Volume 71, February 2014
ISSN 0360-1315
ELSEVIER
Computers & Education
An International Journal
Editors:
Rachelle S. Heller
Jean D.M. Underwood
Chin-Chung Tsai
MAKE AN IMPACT
Impact Factor
2.617
GET READ
Available online at www.sciencedirect.com
ScienceDirect

Computers & Education 71 (2014) 257–278

Contents lists available at ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Learning from the folly of others: Learning to self-correct by monitoring the reasoning of virtual characters in a computer-supported mathematics learning environment

Sandra Y. Okita*

Teachers College, Columbia University, 525 West 120th Street, New York, NY 10027, United States

A R T I C L E I N F O

Article history:
Received 28 March 2013
Received in revised form
18 September 2013
Accepted 24 September 2013

Keywords:
Metacognition
Teaching/learning strategies
Peer learning
Computer-supported learning
Elementary school mathematics

A B S T R A C T

Two studies examined the social basis of self-assessment for learning through the application of creative computer tools that can help students assess and self-correct their own learning. Students are not usually inclined to check their own answers, but they find it relatively motivating to catch other people's mistakes. A total of 62 students, ranging in age from nine to 11, participated in two studies that tested the hypothesis that monitoring "someone" else (i.e., computer character) can help students learn to self-assess their own learning. Two computer-supported learning environments, i.e., "Doodle Math" and "Puzzle Math," were developed as training environments for monitoring. The environments also allowed a direct comparison between self training and self-other training. In the training environment, a computer character, "ProJo," openly displayed its reasoning when solving math problems and allowed children to "look for mistakes." The students in self training solved all the problems on their own, while the students in self-other training worked with the computer character, ProJo, taking turns to solve problems and monitor for any mistakes. The measures on calculation time and accuracy showed that self-other training might be an effective way to help students develop metacognitive skills to self-correct and improve performance in elementary mathematics. The log file tracked the students' progress in various data forms and displayed evidence that self-other training students monitored and self-corrected more than students who experienced self training.

1. Introduction

A critical skill in helping children continue to learn after they leave school is the ability to self-assess the progress of one's learning. This is because students need to be able to "learn for themselves" and make informed decisions in the future. This metacognitive skill of self-assessing and self-correcting also may help students address their difficulties in mathematics at the elementary grade level. Often, students understand the principles and procedures, but they make careless mistakes. These mistakes can range from process problems and precedence errors to substitution and procedural errors, which are common among elementary and middle-school students. Repeating these errors can be quite serious because it often can lead to long-term difficulties with mathematics (Watson, 1980) and the development of a negative perception of math and science (Steele, 2003).

Careless mistakes can easily be avoided if children learn to check their answers. Metacognitive skills to self-assess and self-correct are critical, but teachers have little time to spend on these problems in school, as habit-correcting behavior requires personal attention, time, and consistency on the part of the enforcer (Blando, Kelly, Schneider, & Sleeman, 1989; Wells & Coffey, 2005). One way for children to minimize mistakes and avoid discouragement is to monitor their own thinking and activity. This way, they can catch potential mistakes or avoid confusion. However, the thought of checking their answers may not occur voluntarily in children unless they have developed metacognitive skills to recognize confusion and articulate their reasoning to some degree.

This research examines the hypothesis that children learn to self-monitor by monitoring other people, and with practice, they turn this external monitoring inward. By practicing the metacognitive strategy on others, children develop awareness and proficiency in monitoring,

* Tel.: +1 212 678 4165, +1 650 799 6439 (Cell); fax: +1 212 678 8227.
E-mail addresses: so2269@columbia.edu, okita@tc.columbia.edu.

0360-1315/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compedu.2013.09.018

and ideally, they turn the external monitoring behavior inward after the behavior becomes natural. One inspiration for this theory was that self-monitoring is cognitively demanding because children confront the dual task of solving and checking problems at the same time. Letting children monitor other people's problem-solving activities may alleviate some of the pressure of the dual demands, because children do not need to fully attend to solving the problem themselves when they are observing others.

The second inspiration for the theory was evidence that children find it relatively easy and motivating to catch other people's mistakes, even if they are not inclined to catch their own. This led to the thinking that monitoring may be easier if practiced on others than on oneself. Children may find it difficult to be attentive to their own mistakes when they are concentrating on a problem or task. However, they may find it relatively easy to catch other people's mistakes (Gelman & Meck, 1983).

The third inspiration was the strength of technology in creating specifiable environments in which the system can simulate the thoughts an individual might use to reason about a situation. This is different from most simulations that depict empirical sequences of actions that resemble the phenomena the system presents (e.g., cellular processes displaying changes over time, continental drift). For example, Chin, Dohmen, and Schwartz (2013) created a series of software environments called 'Teachable Agents' that visually modeled how to reason through causal chains. Monitoring the computer character's reasoning helped students learn relational predicates and recognize gaps in their own understanding (Okita & Schwartz, 2013). The ideal, computer-supported learning environment would be a situation in which students could practice their monitoring skills externally on a computer character that openly solved math problems. Then, students could monitor the computer character's reasoning when solving math problems. Developing monitoring skills on the external plane (e.g., artifacts, agents) may put children in a better position to internalize these skills. This type of support seems ideal in a computer-supported learning environment that can assist students as they practice monitoring and self-correcting.

The research presented here proposes a somewhat self-driven, but social approach. Two studies examined the hypothesis that helping children learn to monitor other people's problem solving can, in turn, help them learn to monitor their own problem solving and learning. In the context of learning, self-monitoring usually depends on self-assessments. Self-other monitoring is a particularly powerful form of self-assessment in which students assess the knowledge of someone else and, implicitly, assess their own knowledge. While the belief was that this work would lead to the improvement of all students, another interest was to create model technologies that demonstrated how to implement conditions that train students in conducting self-assessment and that generate interest in mathematics among elementary school students in underserved communities.

The following sections include an analytical review of the relevant background literature. In the first experiment, a game-like learning environment, "Doodle Math," was developed to allow a direct comparison between self-other training and self training. The second experiment examined self-other training and self training behaviors over a slightly longer intervention. The new game-like environment, "Puzzle Math," consisted of shorter subtasks to make the reasoning process more explicit for students to monitor and to add a detection feature that recorded the students' self-correcting behaviors. In both studies, the computer character, "ProJo," was designed to openly display its reasoning when solving math problems, allowing children to "check for mistakes."

2. Background

One question that inspired this research was whether the thought of monitoring occurs automatically or whether it is a metacognitive skill that must be learned. The literature addressed different ways in which children learned to monitor and indicated that children might find it relatively easy to catch other people's mistakes, even if they were not inclined to catch their own. This led to the concept that monitoring might be easier if practiced on others' mistakes rather than on one's own mistakes. The section includes background literature on (a) why the dual task of problem solving and monitoring may be difficult for children due to the intuitive and analytical modes of thinking in elementary mathematics, (b) different ways children learn to develop metacognitive monitoring skills in peer learning, and (c) the strength of computer-supported learning environments for practicing monitoring skills.

2.1. Intuitive and analytical modes of thinking and problem solving in elementary mathematics

In the domain of mathematics education, Stanovich and West (2003) found that students were easily susceptible to distractions and irrelevant external cues when solving simple mathematics problems. Gilovich, Griffin, and Kahneman (2002) examined the dual-process theory as a unified framework for explaining people's decision-making and problem-solving behaviors and for explaining the reasons behind responses that contradict the norm (Stanovich & West, 2000, 2003). Dual-process theory (Kahneman, 2002) involves cognition and behavior, operating in parallel in two different modes. Kahneman identified the modes as two different types of cognitive processes, i.e., intuition (referred to as System 1) and reasoning (referred to as System 2). The operations of System 1 (S1) are fast, automatic, usually based on habit, and difficult to control or modify. The operations of System 2 (S2) monitor mental operations that usually involve rule-governed conscious judgments (Stanovich & West, 2000). Usually, S1 and S2 work together, but there are situations in which S1 produces quick, automatic, non-normative responses, while S2 may or may not intervene as the monitor or critic. According to Leron and Hazzan (2006), S1 and S2 in dual-process theory have strong similarities that correspond to intuitive and analytical modes of thinking relevant to mathematical problem solving. The two systems differ mainly with regard to accessibility and how fast and how easily things come to mind. Different nuances exist in dual-process theory, but the work here is focused mainly on the generic framework by Kahneman (2002), Stanovich and West (2000), and Leron and Hazzan (2006), which applies well to the area of mathematics education.

Kahneman (2002) found that both S1 and S2 are prone to errors under complex and abstract conditions, and that S1 can be distracted by irrelevant external clues. Kahneman gave a striking example in elementary mathematics of how S1 can generate a cognitive illusion that overrides the usual analytical reasoning of S2 in people who understand the concept but give fast, reactive responses without thinking. Kahneman found that, when college students were asked to solve this problem: *A baseball bat and ball together cost $1.10. The bat costs one dollar more than the ball. How much does the ball cost?*, they showed an initial tendency to answer 10 cents, because the sum $1.10 could easily separate into $1 and 10 cents, and 10 cents sounded right. Kahneman mentioned that the description of the problem and the rough approximation that sounds "about right" causes S1 to jump to the answer of 10 cents. The result showed that people yielded to this immediate impulse, and under certain circumstances, ignored relevant facts for the sake of immediacy (or competence). This explanation was well described by Kahneman (2002): "The surprisingly high rate of errors in this easy problem illustrates how lightly the output of S1 is monitored by S2: people are not accustomed to thinking hard and are often content to trust a plausible judgment that quickly comes to mind." (pp. 451–452).

The post-interview revealed that, for those students who answered correctly, S1 had initially jumped to the 10 cents, but in the next moment, their S2 monitoring interfered and made the necessary adjustments. This led to the thought that, even in a situation in which S2 is dormant, students should be able to answer correctly if their S1 response were challenged. Thus, the researchers in elementary mathematics education also have been concerned with children's intuitive modes of thinking when solving mathematics problems and have explored ways to strengthen their S2 thinking processes (Fischbein, 1987). According to Leron and Hazzan (2006), the errors must be addressed as combined failures of both S2 and S1. This is because S2 should monitor the operation of S1 (its standard role) as well as the S1/S2 interaction, meaning S2 has to monitor its own functioning in monitoring S1.

Researchers in elementary mathematics education have been concerned with children's intuitive and analytical modes of thinking when solving mathematics problems (Stavy & Tirosh, 2000). Mathematics difficulties at elementary grade levels often involve children who seem to understand but make careless mistakes (e.g., adding instead of subtracting). The standard approach may be to analyze the mathematical or logical difficulties students face, assuming there is some deficiency in their mathematical knowledge. An alternative approach would be to suspect that an intuitive response took place and examine ways in which the student's metacognitive skills can be strengthened to resist distracting, external cues. In this paper, we describe the similarity that was found between the dual-process framework and the dual task of problem solving and monitoring. This dual task of problem solving requires children to be aware of the formal meaning and implications of the mathematical concepts and, at the same time, to be aware of the underlying intuitions. For example, when doing a mathematics problem, one ideally runs a systematic process that computes a precise answer and a second process that does a quick check to estimate an approximate answer. If the answers are too far apart, the debugging process should be triggered in which a set of activities is used to resolve the discrepancy.

Monitoring may be cognitively demanding, because children must confront the dual task of problem solving and problem checking at the same time. Children may find it difficult to be attentive to their own mistakes when they are concentrating on a problem or task. However, they may find it relatively easy to catch other people's mistakes. Letting children monitor other people's problem solving may alleviate the dual demands, because they no longer have to attend fully to solving the problem themselves. For example, Gelman and Meck (1983) found that children were not good at checking their own work, but they were better at pointing out others' mistakes. Several studies have explored whether pre-school children could check for errors when applying counting principles. Children were asked to help teach a puppet to count and check for any errors the puppet made when counting a particular sequence. The results showed that nearly all the children recognized when the puppet was wrong, and most of the children were able to offer some explanation that indicated an understanding of the violated principle. The study also provided evidence that children were able to check and correct trials even beyond their counting ability when they did not have to do the work themselves, i.e., they only monitored the work. Children find it relatively easy and motivating to catch other people's errors, even if they are not inclined to catch their own. This inspired the two studies in this paper to assess whether monitoring others is easier than monitoring oneself.

2.2. *The development of metacognitive monitoring skills through peer learning*

Metacognition has been identified as a critical process for supporting students' abilities to solve problems and to learn (Bransford, Brown, & Cocking, 2000). Within metacognition, the task of monitoring may take the form of checking, planning, inferring, self-interrogation, and introspection. Brown (1987) described two-component processes that are similar to the dual task of problem solving and problem checking. One component of metacognition is the ability to monitor one's cognitive activities (e.g., detect failure to comprehend). A second component is the ability to take appropriate regulatory steps when a problem has been detected. These steps can include internal regulation (e.g., slow down while reading difficult material) and outward action (e.g., self-correct, consult resources).

There are different viewpoints concerning how metacognition develops and how to promote that development. Piaget inspired an early theory about the development of metacognitive skills that suggests that children develop abilities to self-monitor as they work to resolve their own internal confusion and conflicts. From this perspective, one way of helping children learn to self-monitor is to put them in situations that will lead to cognitive conflict. One behavior that may lead to cognitive conflict is self-explanation. The act of explaining reveals gaps in one's understanding that may develop into frequent checking behaviors. Chi and De Leeuw (1994) examined eighth grade students who read a passage on the human circulatory system. The self-explaining group was asked to read each sentence aloud and explain what it meant to them. The control group was not asked to read the passage twice without explaining it. The results showed that the experimental group improved its understanding of the content significantly and removed gaps and conflicts when creating mental structures. Webb (1989) found that giving elaborate explanations helped students identify missing knowledge, inconsistencies, and clarifications from different perspectives. Both Chi and Webb pointed out that one element of explanation is noticing or "checking" inconsistencies, and that good explainers clear inconsistencies by looking back at the resources to learn more about the domain. The results also indicated that not all students self-explain spontaneously and that students may need some help in learning such skills. This has not gone unnoticed in mathematics education, and many researchers have stressed the importance of reflecting or "looking back" on students' mathematical solutions (Mullins, Rummel, & Spada, 2011; Polya, 1973; Schoenfeld, 1985, 1987; Schon, 1983, 1987).

Another class of theory, inspired by Vygotsky (1978), proposes that children can learn to self-monitor by passively observing other people who self-monitor (Harris & Want, 2005; Okita & Schwartz, 2006; Palincsar & Brown, 1984). In this situation, presenting children with models of others who overtly self-monitor may lead to improvements. Here, self-monitoring is a gradual internalization through the observation of social activities. For example, Rogoff, Mistry, Göncü, and Mosier (1993) studied how children in different cultures observe and learn by situating themselves in a social context in which wide ranges of events occur (e.g., basket weaving in groups). Some patterns may involve children who observe their parents engage in activities until they themselves gradually take initiative to learn and perform on their own (Campione, Brown, Ferrara, & Bryant, 1984). From this perspective, metacognition can be viewed as a social practice that involves social, environmental, and self-influences (Resnick, Lenive, & Teasley, 1993; Zimmerman, 1995). Another example of social practice may be the apprenticeship view of how a novice apprentice learns from observing and monitoring experts in a distributed-task environment (Wertsch, 1978).

In language development, Clark (1995) found preschoolers showed impressive monitoring of their own use of language by spontaneously correcting their mistakes in pronunciation, grammar, and naming objects. In other areas, self-monitoring in children has been more challenging. Geary (2002) stated that child-initiated monitoring activities in the use of language are primary abilities that differ from secondary abilities, such as reading, writing, and mathematical calculations, which involve some level of mastery, as well as procedural and conceptual competencies. Geary pointed out that the acquisition of monitoring activities in secondary abilities in cognition does not normally occur without some formal or informal instruction (Geary, 2002).

Recent theories have suggested how metacognitive processes can be stimulated by probing and peer learning, indicating that metacognition is part of a collaborative learning process in which peers may play a central role (Hurme, Palone, & Järvelä, 2006). In a study with younger children, Markman (1977) worked with students in the first through third grades to examine the developmental changes in monitoring and how they became aware of their comprehension failures. Markman used a number of checking probes that prompted the children to reflect on their comprehension level (e.g., "Can you tell me how to play?;" "Are you sure?;" "Did I tell you everything you need to know?"). The idea of checking was difficult for students to grasp. Results showed that the third graders were aware of their lack of understanding and noted the inadequacy of the instructions through probing.

Azmitia (1996) pointed out several ways that peers could influence the acquisition and revision of knowledge. For example, peers may prompt revision by focusing attention on information that people would not otherwise consider and by forcing people to question or explain their views. Unfortunately, most students do not exhibit positive helping behaviors spontaneously (Roscoe & Chi, 2007). One technique to facilitate students to give help, is to use a reciprocal schema in which one student is given some knowledge in a particular domain and is told to regulate the problem solving of another student; then, the roles are reversed, and the other student becomes the "expert" (Dillenbourg & Jermann, 2007). As part of their role, the expert must monitor their partner's problem solving and offer appropriate help. Other dyadic peer-learning activities include successful cases in the area of reciprocal teaching by Palincsar and Brown (1984), mutual peer tutoring by King, Staffieri, and Adelgais (1998), reciprocal peer tutoring by Fantuzzo, Riggio, Connelly, and Dimeff (1989), the role of peer influences on cultivating mathematical success in urban high school students (Walker, 2006), and young children finding mistakes in puppets when solving elementary mathematics problems (Cauley, 1988).

Palincsar and Brown's (1984) reciprocal teaching model consisted of opportunities for students to learn to monitor each other's comprehension. As a result, they found that a strong emphasis on monitoring improved students' abilities to learn more deeply when reading the subject matter. Similarly, Russell and Ginsburg (1984) conducted a study with fourth grade students with similar IQs, and they found that the ability to monitor other people's reasoning was correlated with the students' larger pattern of performance on achievement tests. In the research described in this paper, it was proposed that helping children learn to monitor other people's problem solving can, in turn, help them learn to monitor their own problem solving and learning. The literature appears to support this hypothesis in that externally-directed monitoring seems to help students eventually turn the monitoring behavior inwards.

2.3. Computer-supported learning environments for elementary mathematics learning

Many studies have explained how computer-learning environments can assist metacognitive learning activities (Kreijns, Kirschner, & Jochems, 2003). The earlier generation of computer-supported, mathematics-learning environments (e.g., BUGGY) involved tools that monitored student performance closely to diagnose and remediate students' mathematical errors (Brown & Burton, 1978). More recent work has moved beyond diagnosis and incorporated strengths from computational models that represent student thinking and cognition (Martin, Mitrovic, Mathan, & Koedinger, 2011).

Another area of research is intelligent tutoring systems. Technology has changed the way tutoring can be structured in that computer systems and computer agents can be used to facilitate the process. The literature concerned with tutoring has shown that much of the overall effectiveness of tutoring is attributed to the tutors' pedagogical skills (Chi, Silver, Jeong, Yamauchi, & Hausmann, 2001; Merrill, Reiser, Ranney, & Trafton, 1992). Pedagogical skills may refer to a variety of tutoring tactics, such as selecting the next problem for the students to work on, giving explanations and feedback, controlling the timing of feedback, eliciting and prompting explanations, and scaffolding. The focus on such pedagogical skills has led to the development of computer-tutoring systems that are more tailored and adaptive to students' needs (Walker, Rummel, & Koedinger, 2011). Intelligent cognitive tutors have been used extensively to teach mathematics in classrooms and improve the acquisition of knowledge (Koedinger & Aleven, 2007; Koedinger, Anderson, Hadley, & Mark, 1997). While these intelligent tutoring systems are likely to match the knowledge organization of the novice learner, novice learners find it difficult to use such systems to identify the gap between their knowledge and the knowledge of the system. Students may notice that their expectation was wrong, but they may have trouble identifying what led to the mismatch. Their interpretation of the source of any discrepancy will be quite difficult, because students will have trouble finding the overlap between the organization of their knowledge and the behaviors of the system. As a result, students may end up treating the system more a like supervised assessment. They may learn that their expectation is wrong at a given point, but they may not have many resources for self-correcting. Supervised assessment is useful, but it does not always provide hints for what to do if one is wrong. There is research on intelligent tutoring approaches to provide the required resources (Wenger, 1987), but intelligent tutoring environments have not adequately solved the problem of helping students learn to self-assess (Anderson, Corbett, Koedinger, & Pelletier, 1995). Perhaps tutoring systems do too much monitoring for the student without a strong model for how to encourage self-monitoring.

A learning environment that openly displays the reasoning process may help students identify the gap between their knowledge and the knowledge of the system, and it may provide hints for what to do when one is wrong. Seifert and Hutchins (1992) conducted a study that observed crewmembers on the navigation deck of a U.S. naval where experienced crewmembers used novice errors to provide instruction so that novice crewmembers could learn on the job. In addition to checking and responding to errors, the novice crewmembers were required to understand how the error was generated. Experienced crewmembers helped the novice crewmembers understand how the errors were generated by modeling the reasoning processes of the novice who committed the error, thereby revealing the cause. Another critical element involved creating a workspace that allowed team members to witness the performance of others so they had the opportunity to monitor errors by others. Seifert and Hutchins (1992) also stressed that novice learning from errors was not automatic and that careful design was required to facilitate novices' ability to notice and recover from errors.

An alternative way to help students see the gap between their own knowledge and the knowledge of the system may be to create a specifiable, computerized, learning environment in which the system can simulate the thoughts an individual might use to analyze a situation. This is different from most simulations, which depict empirical sequences of actions that resemble the phenomena the system presents (e.g., displaying changes over time, continental drift). For example, Chin et al., (2013), created a series of software environments, called 'Teachable Agents' (TA), that visually model how to reason through causal chains. Teachable Agents are a type of pedagogical agent (Baylor, 2007) in which students teach the computer rather than the computer teaching the student or serving as a learning companion (Chou, Lin, & Chan, 2002). TAs have been used to visually teach qualitative, causal relationships in science (Biswas, Leelawong, Schwartz, Vye, & TAG-V, 2005; Chin et al., 2010; Leelawong & Biswas, 2008), taxonomies (Chin et al., 2013), and mathematical topics (Blair, Schwartz, Biswas, & Leelawong, 2007; Matsuda et al., 2012; Pareto, Haake, Lindström, Sjödén, & Gulz, 2012). For example, in a game-like environment, such as 'Betty's Brain' (Biswas et al., 2005; Chin et al., 2010), students used a computerized concept map to teach causal relationships to a pedagogical computer character. Then, the students monitored the computer character as it reasoned through the links in the map that highlighted how entities influenced each other and reached a conclusion. Monitoring the computer character's reasoning helped students learn relational predicates and gaps in their own understanding.

In addition, the advancement of virtual-reality technology and augmented-reality technology is making the self-monitoring process more visible. For example, in a virtual-simulation environment, Bailenson et al. (2008) combined the unique characteristic of virtual reality at both the environmental and social-interaction level. In their study, the students simulated Tai Chi movements in a virtual-learning environment. The students and the instructor were physically separated, but they were connected remotely online through the Internet. The students and instructor engaged in Tai Chi movements through their virtual avatars. The student-controlled avatar followed the Tai Chi movements of the instructor-controlled avatar in real time. The student also was able to overlap his avatar onto the instructor's avatar to adjust his movements in real time.

To help students learn to self-assess and self-correct, the research presented in this paper proposes an alternative approach that is both self-driven and social. The children practiced metacognitive monitoring skills on a computer character. The studies examined the hypothesis that openly monitoring the reasoning process of "someone" else (i.e., the computer character) could help students learn to self-assess their own learning. The ideal computer-supported learning environment would be a situation in which students could practice their monitoring skills on a computer character that openly solved mathematics problems. Students could then monitor the computer character's reasoning when it solves the mathematics problems. By developing monitoring skills on the external plane (e.g., artifacts, agents), children may be in a better position to internalize these skills. This type of support seems ideal in a computer-supported learning environment that assists students in practicing monitoring and self-correcting.

2.4. Overview of the two studies

The literature provided three points that were important in designing this research. First, the literature suggested that the dual task of problem solving and checking might be too cognitively demanding for children who had not yet developed the necessary skills to monitor their thought processes. One way to develop monitoring skills was to alleviate such dual demands by letting children monitor their peers solving problems, because the children found it easier to catch other people's mistakes than their own. The study by Gelman and Meck (1983) provided evidence that children were able to check and correct trials beyond their counting ability when they only had to monitor the work rather than having to do the work themselves.

Second, the literature suggested that the behavior of checking was not automatic and that some kind of treatment or training was required. Geary (2002) pointed out that child-initiated monitoring activities are more common in language use (e.g., correcting pronunciation, grammar) which is a primary ability, but they differ from secondary abilities, such as reading, writing, and mathematical calculations, because the acquisition of monitoring in secondary abilities requires some level of mastery in both procedural and conceptual competencies, and this does not occur without some formal or informal instruction.

Third, the literature stressed that novices learning from errors was not automatic and that a careful design of the environment was necessary to facilitate noticing and recovering from errors (Seifert & Hutchins, 1992). Their research suggested creating a learning workspace in which students can monitor the reasoning process of others, which helps novice learners understand how the errors were generated and the reasoning processes that reveal the causes of the errors.

Two studies examined the hypothesis that helping children learn to monitor other people's problem solving can, in turn, help them learn to monitor their own problem solving and learning. Ideally, monitoring somebody else's work should prompt the child to use her or his own thinking process to generate a solution and then compare it with the other person's solution. In other words, focusing on monitoring others would give the children the opportunity to develop the awareness and capacity required to compare their solutions against others' solutions, and, as these abilities develop, it would be easier to turn this capacity inward. This led to the design and development of two model technologies that demonstrated how to implement conditions that facilitate self-other assessments. In both environments, a computer character called "ProJo" was designed to openly display its reasoning when solving mathematics problems, allowing children to "look for mistakes." In this workspace, students could practice metacognitive skills by monitoring the computer character. While the students are monitoring, the computer character, ProJo, applies techniques of prompting and reciprocal schema in which the students were told to regulate the problem solving of a computer character. This means that the computer character would take the role of a peer learner designed to solve mathematics problems that engage the student in catching mistakes. The learning workspace would allow students to openly observe the reasoning of the peer character when solving mathematics problems. When monitoring becomes an inherent part of problem solving (S1/S2 interaction), students may be able to avoid the pitfalls of rapid, reactive responses (i.e., careless mistakes).

The first experiment was a between-subjects design that directly compared self-other training (experimental condition) to self training (control condition) in which students in self training solved problems on their own, while students in the self-other training took turns solving problems with the computer character while monitoring for any mistakes. The study examined whether monitoring a computer character helped alleviate the dual demands of problem solving and checking, making it easier to monitor someone other than oneself. The study also evaluated how self-other training influenced the performance of the students in solving mathematics problems and whether the behavior of the computer character could be executed in ways that maximize the performance of the students. In the first study, a learning workspace called "Doodle Math" was developed, in which students openly observed the reasoning process of the computer character as it worked out mathematics problems. During the training session, the students in the self-other training group showed evidence of a late-gain effect, which implied that a slightly longer intervention might be needed for results to appear on the posttest. Another limitation was that the current study fell short in determining whether the students were actually monitoring or just copying/modeling behavior.

The second experiment was also a between-subjects design that directly compared self-other training (experimental condition) to self training (control condition). It involved a longer intervention (more than double the time compared to the first experiment) to determine whether more sustained practice at monitoring would lead to stronger effects. The experiment addressed many of the issues and limitations from the first experiment, i.e., focusing on extending the training time by doubling the hours, increasing the number of practice problems, using shorter subtasks to make the reasoning process short and explicit, and adding a logging feature to detect the students' self-corrections. The new learning environment "Puzzle Math" was developed in order to increase the number of problems through shorter subtasks to make the reasoning process more explicit and clear. Also, it added a new detection feature to measure the students' self-correcting behaviors. The second study demonstrated that extended training provided a longer-lasting effect on the posttest. The students in the self-other training had a higher accuracy score throughout the session, and, by the posttest, the difference was significant. Students in the self-other training showed more self-correction with higher accuracy than students in the self training.

3. Experiment 1: self training vs. self-other training

The first experiment attempted to answer the following research questions: (a) Did monitoring a computer character help alleviate the dual demands of problem solving and checking, making it easier to monitor someone other than oneself? (b) Did self-other training in monitoring help students learn and improve their performance in mathematics problem solving? (c) How could the computer character's behaviors be executed in ways that assist training and maximize students' mathematics performance?

The experiment directly compared self-other training to self training. The self training group was the control condition in which the computer system provided "right" or "wrong" feedback. (See Fig. 1; if "right," students see a dinosaur sticker appear, and, if "wrong," students see an "X" mark over the dinosaur sticker.) This was immediate feedback on their performance that served as self training. The self-other training group was the experimental condition in which treatment involved students' taking turns with the computer character in solving problems, and students' practicing their monitoring skills on the computer character by checking for mistakes. (See Fig. 2; students monitored the computer character and pointed out a mistake.) Students in the self-other training also received "right" or "wrong" feedback. This turn-taking structure was important in applying the technique of prompting and reciprocal schema as the treatment in the self-other training condition. As the literature suggests, Cassell (2004) demonstrated a storytelling system in which young children and an embodied conversational agent took turns creating a narrative and passively listening, which improved the children's linguistic skills. Permitting other people to take the initiative offers alternatives to one's own inertia. It also generates "projective" feedback by revealing the variations another person applies to one's ideas. Coaches, for example, may learn a great deal by watching their players take the initiative to adjust a play during a game. In contrast, Barron (2003) found that small groups that blocked the initiative of one of its members often failed to capitalize on the correct ideas that the individual provided, and, therefore, the group members did not perform well. Interactions characterized by an imbalance of initiative led to less learning.

3.1. Methods

3.1.1. Participants

One objective for creating model technologies to train students in monitoring was to provide tools for underserved communities and schools where teachers have little time to closely monitor student progress and problem solving behaviors. For this reason, participants were recruited from a local public school in a low socioeconomic status (SES) underserved community in Harlem, located in upper Manhattan. At the time of the study, the school's New York City (NYC) progress report grade was rated "C" (with "A" the highest and "F" as the

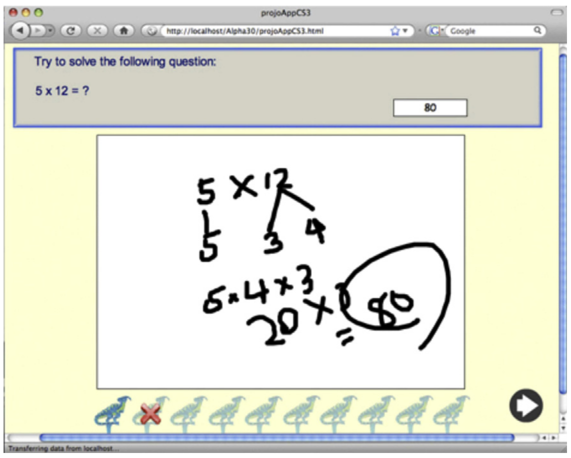

projoAppCS3
http://localhost/Alpha30/projoAppCS3.html
Google
Try to solve the following question:
5 x 12 = ?
80
5 x 12
5 3 4
5x4x3
20 x 3
= 80
Transferring data from localhost...

Fig. 1. Self training in Doodle Math.

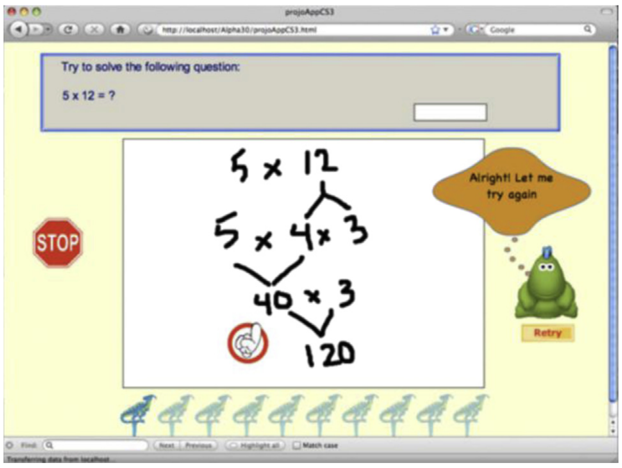

projoAppCS3
Try to solve the following question:
5 x 12 = ?
5 x 12
5 x 4 x 3
40 x 3
120
STOP
Alright! Let me try again
Retry

Fig. 2. Self-other training with ProJo in Doodle Math.

lowest possible grade), and the percentage of students meeting or exceeding the state standard for test results in mathematics were 15–20% below the state average. The study consisted of 40 students in the fourth grade ($N = 19$) and fifth grade ($N = 21$) between 9 and 11 years of age (female, $N = 23$; male, $N = 17$). Each student engaged in a one-hour session that consisted of 15 min of preparation time (getting used to the experimental setting, as seen in Fig. 3) and 45 min of monitoring treatment on a laptop computer.

3.1.2. Design and procedure

This study was a 2 × 1, between-subjects design in which the independent variable was training type (self training vs. self-other training). The students were assigned randomly to one of two conditions, i.e., self training (control) or self-other training (experimental). Each student was seated separately in front of a laptop computer (Fig. 3). Students were familiar with the basic concepts of multiplication and division. They were introduced to an unfamiliar mathematical trick on divisibility rules for multiplication (Fig. 4). The students took a test that consisted of eight multiplication problems that had to be solved using the mathematical trick on divisibility rules. The results from the test were used to create the materials for each student's treatment session. Students started the training session the following day. In the self training condition, the computer character ProJo was not present (Fig. 1), whereas, in the self-other training condition, the computer character ProJo appeared in the learning environment (Fig. 2). The training session had 10 trials (one mathematics problem for each trial). Please see Fig. 5, which shows the research design and procedural flow for experiment 1. Students in the self training solved 10 trials on their own.

In self-other training, the student and computer character ProJo took turns solving problems. In total, the students in the self-other training condition solved five trials on their own and monitored ProJo for the remaining five trials (Fig. 5). ProJo was introduced to students as a peer-like computer character that was one school grade below that of the student. The cover story was that the older peer (the student) would look after the younger peer, ProJo, to make sure it did not make any careless mistakes. When it was ProJo's turn, he would say, "I think I learned a lot by watching you play. Let me try. I don't like mistakes, so if you can catch me when I make a mistake that would be great." If ProJo made a mistake, the student was asked to press the "Stop" button and drag the "hand" icon to the place on the screen where the mistake occurred (Fig. 2). ProJo solved problems that were sometimes correct and sometimes incorrect, and the student would monitor and catch (or not catch) the mistake. Each student was responsible for monitoring and catching any mistakes ProJo made. After the training session, a test that consisted of eight multiplication problems was administered to both conditions, and the study was concluded.

3.2. Materials and measures

The computer-supported learning environment "Doodle Math" was developed to help students practice their monitoring skills on a computer character. Doodle Math was implemented in Flash, Java, and Ruby, and operated on Windows7 and XP operating systems as a web-based application. Students used a pen and tablet device (Fig. 3) to write their calculations, which appeared on a laptop screen (Fig. 1). A

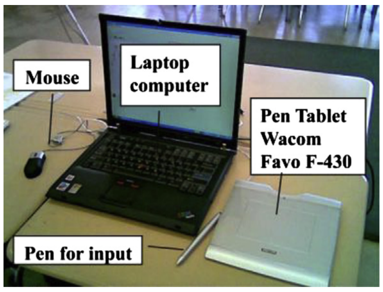
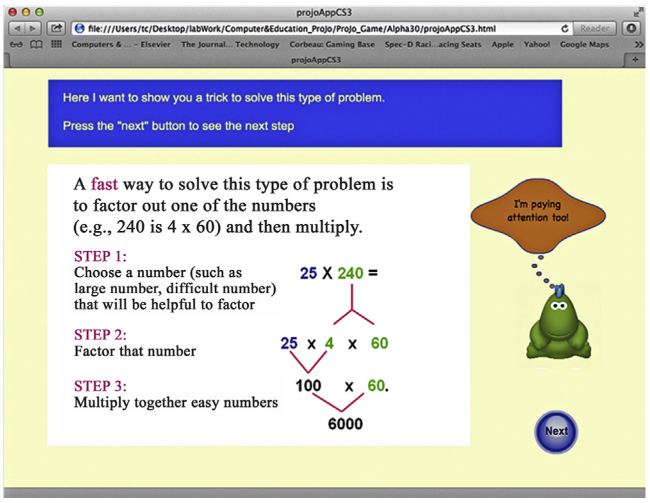


Fig. 3. Experimental setting.


projoAppCS3
file:///Users/tc/Desktop/labWork/Computer&Education_ProJo/ProJo_Game/Alpha30/projoAppCS3.html
I'm paying attention too!
Next

Here I want to show you a trick to solve this type of problem.

Press the "next" button to see the next step

A fast way to solve this type of problem is to factor out one of the numbers (e.g., 240 is 4 x 60) and then multiply.

STEP 1:
Choose a number (such as large number, difficult number) that will be helpful to factor

25 X 240 =

STEP 2:
Factor that number

25 x 4 x 60

STEP 3:
Multiply together easy numbers

100 x 60.

6000

Fig. 4. Divisibility rules of multiplication.

pedagogical computer character called "ProJo" was an interactive dinosaur character that openly displayed its reasoning by writing out the calculations on the screen when solving mathematics problems (Fig. 2). The hand-written calculations were displayed as an animation that appeared as though ProJo were writing in real time. The Doodle Math environment displayed mathematics problems, gave feedback, recorded screen shots, and kept a log file that recorded students' progress (accuracy and time taken to solve problems).

3.2.1. *Pretest and posttest*

The key dependent measures for this study included accuracy scores and calculation time on 1) the pretest and posttest and 2) mathematics performance on problems administered during the monitoring treatment session (Table 1). The pretest was used to examine the current mathematics level of the students and to identify the calculation mistakes each student made. The student's mistakes were later used in the monitoring treatment session (for the self-other training condition only). The relative effects of self-other training compared to self training were evaluated using calculation time and accuracy scores from pre- to posttest and student performance scores during the treatment session. The pretest and posttest each consisted of eight problems that were general calculation problems and commonly known problems where students often make careless mistakes (e.g., losing track while solving problems, mixing calculations during the procedure, such as adding the final step instead of multiplying). The coefficient alpha values for the resulting pretest and posttest measures were 0.78 and 0.70, respectively.

3.2.2. *Performance during monitoring treatment session*

The treatment session consisted of 10 trials for both the self training and self-other training conditions. In the self training condition, the five trials consisted of solving medium-level multiplication problems using the mathematical trick on divisibility rules (trial numbers 1, 3, 5, 7, and 9). The remaining five trials consisted of two problems selected from the pretest that the student solved incorrectly (trial numbers 2 and 10); two medium level multiplication problems (trial numbers 4 and 8); and one general problem for which students were known to make common, careless, calculation mistakes (trial number 6). If there were less than two incorrect problems from the pretest, the screen shot was examined to find problems with which the student had shown signs of struggling.

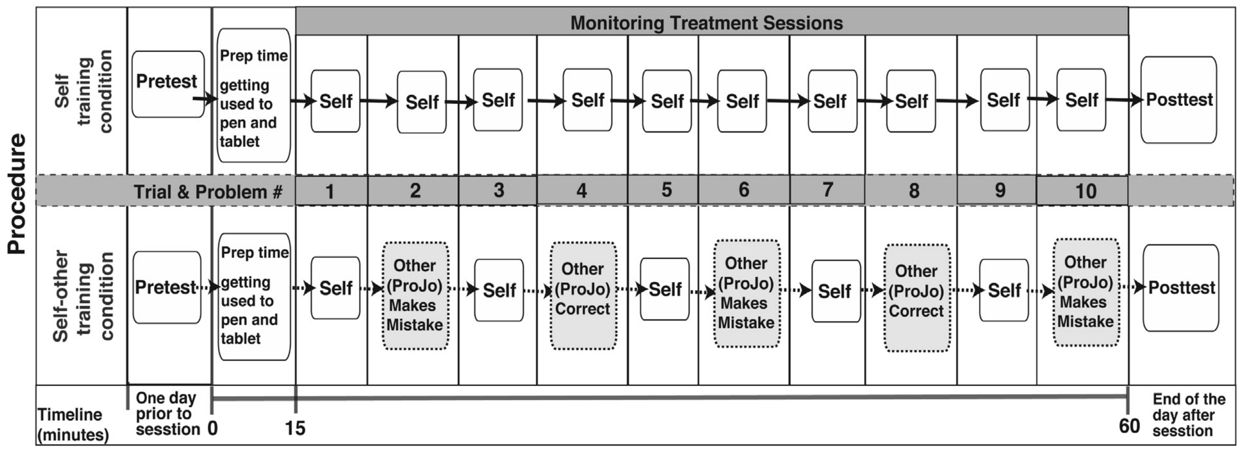

Procedure
Monitoring Treatment Sessions
Self training condition
Pretest
Prep time getting used to pen and tablet
Self
Self
Self
Self
Self
Self
Self
Self
Self
Self
Posttest
Trial & Problem #
1
2
3
4
5
6
7
8
9
10
Self-other training condition
Pretest
Prep time getting used to pen and tablet
Self
Other (ProJo) Makes Mistake
Self
Other (ProJo) Correct
Self
Other (ProJo) Makes Mistake
Self
Other (ProJo) Correct
Self
Other (ProJo) Makes Mistake
Posttest
Timeline (minutes)
One day prior to sesstion
0
15
60
End of the day after session

Fig. 5. Research design and procedure for experiment 1.

Table 1
Description of measures used in pretest, during the treatment session, and posttest.

Prior-to-Intervention measures				
Pretest	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
• Eight problems • General multiplication • Medium level	• Assess student's mathematics level prior to study • Identify problems solved incorrectly for use in treatment session	Problem solver (Solve on own)	• Assess student's mathematics level prior to study • Identify problems solved incorrectly for use in treatment session	Problem solver (Solve on own)
Monitoring treatment session				
Treatment problems	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
Trials# 1, 3, 5, 7, 9• Five problems • General multiplication • Medium level	• Assess student's mathematics performance from self-other training (i.e., accuracy and calculation time) • See immediate "before and after" effects on student performance after monitoring practice w/computer character	Problem solver (Solve on own)	• Assess student's mathematics performance from self training (i.e., accuracy and calculation time)	Problem solver (Solve on own)
Trials# 2, 10• Two problems • Problems selected from the pretest that the student solved incorrectly	• See if students can monitor and catch their own mistake if the computer character they observe makes the same mistakes	Peer observer (Monitor ProJo)	• Assess if students can solve problems that were previously solved incorrectly on the pretest (i.e., accuracy and calculation time)	Problem solver (Solve on own)
Trial# 6• One problem • General problem known for students to make common careless calculation mistakes	• See if students can monitor and catch mistakes that are different from their own mistakes	Peer observer (Monitor ProJo)	• Assess if students can solve a general problem where students often made careless mistakes (i.e., accuracy and calculation time)	Problem solver (Solve on own)
Trials# 4, 8• Two problems • General multiplication • Medium level	• See if students can monitor and identify when problems were solved correctly by the computer character	Peer observer (Monitor ProJo)	• Assess if students can solve general problems (i.e., accuracy and calculation time)	Problem solver (Solve on own)
Post-treatment session				
Posttest	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
• Eight problems • General multiplication • Medium level • Consists of problems for which students have been known to make careless mistakes	• Assess student's post-treatment mathematics performance in self-other training (i.e., accuracy and calculation time)	Problem solver (Solve on own)	• Assess student's post-treatment mathematics performance in self training (i.e., accuracy and calculation time)	Problem solver (Solve on own)

The self-other training condition also consisted of 10 trials (Table 1). The problems in trials 1, 3, 5, 7, and 9 were the same general multiplication problems as the self training condition, but the students' performances on these problems were informative to see the immediate before and after effects of monitoring. In trials 2, 4, 6, 8, and 10, the student monitored the computer character ProJo for any mistakes. During these trials, ProJo solved mathematics problems that were sometimes correct and sometimes incorrect, so the student did not automatically expect ProJo to always make a mistake. ProJo came from the word "projection," since the student's own mistakes from the pretest were used to create some of ProJo's calculation mistakes. ProJo would display two calculation mistakes similar to the problems that were solved incorrectly by students on the pretest (trials 2 and 10) and one general problem for which students were known to have made common, careless, calculation mistakes (trial number 6). Students in the self-other training condition were unaware that the two mistakes by ProJo (replayed as ProJo's own) were actually their own mistakes on the pretest (e.g., mix-up procedure, lose track). The problems were selected individually for each student. ProJo also correctly solved two problems (trials 4 and 8). The monitoring treatment session consisted of 10 mathematics problems, and the resulting coefficient alpha for this measure was 0.68. One reason for the slightly lower alpha score may be due to the fact that the problems used in trials 2 and 10 were custom created for each student (i.e., trials 2 and 10 were problems from the pretest that the student solved incorrectly).

3.2.3. Scoring

Scoring on accuracy was calculated automatically and confirmed manually. Students in both conditions were asked to use the pen and tablet to write out the calculations, circle their written answer, and type their answer into the white box on the screen (Fig. 1). The students could not move on to the next question unless the answer was typed into the box. The typed answer was scored automatically by the computer system. To make sure that students did not type in a random answer, screen shots of students' solving problems during the treatment sessions were also examined. The screen shots showed students' reasoning and calculation methods when solving problems (e.g., did they follow instructions, where did they make a mistake, did they catch ProJo's mistakes?). If there were any questions about the screen shot (e.g., two circled answers, disorganized), the student's log file was used to play back the student's handwritten calculation motion. The log file recorded the handwritten calculation (from the use of the pen and tablet device) for each student by keeping a log of the data points (e.g., *x*, *y* coordinates from the pen and tablet when drawing on the white screen area). The log file allowed researchers to play back the reasoning process and calculation in the student's original handwriting and timing. This was used to see when and where in the reasoning process the student made the mistake or showed any hesitation. For the self training condition, scores for all 10 trials were based on the student's accuracy when solving problems. In the self-other training condition, scores for five trials were based on student's accuracy when solving problems, and, for the remaining five trials, accuracy was based on whether the student was able to catch ProJo's mistakes and accurately point out where in the calculation the mistakes occurred (Fig. 2).

3.3. Results

The results section will be structured around the three research questions of this study: (a) Did monitoring a computer character help alleviate the dual demands of problem solving and checking, making it easier to monitor someone other than oneself? (b) Did self-other training in monitoring help students learn improve performance in mathematics problem solving? (c) How could the computer character's behaviors be executed in ways that assist training and maximize student's mathematics performance?

3.3.1. Alleviating the dual demands of problem solving and checking through monitoring

The first research question consisted of two parts. The first part of the analysis explored whether monitoring a computer character helped alleviate the dual demands of problem solving and checking, and the second part examined whether monitoring a computer character made it easier for students to monitor someone other than oneself. The first part of the analysis looked at the accuracy and calculation time for trials 1, 3, 5, 7, and 9, in which students in both conditions solved the problems on their own. For students in the self-other training condition, the independent performance on these trials measured the immediate "before and after effects" of monitoring the computer character ProJo.

The relative effects of self-other training and self training were examined to see more closely what was going on during the training session. The "odd" number trials in which students from both conditions solved the problems on their own (Trials 1, 3, 5, 7, and 9) were compared across the two conditions.

The initial thought was that students in the self-other training would "slow down" when solving problems on their own. During the training session, students in the self-other training (Fig. 6 dotted line) showed a more gradual decrease in time than the self training students (Fig. 7 dotted line). The students in self training increased their calculation speed immediately after the first trial. Students in both trainings spent the most time on the first problem, possibly becoming acquainted with the environment. The effects on accuracy and time were short-lived, with the two trainings showing little difference by the end of the session. When calculation time and accuracy were compared across the five trials, the results showed that students in self training immediately increased their speed (i.e., took less time), but they made mistakes over time. Students in the self-other training slowed down (i.e., spent more time) after the first exposure to the computer character, but they gradually increased both speed and accuracy over time (Fig. 6). One possible interpretation is that students worked hard to internalize the monitoring, which paid off on later problems, a performance similar to the U-shaped curve (Karmiloff-Smith, 1979). The results were not sufficiently strong to generalize this interpretation, but they did show a promising trend.

The second part of the analysis looked at the accuracy for trials 2, 4, 6, 8, and 10 and examined whether monitoring a computer character was easier (self-other training condition) than monitoring oneself (self training condition). The trials were organized to see the effects of self-other training in different situations, i.e., (a) identify general calculation mistakes, (b) identify when the problems were solved correctly, and (c) identify their own mistakes if (unconsciously) presented as ProJo's mistake.

The analysis looked at whether it was easier to monitor and catch someone else's mistake (i.e., computer character ProJo) than to monitor and catch one's own mistakes when problem solving. This time, the analysis compared the scores from the "even" numbered trials, i.e., 2, 4, 6, 8, and 10 (Fig. 5). In the self training group, scores were based on accuracy, and, in the self-other training, scores were based on whether the students were able to catch ProJo's mistake and accurately point out where in the calculation the mistake occurred. Looking at the results, a significant difference can be observed between the two groups (Fig. 8). The self-other training group scored higher than the self training group at (Self training, $M = 58\%$, Self-other training, $M = 74\%$, $t(38) = 1.98$, $p < 0.05$).

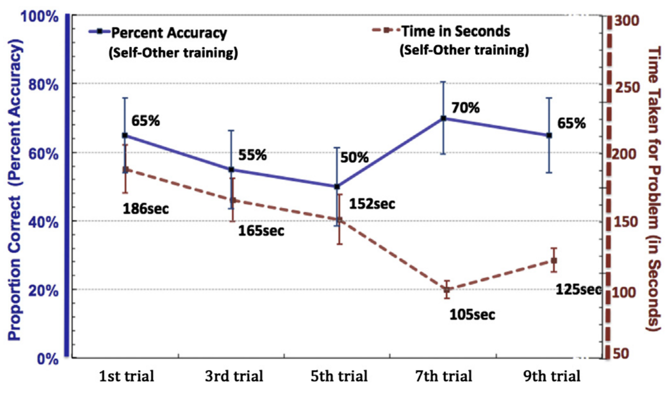


Fig. 6. Accuracy and calculation time for self-other training.

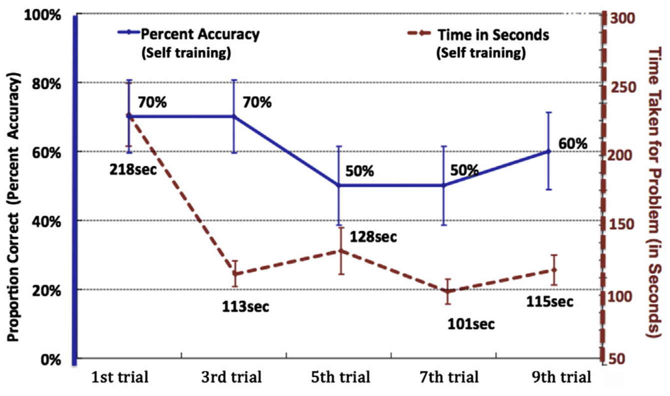


Fig. 7. Accuracy and calculation time for self training.

The self-other training group performed significantly better in identifying general mistakes in calculation problems (catching ProJo's mistake) than the self training group that solved the problems on their own (Self training $M = 55\%$, Self-other training $M = 85\%$, $t(38) = 2.14$, $p < 0.001$). The self-other group had a 100% accuracy rate in identifying when a problem was solved correctly by the computer character, while the students in self training performed significantly lower when solving the same problems on their own (Self training $M = 62\%$, Self-other training $M = 100\%$, $t(38) = 3.94$, $p < 0.001$).

Conversely, students in the self-other training had trouble catching ProJo's mistakes when the mistakes were similar to the ones they had made on the pretest. Although the difference was not significant, the self-other training group performed worse (couldn't catch ProJo's mistake) than the self training group in which the students solved the same problems on their own (Self training $M = 55\%$, Self-other training $M = 43\%$). One possible explanation is that general calculation errors are "matter-of-fact" types of problems, and students can easily remember their mathematics facts and compare them to ProJo's computations. In contrast, for similar mistakes (mix-up and losing track of problems), the errors were more procedural, and students may need to follow the procedure and repeatedly check it against their own answer. This may be more difficult because they are new to the procedures, whereas mathematics facts may be more familiar. An alternative explanation is that the calculation errors differed from the mistakes the students would make, so they were easy for them to catch. The results showed that ProJo was successful in making students (a) identify general calculation mistakes, (b) identify when the problems were solved correctly, but (c) ProJo was somewhat unsuccessful at making students notice their own mistakes. The critical limitation was that ProJo was successful at replaying errors but unsuccessful in making the students notice their own mistakes.

3.3.2. *Effect of self-other training compared to self training on mathematical performance*

To answer the second research question, i.e., whether self-other training in monitoring helped improve students' performance, the pretest and posttest results on accuracy and calculation time were compared.

A paired-sample *t*-test was conducted to compare the accuracy score from pretest to posttest. The average accuracy score in the self training group did not differ much from the pretest ($M = 51\%$, SD = 0.37) to the posttest ($M = 53\%$, SD = 0.34) at $t(19) = -0.14$, $p = 0.89$. The self-other training group did show improvement from the pretest ($M = 50\%$, SD = 0.29) to the posttest ($M = 65\%$, SD = 0.29), but this was not significant $t(19) = -1.75$, $p = 0.09$.

The average calculation time in the self training group decreased from the pretest ($M = 124$ s, SD = 41.02) to the posttest ($M = 111$ s, SD = 45.54) at $t(19) = 0.93$, $p = 0.36$, while the self-other training group increased from the pretest ($M = 113$ s, SD = 29.26) to the posttest

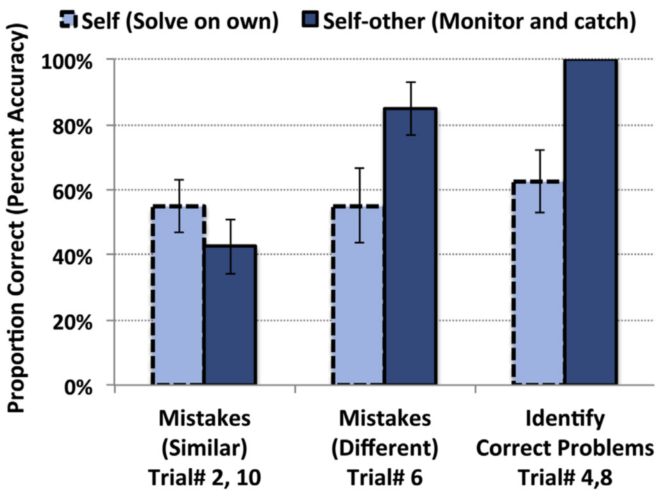


Fig. 8. Accuracy when monitoring ProJo compared to solving problems on one's own.

($M = 120$ s, SD = 45.28) at $t(19) = -0.50$, $p = 0.63$. Overall, there were no significant differences between the two training groups on accuracy and calculation time.

Though not significant, the students in the self-other training group did increase their mathematics performance from pre- to posttest, and they spent more time solving the problems than the self training group.

3.3.3. *Examining computer character behaviors to assist training and maximize performance*

To answer the third research question of how the computer character's behaviors could be executed in ways that assist training and maximize student's mathematics performance, a conditional probability analysis was conducted. This analysis connected student performance within the monitoring treatment session with their final performance on the posttest. The sheer number of possible patterns across 10 in-treatment session problems makes this difficult, given the small sample size of 40 participants split over two conditions (Fig. 9). The following is one strategy for simplifying the analysis: The sample size was too small for meaningful statistical analysis, but descriptive analyses can still be productive.

The mathematics problems during the training session contained three types of potential mistakes, i.e., mix-up, general calculation, and losing track. When students completed the study, they had three scores for each of the types of potential mistakes (three scores for mix-up, three scores for general calculation, and three scores for losing track). Each of the three scores was from a different time in the study. The first score was always from the "odd" numbered trials in which students in both conditions solved the problems; the second score was always from the "even" numbered trials that were solved by either the student (self training) or ProJo (self-other training); and the third score was from the posttest. The first, second, and third scores for each student were tallied for each type of mistake. The R stands for getting the problem right, and W stands for getting it wrong. Students who got the first, second, and third scores all correct for the mix-up questions were labeled as Right, Right, Right (RRR); students who got the first wrong but the second and third correct for the losing track questions were labeled as Wrong, Right, Right (WRR). After tallying for each student, the score was represented in one of eight ways (Fig. 9), representing all combinations possible. The scores from each of the three types of questions were collapsed for each student. As you can see, Fig. 8 shows the number of students who scored right (R) or wrong (W) across problems with three types of potential mistakes (a total of 20 students per condition × three types of potential mistakes = 60 scores per condition). In each training, there were eight different ways a student could have scored (e.g., RRR, RRW, RWW, WWW, WWR, WRR, WRW, and RWR). The left side of Fig. 9 shows how students in self training performed, and the right side shows how students in the self-other training performed. What was surprising was that the same number of students in each group got the problem right and wrong on the first score. This was purely coincidental, but it also demonstrated that the two training groups were equal at the beginning of the tests. In Fig. 9, the results within the training session were connected to the posttest performance. The results showed that, in both training groups, 60% of the students answered the first problem correctly, and 40% answered it incorrectly. When the self training students continued to solve the second problem on their own, 64% solved the problem correctly, while 36% solved it incorrectly. In the self-other training group, students went on to monitor ProJo solving the second problem, and only 47% of the students were able to catch the mistakes, with 53% of the students being unable to catch the mistakes ProJo made.

So why does this matter? This type of analysis told us more than how the students performed during the training session. It also told us how that performance influenced their posttest scores, and it gave us an opportunity to answer other questions, i.e., if the students get the first question wrong, would they benefit more if they solved another problem on their own or monitored a computer character making mistakes? In what environment (self or self-other training) is the student more likely to score well on the posttest. Below, Fig. 10 shows how students in each type of training scored on the first question and how they performed on the posttest. For example, from Fig. 10, certain considerations can be accounted for by such analysis (Table 2).

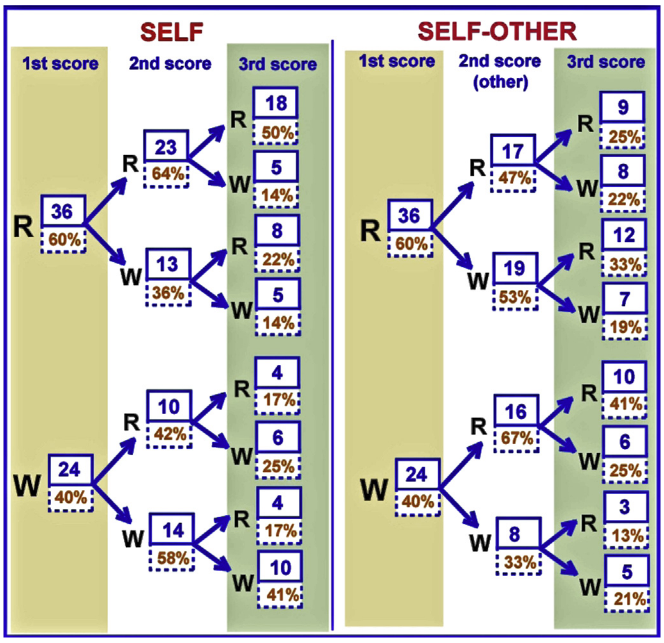


Fig. 9. Average of first, second, and third score of students during training and posttest.

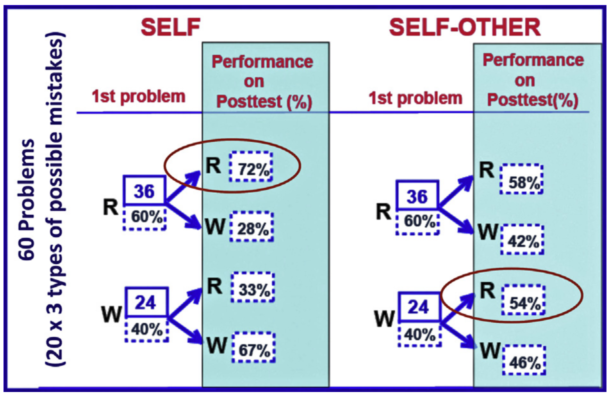


Fig. 10. Students' scores prior to training and students' performances on the posttest after training.

These tentative results suggest the practical hypothesis (Table 2) that if a student gets a problem right, the student should solve another problem on his or her own. If the student gets a problem wrong, the student should monitor a computer character. Although there are more possible patterns across the 10 in-treatment session problems that should be explored with a larger sample size, the results from this analysis can help design the decision making so ProJo can dynamically choose the most effective problem and environment to improve the student's posttest score. These findings have some implications for a scenario in which a more sophisticated version of ProJo could be designed so that the system makes real-time decisions to have the student monitor an incorrect ProJo, a correct ProJo, or simply continue working on his or her own. There may be a need to sequence the computer character's mistakes to further improve student learning. By identifying unfavorable problem sequencing, ProJo can have important implications for the learner.

3.4. Discussion

Overall, the evidence suggested that children learn to self-monitor by monitoring others, but the results were not definitive. After comparing the calculation time and accuracy across the trials, the results showed that students in self training immediately increased their speed but did worse over time. Students in the self-other training showed a "late-gain" effect, in that they slowed down following the first exposure to the computer character, after which they gradually increased their speed and improved their accuracy in solving problems. The training was somewhat limited in duration (15 min of preparatory time getting the laptop set up and students getting used to using the pen and tablet, followed by a 45-min training session). A slightly longer intervention may determine whether more sustained practice at monitoring would lead to stronger effects.

The mathematics problems used in the first study involved divisibility rules that required completion of several subtasks (which were quite implicit) to complete the main task of coming up with the final answer. Some students made procedural mistakes (i.e., mixing up numbers, losing track). Sometimes students looked at the problem 25×240 and immediately saw that 240 could be broken down to 4×60 to make the calculation easier when multiplying by 25. The students proceeded to multiply $25 \times 4 = 100$, and then multiplied 100 by 60 to get the correct answer of 6000. At other times, the students looked at the problem (e.g., 25×240) and broke the problem all the way down to $5 \times 5 \times 2 \times 2 \times 2 \times 2 \times 3 \times 5$ before picking and calculating the values, and this increased the likelihood that they would lose track and mix up the numbers. In the same way, monitoring the computer character's multi-step reasoning process may have been difficult or too implicit for the students to follow. Making each subtask more explicit, as pieces that form a puzzle, may help students avoid procedural mistakes (e.g., mix-up, losing track).

Another limitation is that the current study fell short in determining whether the students were monitoring or just copying/modeling behavior. The second experiment extended the session to a full hour of training each day, for two consecutive days, to determine whether this would sustain monitoring practice and improve accuracy. A new testing environment, "Puzzle Math," was implemented for the second study in order to include a detection feature to determine whether students were monitoring their work and self-correcting.

4. Study 2: extended sessions of self and self-other training and detecting self-correcting behavior

The second experiment was also a between-subjects design that directly compared self-other training (experimental condition) to self training (control condition), and incorporated many of the lessons learned from the first experiment. The second experiment had a longer

Table 2
Considerations made from conditional probability analysis.

If the student gets the problem **RIGHT** on the first question, the next problem may…	
HELP the student's score on the posttest if he or she is given…	Another problem to solve on his or her own (Self training: 72% accuracy on posttest).
HARM the student's score on the posttest if he or she is given…	A computer character to monitor that makes a mistake solving a mathematics problem (Self-other training: 58% accuracy on posttest, which is worse than solving the problem again on one's own, which is 72%).
If the student gets the problem **WRONG** on the first question, the next problem may…	
HELP the student's score on the posttest if he is she is given	A computer character to monitor that makes a mistake solving a mathematics problem (Self-other training: 54%, which is better than doing it again on one's own at 33%).
HARM the student's score on the posttest if he or she is given	Another problem to solve on his or her own (67% scored incorrect on the posttest).

intervention that consisted of a full hour of training each day, for two consecutive days (more than double the time in the first experiment) to determine whether more sustained practice at monitoring would lead to stronger effects. The new learning environment, "Puzzle Math," consisted of more sub-problems that made up a larger puzzle where the reasoning process was short and explicit. The learning environment also included a logging feature that monitored and detected when students were self-correcting. The second experiment attempted to answer the following research questions: (a) Did longer self-other training sessions with the computer character improve mathematics performance? (b) Did self-other training sessions lead to more accurate self-correcting behaviors?

4.1. Methods

4.1.1. Participants

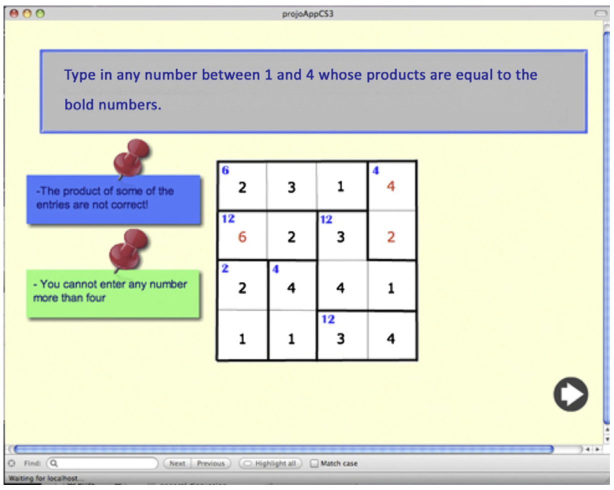
As in the first experiment, the second experiment also focused on creating a mathematics-training environment to develop learning tools for underserved communities. The students in this study were recruited from a different school, but in the same district. This school was also from a low SES, underserved community. At the time the study was conducted, the school's NYC progress report grade was rated "C" (with "A" being the highest grade, and "F" being the lowest possible grade). Similarly, the percentage of students who met or exceeded the state standard for mathematics test results was 15–20% below the state average. The study consisted of 22 students in the fifth grade ($N = 16$) and sixth grade ($N = 6$) between the ages of nine and 11 (females, $N = 11$, males, $N = 11$). The students engaged in a one-hour training session each day for two consecutive days.

4.1.2. Design and procedure

This study was a 2×1, between-subjects design in which the independent variable was the training type (self training or self-other training). The students were assigned randomly to one of two conditions, i.e., self training (control) or self-other training (experimental). Each student was seated separately in front of a laptop computer. Students were familiar with the basic concepts of the divisibility rules for multiplication and division. First, the students were allowed to become familiar with the "Puzzle Math" environment. Then, the students took a pretest (a day before the training treatment sessions) that consisted of two puzzles that were made from a small subset of six to eight mathematics problems on divisibility rules (Figs. 11 and 12).

The students started the training treatment the following day. In the self training condition, the computer character ProJo was not present (Fig. 11), whereas in the self-other training condition, ProJo appeared in the learning environment (Fig. 12). The training treatment sessions included seven puzzles. The research design and procedural flow is described in Fig. 13. Students in the self training condition solved all seven puzzles on their own. In self-other training, the students and computer character ProJo took turns solving problems. In total, the students in the self-other training condition solved five puzzles on their own and monitored ProJo for the remaining two puzzles (Fig. 13). ProJo was introduced to students as a peer-like computer character that was one school grade below their grade i.e., if the student was in fifth grade, ProJo was in fourth grade. The cover story was that the older peer (the student) would look after the younger peer (ProJo) to make sure it did not make any careless mistakes. When it was ProJo's turn, he would say, "I think I learned a lot by watching you play. Let me try. I don't like mistakes, so if you can catch me when I make a mistake that would be great." If ProJo made a mistake, the student was asked to press the "Stop" button, and drag the "hand" icon to the place on the puzzle where the mistake occurred (Fig. 12). The puzzle consisted of a small subset of six to eight mathematics problems, and ProJo would solve most of the subset problems correctly, but would make one mistake within each puzzle, and the student who was monitoring ProJo was supposed to catch the mistake. (See Fig. 12; the mistake pointed out by student is indicated by the red circle (in the web version).) Each student was responsible for monitoring and catching any mistakes ProJo made.

On the second day of the training treatment, the procedure was repeated, and students in both conditions solved another set of seven puzzles; those in the self training condition solved all seven puzzles on their own, and those in the self-other training condition took turns with ProJo, solving five puzzles on their own and monitoring ProJo as it solved the remaining two puzzles. The posttest was completed on the day following the two-day, training treatment sessions. The posttest consisted of three puzzles with each puzzle consisting of a small subset of six to eight mathematics problems on divisibility rules.


projoAppCS3

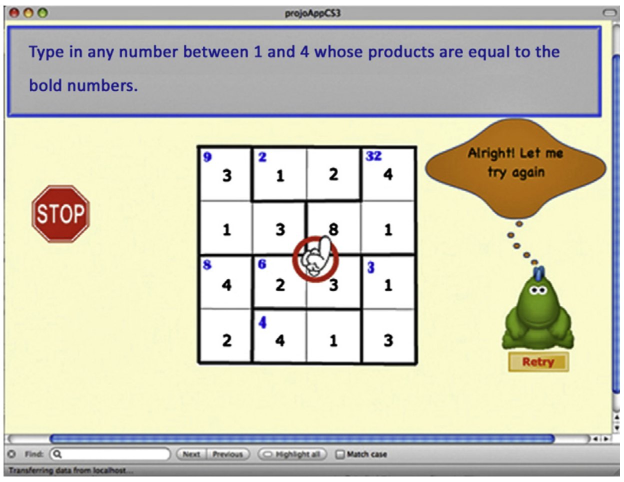
Type in any number between 1 and 4 whose products are equal to the bold numbers.

-The product of some of the entries are not correct!

- You cannot enter any number more than four

6 2	3	1	4 4
12 6	2	12 3	2
2 2	4 4	4	1
1	1	12 3	4

Fig. 11. Self training in Puzzle Math.


Type in any number between 1 and 4 whose products are equal to the bold numbers.

Alright! Let me try again

projoAppCS3
STOP
Retry
Find: Next Previous Highlight all Match case
Transferring data from localhost...

Fig. 12. Self-other training in Puzzle Math.

4.2. Materials and measures

The learning environment, Puzzle Math, incorporated many of the lessons learned from the first study, i.e., (a) a longer intervention that consisted of a full hour of training each day, for two consecutive days (more than double the time of the first experiment), (b) using puzzles consisting of sub-problems that make up a larger puzzle to make the reasoning process short and explicit, (c) increasing the number of practice problems following the monitoring session for the self-other training session, and (d) the addition of a logging feature that monitors/detects when students are self-correcting.

The testing environment "Puzzle Math" was created to make the reasoning process more explicit. The divisibility rule of multiplication was applied when solving the puzzle, providing multiple pathways to solve the same problem (e.g., 12 can be factored by 1 × 12,

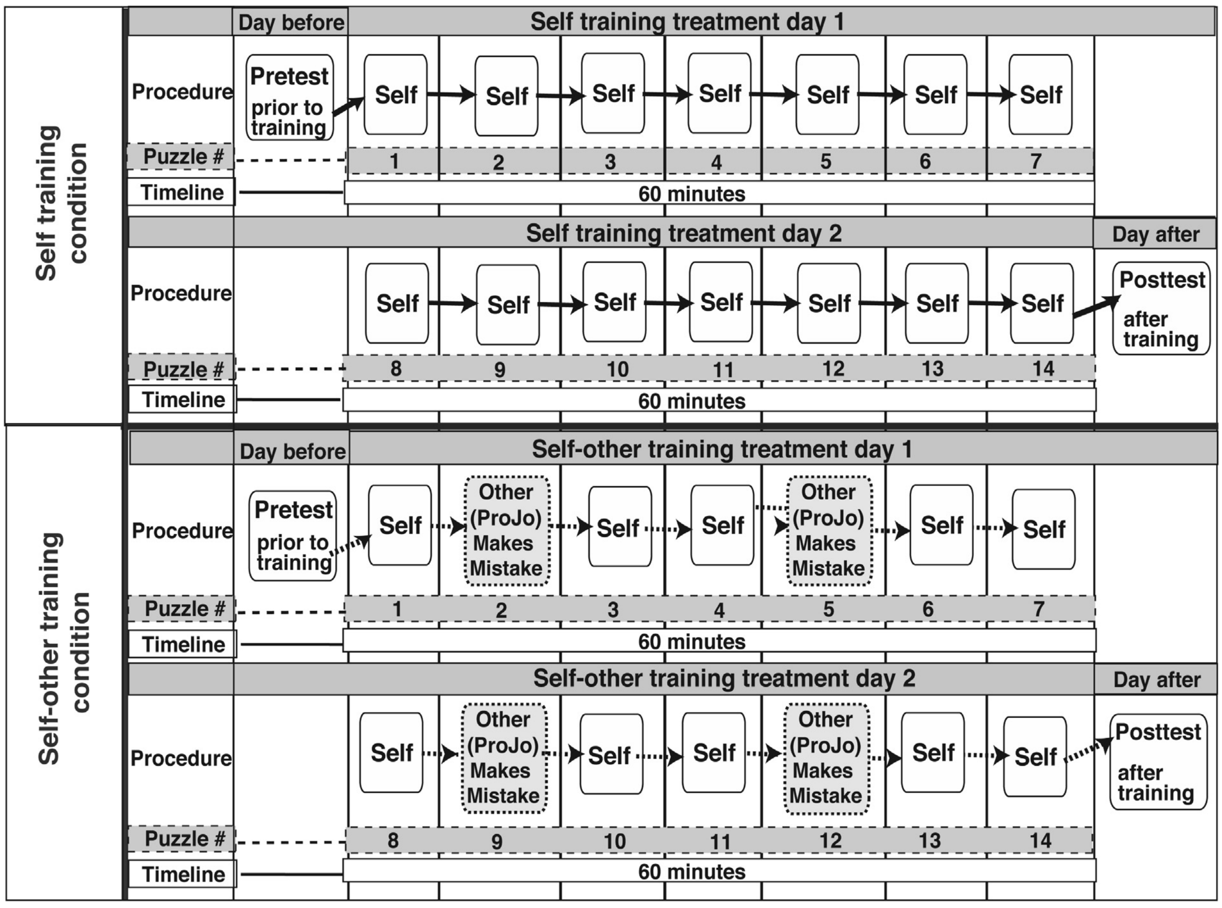

Self training condition
Day before
Self training treatment day 1
Procedure
Pretest prior to training
Self
Puzzle #
1 2 3 4 5 6 7
Timeline
60 minutes
Self training treatment day 2
Day after
Posttest after training
8 9 10 11 12 13 14
Self-other training condition
Self-other training treatment day 1
Other (ProJo) Makes Mistake
Self-other training treatment day 2

Fig. 13. Research design and procedure for experiment 2.

Table 3
Description of measures used in the pretest during the treatment sessions in days 1 and 2, and in the posttest.

Prior-to-Intervention measures				
Pretest	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
• Two puzzles • One easy level • One medium level • Subset of six to eight problems in each puzzle • General divisibility rules	• Assess student's mathematics performance level prior to study	Problem solver (Solve on own)	• Assess student's mathematics performance level prior to study	Problem solver (Solve on own)
Training treatment day 1 and day 2				
Puzzle used in treatment	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
Puzzle#: **1, 3, 4, 6, 7 (Day 1)** **8, 10, 11, 13, 14 (Day2)**• 10 puzzles ■ Four easy level (Calculation only) ■ Six medium level (Calculation and Rule) • Subset of six to eight problems in each puzzle • Includes log tracking of self-correcting behavior • General divisibility rules	• Assess student's mathematics performance from self-other training (i.e., accuracy) • See immediate "before and after" effects after monitoring practice w/computer character • Log file to monitor self-correcting behavior	Problem solver (Solve on own)	• Assess student's mathematics performance from self training (i.e., accuracy) • See log file whether students learned to self-assess and self-correct when solving puzzles	Problem solver (Solve on own)
Puzzle# **2, 5 (Day 1)** **9, 12 (Day2)**• Four puzzles ■ Two easy level (Calculation only) ■ Two medium level (Calculation and Rule) • Subset of six to eight problems in each puzzle • General divisibility rules	• See if students can monitor the computer character solve the puzzle, and catch any mistakes	Peer observer (Monitor ProJo)	• Assess student's mathematics performance from self training (i.e., accuracy)	Problem solver (Solve on own)
Post-treatment session				
Posttest	Self-other training		Self training	
	Purpose	Role of student	Purpose	Role of student
• Three puzzles ■ One easy level ■ Two medium level • Subset of six to eight problems in each puzzle • General divisibility rules	• Assess student's mathematics performance post treatment in self-other training (i.e., accuracy, time)	Problem solver (Solve on own)	• Assess student's mathematics performance in self training, post treatment (i.e., accuracy, time)	Problem solver (Solve on own)

2×6, 3×4, and $3 \times 2 \times 2$). Puzzle Math has a set of six to eight problems in which the divisible numbers are small, and the number of factors are limited. The computer character's reasoning can be monitored in steps through short subtasks (e.g., pieces of puzzle) that are part of a larger task (e.g., solve the entire puzzle), making the computer character's reasoning more explicit and easier for the student to follow.

As the treatment progressed, restrictions were imposed on the divisibility rules for each puzzle (e.g., type in any number between 1 and 9, where the product equals the bold numbers). The restrictions became slightly more challenging as the puzzle game progressed (Fig. 12), where the easy level rule was Calculation only: use any number whose products equal the bold number. The medium level rule was Calculation and Rule: Use any number between 1 and 4 whose products equal the bold number. The dependent variables included accuracy (percentage) and monitoring (frequency of self-correction). To evaluate students' performance from day 1 and day 2 of training, accuracy measures and screen shots were taken during the pretest, posttest, and training session, and detection log files were monitored. To evaluate whether the students were monitoring their work, the frequency of self-corrective behaviors was recorded in the student's log file (Fig. 15).

4.2.1. *Pretest and posttest*

Prior to the study, a pretest was conducted to examine the performance level of the students working on the mathematics puzzle (Table 3). The pretest consisted of two puzzles that were made from a small subset of six to eight mathematics problems on divisibility rules. The pretest included one easy-level puzzle and one medium-level puzzle. The posttest attempted to determine whether the two-day training on learning and behavior (self training or self-other training) had an overall effect. The posttest consisted of three puzzles, one easy-level puzzle and two medium-level puzzles. The alpha coefficients for the resulting pretest and posttest measures were 0.92 and 0.85, respectively.

4.2.2. *Performance during training treatment sessions on day 1 and 2*

The treatment sessions consisted of the same 14 puzzles (seven puzzles on day 1 and seven puzzles on day 2) for both the self training and self-other training conditions (Table 3). The monitoring treatment session consisted of 14 mathematical puzzles for both conditions (over the two days), and the resulting alpha coefficient for the mathematical puzzle measure was 0.86.

On the first day of treatment (Fig. 13), the students in the self training condition engaged in three puzzles (puzzle numbers 1, 2, and 3) at the easy-level divisibility rule and four puzzles (puzzle numbers 4, 5, 6, and 7) at the medium-level divisibility rule. The students in the self-other training condition took turns with the computer character ProJo when solving the puzzles. The student solved a total of five puzzles (two at the easy level and three at the medium level of difficulty), and ProJo solved two puzzles (one at the easy level and one at the medium level) while the student monitored for any mistakes. The puzzles that the student solved independently from ProJo (puzzle numbers 1, 3, 4, 6, and 7) were designed to indicate the immediate before and after effects of monitoring. The student monitored the computer character ProJo as it solved puzzles 2 and 5. The puzzles were organized in a way that increased the number of practice problems the student solved on her or his own following the monitoring session with ProJo (i.e., two puzzles for the student to solve independently after monitoring ProJo). ProJo solved the small subset problems that were sometimes correct and sometimes incorrect within each puzzle, e.g., Fig. 12 indicated that ProJo made a mistake by calculating 4×8 for the value 32, when the divisibility rule said to use “numbers between 1 and 4”. On the second day of treatment, the same procedure was repeated for both conditions during the training treatment session, and that was followed by the posttest (Fig. 13).

4.2.3. *Scoring*

One puzzle can consist of six to eight short multiplication problems. For example, the puzzle shown in Fig. 14 is made up of seven small mathematics problems (e.g., 2 can be factored to 2×1; 9 can be factored to $3 \times 3 \times 1$). Each puzzle came with a restricting rule: “Type in any number between 1 and 4 whose products are equal to the bold number.” There were three accuracy scores for each puzzle: (a) the number of problems for which only the calculation was correct, (b) the calculation was correct and the rules were followed, and (c) whether the student got the entire puzzle correct. For example, the puzzle in Fig. 10 consisted of three accuracy scores, i.e., (a) 7 points out of 7 because the calculations for all seven problems were correct, (b) 6 points out of 7 because even though the calculation was correct, the student did not follow the rule (numbers between 1 and 4 only) for 32 ($4 \times 8 \times 1$), and (c) 0 out of 1 (because the entire puzzle must be correct for both calculation and rule). The monitor detection log was a script that ran in the background of the application keeping track of how many times the student self-corrected and changed her or his answer while solving the puzzle (Fig. 15).

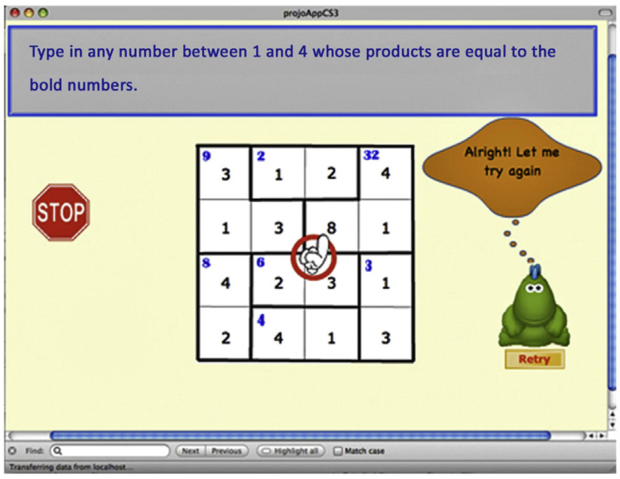
4.3. *Results*

The results section will be structured around the two research questions: (a) Did longer self-other training sessions with the computer character improve mathematics performance? (b) Did self-other training sessions lead to more accurate self-correcting behaviors?

4.3.1. *Effect of longer training sessions on students’ mathematics performance*

The accuracy of the data on student performance was categorized into three different scores. The pretest showed no difference in the groups prior to the study (Fig. 16). The three accuracy scores during the training session over the two days are shown in Fig. 17. Overall, students in the self-other training had a higher accuracy score than those in self training. The students in the self-other training seemed to improve at a higher rate from training day 1 to day 2. The differences between the two groups in performance (during the training session) started to appear by the end of the second day. The accuracy of “calculation” between the two groups approached significance at (Self training $M = 68\%$, Self-other training $M = 87\%$) $t(20) = -1.825$, $p = 0.083$. For the accuracy of following “rules and calculation,” the differences between the two groups approached significance at (Self training $M = 59\%$, Self-other training $M = 78\%$) $t(20) = -1.833$, $p = 0.082$.

By the time the students took the posttest, there was a significant difference between the two groups, with the self-other training group performing better than the self training group. The posttest accuracy scores for the calculation-only problems were significantly different between the two trainings (Self training $M = 55\%$, Self-other training $M = 96\%$) $t(20) = -3.28$, $p < 0.05$. Even with the rule restrictions, there



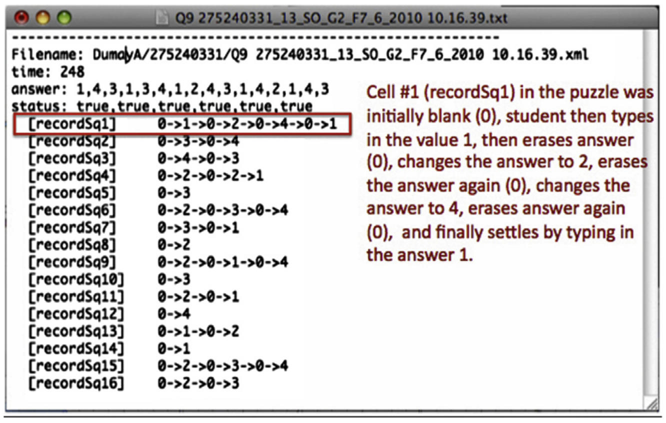
Type in any number between 1 and 4 whose products are equal to the bold numbers.

projoAppCS3
STOP
Alright! Let me try again
Retry
Find: Next Previous Highlight all Match case
Transferring data from localhost...

Scoring Method (example)
Rule Given: Type in any number between 1 and 4 whose products are equal to the bold blue numbers

Calculation Only: Number of sub-problems where only the calculations are correct.	**Score out of 7:** 7
Calculation and Rule: Number of sub-problems where calculations are correct, AND rules followed.	**Score out of 7:** 6
Entire Puzzle: Entire puzzle is correct.	**Score out of 1:** 0

Fig. 14. Scoring method example (for puzzle shown on left).



```
------------------------------------------------------------
Filename: DumayA/275240331/Q9 275240331_13_SO_G2_F7_6_2010 10.16.39.xml
time: 248
answer: 1,4,3,1,3,4,1,2,4,3,1,4,2,1,4,3
status: true,true,true,true,true,true
  [recordSq1]     0->1->0->2->0->4->0->1
  [recordSq2]     0->3->0->4
  [recordSq3]     0->4->0->3
  [recordSq4]     0->2->0->2->1
  [recordSq5]     0->3
  [recordSq6]     0->2->0->3->0->4
  [recordSq7]     0->3->0->1
  [recordSq8]     0->2
  [recordSq9]     0->2->0->1->0->4
  [recordSq10]    0->3
  [recordSq11]    0->2->0->1
  [recordSq12]    0->4
  [recordSq13]    0->1->0->2
  [recordSq14]    0->1
  [recordSq15]    0->2->0->3->0->4
  [recordSq16]    0->2->0->3
```

Cell #1 (recordSq1) in the puzzle was initially blank (0), student then types in the value 1, then erases answer (0), changes the answer to 2, erases the answer again (0), changes the answer to 4, erases answer again (0), and finally settles by typing in the answer 1.

Fig. 15. Record of students' self-correcting.

was a significant difference between the two groups, with the self-other training group scoring higher than the self training group at (Self training $M = 43\%$, Self-other training $M = 88\%$) $t(20) = -3.60$, $p < 0.05$.

4.3.2. *Effect of self-other training sessions on students' self-correcting behaviors*

The monitor detection log recorded the number of incidents in which the students self-corrected during problem solving and changed their answers before finishing the puzzle (Fig. 18). In addition to the number of times the students self-corrected, the analysis indicated how many of those corrections were actually correct. The results showed that self-other training had significantly more cases of self-correction than self training (Self training $M = 9.6$, Self-other training $M = 32.6$) $t(20) = -2.25$, $p < 0.05$. There was also a significant difference between the two groups on the accuracy of the self-correction (Self training $M = 8.6$, Self-other training $M = 18.8$) $t(20) = -3.03$, $p < 0.05$. Although the number of self-corrections decreased slightly from day 1 to day 2, the accuracy in self-correction increased from day 1 to day 2 for the self-other training.

4.4. *Discussion*

The accuracy score on the posttest indicated that students in the self-other training group performed better than the students in the self training group (Fig. 16). This also was observed during the training session, where self-other training had a higher performance rate from day 1 to day 2 (Fig. 17). The difference between the two trainings was not significant until the posttest, and this may have been attributable to the late-gain effect. Students in the self-other training experience improved their performance even when they were no longer in the context of the computer character, i.e., the posttest was a pencil/paper task. As an indicator of monitoring, the number of self-corrections was significantly greater in the self-other training (Fig. 18). Although the number of incidents decreased slightly over time, the quality of self-correction increased for self-other training, since the accuracy increased from day 1 to day 2; however, in the self training group, both self-correction and accuracy decreased from day 1 to day 2.

5. General discussion

In the first study, using Doodle Math was a between-subjects design in which the students in self training solved problems on their own, while self-other training students took turns with ProJo and monitored the computer character for any mistakes. The study demonstrated some suggestive findings. It was easier for students to catch the computer character ProJo's mistakes than to catch their own. However, students found it difficult to catch mistakes that were similar to their own. A late-gain effect was found, such that students in the self-other training initially slowed down, started out with less accuracy, and, by the end of the training, scored slightly above the self training group. The results were interesting because students in self-other training did not learn less, even though half of their time was spent monitoring ProJo instead of explicitly solving the problems themselves.

The second study with Puzzle Math addressed many of the issues and limitations in the first study, i.e., focusing on extending the training time, increasing the number of practice problems, using shorter subtasks to make the reasoning process short and explicit, and adding a logging feature to detect the students' self-corrections. The second study demonstrated that extended training seemed to provide a longer-lasting effect on the posttest. The students in the self-other training had a higher accuracy score throughout the session, and, by the time the posttest was taken, the difference was significant. Students in the self-other training showed more self-correction with higher accuracy than students in the self training.

The two studies have practical implications for both students and teachers. Through the process of self-other monitoring, which was proposed as being easier than self-monitoring, students learned to self-assess, which not only helped them learn the topic at hand, but also prepared them to self-monitor and correct their own mistakes.

An immediate implication is that the results can be used to address common mathematics difficulties at the elementary grade level. Careless mistakes often involve students who understand but make simple mistakes. These mistakes can range from process problems and precedence errors to substitution and procedural errors, which are all common among elementary and middle school students. Repeating these errors can be quite serious because it often leads to long-term difficulties with mathematics (Watson, 1980) and the development of negative perceptions of mathematics and science (Steele, 2003). Confusion and discouragement from careless mistakes were minimized

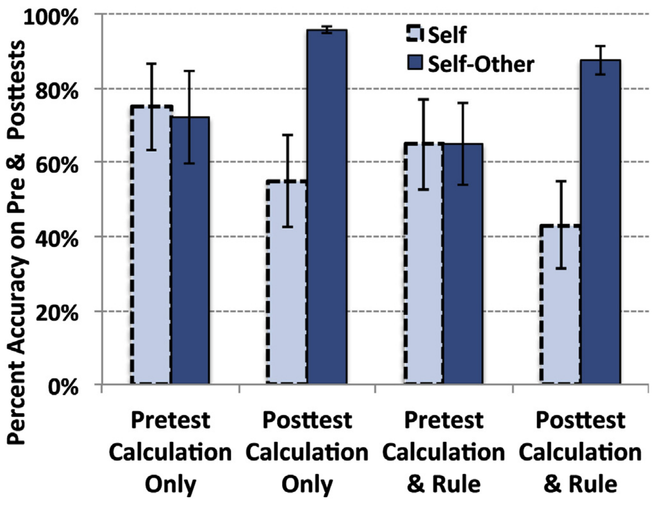


Fig. 16. Pretest and posttest scores on accuracy.

when children learned to check and self-correct their answers. Teachers have little time to spend on these problems in school, since correcting habitual behaviors requires personal attention, time, and consistency on the part of the enforcer (Blando et al., 1989; Wells & Coffey, 2005). Developing metacognitive skills to self-correct may help remove one key roadblock to young students' interest in mathematics-related subjects.

A long-term implication is helping students continue to learn and improve on their own. Rather than using a computerized peer, some may argue that the use an external influence, e.g., a teacher, a coach, a parent, a peer, or an intelligent tutor, to point out places for improvement may suffice. However, students may not always have a "knowledgeable other" at their side, and once they leave school, students would need to learn without supervision. In underserved communities, students may have limited access to good external influences or a "knowledgeable other." A more practical approach would be for students to "learn for themselves" and develop the ability to "self-assess and monitor" their understanding, self-correct mistakes, and make informed decisions in the future.

This research also has practical implications for teachers in formal instruction. When assisting teachers with student-centered instruction, the findings from the two studies further explore the use of automated assessments. For example, the computer can automatically assess each student's performance, self-correcting behavior, and provide useful formative information to the teacher (and student) without the need for separate testing that takes away from instructional time and burdensome grading (Barksdale-Ladd & Thomas, 2000). The two mathematics learning environments can be useful tools for schools in underserved communities in which teachers have high teaching loads with little time to closely monitor students' progress and learning behaviors in both formal and informal practices (e.g., homework).

As a tool, the future interest is to ensure that these learning environments (i.e., Doodle Math, and Puzzle Math) are optimized to create model technologies that demonstrate how to implement conditions that train students how to conduct self-assessment. The specific analysis in the findings (i.e., conditional probability analysis) has helped suggest some practical ways to design individualized, online, learning instructions. The descriptive analysis from the first study led to two suggestions for how the computer character's errors could be arranged to increase or decrease the students' performance on the posttest. The analysis suggests that, if the students get a problem right, then it is better to let them solve another one on their own rather than monitor whether a computer character makes a mistake. However, if the students get a problem wrong, it is better to have them monitor a computer character's mistakes rather than solve another problem on their own. The analysis was based on the different possible combinations of error sequences from training to posttest performance for a few specific types of problems. Learning implications are inextricably related to the exploration of how people interact with, around, and through these pedagogical, peer-like characters. According to Koschmann (2002), technological functions and implications reveal

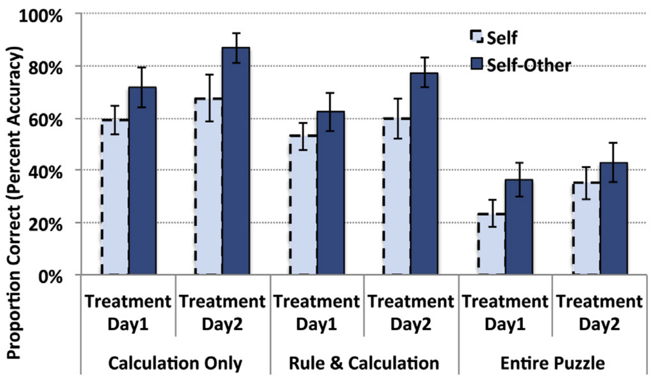


Fig. 17. Accuracy score by training over two days.

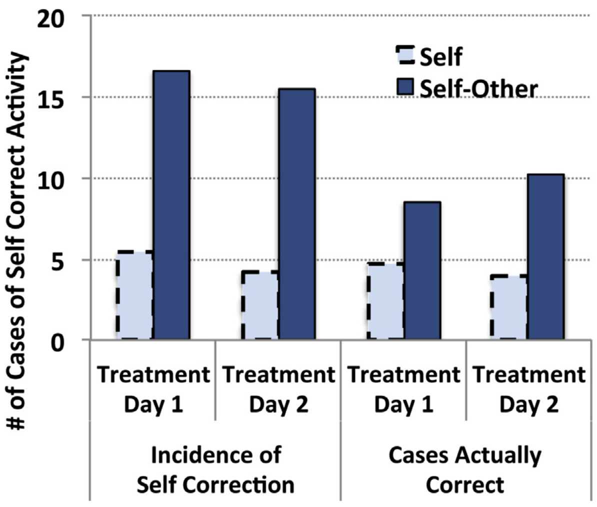


Fig. 18. Number of self-correcting behaviors observed during the training.

themselves based on how humans use them and that it is the learner who constitutes meanings from the features of computer applications. Still, very little is known about how to design adaptive support/responses to create a positive effect on academic performance. The conditional probability analysis from the first study illustrated how specific features in computer characters can be executed in a way that maximizes students' learning.

Another contribution is the implementation of two computer-assisted learning environments (i.e., Doodle Math, Puzzle Math), in which self-other training could be directly compared to self training, providing a testing ground in which students can practice monitoring with a computer character. In examining ways to design features to maximize student performance in computer-assisted learning environments, ProJo provided a controlled interactive test bed that could isolate and adjust factors (e.g., how to arrange a computer character's errors) and keep track of students' progress and achievements in various data forms (e.g., log file, screen shots, calculation time, and accuracy scores) that may help better assess students' progress.

The work provided initial evidence that self-other training may be an effective way to help students develop metacognitive skills. No doubt, a larger-scale study is needed to create a more effective and robust system. Nevertheless, the small sample studies did provide templates for larger, more ambitious studies, and the results appear to indicate that monitoring a computer character changes students' behavior in ways that merit follow-up.

6. Future work

Two directions for future research involve research and technical development. With regard to research, there are two types of useful studies, i.e., learning effectiveness studies that determine whether self-other training leads to effective learning and causal studies that try to determine why there are changes. For learning effects, a simple next step would be to increase the sample size to further examine the effects and performance. The training sessions for the two studies were limited to 1–2 days, and the studies used only the divisibility rules of multiplication in elementary mathematics. A study that may promote further learning effects should use a longer intervention that lasts a few months and covers several topics. Such a study would determine whether a more sustained practice at self-other training leads to stable effects and improved learning, even when students are no longer in the context of a pedagogical, computer character-monitoring environment.

The second direction for future research is technical development. ProJo has a very simple interface and appearance. One area of improvement may be to increase the presence of ProJo by providing better graphics and voice, as well as a more elaborative dialog interaction. A more sophisticated version of ProJo could be designed in which the system makes real-time decisions to have the student monitor an incorrect computer character, monitor a correct computer character, or simply keep working on her or his own. In this practical hypothesis, there is still a need for extensive empirical work. The extant studies did not sample the effects of correct computer character behaviors or more variation in incorrect computer character behaviors across the different types of errors. Once all the possible patterns are covered and effective computer character behaviors are revealed (e.g., which patterns increase or decrease students' learning), the next step would be to create a dynamic ProJo that can choose the most effective problem for the student in real time.

7. Summary

In this competitive world, success depends on continued learning, often without supervision. A future skill needed for students is to learn for themselves and make informed decisions. A critical ingredient of being able to learn for oneself is the ability to self-assess one's learning progress. The two studies created model technologies that demonstrated how to implement conditions that realize self-other assessments and examined the novel hypothesis that observing "someone" else who knows little more than oneself can be better for learning than just doing things oneself. Thus, "self-other" assessments are an important way in which children can learn to self-assess. A total of 62, students, in the age range of nine to 11, participated in two experiments that directly compared the results of self training and self-other training. In the training environment, a computer character, "ProJo," openly displayed its reasoning when solving mathematics problems and allowed children to “look for mistakes.” The students in the self training group solved all the problems on their own, while the students in the self-other training group worked with the computer character ProJo, taking turns to solve problems and monitor for any mistakes. The measures on calculation time and accuracy showed that self-other training might be an effective way to help students develop metacognitive skills to self-correct and improve performance in elementary mathematics. The log file tracked students’ progress in various data forms, displaying evidence that students in the self-other training group monitored and self-corrected more than students in the self training group. ProJo provides one instance of technology using pedagogical computer characters that take on the role of peer learners to students in developing monitoring skills. The results also suggested possible ways to design adaptive systems to maximize academic performance and provided two computer-assisted learning environments (i.e., Doodle Math and Puzzle Math) that tracked students’ progress and achievements in various data forms.

References

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: lessons learned. *Journal of Learning Sciences, 4*(2), 167–207.

Azmitia, M. (1996). Peer interactive minds: developmental, theoretical, and methodological issues. In P. B. Baltes, & U. M. Staudinger (Eds.), *Interactive minds: Life-span perspectives on the social foundation of cognition* (pp. 133–162). Cambridge University Press.

Bailenson, J. N., Patel, K., Nielsen, A., Bajscy, R., Jung, S., & Kurillo, G. (2008). The effect of interactivity on learning physical actions in virtual reality. *Media Psychology, 11*, 354–376.

Barksdale-Ladd, M. A., & Thomas, K. F. (2000). What’s at stake in high-stakes testing: teachers and parents speak out. *Journal of Teacher Education, 51*(5), 384–397.

Barron, B. (2003). When smart groups fail. *Journal of the Learning Sciences, 12*(3), 307–359.

Baylor, A. L. (2007). Pedagogical agents as a social interface. *Educational Technology, 47*(1), 11–14.

Biswas, G., Leelawong, K., Schwartz, D., Vye, N., & TAG-V. (2005). Learning by teaching: a new agent paradigm for educational software. *Applied Artificial Intelligence, 19*, 363–392.

Blair, K., Schwartz, D., Biswas, G., & Leelawong, K. (2007). Pedagogical agents for learning by teaching: teachable agents. *Education Technology, 47*, 56–61.

Blando, J. A., Kelly, A. E., Schneider, B. R., & Sleeman, D. (1989). Analyzing and modeling arithmetic errors. *Journal for Research in Mathematics Education, 20*(3), 301–308.

Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn* (expanded ed.). Washington, D.C: National Academy Press.

Brown, A. (1987). Metacognition, executive control, self-regulation and other more mysterious mechanisms. In F. E. Weinert, & R. H. Kluwq (Eds.), *Metacognition, motivation and understanding*. New Jersey: Lawrence Erlbaum Associates.

Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science, 2*, 155–192.

Campione, J. C., Brown, A. L., Ferrara, R. A., & Bryant, N. R. (1984). The zone of proximal development: implications for individual differences in learning. In B. Rogoff, & J. V. Wertsch (Eds.), *New directions for child development: No. 23: Children’s learning in the zone of proximal development* (pp. 77–91). San Francisco, CA: Jossey Bass.

Cassell, J. (2004). Towards a model of technology and literacy development: story listening systems. *Journal of Applied Developmental Psychology, 25*(1), 75–105.

Cauley, K. M. (1988). Construction of logical knowledge: study of borrowing in subtraction. *Journal of Educational Psychology, 80*(2), 202–205.

Chi, M. T. H., & De Leeuw, N. (1994). Eliciting self-explanations improves understanding. *Cognitive Science, 18*(3), 439–477.

Chi, M. T. H., Silver, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25*, 471–533.

Chin, D. B., Dohmen, I. M., Cheng, B. H., Oppezzo, M. A., Chase, C. C., & Schwartz, D. L. (2010). Preparing students for future learning with teachable agents. *Educational Technology Research and Development, 58*, 649–670.

Chin, D. B., Dohmen, I. M., & Schwartz, D. L. (2013). Young children can learn scientific reasoning with teachable agents. *IEEE, Transactions on Learning Technologies, 6*(3), 248–257.

Chou, C. Y., Chan, T. W., & Lin, C. J. (2002). An approach of implementing general learning companions for problem solving. *IEEE Transactions on Knowledge and Data Engineering, 14*(6), 1376–1386.

Clark, E. V. (1995). Later lexical development and word formation. In P. Fletcher, & B. MacWhinney (Eds.), *The handbook of child language* (pp. 393–412). Cambridge: Basil Blackwell.

Dillenbourg, P., & Jermann, P. (2007). Designing integrative scripts. In F. Fischer, H. Mandl, J. Haake, & I. Kollar (Eds.), *Scripting computer-supported communication of knowledge – Cognitive, computational and educational perspectives* (pp. 275–301). New York: Springer.

Fantuzzo, J. W., Riggio, R. E., Connelly, S., & Dimeff, L. A. (1989). Effects of reciprocal peer tutoring on academic achievement and psychological adjustment: a component analysis. *Journal of Educational Psychology, 81*(2), 173–177.

Fischbein, E. (1987). *Intuition in science and mathematics: An educational approach*. Dordrecht, The Netherlands: Reidel.

Geary, D. (2002). Principles of evolutionary educational psychology. *Learning and Individual Differences, 12*, 317–345.

Gelman, R., & Meck, E. (1983). Preschoolers’ counting: principles before skill. *Cognition, 13*, 343–359.

Gilovich, T., Griffin, D., & Kahneman, D. (2002). *Heuristics and biases: The psychology of intuitive judgment*. Cambridge University Press.

Harris, P., & Want, S. (2005). On learning what not to do: the emergence of selective imitation in young children’s tool use. In S. Hurley, & N. Chater (Eds.), *Perspectives on imitation: From neuroscience to social science*. Cambridge, MA: The MIT Press.

Hurme, T., Palone, T., & Järvelä, S. (2006). Metacognition in joint discussions: an analysis of the patterns of interaction and the metacognitive content of the networked discussions in mathematics. *Metacognition Learning, 1*, 181–200.

Kahneman, D. (2002). In T. Frangsmyr (Ed.), *Maps of bounded rationality: A perspective on intuitive judgment and choice* (pp. 416–499). Les Prix Nobel. (Nobel Prize Lecture, December 8). Also accessible at http://www.nobel.se/economics/laureates/2002/kahnemann lecture.pdf.

Karmiloff-Smith, A. (1979). Micro- and macro- developmental changes in language acquisition and other representational systems. *Cognitive Science, 3*, 91–118.

King, A., Staffieri, A., & Adelgais, A. (1998). Mutual peer tutoring: effects of structuring tutorial interaction to scaffold peer learning. *Journal of Educational Psychology, 90*, 134–152.

Koedinger, K. R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review, 19*, 239–264.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8*, 30–43.

Koschmann, T. (2002). Dewey’s contribution to the foundations of CSCL research. In G. Stahl (Ed.), *Computer support for collaborative learning: Foundations for a CSCL community: Proceedings of CSCL, 2002* (pp. 17–22). Boulder CO: Lawrence Erlbaum.

Kreijns, K., Kirschner, P. A., & Jochems, W. (2003). Identifying the pitfalls for social interaction in computer-supported collaborative learning environments: a review of the research. *Computers in Human Behavior, 19*, 335–353.

Leelawong, K., & Biswas, G. (2008). Designing learning by teaching agents: the Betty’s Brain system. *International Journal of Artificial Intelligence in Education, 18*(3), 181–208.

Leron, U., & Hazzan, O. (2006). The rationality debate: application of cognitive psychology to mathematics education. *Educational Studies in Mathematics, 62*(2), 105–126.

Markman, E. (1977). Realizing that you don’t understand: elementary school children’s awareness of inconsistencies. *Child Development, 48*, 986–992.

Martin, B., Mitrovic, T., Mathan, S., & Koedinger, K. R. (2011). Evaluating and improving adaptive educational systems with learning curves. *User Model User-Adapted Interaction, 21*, 249–283.

Matsuda, N., Cohen, W. W., Koedinger, K. R., Keiser, V., Raizada, R., Yarzebinski, E., et al. (2012). Studying the effect of tutor learning using a teachable agent that asks the student tutor for explanations. In M. Sugimoto, V. Aleven, Y. S. Chee, & B. F. Manjon (Eds.), *Proceedings of the international conference on digital game and intelligent toy enhanced learning (DIGITEL 2012)* (pp. 25–32). Los Alamitos, CA: IEEE Computer Society.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: a comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences, 2*, 277–306.

Mullins, D., Rummel, N., & Spada, H. (2011). Are two heads always better than one? Differential effects of collaboration on students’ computer-supported learning in mathematics. *International Journal of Computer-Supported Collaborative Learning, 6*, 421–443.

Okita, S. Y., & Schwartz, D. L. (2006). When observation beats doing: learning by teaching. In S. Barab, K. Hay, & D. Hickey (Eds.), *Proceedings of the 7th international conference of the learning sciences* (Vol. 1; pp. 509–515). New Jersey, Bloomington, USA: Lawrence Erlbaum Associates.

Okita, S. Y., & Schwartz, D. L. (2013). Learning by teaching human pupils and teachable agents: the importance of recursive feedback. *Journal of the Learning Sciences, 22*(3), 375–412.
Palincsar, A. S., & Brown, A. L. (1984). Reciprocal teaching of comprehension fostering and comprehension-monitoring activities. *Cognition and Instruction, 2*(2), 117–175.
Pareto, L., Haake, M., Lindström, P., Sjödén, B., & Gulz, A. (2012). A teachable agent based game affording collaboration and competition: evaluating math comprehension and motivation. *Educational Technology Research and Development, 60*, 723–751.
Polya, G. (1973). *How to solve it?* Princeton, New Jersey: Princeton University Press.
Resnick, L. B., Lenive, J. M., & Teasley, S. D. (1993). *Perspectives on socially shared cognition*. Washington: American Psychological Association.
Rogoff, B., Mistry, J., Göncü, A., & Mosier, C. (1993). Guided participation in cultural activity by toddlers and caregivers. *Monographs of the Society for Research in Child Development, 58*(236).
Roscoe, R. D., & Chi, M. (2007). Understanding tutor learning: knowledge-building and knowledge-telling in peer tutors' explanations and questions. *Review of Educational Research, 77*(4), 534–574.
Russell, R. L., & Ginsburg, H. P. (1984). Cognitive analysis of children's mathematics difficulties. *Cognition and Instruction, 1*(2), 217–244.
Schoenfeld, A. H. (1985). *Mathematical problem solving*. New York: Academic Press.
Schoenfeld, A. H. (1987). What is all the fuss about metacognition? In A. H. Schoenfeld (Ed.), *Cognitive science and mathematics education* (pp. 189–215) Lawrence Erlbaum Associates.
Schon, D. A. (1983). *The reflective practitioner*. Basic Books.
Schon, D. A. (1987). *Educating the reflective practitioner: Towards a new design for teaching and learning in the profession*. San Francisco: Jossey-Bass.
Seifert, C. M., & Hutchins, E. L. (1992). Error as opportunity: learning in a cooperative task. *Human–Computer Interaction, 7*, 409–435.
Stanovich, K. E., & West, R. F. (2000). Individual differences in reasoning: implications for the rationality debate. *Behavioral and Brain Sciences, 23*, 645–726.
Stanovich, K. E., & West, R. F. (2003). Evolutionary versus instrumental goals: how evolutionary psychology misconceives human rationality'. In D. E. Over (Ed.), *Evolution and the psychology of thinking: The debate* (pp. 171–230). Psychology Press.
Stavy, R., & Tirosh, D. (2000). *How students (mis-)understand science and mathematics: Intuitive rules*. Teachers College Press.
Steele, J. (2003). Children's gender stereotypes about math: the role of stereotype stratification. *Journal of Applied Social Psychology, 33*, 2587–2606.
Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Cambridge, MA: Harvard University Press.
Walker, E. N. (2006). Urban high school academic communities and their effects on mathematics success. *American Educational Research Journal, 43*(1), 43–73.
Walker, E., Rummel, N., & Koedinger, K. R. (2011). Designing automated adaptive support to improve student helping behaviors in a peer tutoring activity. *International Journal of Computer-Supported Collaborative Learning, 6*(2), 279–306.
Watson, I. (1980). Investigating errors of beginning mathematicians. *Educational Studies in Mathematics, 11*(3), 319–329.
Webb, N. M. (1989). Peer interaction and learning in small groups. *International Journal of Educational Research, 13*, 21–39.
Wells, P. J., & Coffey, D. C. (2005). Are they wrong? Or did they just answer a different question? *Teaching Children Mathematics, 12*(4), 202–207.
Wenger, E. (1987). *Artificial intelligence and tutoring systems*. Los Altos, California: Morgan Kaufmann Publishers.
Wertsch, J. V. (1978). Adult-child interaction and the roots of metacognition. *The Quarterly Newsletter of the Institute for Comparative Human Development, 2*, 15–18.
Zimmerman, B. J. (1995). Self-regulation involves more than metacognition: a social cognitive perspective. *Educational Psychologist, 30*, 217–221.