.la/post/programming-museum-creative-coding-art-mediation. Accessed: 2021-5-21.

[8] [n.d.]. Scratch - Imagine, Program, Share. https://scratch.mit.edu/. Accessed: 2021-5-21.

[9] [n.d.]. Stuart Swann on Twitter: "Love this version of @mres ‘s Creative Learning Spiral that I stumbled across. https://t.co/bYKMzxmb9W" / Twitter. https://twitter.com/stuartswann/status/1396022891117006852. (Accessed on 05/23/2021).

[10] Edith K Ackermann. 2005. Playthings that do things: a young kid's incredibles!. In *Proceedings of the 2005 conference on Interaction design and children*. ACM, 1–8.

[11] Safinah Ali, Tyler Moroso, and Cynthia Breazeal. 2019. Can Children Learn Creativity from a Social Robot?. In *Proceedings of the 2019 on Creativity and Cognition* (San Diego, CA, USA) *(Camp;C '19)*. Association for Computing Machinery, New York, NY, USA, 359–368. https://doi.org/10.1145/3325480.3325499

[12] Patrícia Alves-Oliveira, Patrícia Arriaga, Ana Paiva, and Guy Hoffman. 2017. YOLO, a Robot for Creativity: A Co-Design Study with Children. In *Proceedings of the 2017 Conference on Interaction Design and Children* (Stanford, California, USA) *(IDC '17)*. Association for Computing Machinery, New York, NY, USA, 423–429.

[13] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N Bennett, Kori Inkpen, et al. 2019. Guidelines for human-ai interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

[14] Morgan G Ames. 2015. Charismatic technology. In *Proceedings of The Fifth Decennial Aarhus Conference on Critical Alternatives*. Aarhus University Press, 109–120.

[15] Christopher Andrews. 2019. An Embodied Approach to AI Art Collaboration. In *Proceedings of the 2019 on Creativity and Cognition* (San Diego, CA, USA) *(C&C '19)*. Association for Computing Machinery, New York, NY, USA, 156–162.

[16] Cecilia R Aragon, Sarah S Poon, Andrés Monroy-Hernández, and Diana Aragon. 2009. A tale of two online communities: fostering collaboration and creativity in scientists and children. In *Proceedings of the seventh ACM conference on Creativity and cognition* (Berkeley, California, USA) *(C&C '09)*. Association for Computing Machinery, New York, NY, USA, 9–18.

[17] A Bandura. 1977. Self-efficacy: toward a unifying theory of behavioral change. *Psychol. Rev.* 84, 2 (March 1977), 191–215.

[18] Baptiste Barbot, Todd I Lubart, and Maud Besançon. 2016. "Peaks, slumps, and bumps": Individual differences in the development of creativity in children and adolescents. *New Dir. Child Adolesc. Dev.* 2016, 151 (2016), 33–45.

[19] Simon Baron-Cohen. 1999. *The evolution of a theory of mind.* na.

[20] Christoph Bartneck, Dana Kulić, Elizabeth Croft, and Susana Zoghbi. 2009. Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots. *International journal of social robotics* 1, 1 (2009), 71–81.

[21] Benjamin B Bederson, Jesse Grosjean, and Jon Meyer. 2004. Toolkit design for interactive structured graphics. *IEEE Transactions on software engineering* 30, 8 (2004), 535–546.

[22] Ronald A Beghetto. 2006. Creative Self-Efficacy: Correlates in Middle and Secondary Students. *Creat. Res. J.* 18, 4 (Oct. 2006), 447–457.

[23] Elin A Björling, Kyle Thomas, Emma J Rose, and Maya Cakmak. 2020. Exploring Teens as Robot Operators, Users and Witnesses in the Wild. *Front Robot AI* 7 (Feb. 2020), 5.

[24] Paulo Blikstein. 2013. Gears of our childhood: constructionist toolkits, robotics, and physical computing, past and future. In *Proceedings of the 12th international conference on interaction design and children.* 173–182.

[25] Paulo Blikstein. 2014. Digital fabrication and 'making'in education. In *FabLab.* transcript-Verlag, 203–222.

[26] Paulo Blikstein and Dennis Krannich. 2013. The makers' movement and FabLabs in education: experiences, technologies, and research. In *Proceedings of the 12th international conference on interaction design and children.* 613–616.

[27] Benjamin S Bloom et al. 1956. Taxonomy of educational objectives. Vol. 1: Cognitive domain. *New York: McKay* 20 (1956), 24.

[28] Benedict du Boulay, Tim O'Shea, and John Monk. 1981. The black box inside the glass box: presenting computing concepts to novices. *Int. J. Man. Mach. Stud.* 14, 3 (April 1981), 237–249.

[29] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[30] K Brennan. 2015. Beyond right or wrong: Challenges of including creative design activities in the classroom. *Journal of Technology and Teacher Education* (2015).

[31] Karen Brennan, Andrés Monroy-Hernández, and Mitchel Resnick. 2010. Making projects, making friends: online community as catalyst for interactive media creation. *New Dir. Youth Dev.* 2010, 128 (2010), 75–83.

[32] K A Brennan. 2013. Best of both worlds: Issues of structure and agency in computational creation, in and out of school. (2013).

[33] Urie Bronfenbrenner. 1994. Ecological models of human development. *Readings on the development of children* 2, 1 (1994), 37–43.

[34] Amy Bruckman. 1998. Community Support for Constructionist Learning. *Comput. Support. Coop. Work* 7, 1 (March 1998), 47–86.

[35] Lindsey Jacquelyn Byom and Bilge Mutlu. 2013. Theory of mind: Mechanisms, methods, and new directions. *Frontiers in human neuroscience* 7 (2013), 413.

[36] Elizabeth Charters. 2003. The use of think-aloud methods in qualitative research an introduction to think-aloud methods. *Brock Education: A Journal of Educational Research and Practice* 12, 2 (2003).

[37] Christmas Experiments Team. [n.d.]. Legends. https://christmasexperiments.com/2018/23/legends. Accessed: 2021-5-21.

[38] Chen Chunhan, Tang Yihan, Xie Tianyi, and Druga Stefania. 2019. The Humming Box: AI-powered Tangible Music Toy for Children. In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts.* 87–95. https://doi.org/10.1145/3341215.3356990

[39] Anne Cloonan. 2008. Multimodality pedagogies: A multiliteracies approach. *International journal of learning* 15, 9 (2008), 159–168.

[40] J Daniel Couger. 1996. *Creativity & innovation in information systems organizations.* Boyd & Fraser Publishing Company.

[41] Mihaly Csikszentmihalyi, Sami Abuhamdeh, Jeanne Nakamura, et al. 1990. Flow.

[42] M Berry David. 2008. A Contribution Towards A Grammar of Code. *Fibreculture journal* 13 (Jan. 2008).

[43] N Davis, C Hsiao, K Y Singh, B Lin, and B Magerko. 2017. Quantifying Collaboration with a Co-Creative Drawing Agent. *ACM Trans. Interact. Intell. Syst.* 7, 4 (Dec. 2017), 1–25.

[44] Nicholas Davis, Holger Winnemöller, Mira Dontcheva, and Ellen Yi-Luen Do. 2013. Toward a cognitive theory of creativity support. In *Proceedings of the 9th ACM Conference on Creativity & Cognition* (Sydney, Australia) *(C&C '13)*. Association for Computing Machinery, New York, NY, USA, 13–22.

[45] Dominik Dellermann, Adrian Calma, Nikolaus Lipusch, Thorsten Weber, Sascha Weigel, and Philipp Ebel. 2019. The future of human-AI collaboration: a taxonomy of design knowledge for hybrid intelligence systems. In *Proceedings of the 52nd Hawaii International Conference on System Sciences.*

[46] Manuj Dhariwal and Shruti Dhariwal. 2020. Let's Chance: Playful Probabilistic Programming for Children. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–7.

[47] Paul E Dickson, Neil C C Brown, and Brett A Becker. 2020. Engage Against the Machine: Rise of the Notional Machines as Effective Pedagogical Devices. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (Trondheim, Norway) *(ITiCSE '20)*. Association for Computing Machinery, New York, NY, USA, 159–165.

[48] Betsy DiSalvo and Amy Bruckman. 2011. From interests to values. *Commun. ACM* 54, 8 (Aug. 2011), 27–29.

[49] Betsy DiSalvo, Mark Guzdial, Charles Meadows, Ken Perry, Tom McKlin, and Amy Bruckman. 2013. Workifying games: successfully engaging african american gamers with computer science. In *Proceeding of the 44th ACM technical symposium on Computer science education* (Denver, Colorado, USA) *(SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 317–322.

[50] Andrea A DiSessa. 2001. *Changing minds: Computers, learning, and literacy*. Mit Press.

[51] Steven P Dow, Alana Glassco, Jonathan Kass, Melissa Schwarz, Daniel L Schwartz, and Scott R Klemmer. 2011. Parallel prototyping leads to better design results, more divergence, and increased self-efficacy. *ACM Trans. Comput.-Hum. Interact.* 17, 4 (Dec. 2011), 1–24.

[52] Stefania Druga. 2018. *Growing up with AI: Cognimates: from coding to teaching machines*. Ph.D. Dissertation. Massachusetts Institute of Technology.

[53] Stefania Druga and Amy J. Ko. 2021. How do children's perceptions of machine intelligence change when training and coding smart programs? *Interaction Design for Children ACM* (2021). https://doi.org/10.1145/3459990.3460712

[54] Stefania Druga, Sarah T Vu, Eesh Likhith, and Tammy Qiu. 2019. Inclusive AI literacy for kids around the world. In *Proceedings of FabLearn 2019*. ACM, 104–111.

[55] Allison Druin. 2002. The role of children in the design of new technology. *Behaviour and information technology* 21, 1 (2002), 1–25.

[56] Michael Eagle and Tiffany Barnes. 2008. Wu's castle: teaching arrays and loops in a game. In *Proceedings of the 13th annual conference on Innovation and technology in computer science education* (Madrid, Spain) *(ITiCSE '08)*. Association for Computing Machinery, New York, NY, USA, 245–249.

[57] Steven Earnshaw. 2007. *The handbook of creative writing*. Edinburgh University Press.

[58] Ron Eglash, Juan E Gilbert, and Ellen Foster. 2013. Toward culturally responsive computing education. *Commun. ACM* 56, 7 (July 2013), 33–36.

[59] Mary Fawcett and Penny Hay. 2004. 5x5x5 = Creativity in the Early Years. *The international journal of art & design education* 23, 3 (Oct. 2004), 234–245.

[60] Diana Franklin, David Weintrop, Jennifer Palmer, Merijke Coenraad, Melissa Cobian, Kristan Beck, Andrew Rasmussen, Sue Krause, Max White, Marco Anaya, and Zachary Crenshaw. 2020. Scratch Encore: The Design and Pilot of a Culturally-Relevant Intermediate Scratch Curriculum. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 794–800.

[61] Paulo Freire. 2013. *Pedagogy of the oppressed*. Routledge.

[62] Jonas Frich, Lindsay MacDonald Vermeulen, Christian Remy, Michael Mose Biskjaer, and Peter Dalsgaard. 2019. Mapping the Landscape of Creativity Support Tools in HCI. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19, Paper 389)*. Association for Computing Machinery, New York, NY, USA, 1–18.

[63] Howard Gardner and Katie Davis. 2013. 6. Acts (And Apps) of Imagination Among Today's Youth. In *The App Generation*. Yale University Press, 120–154.

[64] John Taylor Gatto. 2003. *The underground history of American education*. Oxford Village Press New York, NY.

[65] Clifford Geertz. 2008. Thick description: Toward an interpretive theory of culture. In *The cultural geography reader*. Routledge, 41–51.

[66] David J Gilmore. 1991. Models of debugging. *Acta Psychol.* 78, 1 (Dec. 1991), 151–172.

[67] Rachel Gockley, Allison Bruce, Jodi Forlizzi, Marek Michalowski, Anne Mundell, Stephanie Rosenthal, Brennan Sellner, Reid Simmons, Kevin Snipes, Alan C Schultz, et al. 2005. Designing robots for long-term social interaction. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 1338–1343.

[68] Saul Greenberg. 2004. Collaborative physical user interfaces. (2004).

[69] J P Guilford. 1967. Creativity: Yesterday, today and tomorrow. *J. Creat. Behav.* 1, 1 (Jan. 1967), 3–14.

[70] Philip J Guo. 2015. Codeopticon: Real-Time, One-To-Many Human Tutoring for Computer Programming. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 599–608.

[71] Mark Guzdial. 1994. Software-Realized Scaffolding to Facilitate Programming for Science Learning. *Interactive Learning Environments* 4, 1 (Jan. 1994), 001–044.

[72] Mark Guzdial. 2015. Learner-Centered Design of Computing Education: Research on Computing for Everyone. *Synthesis Lectures on Human-Centered Informatics* 8, 6 (Nov. 2015), 1–165.

[73] David Hammer. 1996. Misconceptions or p-prims: How may alternative perspectives of cognitive structure influence instructional perceptions and intentions. *The journal of the learning sciences* 5, 2 (1996), 97–127.

[74] Idit Harel. 1990. Children as software designers: A constructionist approach for learning mathematics. *Journal of Mathematical Behavior* 9, 1 (1990), 3–93.

[75] Cindy E Hmelo and Mark Guzdial. 1996. Of black and glass boxes: Scaffolding for doing and learning. (1996).

[76] Glynda A Hull and Mark Evan Nelson. 2005. Locating the semiotic power of multimodality. *Written communication* 22, 2 (2005), 224–261.

[77] Ivan Illich, Ivan Illich, Ivan Illich, and Ivan Illich. 1971. Deschooling society.

[78] Yasmin Kafai, Deborah Fields, and Kristin Searle. 2014. Electronic textiles as disruptive designs: Supporting and challenging maker activities in schools. *Harvard Educational Review* 84, 4 (2014), 532–556.

[79] Yasmin B Kafai. 2016. From computational thinking to computational participation in K–12 education. *Commun. ACM* 59, 8 (2016), 26–27.

[80] Yasmin B Kafai and Quinn Burke. 2014. *Connected code: Why children need to learn programming.* Mit Press.

[81] Yasmin B Kafai and Cynthia Carter Ching. 2001. Affordances of collaborative software design planning for elementary students' science talk. *The Journal of the Learning Sciences* 10, 3 (2001), 323–363.

[82] Yasmin B Kafai, Eunkyoung Lee, Kristin Searle, Deborah Fields, Eliot Kaplan, and Debora Lui. 2014. A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles. *ACM Transactions on Computing Education (TOCE)* 14, 1 (2014), 1–20.

[83] Pegah Karimi, Kazjon Grace, Mary Lou Maher, and Nicholas Davis. 2018. Evaluating Creativity in Computational Co-Creative Systems. (July 2018). arXiv:1807.09886 [cs.AI]

[84] Pegah Karimi, Jeba Rezwana, Safat Siddiqui, Mary Lou Maher, and Nasrin Dehbozorgi. 2020. Creative sketching partner: an analysis of human-AI co-creativity. In *Proceedings of the 25th International Conference on Intelligent User Interfaces* (Cagliari, Italy) *(IUI '20)*. Association for Computing Machinery, New York, NY, USA, 221–230.

[85] Maciej Karwowski and James C Kaufman. 2017. *The Creative Self: Effect of Beliefs, Self-Efficacy, Mindset, and Identity.* Academic Press.

[86] James C Kaufman and Ronald A Beghetto. 2009. Beyond Big and Little: The Four C Model of Creativity. *Rev. Gen. Psychol.* 13, 1 (March 2009), 1–12.

[87] Annie Kelly. 2019. Code Against the Machine: Design-based Research Towards More Equitable Computational Tools for Performing Artists. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. Association for Computing Machinery, New York, NY, USA, 337–338.

[88] Joy Kim, Maneesh Agrawala, and Michael S Bernstein. 2017. Mosaic: Designing Online Creative Communities for Sharing Works-in-Progress. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing* (Portland, Oregon, USA) *(CSCW '17)*. Association for Computing Machinery, New York, NY, USA, 246–258.

[89] A Ko and B Myers. 2008. Debugging reinvented. *2008 ACM/IEEE 30th International Conference* (2008).

[90] Amy Ko and Brad Myers. 2008. Debugging reinvented. In *2008 ACM/IEEE 30th International Conference on Software Engineering*. IEEE, 301–310.

[91] Amy J. Ko, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scaffidi, Joseph Lawrance, Henry Lieberman, Brad Myers, et al. 2011. The state of the art in end-user software engineering. *ACM Computing Surveys (CSUR)* 43, 3 (2011), 1–44.

[92] A J Ko, B A Myers, and H H Aung. 2004. Six learning barriers in end-user programming systems. *Proc. IEEE Symp. Vis. Anal. Sci. Technol.* (2004).

[93] Kyu Han Koh, Ashok Basawapatna, Vicki Bennett, and Alexander Repenning. 2010. Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning. In *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*. ieeexplore.ieee.org, 59–66.

[94] Arie W Kruglanski and Anna Sheveland. 2012. Thinkers' personalities: On individual differences in the processes of sense making. *The SAGE handbook of social cognition* (2012), 474–493.

[95] Jaron Lanier. 2010. *You are not a gadget: A manifesto.* Vintage.

[96] Lotta C Larson and Teresa Northern Miller. 2011. 21st century skills: Prepare students for the future. *Kappa Delta Pi Record* 47, 3 (2011), 121–123.

[97] Jean Lave, Etienne Wenger, et al. 1991. *Situated learning: Legitimate peripheral participation.* Cambridge university press.

[98] Michael J Lee. 2020. Auto-generated game levels increase novice programmers' engagement. *J. Comput. Sci. Coll.* 36, 3 (Oct. 2020), 70–79.

[99] Michael J Lee, Faezeh Bahmani, Irwin Kwan, Jilian LaFerte, Polina Charters, Amber Horvath, Fanny Luor, Jill Cao, Catherine Law, Michael Beswetherick, et al. 2014. Principles of a debugging-first puzzle game for computing education. In *2014 IEEE symposium on visual languages and human-centric computing (VL/HCC)*. IEEE, 57–64.

[100] Michael J Lee, Faezeh Bahmani, Irwin Kwan, Jilian LaFerte, Polina Charters, Amber Horvath, Fanny Luor, Jill Cao, Catherine Law, Michael Beswetherick, Sheridan Long, Margaret Burnett, and Amy J Ko. 2014. Principles of a debugging-first puzzle game for computing education. In *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 57–64.

[101] Michael J Lee and Joseph Chiou. 2020. Animated hints help novices complete more levels in an educational programming game. *J. Comput. Sci. Coll.* 35, 8 (April 2020), 136–145.

[102] Michael J Lee and Amy J Ko. 2015. Comparing the effectiveness of online learning approaches on CS1 learning outcomes. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research.* ACM, 237–246.

[103] Yong Jae Lee, C Lawrence Zitnick, and Michael F Cohen. 2011. ShadowDraw: real-time user guidance for freehand drawing. *ACM Trans. Graph.* 30, 4 (July 2011), 1–10.

[104] Yong Jae Lee, C Lawrence Zitnick, and Michael F Cohen. 2011. Shadowdraw: real-time user guidance for freehand drawing. *ACM Transactions on Graphics (TOG)* 30, 4 (2011), 1–10.

[105] Anaïs Leroy, Margarida Romero, and Laura Cassone. 2021. Interactivity and materiality matter in creativity: educational robotics for the assessment of divergent thinking. *Interactive Learning Environments* (Jan. 2021), 1–12.

[106] Stanley Letovsky. 1987. Cognitive processes in program comprehension. *Journal of Systems and software* 7, 4 (1987), 325–339.

[107] Raymond Lister. 2016. Toward a Developmental Epistemology of Computer Programming. In *Proceedings of the 11th Workshop in Primary and Secondary Computing Education* (Münster, Germany) *(WiPSCE '16)*. Association for Computing Machinery, New York, NY, USA, 5–16.

[108] Dastyni Loksa, Benjamin Xie, Harrison Kwik, and Amy J Ko. 2020. Investigating Novices' In Situ Reflections on Their Programming Process. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 149–155.

[109] Duri Long and Brian Magerko. 2020. What is AI Literacy? Competencies and Design Considerations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–16.

[110] Pedro Lucas and Carlos Martinho. 2017. Stay Awhile and Listen to 3Buddy, a Co-creative Level Design Support Tool. In *ICCC*. computationalcreativity.net, 205–212.

[111] Carmen Luke. 2003. Pedagogy, connectivity, multimodality, and interdisciplinarity. *Reading research quarterly* 38, 3 (2003), 397–403.

[112] John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)* 10, 4 (2010), 1–15.

[113] Michael Michalko. 2006. *Thinkertoys: A handbook of creative-thinking techniques*. Random House Digital, Inc.

[114] Matthew B Miles and A Michael Huberman. 1984. Drawing valid meaning from qualitative data: Toward a shared craft. *Educational researcher* 13, 5 (1984), 20–30.

[115] Brad A Myers, Richard G McDaniel, Robert C Miller, Alan S Ferrency, Andrew Faulring, Bruce D Kyle, Andrew Mickish, Alex Klimovitski, and Patrick Doane. 1997. The Amulet environment: New models for effective user interface software development. *IEEE Transactions on software engineering* 23, 6 (1997), 347–365.

[116] Ulric Neisser. 2014. *Cognitive psychology: Classic edition*. Psychology Press.

[117] Greg L Nelson, Andrew Hu, Benjamin Xie, and Amy J Ko. 2019. Towards validity for a formative assessment for language-specific program tracing skills. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*. 1–10.

[118] Greg L Nelson, Benjamin Xie, and Andrew J Ko. 2017. Comprehension first: evaluating a novel pedagogy and tutoring system for program tracing in CS1. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*. 2–11.

[119] Daniel Ness and Stephen J Farenga. 2016. Blocks, bricks, and planks: Relationships between affordance and visuo-spatial constructive play objects. *Am. J. Play* 8, 2 (2016), 201–227.

[120] Seymour Papert. 1980. *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc.

[121] Seymour Papert. 1993. *The children's machine: Rethinking school in the age of the computer.* ERIC.

[122] Seymour Papert and Idit Harel. 1991. Situating constructionism. *Constructionism* 36, 2 (1991), 1–11.

[123] Hae Won Park, Rinat B Rosenberg-Kima, Maor Rosenberg, Goren Gordon, and Cynthia Breazeal. 2017. Growing Growth Mindset with a Social Robot Peer.. In *HRI*. 137–145.

[124] Michael Quinn Patton. 1990. *Qualitative evaluation and research methods*. SAGE Publications, inc.

[125] William Christopher Payne, Yoav Bergner, Mary Etta West, Carlie Charp, R Benjamin Benjamin Shapiro, Danielle Albers Szafir, Edd V Taylor, and Kayla DesPortes. 2021. danceON: Culturally Responsive Creative Computing. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21, Article 96)*. Association for Computing Machinery, New York, NY, USA, 1–16.

[126] Michael A Peters, Simon Marginson, and Peter Murphy. 2009. *Creativity and the global knowledge economy*. Peter Lang.

[127] Mike Petrich, Karen Wilkinson, and Bronwyn Bevan. 2013. It looks like fun, but are they learning? In *Design, make, play*. Routledge, 68–88.

[128] Jean Piaget. 1951. *The child's conception of the world*. Number 213. Rowman & Littlefield.

[129] Linda Pound. 2017. *How Children Learn-Book 1: From Montessori to Vygosky-Educational Theories and Approaches Made Easy*. Vol. 1. Andrews UK Limited.

[130] Lauren B Resnick, Merrilee Salmon, Colleen M Zeitz, Sheila Haley Wathen, and Mark Holowchak. 1993. Reasoning in conversation. *Cognition and instruction* 11, 3-4 (1993), 347–364.

[131] Mitchel Resnick. [n.d.]. Give p's a chance: Projects, peers, passion, play. https://web.media.mit.edu/~mres/papers/constructionism-2014.pdf. Accessed: 2021-5-19.

[132] Mitchel Resnick. 2007. All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten. In *Proceedings of the 6th ACM SIGCHI Conference on Creativity & Cognition*. 1–6.

[133] Mitchel Resnick. 2008. Sowing the seeds for a more creative society. *Learning & Leading with Technology* 35, 4 (2008), 18–22.

[134] Mitchel Resnick, Robbie Berg, and Michael Eisenberg. 2000. Beyond black boxes: Bringing transparency and aesthetics back to scientific investigation. *The Journal of the Learning Sciences* 9, 1 (2000), 7–30.

[135] Mitchel Resnick, Amy Bruckman, and Fred Martin. 1996. Pianos not stereos: Creating computational construction kits. (1996).

[136] Mitchel Resnick and Ken Robinson. 2017. *Lifelong Kindergarten: Cultivating Creativity Through Projects, Passion, Peers, and Play*. MIT Press.

[137] Mitchel Resnick and Eric Rosenbaum. 2013. Designing for tinkerability. *Design, make, play: Growing the next generation of STEM innovators* (2013), 163–181.

[138] Barbara Rogoff, Behnosh Najafi, and Rebeca Mejía-Arauz. 2014. Constellations of cultural practices across generations: Indigenous American heritage and learning by observing and pitching in. *Human Development* 57, 2-3 (2014), 82–95.

[139] Marlene Scardamalia and Carl Bereiter. 1994. Computer support for knowledge-building communities. *The journal of the learning sciences* 3, 3 (1994), 265–283.

[140] Donald A Schon. 1983. *Reflective practitioner*. Vol. 5126. Basic books.

[141] David Williamson Shaffer and Mitchel Resnick. 1999. " Thick" authenticity: New media and authentic learning. *Journal of interactive learning research* 10, 2 (1999), 195–216.

[142] Ben Shneiderman. 2002. Creativity support tools. *Commun. ACM* 45, 10 (2002), 116–120.

[143] Ben Shneiderman. 2007. Creativity support tools: accelerating discovery and innovation. *Commun. ACM* 50, 12 (Dec. 2007), 20–32.

[144] Charles Percy Snow. 2012. *The two cultures.* Cambridge University Press.

[145] Amber Solomon, Vanessa Oguamanam, Mark Guzdial, and Betsy DiSalvo. 2019. Making CS Learning Visible: Case Studies on How Visibility of Student Work Supports a Community of Learners in CS Classrooms. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (Aberdeen, Scotland Uk) *(ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 161–167.

[146] Woonhee Sung, Junghyun Ahn, and John B Black. 2017. Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning* 22, 3 (2017), 443–463.

[147] L M Terman and M A Merrill. 1937. Measuring intelligence: A guide to the administration of the new revised Stanford-Binet tests of intelligence. 461 (1937).

[148] Pamela Tierney and Steven M Farmer. 2002. Creative Self-Efficacy: Its Potential Antecedents and Relationship to Creative Performance. *Acad. Manage. J.* 45, 6 (2002), 1137–1148.

[149] E Paul Torrance. 1981. Creative teaching makes a difference. *Creativity: Its educational implications* (1981), 99–108.

[150] Moran Tsur and Natalie Rusk. 2018. Scratch Microworlds: Designing Project-Based Introductions to Coding. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. Association for Computing Machinery, New York, NY, USA, 894–899.

[151] Sherry Turkle. 2011. *Evocative Objects: Things We Think With.* MIT Press.

[152] Sherry Turkle and Seymour Papert. 1992. Epistemological pluralism and the revaluation of the concrete. *Journal of Mathematical Behavior* 11, 1 (1992), 3–33.

[153] Lev Semenovich Vygotsky. 2004. Imagination and Creativity in Childhood. *Journal of Russian & East European Psychology* 42, 1 (Jan. 2004), 7–97.

[154] Qiaosi Wang, Koustuv Saha, Eric Gregori, David Joyner, and Ashok Goel. 2021. Towards Mutual Theory of Mind in Human-AI Interaction: How Language Reflects What Students Perceive About a Virtual Teaching Assistant. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21, Article 384)*. Association for Computing Machinery, New York, NY, USA, 1–14.

[155] Chao Zhang, Cheng Yao, Jianhui Liu, Zili Zhou, Weilin Zhang, Lijuan Liu, Fangtian Ying, Yijun Zhao, and Guanyun Wang. 2021. StoryDrawer: A Co-Creative Agent Supporting Children's Storytelling through Collaborative Drawing. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–6.