Volume 12 Number 2

The TangibleK Robotics Program: Applied Computational Thinking for Young Children

Marina U. Bers
Tufts University

Abstract

This article describes the TangibleK robotics program for young children. Based on over a decade of research, this program is grounded on the belief that teaching children about the human-made world, the realm of technology and engineering, is as important as teaching them about the natural world, numbers, and letters. The TangibleK program uses robotics as a tool to engage children in developing computational thinking and learning about the engineering design process. It includes a research-based, developmentally appropriate robotics kit that children can use to make robots and program their behaviors. The curriculum has been piloted in kindergarten classrooms and in summer camps and lab settings. The author presents the theoretical framework that informs TangibleK and the "powerful ideas" from computer science and robotics on which the curriculum is based, linking them to other disciplinary areas and developmental characteristics of early childhood. The article concludes with a description of classroom pedagogy and activities, along with assessment tools used to evaluate learning outcomes.

Introduction

We are surrounded by technology. Yet, in the early grades, children learn very little about this subject. Although it is common to see young children using cardboard or recycled materials to build cities and bridges and become "little engineers" (Bers, 2008b), early childhood curriculum has for decades focused on literacy and numeracy, with some recent attention to science. The TangibleK robotics program described in this paper is grounded on the belief that teaching children about the human-made world is as important as teaching them about the natural world, numbers, and letters (Bers, 2008a). However, what is unique to our human-made world today is the fusion of electronics and software with mechanical structures—the discipline of robotics.

Background of TangibleK Robotics

Robots and Robotics

Robots are no longer science fiction creatures; they can be found in many places. They have different forms—from industrial to humanoid—performing autonomous or preprogrammed tasks, such as radioactive waste clean-up, surgical procedures, and automobile production. No consensus exists regarding which machines qualify as robots, but there is agreement that robots may do some or all of the following activities—move around, operate a mechanical limb, sense and manipulate their environment, and exhibit intelligent and/or social behaviors that mimic humans or other animals (Craig, 2005).

The discipline of robotics provides opportunities for young children to

learn about mechanics, sensors, motors, programming, and the digital domain. With the growing popularity of robotics, the use of educational robotic kits is becoming widespread in high schools, middle schools, and elementary schools (Rogers, Wendell, & Foster, 2010). In this paper, I describe a robotic kit for children as involving two elements—construction in the physical world and programming the behaviors of the constructed artifact to be interactive and respond to stimuli. Educational robotic kits have been developed by commercial companies such as LEGO and by research laboratories in different universities (Rogers & Portsmore, 2004; Martin, Mikhak, Resnick, Silverman, & Berg, 2000; Rusk, Resnick, Berg, & Pezalla-Granlund, 2008).

In the DevTech research lab at Tufts University, with funding from the National Science Foundation, we are focusing on developing robotic kits that are developmentally appropriate for use in early childhood education (Horn, Crouser, & Bers, in press). At the same time, we are developing curriculum and methodologies to bring robotics into classrooms (Bers & Horn, 2010; Bers, 2008a) and studying learning outcomes of the classroom use of robotics. Our approach engages young children in playful learning by inviting them to build their own robotic projects, such as cars that follow a light, elevators that work with touch sensors, and puppets that can play music (Bers, 2010). Young children can become engineers by exploring gears, levers, motors, sensors, and programming loops; they can become storytellers by creating their own constructions that move in response to their environment (Bers, 2008b; Wang & Ching, 2003).

Robotics can be a gateway to learning applied mathematical concepts, the scientific method of inquiry, and problem solving (Rogers & Portsmore, 2004). Moreover, robotics invites children to participate in social interactions and negotiations while playing to learn and learning to play (Resnick, 2003). Educational robotic kits are a new generation of learning manipulatives building on the tradition of Montessori and Fröebel, whose early "manipulatives" and "gifts" were designed for children to develop a deeper understanding of such concepts as number, size, and shape (Brosterman, 1997). Today, most early childhood settings include Cuisenaire rods, pattern blocks, DigiBlocks, and other manipulatives, carefully designed to help children build and experiment. Recently, “digital manipulatives” expand the range of concepts that children can explore; researchers have embedded computational power into toys such as blocks, beads, and balls, so young children can learn about dynamic processes and "systems concepts," such as feedback and emergence that were previously considered too advanced for them (Resnick, Berg, & Eisenberg, 2000).

Within this tradition, robotics presents an opportunity to introduce children to the world of technology and engineering. Robotic manipulatives invite children into activities that develop fine motor skills and hand-eye coordination and into activities that involve collaboration and teamwork. They also provide a concrete and tangible way to understand abstract ideas (Bers, 2008a). For example, while playing with mechanical parts to design their robotic creatures, children explore levers, joints, and motors, and they build simple machines. By adding gears to their machines, they begin to explore the mathematical concept of ratio.

Computer Programming, Computational Thinking, and Children

Working with robotics involves more than building physical artifacts. Bringing robots to “life” involves computer programming. Thus, children learn to create computer programs—algorithms or sequences of instructions that allow robots to move and to sense and respond to their environment.

Work on children’s computer programming began several decades ago at the MIT Artificial Intelligence Lab, which later became the Logo lab, when Seymour Papert developed a floor turtle that children could control using the text-based Logo programming language (Bers, 2008a). Recent research has shown that children as young as 4 years old can understand basic concepts of computer programming and can build simple robots (Bers, Ponte, Juelich, Viera, & Schenker, 2002;

Cejka, Rogers, & Portsmore, 2006). Early studies with Logo showed that when introduced in a structured way, computer programming can help young children improve visual memory and basic number sense, as well as develop problem-solving techniques and language skills (Clements, 1999). Work by Papert (1980) and by Resnick (1996) has also shown that learning how to program may result in changes to the ways people think.

Computational thinking is a type of analytical thinking that shares many similarities with mathematical thinking (e.g., problem solving), engineering thinking (designing and evaluating processes), and scientific thinking (systematic analysis). The term grew out of the pioneering work of Papert and colleagues on design-based constructionist programming environments; it refers to ways of algorithmically solving problems and to the acquisition of technological fluency (Papert, 1980, 1993). The foundation for computational thinking is abstraction—abstracting concepts from cases and evaluating and selecting the "right" abstraction. It relies on selection of inputs (manipulation of variables and computational instructions), observation of outputs (outcome data), and decomposition of what happens in between. Computational thinking involves the abilities to abstract from computational instructions (programming languages) to computational behaviors, to identify potential "bugs" and places for errors, to decide which details in the input-computation-output algorithm to highlight and retain and which to discard (Wing, 2006).

Previous work on computational thinking in young elementary school-age children can be found in the research literature on constructionist programming environments (Repenning, Webb, & Ioannidou, 2010; Resnick et al., 2009). Wing (2006) describes computational thinking as a fundamental skill for everyone, not just for computer scientists:

> To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, … computing and computers facilitate the spread of computational thinking. Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. (Wing, 2006, p. 33)

In sum, computational thinking refers to a range of mental tools that reflect the breadth of the field of computer science.

TangibleK Robotics Program

Program Goals and Research Questions

TangibleK Robotics is an educational robotics program that has been piloted with children and teachers in prekindergarten to second grade. It consists of curriculum, assessment tools, and a robotics construction kit with a developmentally appropriate interface. The curriculum and the kit are specifically aimed at teaching a particular subset of mental tools to young children—powerful ideas and skills that are useful when applying computational thinking in a robotic context.

Not enough is known at present about how working with new technologies might promote computational thinking in young children and what kinds of learning trajectories lead to the best outcomes. These are the goals of the TangibleK research program, which explores both learning aspects and computer interface design issues such as tangible programming languages.

Three research questions are at the core of work on TangibleK:

- What are young children's trajectories in computational thinking when exposed to a robotics educational program?
- What concepts and skills are involved in robotics programming that young children can develop, and what support mechanisms (in terms of both curriculum and technology) do they need?
- What design elements should a developmentally appropriate robotics kit include to engage young children in a successful learning experience?

It is beyond the scope of this paper to address these research questions; instead, we describe the TangibleK educational program that makes possible our exploration of those questions. More information on this program and access to the curriculum and the technology, as well as to research papers, can be found at http://ase.tufts.edu/DevTech/tangiblek/.

Positive Technological Development (PTD): The Theoretical Foundation of the Program

The Positive Technological Development (PTD) framework forms the theoretical foundation of the program. PTD is an interdisciplinary approach that integrates research in applied developmental science and positive youth development with ideas from computer-mediated communication, computer-supported collaborative learning, and constructionist learning with technology. PTD is a natural extension of the computer literacy and the technological fluency movements that have influenced educational technology (Pearson & Young, 2002), adding psychosocial and ethical components to the cognitive ones (Bers, 2008a; Bers, 2006; Bers, Doyle-Lynch, & Chau, in press). PTD examines the developmental tasks of a child growing up in our digital era and provides a model for developing and evaluating technology-rich programs. It informs the design and evaluation of technology-based educational programs and experiences that are aimed at helping young people use technology in positive ways to learn new things, to express themselves creatively, to communicate, to care for themselves and others, and to contribute to a community, while developing their own sense of identity grounded on personal and moral values. Two bodies of work—constructionism and Positive Youth Development—have influenced the PTD framework.

Constructionism. Constructionism is situated in the intellectual trajectory started in the 1960s by the MIT Logo Group, under the direction of Seymour Papert (Bers et al., 2002). Constructionism has four "pillars." First is the belief in the "learn-by-doing" approach to education. Strongly based on Piaget's constructivism, constructionism emphasizes the use of new technologies to help children learn by making, by actively inquiring, and by playing. The second constructionist "pillar" is recognition of the importance of objects for supporting the development of concrete ways of thinking and learning about abstract phenomena. Computers and robotic construction kits have a salient role as powerful tools to design, create, and manipulate objects in both the real and the virtual world. The third pillar of constructionism is the understanding that powerful ideas can empower the individual. A "powerful idea" is a central concept within a domain that is at once epistemologically and personally useful, interconnected with other disciplines, and has roots in intuitive knowledge that a child has internalized over a long period of time (Papert, 2000; Bers et al., 2002). Constructionists envisioned the computer as a powerful carrier of new ideas that afford new ways of thinking, new ways of putting knowledge to use, and new ways of making personal and epistemological connections with other domains of knowledge (Papert, 2000). The fourth constructionist pillar is recognition of the significance of self-reflection.From a constructionist perspective, the programming of a computer is a powerful way to gain new insights into how the mind works and to reflect about one's own thinking process and one's intellectual and emotional relationship to knowledge (Papert, 1993; Kafai & Resnick, 1996).

Papert's constructionism became widespread in the world of education in 1980 with the publication of

his pioneering book *Mindstorms: Children, Computers, and Powerful Ideas* (Papert, 1980). In *Mindstorms*, Papert advocated giving children opportunities to program computers as a way to learn about mathematics and, more importantly, to learn about their own learning. Through the process of designing and debugging computer programs (external objects to reflect upon), children would develop not only computational thinking but also a metacognitive approach to problem solving and learning.

Positive Youth Development (PYD). Research on TangibleK robotics focuses on the dynamic relations between individuals and contexts, emphasizing the strengths and assets of young people, rather than focusing on diminishing or preventing their risk-taking behaviors (Damon, 2004; Larson, 2000; Theokas & Lerner, 2006; Scales, Benson, Leffert, & Blyth, 2000). The term "positive" connotes the promotion of valued characteristics and activities (i.e., developmental assets) that would lead a young person toward a good developmental trajectory (i.e., development toward improvement of oneself and society). Lerner, Almerigi, Theokas, & Lerner (2005) frame the developmental assets of the PYD model as "six Cs": competence, confidence, character, connection, caring, and contribution, which are conceived as pathways to promote thriving and healthy communities.

Informed by both constructionism and Positive Youth Development, Positive Technological Development (PTD) is a multidimensional framework that makes "the six Cs" relevant in our digital world. It emphasizes not only developmental assets but also positive behaviors supported by the technology—content creation, creativity, communication, collaboration, community building, and choices of conduct (Bers et al., in press). As a framework for the design and implementation of educational programs, PTD takes into consideration the learning environment and the pedagogical practices, cultural values, and rituals that mediate teaching and learning (Rogoff, 2003; Rogoff, Turkanis, & Bartlett, 2001). Elements of the framework presented in Figure 1 will be described later.

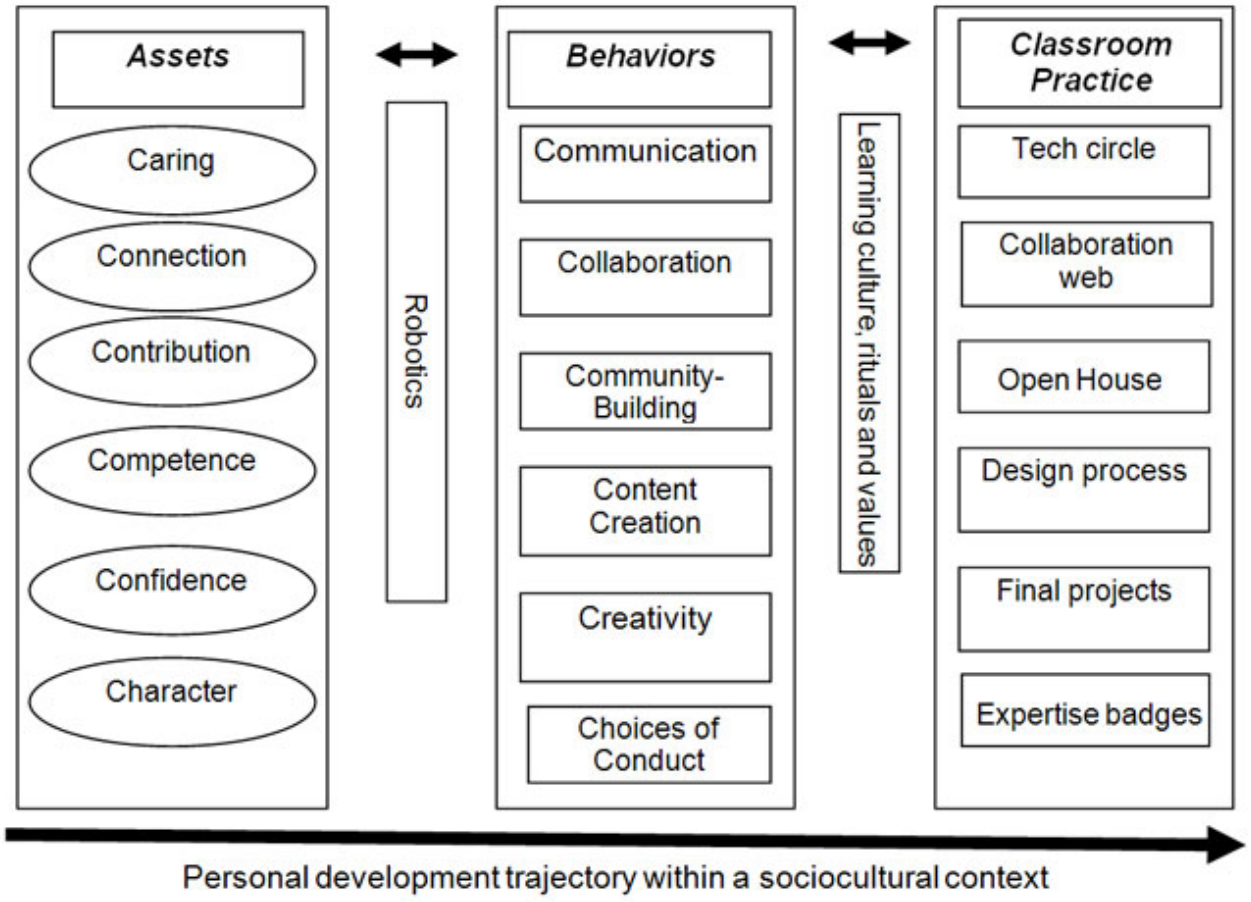

Figure 1: The PTD framework, including Assets,

Behaviors, and Classroom Practices.

TangibleK and the PTD Framework

As Figure 1 suggests, the TangibleK robotics program addresses "the six Cs," or assets, of Positive Youth Development (listed in the left-hand column) through specific classroom practices (right-hand column) that provide opportunities for young learners to engage in specific behaviors associated with working with technology (center column).

Content Creation. TangibleK involves content creation: A child makes a robotic artifact and programs its behaviors. The engineering design process of building and the computational thinking involved in programming foster the child's *competence* in computer literacy and technological fluency. The classroom practice of having children keep design journals during the process of creating robots helps make transparent to the children (as well as teachers and parents) their own thinking, their learning trajectories, and the project's evolution over time. Like the scientific method, the formal steps of the engineering design process—posing a problem, doing research, planning, developing a prototype, testing, redesigning, and sharing solutions—give students a tool for systematically addressing a problem. TangibleK design journals may provide more- or less-structured paths for children to navigate the process from idea to product by scaffolding these formal steps. A journal may have worksheets to address all steps of the design process or simply white pages to invite imagination; at best, they have a combination of both. This individualization is important. Some children need constraints and top-down planning in order to work effectively. Others do not like to plan in advance. They might belong to a group of learners characterized as "tinkerers" and "bricoleurs" (Turkle & Papert, 1992) who engage in dialogues and negotiations with the technology; they enjoy working bottom-up, messing around with the materials to come up with ideas as they create, design, build, and program. Both epistemological styles are conducive for building competence in the technological domain.

Creativity. The TangibleK approach is based on the promotion of creativity, as opposed to efficiency, in problem solving; the approach is informed by the original meaning of the word *engineering*, which derives from the Latin *ingenium* meaning "innate quality, mental power, clever invention" (*Random House Webster's Unabridged Dictionary,* 2006). The program integrates media such as LEGO pieces, motors, sensors, recyclable materials, arts and crafts materials, and graphical elements from the programming language. In the process of solving technical problems in creative ways with these media, children develop *confidence* in their learning potential. However, clever or creative projects may be difficult to make, and the process can be frustrating. After many tries, the jaw of a child's robotic crocodile still may not open, or her car may break every time it turns to the left. To avoid frustration, some teachers carefully choose the projects for children to work on or provide step-by-step directions. Such a strategy may shelter children from what Alan Kay calls the "hard fun" of creative learning (Kay, 2003). Instead, the TangibleK approach aims to help children learn to manage frustration—an important step toward the development of confidence in one's ability to learn. The learning environment is set up to create a culture in which it is expected that things may not work, and in which succeeding the first time is seen as a rarity, perhaps as a sign that the child might not have challenged herself. As children go through the program, they gradually realize their ability to find solutions by trying multiple times, by using different strategies, or by asking for help (Bers & Horn, 2010).

Collaboration. Most educational robotic programs for older children, such as the National Robotics Challenge and FIRST (For Inspiration and Recognition of Science and Technology), are set up as competitions in which robots have to accomplish a given task, usually with the goal of outperforming other robots. However, research has shown that females do not tend to respond well to teaching strategies that stress competition (Turbak & Berg, 2002); such strategies also might not always be appropriate in the early childhood setting (Bers, 2008a). The TangibleK learning environment,

instead of focusing on competition, promotes having children work in teams, sharing resources, and *caring* about each other. The use of collaboration webs fosters collaboration. At the beginning of each day of work, each child receives, along with the design journal, a personalized printout with his or her photograph in the center of the page and the photographs and names of all other children in the class arranged in a circle surrounding that central photo (see Figure 2). Throughout the day, at the teacher's prompting, each child draws a line from his or her own photo to the photos of the children with whom he or she has collaborated. (Collaboration is defined here as getting or giving help with a project, programming together, lending or borrowing materials, or working together on a common task.) At the end of the week, children write or draw "thank you cards" to the children with whom they have collaborated the most.

Figure 2. Example of a blank collaboration web.

Communication. Communication is an important feature of the TangibleK approach, which includes mechanisms that promote a sense of *connection* between peers or between peers and adults. One feature that encourages communication is technology circles. During technology circles, children and adults stop their work, put their projects on the table or floor, sit down in a circle together, and share the state of their projects (Bers, 2008a). This is similar to other circle times that children are exposed to in kindergarten (Kantor, Elgas, & Fernie, 1989). Technology circles present an opportunity for debugging as a community—that is, for solving technical problems in programming or building. The teacher starts the technology circle by asking children to show their projects and asking questions such as, "What worked as expected and what didn't?", "What are you trying to accomplish?", "What do you need to know in order to make it happen?" The teacher then uses children's projects and questions to highlight powerful ideas illustrated by the projects. The curriculum emerges based on what this particular learning community needs to know. This approach provides technical information on-demand, based on emerging needs, and is an alternative to lectures. Technology circles can be called as often as every 20 minutes at the beginning of a project or only once at the end of a day of work, depending on the needs of the children and the teacher's need to introduce new concepts.

Community Building. Community-building techniques in TangibleK programs scaffold support networks that promote each child's *contribution* to the learning environment and community. In the spirit of the Reggio Emilia approach (started in municipal infant-toddler centers and preschools of Reggio Emilia, Italy, after World War II), the children's projects are shared with the community via an open house, demonstration day, or exhibition (Rinaldi, 1998). An open house provides authentic opportunities for children to share and celebrate the processes and tangible products of their

learning with others who are invested in their learning, such as family, friends, and community members. These public displays make learning visible to others and to the children themselves.

Choices of Conduct. TangibleK activities provide opportunities for children to experiment with "what if" questions and consider potential consequences of their own choices. Choices of conduct are not only made by children; teachers also make important decisions that affect what the children do. For example, if the LEGO building pieces are sorted by types and placed in bins in the center of the room (instead of given to each child or group as a presorted robotic kit), children learn to take what they need without depleting the bins of the "most wanted" pieces, such as special sensors or the colorful LEGO minifigures. They also learn how to negotiate for what they need. For teachers using the TangibleK program, helping children develop an inner compass to guide their actions in a just and responsible way is as important as the focus on learning about robotics. The program's emphasis on choices of conduct may provoke examination of values and exploration of *character traits.* Differentiation of roles can be important to the growth of a responsible learning community. In any classroom, for example, one child may learn very quickly about mechanics, while another may become a programming expert, and still another may easily problem-solve or skillfully mediate conflicts among group members. Such children may be assigned "expertise badges" by teachers or by the other children. Those children who are seen as especially skilled at something can make the choice to help classmates build a bigger structure or address other challenges. Children are also encouraged to take on new roles and be flexible; there is a badge for "expert on trying new things."

Overview of the TangibleK Curriculum

The TangibleK curriculum introduces and uses six powerful ideas from computer science in a robotics context in a structured, developmentally appropriate way—robotics, engineering design process, sequencing and control flow, loops and parameters, sensors, and branches. See Table 1 for descriptions of these concepts. Themes vary in the pilot versions of this curriculum—transportation, community, animals—but the powerful ideas of computer science and robotics remain the same. The transportation curriculum is used here as an example.

Table 1

Powerful Ideas, Definitions, Activities, and Disciplinary Connection of the TangibleK Program

Powerful Idea	Definition	Activity	Discipline Connections
Robotics	The engineering discipline that focuses on the creation and programming of robots, machines that can follow instructions and move on their own to perform tasks.	*What Is a Robot?* After an introduction to robotics by looking at different robots and talking about the functions they serve, children build their own robotic vehicles and explore the parts and instructions they can use to program them.	• Engineering • Computer Science
Engineering Design Process	A process used to develop products to solve a need or problem. It has several iterative steps: identifying a need or defining the problem, doing research, analyzing possible solutions, developing the product, communicating, and presenting the product.	*Sturdy building:* Children build a nonrobotic vehicle to take small toy people from home to school. The vehicle needs to be sturdy and able to perform its intended functions. *Design journals:* Children will use the design journals to learn the engineering design process. *All programming activities.* (see below)	• Engineering • Computer science
Sequencing/Control	A sequence of instructions can	*The Hokey-Pokey:* Choose the	

Flow	be described in a program and acted out in order by a robot. Each block has a specific meaning. The order of the blocks is important.	appropriate commands and put them in order to program a robot to dance the Hokey-Pokey.	• Creative storytelling • Organization of ideas • Mathematical proofs • Procedural thinking
Loops and Parameters	A sequence of instructions can be modified to occur over and over again. Control flow commands can be qualified with additional information. For example, loops can be modified to repeat forever or a concrete number of times.	*Again and Again until I Say When:* Students use a pair of loop blocks ("repeat"/"end repeat") to make the robot go forward again and again, infinitely, and then just the right number of times to arrive at a fixed location.	• Cyclical events in nature • Scheduling • Timing and control • Feedback loops • Number sense
Sensors	A robot can use sensors, akin to human sense organs, to gather information from its environment. Sensor information can be used to control when the robot follows given commands.	*Through the Tunnel:* Children use light sensors and commands to program a robot to turn its lights on when its surroundings are dark and vice versa.	• Scientific observations • Cause and effect • Sensors (both human-made and natural)
Branches	At a branch in the program, a robot can follow one set of commands or another depending on the state of a given condition.	*The Robot Decides:* Students program their robots to travel to one of two destinations based on light or touch sensor information.	• Cause and effect • Sensors (both human-made and natural)

The TangibleK curriculum involves a minimum of 20 hours of classroom work, divided into the following structured sessions based on the six powerful ideas identified above:

- Session 1. Sturdy Building (the engineering design process)
- Session 2. What Is a Robot? (robots have special parts to follow instruction)
- Session 3. Hokey-Pokey: sequence of commands (the sequence or order of commands matters)
- Session 4. Again and Again until I Say When (loops and number parameters)
- Session 5. Through the Tunnel (sensors and loops)
- Session 6. The Robot Decides (sensors and branches)

Table 2 describes the six lessons in the curriculum, with learning objectives stated in terms of what the students should understand and what they should be able to do.

Table 2

Objectives for Each of the Six Lessons

Lesson	Students will understand that...	Students will be able to ...
Sturdy Building (engineering design process)	• LEGO bricks and other materials can fit together to form sturdy structures. • The engineering design process is useful for planning and guiding the creation of artifacts.	• Build a sturdy, nonrobotic vehicle using LEGO bricks and other materials. • Use design journals to learn the engineering design process.
What Is a Robot? (robots have special parts to follow instructions)	• Robots need moving parts, such as motors, to be able to perform behaviors specified by a program. • The robotic brain (RCX) has the programmed	• Describe the components of a robot, including the brain (RCX), motors, and wires. • Upload a program to a robot via the

	instructions that make the robot perform its behaviors. • The RCX must communicate with the motors for the motors to function.	tangible blocks or graphical icons, computer interface, and LEGO IR tower. • Build a sturdy, robotic vehicle using LEGO bricks and other materials.
Hokey-Pokey (sequence of commands matters)	• Each icon corresponds to a specific command. • A program is a sequence of commands that is followed by a robot. • The order of the blocks dictates the order in which the robot executes the commands.	• Select the appropriate block corresponding to a planned robot action. • Connect a series of blocks. • Upload a program to the computer and transmit it to a robot.
Again and Again until I Say When (loops and number parameters)	• A command or sequence of commands may be modified to repeat. • Some programming commands, like "Repeat," can be modified with additional information. • A simple program that uses fewer blocks is better than a more complex one that accomplishes the same goal.	• Recognize a situation that requires a program to use loops. • Write a program that loops. • Use parameters to modify the number of times a loop runs before the program stops.
Through the Tunnel (sensors and loops)	• Through the use of sensors, a robot can feel and see its surroundings. • A robot can react to collected data by changing its behavior. • A robot can be programmed to remain on a certain task until a specific condition is met.	• Connect a light or touch sensor to the correct port on the RCX. • Write a program that includes waiting for a specific condition.
The Robot Decides (sensors and branches)	• A robot can "choose" between two sequences of commands depending on the state of a given condition.	• Connect a light or touch sensor to the correct port on the RCX. • Identify a situation that calls for a branched program. • Write a program that uses a branch.

Each session follows a similar basic format: (1) warm-up games to introduce the new concept or powerful idea in a playful way, (2) a building and/or programming task to reinforce the powerful idea underlying the lesson, (3) work on a small project (individually or in pairs) that uses the powerful idea in a new context, (4) technology circle, and (5) assessment.

After the six TangibleK sessions, it is time for the class to create a final project. This is an opportunity to revisit the learned concepts and skills, applying them to a project of their own choice. The length of time for these projects varies according to the classroom and the teachers' goals, expectations, and curricular demands. These final projects will be shared in an open house for the wider community.

Examples of children's TangibleK final projects include a robotic city, a zoo with moving animals, a dinosaur park, a circus, and a garden with robotic flowers responsive to different sensors (Bers, 2008a). These projects all incorporated use of inexpensive recyclable materials. For example, one kindergarten classroom in Boston, after a field trip to the old city, constructed a robotic Freedom Trail, using cardboard boxes to re-create the historical buildings of the city and embedding light sensors and motors into the boxes to bring their buildings to life (Bers, 2008a).

Overview of TangibleK Robotics Technology

One of the research questions that the TangibleK robotics program set out to explore is "What design elements should a developmentally appropriate robotic kit include to engage young children in a

successful learning experience?" The TangibleK program is based on evidence collected during almost a decade of research with young children and robotics (Bers, 2000; Cejka et al., 2006; Bers, 2008a). In that time, different interfaces and technologies have been designed, implemented, and evaluated (Martin et al., 2000), but no steady effort has been made to create interfaces that are developmentally appropriate for young children. The TangibleK program draws from the field of human-computer interaction (HCI) on tangible interfaces that show that offering tangible systems can overcome the inherent limitations of writing computer programs with a keyboard or mouse (Blikstein, Buechley, Horn, & Raffle, 2010).

A tangible programming language, like text-based and icon-based languages, is a tool for giving instructions to a computer. Hybrid systems also exist. With a text-based language, a programmer uses words such as BEGIN, IF, and REPEAT to instruct a computer. This code must be written according to strict, often frustrating, syntactic rules. LOGO is an example of a text-based language for children (Papert, 1980). An icon-based or visual language replaces words with pictures; programs are expressed by arranging and connecting icons on the computer screen with the mouse. Robolab (Rogers & Portsmore, 2004) and SCRATCH (Resnick et al., 2009) are icon-based languages for children. Syntax rules must still be followed but can be conveyed to the programmer through a set of visual cues.

With tangible languages, instead of relying on pictures and words on a computer screen, users arrange and connect physical objects to construct computer programs. Tangible languages exploit physical properties of objects (size, shape, materials, etc.) to express and enforce syntax. The idea of tangible programming was first introduced in the mid-1970s (Perlman, 1976) and was revived nearly two decades later (Suzuki & Kato, 1995). Since then, several tangible languages have been created in research labs around the world (e.g., McNerney, 2004; Wyeth, 2008; Smith, 2007; Horn & Jacob, 2007).

Researchers at Tufts have taken a hybrid approach, allowing children to transition back and forth between screen-based and tangible programming languages (Bers & Horn; 2010). This system, called CHERP (Creative Hybrid Environment for Robotic Programming), allows children to program with interlocking wooden blocks (Figure 3). Children can transition back and forth between the blocks and on-screen programs using icons of the blocks to represent actions for the robots to perform. This hybrid approach allows children to work with multiple representations (Horn et al., in press).

Figure 3. Tangible and on-screen elements of the CHERP language developed at Tufts University.

CHERP uses a collection of image-processing techniques to convert physical programs into digital instructions. Each block in the language is imprinted with a circular symbol called a TopCode (Horn, Bers, & Jacob, 2009). These codes allow the position, orientation, size, shape, and type of each statement to be quickly determined from a digital image. A standard webcam connected to a desktop

or laptop computer takes a picture of the program. A compiler converts the picture into digital code that is downloaded to the robot in a matter of seconds (see Figure 4).

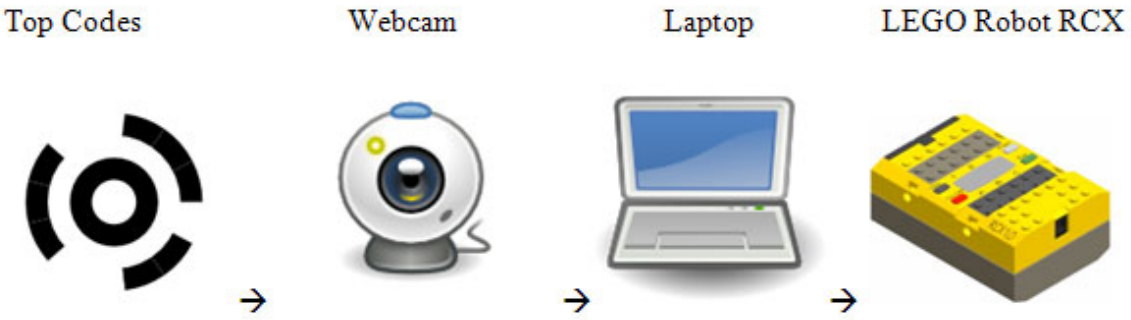

Figure 4. The CHERP download process.

The TangibleK program uses CHERP in combination with different robotic kits. The programs that children create with the tangible blocks can be downloaded to the LEGO RCX brick, an embedded microcomputer or the "robot brain," contained in the LEGO Mindstorms kit. Tufts researchers are exploring the use of other robotic kits that might not include LEGO pieces, thus providing a cheaper alternative. Discussing such technical options is beyond the scope of this paper.

Connections to Curricular Areas

The TangibleK robotics program is explicitly designed to address "the missing middle letters" of STEM in early childhood education—the T (technology) and the E (engineering). However, for the program to be successfully adopted in classrooms, it must integrate and facilitate the introduction of other curricular content, both in terms of themes and disciplinary concepts and skills (Bers et al., 2002). Table 3 shows possible disciplinary connections between powerful ideas from the TangibleK program and other domains of knowledge.

Table 3

Powerful Ideas from Computer Programming with Cross-Disciplinary Connections

Powerful Idea	Description	Disciplinary Connections
Sequencing/Control Flow	A sequence of instructions can be described in a program and acted out in order by a robot.	• literacy: storytelling, "how to" books • symbolic system • setting up controlled experiments • basic explorations of geometry • cause and effect
Loops	Sequences of instructions can be modified to repeat indefinitely or in a controlled way.	• expanded geometry explorations • time • cycles • symbols are used to communicate a message
Parameters	Some instructions can be qualified with additional information.	• expanded geometry explorations • number sense • timing and control
Branches	Some instructions in a program ask questions and, depending on the answer, have a robot do one thing or another.	• cause and effect • logic • expanded scientific observations

		• executive function skills • decision making
Sensors	Devices measure a change in the environment and convert it into a signal that can be read by an observer, by an instrument, or by a robot.	• natural and human-made world • biology: sensing in animals and humans

TangibleK Robotics and Learning Standards

The powerful ideas underlying the TangibleK curriculum are aligned with the standards developed by the International Technology Education Association (ITEA, 2007) (http://www.iteea.org/TAA/PDFs/xstnd.pdf) and the 2006 Massachusetts Science and Technology/Engineering Curriculum Framework (http://www.doe.mass.edu/frameworks/current.html). Both sets of standards emphasize the importance of learning about components of engineering design in their benchmarks for kindergarten through second grade:

- People plan to help get things done. (Standard 2E; K-2)
- Everyone can design solutions to a problem. (Standard 8A; K-2)
- Design is a creative process (that leads to useful products and systems). (Standard 8B; K-2)
- The engineering design process includes identifying a problem, looking for ideas, developing solutions, and sharing solutions with others. (Standard 9A; K-2)
- Asking questions and making observations help a person to figure out how things work. (Standard 10A; K-2)

Table 4 shows connections with concepts and skills that should be addressed in classrooms according to ITEA standards and the Massachusetts Technology and Engineering framework.

Table 4

Connections between TangibleK Concepts and Skills and ITEA Standards and the Massachusetts Technology and Engineering Framework

Powerful Idea	ITEA Standards	MA Technology/Engineering Framework
Building	• Different materials are used in making things. (Std 2D; K-2) • Materials have many different properties. (Std 2J; Gr 3-5) • Some materials can be reused or recycled. (Std 5A; K-2) • Build or construct an object using the design process (K-2)	• Materials both natural and human-made have specific characteristics that determine how they will be used. (Gr PreK-2; Central Concept 1) • Identify and describe the safe and proper use of tools and materials. (TE Gr PreK-2; Std 1.1) • Appropriate materials, tools, and machines enable us to solve problems, invent, (and construct). (TE Gr 3-5 (6-8); Central Concept 1)
Robotics	• Build or construct an object using the design process. (Std 11B; K-2) • Discover how things work. (Std 12A; K-2) • Systems have parts that work together to accomplish a goal. (Std 2B; K-2) • Tools, machines use energy to do	• With teacher direction, use appropriate technology tools [...] to define problems and propose hypotheses. (TL Gr 3-5; Std 3.6)

	work. (Std 16D; Gr 3-5)	
Programming	• Recognize and use everyday symbols. (Std 12C; K-2) • People use symbols when they communicate by technology. (Std 17C; K-2) • The study of technology uses many of the same ideas and skills as other subjects. (Std 3A; K-2)	• Identify and explain how symbols and icons [...] are used to communicate a message. (TE Gr 6-8; Std 3.4)
Sensors	• The natural world and human-made world are different. (Std 1A; K-2)	• Identify and describe characteristics of natural and human-made materials. (K-2) • Human beings and animals use parts of the body as tools. (PreK-2)

Assessment

The PTD framework discussed earlier provides guidelines both for designing the educational program and for evaluating children's learning and development. *Content creation* and *creativity* (the first two "Cs") are evaluated in terms of competence (or level of understanding) and confidence in the domain. The following are used to assess content creation and creativity:

- Student's portfolios—composed of student's design journals, student's programming samples (code), and student's robotic projects. Change over time in the level of sophistication and complexity is assessed.
- Video journals—videotapes made at least three times during the program (e.g., beginning, middle, and end) of children showing what they have been working on and explaining their activities.
- SSS rubric of levels of understanding—a set of questions for the teacher or researcher to complete at the end of each session to assess each child's level of understanding (syntactic, semantic, or systemic, and the transitions among levels) on a scale of 0 to 5, for each of the learning objectives throughout the curriculum and for each of the tasks at different levels of complexity. The following example shows the assessment rubric for session 2, during which children program their robots to dance the hokey-pokey and which introduces the powerful idea that "the order or sequence of instructions matters," a key concept in computational thinking.

> *Part 1:* Circle the corresponding level of achievement for each child for each statement listed. Circle NA if the statement could not be assessed during this activity.

5	4	3	2	1	0
Achieves without assistance	Achieves with minimal assistance	Achieves with periodic assistance	Achieves with significant assistance	Achieves with step-by-step instructions	Does not achieve

A. Works purposefully toward the goal of the activity.	5 4 3 2 1 0 NA
B. Works purposefully toward self-selected goal. If so, what goal?	5 4 3 2 1 0 NA
If working on a self-selected goal, s/he can articulate the goal.	5 4 3 2 1 0 NA
Translates his/her ideas into code for the robot to act out.	5 4 3 2 1 0 NA
Arranges blocks or icons in a syntactically correct sequence to make a functional program.	5 4 3 2 1 0 NA

A. Recognizes incorrect actions or order in a program by reading the program or watching the robot run the program.	5 4 3 2 1 0 NA
B. Has a hypothesis of the problem.	5 4 3 2 1 0 NA
C. Attempts to solve the problem.	5 4 3 2 1 0 NA
Selects correct instructions for the program based on their corresponding actions.	5 4 3 2 1 0 NA
Robot maintains its core integrity while being handled and while it runs programs.	5 4 3 2 1 0 NA
Understands that programs made in the TUI and GUI must be translated and sent to the robot by the computer.	5 4 3 2 1 0 NA

Part 2: Ask each child questions such as, "When you make a program, does it make a difference what order you put the blocks in?" to supplement the students' journals and the teachers' observations of and conversations with students during work and sharing times. Mark the students' level of understanding of how to program a robot along the following criteria.

Syntactic:	Understands the function of individual instructions but not how to choose and assemble them to make a functional program that accomplishes a given goal.
Semantic:	Chooses appropriate instructions for the program and puts them in the right order. Understands that putting the parts together in certain ways creates an overall outcome. May not be able to create a program that completely meets the given goal, or may not realize it when the goal is met.
Systems:	Understands the function of each element and that the order they are put in results in a specific overall outcome. Is able to purposefully put the right instructions in the right order for the program to achieve the given goal.

Figure 5. An SSS rubric.

Most children do not fall neatly into any of these categories but in the transitions between the categories. Current data analysis is being used to construct learning trajectories.

Children's collaboration and communication are evaluated in terms of the levels of caring and connection achieved by the children by analyzing the collaboration webs over time (Lee & Bers, 2010) and the children's participation in the technology circles.

Finally, community building and choices of conduct are evaluated by looking at a child's overall participation and engagement in the TangibleK program and her *contributions* to the learning environment, in particular during the final project presented at the open house. Expertise badges are seen as representative of the child's character traits. Change over time is analyzed.

Assessment in the TangibleK program, unlike most programs focused on technological literacy, addresses not only the cognitive dimension but extends also to the social and the moral dimensions of the child's experience through and with the technology, toward a goal of helping the child develop in an integrated and holistic way.

In terms of research, the evaluation goals guiding the TangibleK robotics program are twofold. The first goal is to provide an evidence-based systematic account of children's learning and use of the powerful ideas identified in the curriculum, employing methods described earlier corresponding to components of the Positive Technology Development (PTD) framework. The second goal is to establish potential learning trajectories in this disciplinary context that consider developmental progressions and epistemological foundations, as well as to establish specific activities or tasks with incremental levels of difficulty, matched to the robotics powerful ideas and to the children's levels of understanding (Clements & Sarama, 2004; Clements & Sarama, 2009). Describing the research aspects of this program is beyond the scope of this article. Future work related to TangibleK robotics will focus on developing new curriculum modules, implementing a less costly robotics system that uses everyday materials, and constructing a solid theoretical model for learning trajectories in this

area.

Conclusion

This is a challenging time for early childhood education. The academic demands of federal mandates may be in opposition to a rising concern about respect for children's developmental needs. In order to advance the technological fluency of our nation's youth, we need to start in the early years. However, there has been no profound reexamination of content that young children are able to learn, in particular in the areas of technology and engineering. Nor do we have a research base sufficient to evaluate what children are able to learn with innovative technologies and how they learn it. The research on and development of the TangibleK robotics program contributes to the research base related to bringing ideas of computer science and engineering into the early childhood classroom. Experience with the TangibleK program to date suggests that given age-appropriate technologies, curriculum, and pedagogies, young children can actively engage in computer programming and robotics activities in ways that are consistent with developmentally appropriate practice.

Acknowledgments

I sincerely thank my students at the Developmental Technologies Research Group at Tufts University: Jordan Crouser, Rachael Fein, Louise Flannery, and Elizabeth Niro, who all worked on different aspects of this project. I also thank my former student and now colleague at Northwestern University, Mike Horn, for developing TERN while working on his Ph.D. at Tufts and for helping me get this work started, and my colleagues Robert Jacob from the Human-Computer Interaction Laboratory and Chris Rogers from the CEEO at Tufts University. Finally, my thanks to the National Science Foundation for support of this research (NSF Grant DRL-0735657 and NSF Career award IIS-0447166). Any opinions, findings, and conclusions or recommendations expressed in this article are those of the author and do not necessarily reflect the views of the National Science Foundation.

References

Bers, Marina Umaschi, & Urrea, Claudia. (2000). Technological prayers: Parents and children exploring robotics and values. In Allison Druin & James Hendler (Eds.), *Robots for kids: Exploring new technolgies for learning* (pp. 194-217). San Francisco: Morgan Kaufmann.

Bers, Marina Umaschi. (2006). The role of new technologies to foster positive youth development. *Applied Developmental Science, 10*(4), 200-219.

Bers, Marina Umaschi. (2008a). *Blocks to robots: Learning with technology in the early childhood classroom.* New York: Teachers College Press.

Bers, Marina U. (2008b). Engineers and storytellers: Using robotic manipulatives to develop technological fluency in early childhood. In Olivia N. Saracho & Bernard Spodek (Eds.), *Contemporary perspectives on science and technology in early childhood education* (pp. 105-125). Charlotte, NC: Information Age.

Bers, Marina Umaschi. (2010). When robots tell a story about culture...and children tell a story about learning (pp. 227-247). In Nicola Yelland (Ed.), *Contemporary perspective on early childhood education.* Maidenhead, UK: Open University Press.

Bers, Marina Umaschi, & Horn, Michael S. (2010). Tangible programming in early childhood: Revisiting developmental assumptions through new technologies. In Irene R. Berson & Michael J. Berson (Eds.), *High-tech tots: Childhood in a digital world* (pp. 49-70). Greenwich, CT: Information Age.

Bers, Marina; Doyle-Lynch, Alicia; & Chau, Clement. (in press). Positive technological development:

The multifaceted nature of youth technology use towards improving self and society. In Cynthia Carter Ching & Brian Foley (Eds.), *Technology, learning, and identity: Research on the development and exploration of selves in a digital world.* Cambridge: Cambridge University Press.

Bers, Marina U.; Ponte, Iris; Juelich, Katherine; Viera, Alison; & Schenker, Jonathan. (2002). Teachers as designers: Integrating robotics into early childhood education. *Information Technology in Childhood Education Annual,* 123-145.

Blikstein, Paulo; Buechley, Leah; Horn, Michael; & Raffle, Hayes. (2010). A new age in tangible computational interfaces for learning. In K. Gomez, L. Lyons, & J. Radinsky (Eds.), *Proceedings of the 9th International Conference of the Learning Sciences* (Vol. 2, pp. 130-132). Chicago: International Society of the Learning Sciences.

Brosterman, Norman. (1997). *Inventing kindergartem.* New York: H.N. Abrams.

Cejka, Erin; Rogers, Chris; & Portsmore, Merredith. (2006). Kindergarten robotics: Using robotics to motivate math, science, and engineering literacy in elementary school. *International Journal of Engineering Education*, *22*(4), 711-722.

Clements, Douglas H. (1999). The future of educational computing research: The case of computer programming. *Information Technology in Childhood Education Annual,* 147-179.

Clements, Douglas H., & Sarama, Julie (Eds.). (2004). Hypothetical learning trajectories. *Mathematical Thinking and Learning* [Special issue]*, 6*(2).

Clements, Douglas H., & Sarama, Julie. (2009). *Learning and teaching early math:The learning trajectories approach*. New York: Routledge.

Craig, John J. (2005). *Introduction to robotics.* Upper Saddle River, NJ: Pearson Prentice Hall.

Damon, William. (2004). What is positive youth development? *Annals of the American Academy of Political and Social Science, 591*(1), 13-24.

Horn, Michael S.; Bers, Marina U.; & Jacob, Robert J. K. (2009, April). *Tangible programming in education: A research approach.* Paper presented at CHI '09. Boston, MA.

Horn, Michael S., & Jacob, Robert J. K. (2007). Designing tangible programming languages for classroom use. In Brygg Ullmer, Albrecht Schmidt, Eva Hornecker, Caroline Hummels, Robert J. K. Jacob, & Elise van den Hoven, *TEI '07: Proceedings of the First International Conference on Tangible and Embedded Interaction* (pp. 159-162). New York: Association for Computing Machinery.

Horn, Michael S.; Crouser, R. Jordan; & Bers, Marina U. (in press). Tangible interaction and learning: The case for a hybrid approach. *Personal and Ubiquitous Computing* [Special issue on tangibles and children].

International Technology Education Association (ITEA). (2007). *Standards for technological literacy: Content for the study of technology* (3rd ed.). Reston, VA: Author.

Kafai, Yasmin B., & Resnick, Michael. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*. Mahwah, NJ: Erlbaum.

Kantor, Rebecca; Elgas, Peggy M.; & Fernie, David E. (1989). First the look and then the sound: Creating conversations at circle time. *Early Childhood Research Quarterly, 4*(4), 433-448.

Kay, Alan C. (2003). Interview with Alan Kay. *Computers in Entertainment, 1*(1), 8.

Larson, Reed W. (2000). Toward a psychology of positive youth development. *American Psychologist, 55*(1), 170-183.

Lee, K. T., Bers, Marina. (2010). *Collaboration by design: Using robotics to foster social interaction in kindergarten.* Manuscript submitted for publication.

Lerner, Richard M.; Almerigi, Jason B.; Theokas, Christina; & Lerner, Jacqueline V. (2005). Positive youth development: A view of the issues. *Journal of Early Adolescence, 25*(1), 10-16.

Martin, Fred; Mikhak, Bakhtiar; Resnick, Mitchel; Silverman, Brian; & Berg, Robbie. (2000). To mindstorms and beyond: Evolution of a construction kit for magical machines. In Allison Druin & James Hendler (Eds.), *Robots for kids: Exploring new technolgies for learning* (pp. 9-33). San Francisco: Morgan Kaufmann.

Massachusetts Department of Education. (2006, October). *Massachusetts science and technology/engineering curriculum framework*. Retrieved October 7, 2010, from http://www.doe.mass.edu/frameworks/current.html

McNerney, Timothy S. (2004). From turtles to tangible programming bricks: Explorations in physical language design. *Personal and Ubiquitous Computing, 8*(5), 326-337.

Papert, Seymour. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Papert, Seymour. (1993). *The children's machine: Rethinking school in the age of the computer*. New York: Basic Books.

Papert, Seymour. (2000). What's the big idea: Towards a pedagogy of idea power. *IBM Systems Journal, 39*(3-4), 720-729.

Pearson, Greg, & Young, Thomas A. (2002). *Technically speaking: Why all Americans need to know more about technology*. Washington DC: National Academy Press.

Perlman, Radia. (1976). Using computer technology to provide a creative learning environment for preschool children. *AI Memo 360: Logo Memo No. 24.* Cambridge, MA: MIT Artificial Intelligence Laboratory.

Random House Webster's Unabridged Dictionary. (2006). New York: Random House.

Repenning, Alexander; Webb, David; & Ioannidou, Andri. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. *SIGCSE 2010: Proceedings of the 41st ACM technical symposium on computer science education*. New York: Association for Computing Machinery.

Resnick, Mitchel. (1996). New paradigms for computing, new paradigms for thinking. In Yasmin B. Kafai & Mitchel Resnick (Eds.), *Constructionism in practice: Designing, thinking, and learning in a digital world*. Mahwah, NJ: Erlbaum.

Resnick, Mitchel. (2003). Playful learning and creative societies. *Education Update*, *8*(6), Retrieved May 1, 2009, from http://web.media.mit.edu/~mres/papers/education-update.pdf

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

Resnick, Mitchel; Maloney, John; Monroy-Hernández, Andrés; Rusk, Natalie; Eastmond, Evelyn; Brennan, Karen; Millner, Amon; Rosenbaum, Eric; Silver, Jay; Silverman, Brian; & Kafai, Yasmin. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60-67.

Rinaldi, Carlina. (1998). Projected curriculum constructed through documentation—Progettazione: An interview with Lella Gandini. In Carolyn Edwards, Lella Gandini, & George Forman (Eds.), *The hundred languages of children: The Reggio Emilia approach—Advanced reflections* (2nd ed., pp. 113-126). Greenwich, CT: Ablex.

Rogers, Chris, & Portsmore, Merredith. (2004). Bringing engineering to elementary school. *Journal of STEM Education, 5*(3-4), 17-28.

Rogers, Chris B.; Wendell, Kristen; & Foster, Jacob. (2010). The academic bookshelf: A review of the NAE Report, "Engineering in K-12 education." .*Journal of Engineering Education, 99*(2), 179-181. Retrieved October 6, 2010, from http://findarticles.com/p/articles/mi_qa3886/is_201004/ai_n53931016/

Rogoff, Barbara. (2003). *The cultural nature of human development*. New York: Oxford University Press.

Rogoff, Barbara; Turkanis, Carolyn Goodman; & Bartlett, Leslee. (2001). *Learning together: Children and adults in a school community*. New York: Oxford University Press.

Rusk, Natalie; Resnick, Mitchel; Berg, Robbie; & Pezalla-Granlund, Margaret. (2008). New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology, 17*(1), 59-69.

Scales, Peter C.; Benson, Peter L.; Leffert, Nancy; & Blyth, Dale A. (2000). Contribution of developmental assets to the prediction of thriving among adolescents. *Applied Developmental Science, 4*(1), 27-46.

Smith, A. C. (2007). Using magnets in physical blocks that behave as programming objects. In Brygg Ullmer, Albrecht Schmidt, Eva Hornecker, Caroline Hummels, Robert J. K. Jacob, & Elise van den Hoven, *TEI'07: Proceedings of the First International Conference on Tangible and Embedded Interaction* (pp. 147-150). New York: Association for Computing Machinery.

Suzuki, Hideyuki, & Kato, Hiroshi. (1995). Interaction-level support for collaborative learning: Algoblock—an open programming language. In John L. Schnase & Edward L. Cunnius (Eds.), *Proceedings of CSCL '95: The First International Conference on Computer Support for Collaborative Learning* (pp. 349-355). Mahwah, NJ: Erlbaum.

Theokas, Christina, & Lerner, Richard M. (2006). Observed ecological assets in families, schools, neighborhoods: Conceptualization, measurement, and relations with positive and negative developmental outcomes. *Applied Development Science, 10*(2), 61-74.

Turbak, Franklyn, & Berg, Robbie. (2002). Robotic design studio: Exploring the big ideas of engineering in a liberal arts environment. *Journal of Science Education and Technology, 11*(3), 237–253.

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

Wang, X. Christine, & Ching, Cynthia Carter. (2003). Social construction of computer experience in a first-grade classroom: Social processes and mediating artifacts. *Early Education and Development, 14*(3), 335-361.

Wing, Jeannette M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. Retrieved June 10, 2010, from http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.pdf

Wyeth, Peta. (2008). How young children learn to program with sensor, action, and logic blocks. *International Journal of the Learning Sciences, 17*(4), 517-550.

Author Information

Marina Umaschi Bers, Ph.D., is an associate professor at the Eliot-Pearson Department of Child Development and the Computer Science Department at Tufts University. Her research involves the design and study of innovative learning technologies to promote positive youth development. At Tufts, Professor Bers heads the interdisciplinary Developmental Technologies research group. Professor Bers received the 2005 Presidential Early Career Award for Scientists and Engineers (PECASE), the highest honor given by the U.S. government to outstanding investigators at the early stages of their careers.

She also received a five-year National Science Foundation (NSF) Young Investigator's Career Award and the American Educational Research Association's (AERA) Jan Hawkins Award for Early Career Contributions to Humanistic Research and Scholarship in Learning Technologies. Over the past 14 years, Professor Bers has conceived and designed diverse technological tools, ranging from robotics to virtual worlds. She conducted studies in after-school programs, museums, and hospitals, as well as schools in the United States and abroad. Her book *Blocks to Robots: Learning with Technology in the Early Childhood Classroom* was published by Teachers College Press in 2008. Her upcoming book *Positive Uses of New Technologies: The Design of Digital Experiences for Youth Learning and Development* will be published in 2012 by Oxford University Press. Dr. Bers is from Argentina, where she did her undergraduate studies in social communication at Buenos Aires University. In 1994, she came to the United States, where she received a master's degree in educational media and technology from Boston University and a master's of science and Ph.D. from the MIT Media Laboratory, working with Seymour Papert. For more on Dr. Bers, please visit http://www.tufts.edu/~mbers01/.

Marina U. Bers, Ph.D.
Associate Professor
DevTech Research Group
Eliot Pearson Department of Child Development
Computer Science Department
Tufts University
105 College Ave
Medford, MA 02155
Telephone: 617-627-4490
Fax: 617-627-3503
Email: Marina.Bers@tufts.edu