Chapter 13
Roles, Collaboration, and the Development of Computational Thinking in a Robotics Learning Environment

P. Kevin Keith, Florence R. Sullivan and Duy Pham

Abstract Robotics is a robust vehicle for supporting the development of computational thinking in students. Educational robotics activities unfold in a multidimensional problem space that requires the integration of programming, building, and environmental aspects of the activity. Students working collaboratively with robotics have the opportunity to adopt roles within the group that are aligned to these multiple dimensions (e.g., programmer, builder, and analyst). Group roles are an important element of all collaborative learning, but especially in a Computer-Supported Collaborative Learning (CSCL) environment, as the roles help to regulate group activity and learning. In this observational, microgenetic case study, we investigated the relationship of the roles middle school-aged girls adopted to the collaborative interactions they engaged in, and, ultimately to the development of computational thinking evidenced as a result of role participation. The video and audiotaped data used in this study were collected at a 1-day introduction to robotics workshop for girls. Our mixed methods approach included sequential and qualitative analysis of the behavioral and verbal interactions of two groups of girls (n = 6) who participated in the workshop. Our results indicate that the emergence of distinct roles correlates with periods of collaboration and periods of parallel solo work, which, in turn, had an impact on student's engagement in computational thinking including solution planning, algorithmic and debugging operations, and the design of the robotic device. Moreover, students who engaged in greater levels of collaboration selected more difficult challenges to solve within the robotics environment. Suggestions for future research are provided.

Keywords Robotics · Collaboration · Computational thinking · Girls · Roles

P. Kevin Keith · F. R. Sullivan (✉) · D. Pham
College of Education, University of Massachusetts, Amherst, 813 N. Pleasant St., Amherst, MA 01003, USA
e-mail: fsullivan@educ.umass.edu

P. Kevin Keith
e-mail: pkk@umass.edu; pkkeith@educ.umass.edu

D. Pham
e-mail: dpham@umass.edu

S.-C. Kong and H. Abelson (eds.), *Computational Thinking Education*,
https://doi.org/10.1007/978-981-13-6528-7_13

13.1 Introduction

In this chapter, we examine the relationship of group roles to collaboration and computational thinking in an educational robotics setting. Robotics is an immersive activity (Rieber, 2005) that unfolds in a multidimensional problem space, featuring programming, building, and environmental aspects. Due to this multidimensionality, robotics is an activity that lends itself well to collaborative group learning; students can take on different roles including programmer, builder, and analyst in a team effort to solve the robotics problem. Moreover, the external, 3D nature of two of the three activities (building and testing in the environment), supports reasoning in the third, 2D activity (programming). Robotics is an activity of growing interest for children and youth as evidenced by their global participation in the FIRST Lego League (FLL) (First Lego League, n.d.) Moreover, our recent literature review of robotics studies indicates that it is a collaborative activity that has the potential to develop students' computational thinking abilities (Sullivan & Heffernan, 2016). Yet, while this potential exists, there is little research specifically devoted to understanding the relationship of student collaborative learning with robotics and the development of computational thinking.

Meanwhile, the development of computational thinking is increasingly recognized as an important ability for life in the new millennium. Indeed, in the context of the US, there is a national movement to integrate computer science education into K12 education (National Science Foundation, 2016). This movement is powered by the increasingly ubiquitous nature of computing and computer science in all fields, the growing demand for computer scientists, (US Bureau of Labor Statistics, 2017), and a more diverse corps of computer scientists in the US workforce. Our work is aimed at bringing computer science learning opportunities to girls, in order to improve their awareness and knowledge of computer science, as well as to engage their interest in the field.

In this chapter, we report on a recent study in which we examined how collaborative arrangements in the multidimensional problem space of robotics influences group participation and engagement in computational thinking for middle school-aged girls. As will be demonstrated through our analysis, students who share the main roles afforded by the robotics environment (i.e., programmer, builder, and analyst) were able to develop a greater understanding of robotics and engage in computational thinking more frequently.

We chose to work with middle school-aged girls, and it is at this age that people begin to explore the possibility of future identities (Ji, Lapan & Tate, 2004). And, it is through exposure to an area of study, and opportunities for accomplishments within that area that support a feeling of self-efficacy and further interest in the field (Ali, Brown, & Loh, 2017). Therefore, in conducting research with middle school-aged girls, we aim to create knowledge that will support the development of successful educational interventions for girls that will improve girls interest and participation in computer science. Our chapter is organized in the following way: first, we provide a brief definition of computational thinking from the literature, we then provide

an overview of research on educational robotics, how it relates to computational thinking, and to collaborative learning and role adoption as an emergent quality of participation. We then present our research questions, our methods for investigating these questions, our results, and our interpretation of these results. We end with suggestions for practice and future research.

13.1.1 Computational Thinking

Computational Thinking (CT) reflects the habits of the mind engaged in by computer scientists including, but not limited to "…solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006). CT can be thought of as a broad foundation consisting of the heuristics used by computer scientists and as a way to think about the diverse thinking skills associated with computing. The Computer Science Teachers Association (CSTA) has disseminated a definition of CT as including: formulating problems in ways that enable us to use a computer to solve them, and automating solutions through algorithmic thinking. They further indicate that these skills are important because they create a tolerance for ambiguity, allow for persistence in working with difficult problems, and provide an opportunity to practice collaborating with others to achieve a common goal (Computer Science Teachers Association, 2016). An understanding of the aspects of teaching CT is important because it has been argued that CT skills are essential skills that should be taught to all school-aged learners (Lee, et al., 2011). Not only because these skills are used by computer scientists, but influence an abundance of other fields (Wing, 2006) and people with a command of these competencies will be better positioned to participate in a world where the computer is ubiquitous (Grover & Pea, 2013).

13.1.2 Educational Robotics and Computational Thinking

Robotics kits are computational manipulatives that enable student engagement in computational thinking and doing (Sullivan & Heffernan, 2016). Designed and programmed robotics devices provide immediate, concrete feedback to students on the efficacy of their programs, thereby promoting reflection and debugging analysis (Sullivan & Heffernan). These activities—designing, programming, testing, and debugging—form a troubleshooting cycle students typically enact as they work with robotics systems (Sullivan, 2011). Hence, robotics is an ideal activity for engendering computational thinking and doing in students.

Novices engaged in robotics study, begin by building a robotic device, usually using blueprints provided by Lego as part of the Mindstorms kit. They then learn how to write simple algorithms to make the robotic device move (Sullivan, 2008). This process is facilitated by the graphical programming language used in the Lego

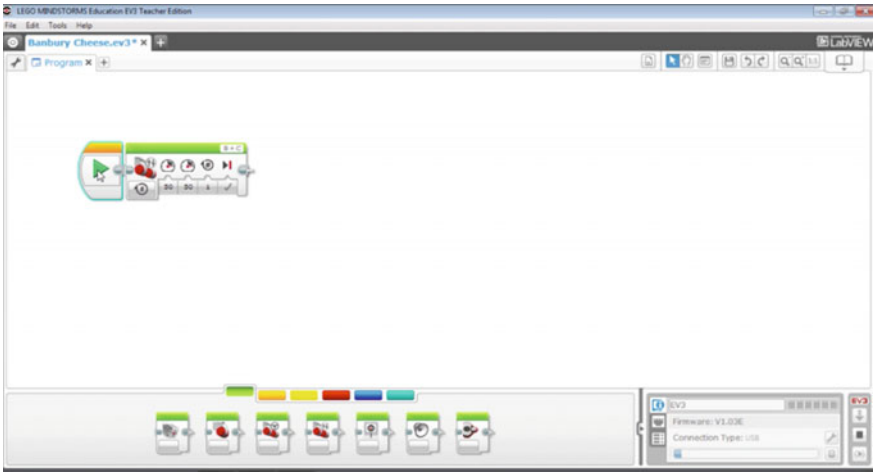

Fig. 13.1 EV3 Mindstorms graphical programming language

Mindstorms robotics kit. This programming language is a modified version of LabVIEW (see Fig. 13.1). LabVIEW is a graphical programming language developed by National Instruments, whose power derives from its ability to make "…computer science accessible to non-programmers. LabVIEW object-oriented programming brings some of the most advanced programming techniques into the grasp of beginning users" (National Instruments, 2014).

The graphical nature of the programming language allows students to immediately begin programming, writing algorithms to move the robot. The physical nature of the robotic device provides students with concrete and immediate feedback on the efficacy of the algorithm they have written (Sullivan & Heffernan, 2016). The iterative nature of the robotic activity, embodied in the troubleshooting cycle (Sullivan, 2011), allows the students to experiment with elements of the programming to broaden and deepen their knowledge of programming and to engage in computational thinking. This is supported as each of the graphical icons in the programming language has variable properties that can be set. As students test their robotic device in the environment, they begin to think about how to more precisely program the movement of the robot within the given environment (e.g., to avoid obstacles). Having the ability to vary the speed of the robot, the angle of a turn, or the distance a robotic device will travel, allows for a more precise program, more complex algorithms, and more practice with computational thinking to solve the robotics problems with algorithms. Therefore, robotics is a very robust activity for stimulating the development of computational thinking.

13.1.3 Collaborative Learning with Robotics: Emergent Roles

When working collaboratively, students take on roles as the group seeks to regulate its work. Group roles are an important element of all collaborative learning, but especially in a Computer-Supported Collaborative Learning (CSCL) environment (Hoadley, 2010), such as robotics. According to Stijbos and De Laat (2010), roles come into being in one of two ways: scripted or emergent. Scripted roles are those that is assigned by a teacher to facilitate the process of collaborative learning, and include precise instructions on how students should interact with one another in the group setting to promote learning. This is contrasted with emergent roles that "emerge spontaneously or are negotiated spontaneously by group members without interference by the teacher or researcher" (p. 496).

Research on scripting collaborative roles for students demonstrates mixed results. While there is a fair amount of support for the efficacy of providing students with scripts for how to interact with one another while working collaboratively (Fischer, Kollar, Mandl, & Haake, 2007; O'Donnell, 1999). There is also research that indicates that scripts lose their efficacy when students are engaged in collaborative learning over longer periods (Rummel & Spada, 2007); and that other approaches, for example, students observing a model of collaborative learning, are more effective than scripts in enabling successful collaborative learning in student groups (Rummel & Spada, 2009). Moreover, providing students scripts for collaborative learning interactions has been criticized as overly directive, depriving students of the opportunity to engage in the type of thinking that will lead to creativity in problem-solving (Dillenbourg, 2002).

Meanwhile, robotics learning environments are multidimensional problem spaces which afford multiple roles that may be taken up in an emergent fashion, including the role of programmer, builder, and analyst. The multiple tools in this problem space can create a situation where students vie for control of the tools through adopting certain roles (Jones & Issroff, 2005). Such vying for control can create tension in the group and interfere with opportunities to learn (Sullivan & Wilson, 2015). Therefore, it is not only the functions of the role that are important, but the shifting of roles, and the negotiation of roles that may also affect student learning (Sullivan & Wilson, 2015; Wise, Saghafian, & Padmanabhan, 2012).

In collaborative learning settings, work is meant to be done by the entire group and through these social interactions new knowledge is built (Vygotsky, 1978). However, collaborative groups do not always work well together, hence, these learning interactions may not take place (Dillenbourg, 1999). Successful collaborative group work requires ongoing, well-coordinated group interactions (Barron, 2003). Barron has argued that the accomplishment of this coordination occurs through a complex array of cognitive and social tasks, which may be categorized into three areas of collaborative interaction: shared task alignment, joint attention, and mutuality. Collaborative groups may demonstrate varying degrees of these interactions, indicating both higher and lower levels of coordination. In this way, the level of coordination

will have an impact on the group's learning outcomes. Recent research has shown that students who have adopted emergent roles in group work can successfully achieve higher levels of coordination (Sullivan & Wilson, 2015). Given this, and the fact that emergent roles may better support creative thinking (Dillenbourg, 2002) scripted collaborative learning was avoided. In our study, students were asked to collaborate, they were not assigned specific roles, but they had access to the three primary roles associated with the activity: programmer, builder, and analyst. Here, we examine how the specific emergent roles the students inhabit impacts both their collaborative interactions and their engagement in computational thinking.

This research proceeds from the Vygotsky (1978) perspective, which holds that through direct interaction with the robotics tools and engagement in collaborative group discussions, students will develop knowledge and ideas about computation. Here, we aim, through descriptive and qualitative analysis, to characterize the nature of girls' computational thinking as they "do" robotics. This analysis will serve as a starting point for designing effective curricular and pedagogical scaffolds for supporting girls' development of computational thinking abilities and computer science knowledge.

Our focus is on how emergent roles in the multidimensional problem space of robotics relates to collaboration and to different types of computational thinking. For the purposes of this study, we define three working modes: working individually, working cooperatively (defined as working jointly toward a solution through a divide and conquer approach), and working collaboratively (defined as working jointly toward a solution through discussion and dialogue). This analysis is accomplished through systematic observational methods and sequential analysis which allows for the understanding of behavior as it unfolds over time (Bakerman & Quera, 2011). The goal of this research is to improve our knowledge of how the emergent roles enabled in a robotics learning space are taken up and how they relate to collaborative interactions, engagement in computational thinking and learning outcomes as measured by the difficulty of challenges undertaken. This knowledge will assist teachers and curriculum developers in creating robotics learning settings that best support student learning in computer science. Our specific research questions are presented below.

13.1.4 Research Questions

RQ1. What are the role transitions made by novice programmers in this study?
RQ2. How do different roles in a robotics programming environment relate to different types of collaboration?
RQ3. How do different types of collaboration in a robotics programming environment relate to different types of computational thinking?
RQ4. How does engagement in specific types of computational thinking relate to student learning of robotics as measured by the difficulty of challenges undertaken?

13.2 Methods

This observational case study took place at a 1-day, all-girl introduction to robotics event called "Girls Connect." The students spent 2 hours learning how to design and program the robot (Lego® EV3), they then spent the remainder of the day working on solving robotics challenges. The participants in the workshop included 17 girls, ages 8–13 (M = 11.725) who attended five different schools in New England. Purposeful sampling was used to select students from various backgrounds and geographic areas from the pool of students who volunteered for the event. Students were drawn from schools that were struggling academically; four of the five schools who sent student participants were not meeting state standards for student academic performance. All of the participants were working with robotics for the first time. The "Girls Connect" event was designed to introduce girls to the FIRST LEGO League (FLL)® in order to stimulate their interest in the FLL and robotics. The workshop featured the FLLs 2011 challenge: "Food Factor." This challenge includes 11 missions of varying degrees of difficulty. The students were allowed to select the mission(s) they wished to solve. In the morning of the 1-day workshop, the girl participants did team building exercises, learned a little about programming, learned about the food factor challenge, and built a basic robotic car from blueprints. After lunch, the girls devoted themselves to solving the robotics missions that were laid out on the food factor challenge board. In our study, the aggregate of missions attempted across all groups was seven. In other words, collectively the six groups of girls attempted to solve seven of the missions. The students were divided into six teams (five teams of 3 and one team of 2); girls from the same schools were on the same team. Each of the six teams were given color-coded t-shirts to wear for the day. For example, one team wore green t-shirts, one yellow, etc. The t-shirts bore the "Girls Connect" logo and were presented both as commemorative gifts to participating girls and also to function as an aid to the researchers in keeping track of who was on which team as the girls roamed about the room.

The data analyzed in this study are drawn from two of the teams who participated. For purposes of analysis, we selected a team that appeared to demonstrate higher levels of coordination (worked collaboratively), and a team that appeared to demonstrate lower levels of coordination (worked in parallel), based on viewing of the videotapes. The purpose of making this selection is to aid analysis of the relationship of role to collaboration and computational thinking. By selecting a more collaborative team and a less collaborative team, we are better able to delineate the relationship of our constructs of interest. The data consists of the afternoon problem-solving activities and discussions, and includes 3 h of video for each of two teams. Pseudonyms of Anna, Becky, and Cindy were used to identify each of the girls on the light blue team, and Janet, Kaylee, and Lisa on the dark blue team. Consent was obtained from the parents of the participants and assent from the participants themselves.

We collected audio and video data at the 1-day event. Each of the girl participants in the study wore a wireless microphone. Each group of girls had their own worktable, a LEGO Mindstorms EV3 robotics kit, and a laptop computer. Two challenge arenas

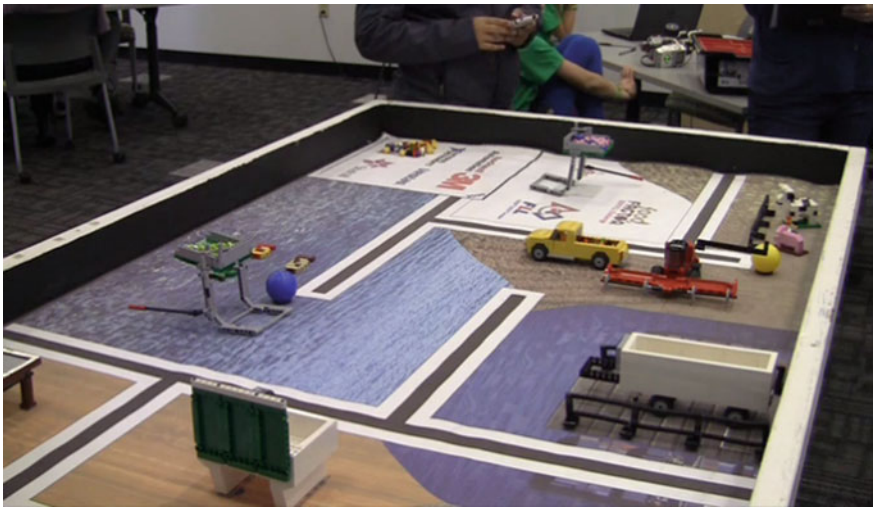

Fig. 13.2 FLL challenge arena

were set up in the room so that the girls could test their solutions (see Fig. 13.2). A stationary video camera was mounted at each group worktable to capture the building and programming of the robots. Two additional cameras were focused on each of the challenge arenas. From these data, we created a video and audio recording of each group's activity and discussion for the day as they moved between their individual worktables and the challenge arenas. See Fig. 13.3 for an illustration of the room and camera setup. A professional transcriptionist transcribed all group talk. We also ran a screen capture program on each groups' laptop. In this way, we collected all of the robotics programming activity engaged in by each group. This data includes the final robotics program(s) created by each group. The data analysis unfolded over four distinct phases including: (1) behavior analysis of the roles students enacted and collaborative interactions observed; (2) discourse analysis of student talk related to computational thinking; (3) descriptive statistics related to the observed roles and collaborations; and (4) learning outcomes analysis based on a challenge difficulty score, role enactment and observed collaboration.

13.2.1 Phase I—Behavior Analysis: Roles and Collaboration

The first phase was to record onset and offset times of certain behaviors as observed on the videos. The role of the student and the type of collaboration were coded as continuous and mutually exclusive. This means that all 3 h of video data were coded (for each student) and no overlapping of the codes occurs. The unit of analysis for this phase of data coding was "shift of focus." In other words, when students

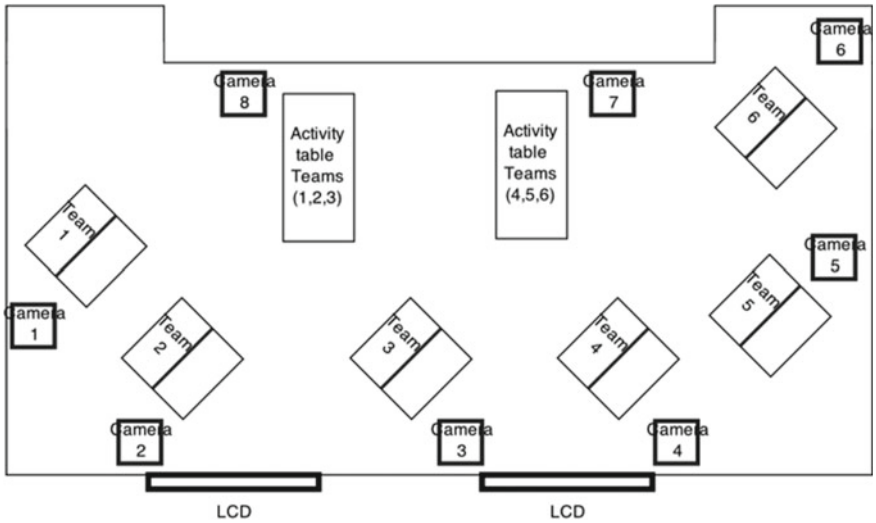

Fig. 13.3 Research video camera setup

switched their attention from one activity to another activity, we initiated a new code (Chi, 1997). The codes for the emergent roles were derived from attending the event and watching the video recording numerous times and included Programmer, Tester, Builder, Analyst, and Other.

The codes for collaboration are based on Forman and Cazden's (1985) codes for participation in groups as markers of coordination, discussed above. They included Parallel (little to no focus on the group), Cooperative (Working together, focused on own results), Collaborative (Working together and sharing ideas), and External (Focused on something outside of the group).

Inter-rater reliability was assessed by training a second coder. A timed-event alignment kappa (Bakerman & Quera, 2011) was used since it allows for tolerances between onset and offset times. Results for inter-rater reliability for the role were $\kappa = 0.83$ and for collaboration were $\kappa = 0.92$. In order to achieve this kappa, the coders reviewed disagreements in their coding and resolved conflicts.

13.2.2 Phase II—Discourse Analysis: Computational Thinking

This phase focused on the discourse related to the robotics activity. First, we segmented the 3-hour conversations into Individual Troubleshooting Cycles (TSC) for each group. As discussed above, the troubleshooting cycle is iterative and consists of building the device, writing a program, testing the program, diagnosing and debug-

Table 13.1 Computational thinking coding system

Major level	Subcategory	Code	Description
Analysis		A	Any planning of a mission solution, including developing ideas or approaches for mission solutions that are specific to the physical space the robot must negotiate, like the initial placement of the robot
Algorithmic thinking	Variable	ATV	When quantity is discussed in determining how far something should turn or move. Often will be thought of as degrees, seconds, or revolutions
	Operation	ATO	When discussing the type of programming block that should be used. Most programs in the data set will only use two blocks, turn or move. Often will be used as turn left, turn right, move forward, move backward
Debugging	Observation	DO	When students describe the movement of the robot during a test, or discuss the outcome of the test in terms of the robots interaction with the pieces on the gaming arena
Designing		D	When working on building additions to the robot. Or any discussion about the design of the robot including physical changes that need to be made

ging the program, and revising the program, or revising the design of the built device (Sullivan, 2011). The transcripts were then coded using a priori codes we developed based on the work of Wing (2006) and Barr and Stephenson (2011). These codes have been synthesized to be relevant for the activities and type of behavior expected and observed for novice programmers in a robotics environment. Table 13.1 presents the coding scheme.

Two undergraduate students were trained in the use of the coding scheme and all utterances from both groups over the course of the challenge period (3 h in the afternoon) were coded. This chapter is concerned with student's computational thinking. Therefore, all comments that were not related to computational thinking were coded as other. Inter-rater reliability was calculated utilizing Krippendorff's alpha (Krippendorff, 2004). Results for inter-rater reliability for the discourse were $\alpha = 0.901$, which indicate that inter-rater reliability for this study was high.

13.2.3 Phase III—Descriptive Statistics: Roles

The first step in phase III was to calculate descriptive statistics to summarize the coded observations of student behavior. The total time and relative duration were

calculated for role and collaboration. Then, once we had segmented, coded, and scored the discourse data, we tabulated the instances of the CT codes in each TSC for each group. As a result, we created one table of data for each group. The rows of the tables were the eight CT codes described above. The columns listed out all the TSCs of the group. In each cell of the tables, we put the number of instances we observed for each CT code at each TSC. Using those tables as input, we conducted one-way ANOVAs to compare the means of the number of instances of the CT codes across all the TSCs. For each group, the null hypothesis we tested was whether the discussions underlying the codes occurred at the same frequency on average across the TSCs. To visually inspect the frequency differences among the groups, we then created boxplots for the instance counts of each CT code for each group. We used those plots to visually assist the comparison of how frequent the discussions were with each group.

The second step in phase III was to calculate the joint probabilities of roles and type of collaboration. A joint probability is the probability that an event will occur given another event. This is also called Lag(0) (Bakerman & Quera, 2011) analysis since the calculation describes the co-occurrence of events in the same time frame. Transitional probabilities, or Lag(1) (Bakerman & Quera, 2011) were also calculated. These show the probability of one event following another event.

13.2.4 Phase IV—Difficulty Score Calculation: Learning Outcomes

The last phase of the analysis included examining the difficulty of the programs that each of the two groups attempted as a proxy for the level of learning each group had achieved. We reasoned that the more confident each of the teams felt over the course of the afternoon, the more likely they would be to select more challenging tasks to accomplish on the game board. We developed a rubric to score these programs. This rubric is based on the difficulty of the mission tasks presented on the FLL challenge arena. There were 11 missions on this particular arena. However, the students in our study only attempted seven of those missions. The missions (also referred to as challenges in this chapter), consisted of moving the robot to different sections of the game board in order to act upon materials placed on the game board. In the Food Factor challenge, students learn about bacteria, overfishing, and other aspects of food safety and food production. The missions relate to these ideas and consist of moving materials, maneuvering around materials, and collecting materials.

The first author on this chapter is a computer science professor and an expert in the field. The first author calculated the difficulty of each of the missions by creating an optimal solution for each task. Due to the fact that students were novices, with no prior programming or robotics experience, the challenge solutions were simple and relied primarily on moving the robot forward and backward and turning the robot. These elements, forward/backward and turning, could be programmed by

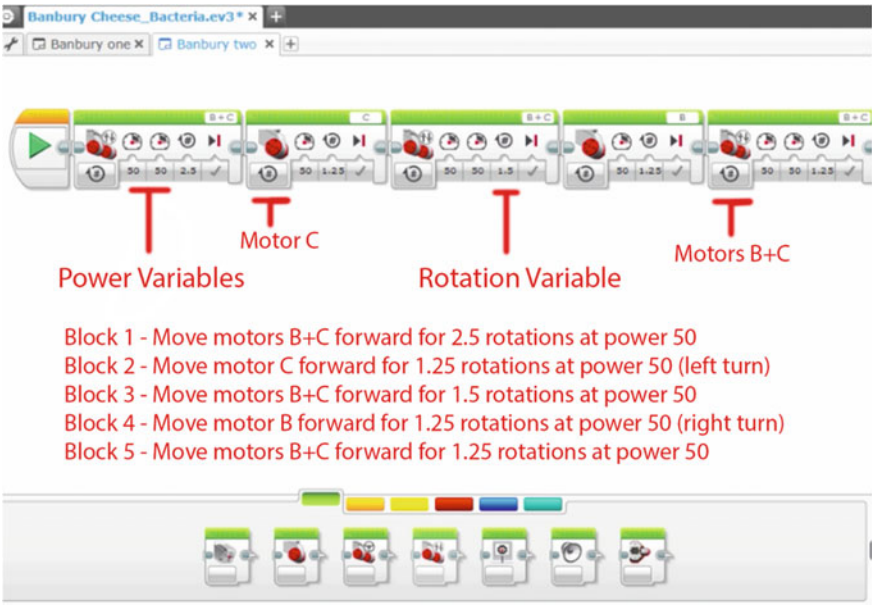

Fig. 13.4 Annotated student program

adjusting power to the motors, and/or setting the number of rotations of the attached wheels on the robotic device. We have annotated an example of a student program, drawn from the data, to highlight the nature of student programs (see Fig. 13.4). The annotations highlight the meaning of the icons in each block, as well as provide an English language translation of the meaning of each block as arranged in the program. After we had developed optimal solutions, we then compared the relative difficulty of each solution based on three variables including: (1) the number of programmed moves forward/backward needed, (2) the number of turns needed, and (3) the distance needed to be crossed on the board to attempt that mission (based on # of wheel rotations). We reasoned that attempting a mission that was far away required more precision in navigating the board to arrive at the mission. Therefore, for the far missions, we doubled the sum of turns + moves to account for the added difficulty presented by the distance. Student programs were then scored based on the difficulty of the mission attempted. Table 13.2 presents the scoring rubric by the mission for the seven missions attempted by participants in the study.

In the final step, we focused on examining the relationship between the frequency of the discussions that are the most relevant to computational thinking and the overall scores for the groups. Those discussions were those that received the codes A, ATO, ATV, D, and DO. For each group, we computed the frequency means of the five codes across all the TSCs. Then, we plotted those means along with the overall program scores in the same scatter plot for each group. To guide the interpretation of the relationship, we used different colors and symbols for each group and each CT code.

Table 13.2 Final program scoring rubric by mission

Mission name	# of turns req. (a)	# of moves req. (b)	Distance traveled (near or far)	Score = (a + b) × 2
Blue Ball	0	2	Near	2
Green Bacteria	0	2	Near	2
Yellow Ball	2	4	Near	6
Harvester	4	6	Near	10
Pink Bacteria	4	6	Near	10
Pizza	2	3	Far	10
Red Bacteria	3	4	Far	14

13.3 Results

13.3.1 Role Transitions

In answer to research question #1—What are the role transitions made by novice programmers in this study?, we present the results of our behavior analysis of transitional probabilities; this gives us an overall view of how the students organized themselves. Transitional probabilities in this analysis are indicated by the arrow (direction) and percentage (probability). The transitional probabilities in this analysis specify the probability of the next role taken up by the student given their current role. For example, in Fig. 13.4, there is a 76% probability that for Anna, the role of succeeding programmer would be tester. The light blue Team (Fig. 13.5) took a divide and conquer approach to solving the programming challenges. This is evidenced by the lack of continuity shown in their paths between roles. Each student has a unique path, indicating variation in student activity. Anna has taken on a primary role of programmer, Becky the role of builder, which left Cindy with no primary individual role, but she did take part in the collaborative role of analyst. The dark blue team (Fig. 13.6) have taken up a more collaborative strategy that is highlighted by the similarity of their role transitions. In other words, in visually examining the role transitions for the dark blue team, there is less variation, indicating that the students were jointly involved in sharing the roles and collaborating.

13.3.2 Collaboration

In answer to research question #2—How do different roles in a robotics programming environment relate to different types of collaboration?, we first determined the amount of time spent in the types of collaboration (parallel or collaboration). No episodes of cooperation were observed. Students were either working together with

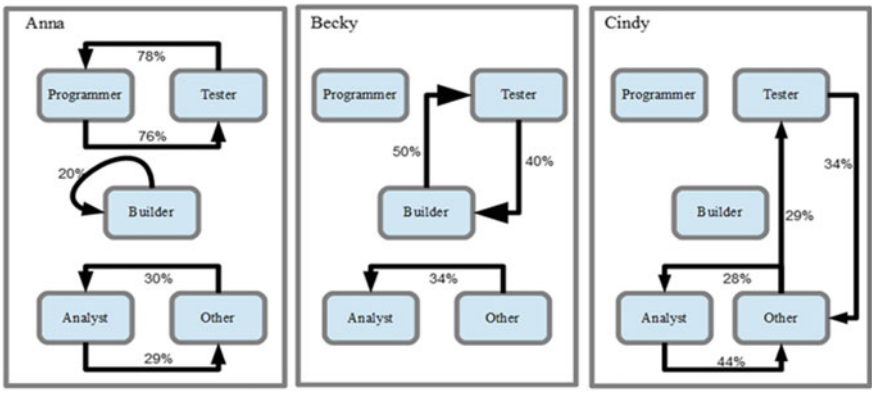

Fig. 13.5 Roles transitions by light blue team

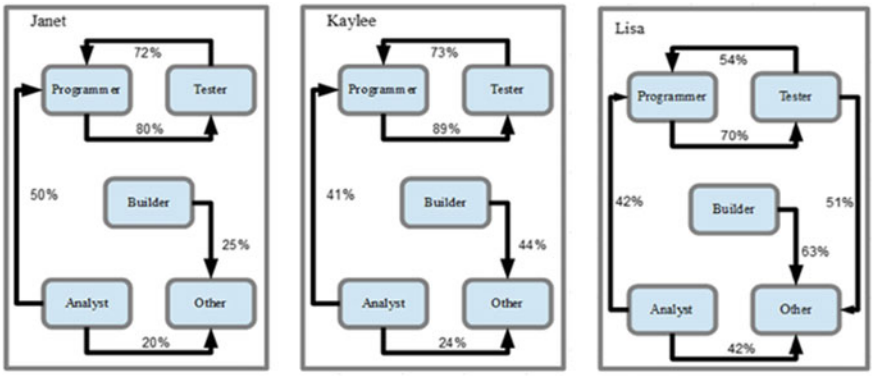

Fig. 13.6 Roles transitions by dark blue team

one set of materials (collaboration) or working alone (individual). We then calculated the duration and relative duration of collaboration for each of the students, presented in Figs. 13.7 and 13.8.

The data indicate that the students on the light blue team spent at least half of the time working collaboratively. However, the students on the dark blue team were jointly attentive at least 80% of the time. The understanding of collaboration is further explored by examining the joint probabilities between the role and the type of collaboration. Notable outcomes from this analysis show that when a team divides up the roles and a member is involved in a primary role, they perform that role in a non-collaborative way. For example, when Anna assumed the role of programmer, there was a 72% chance that she would be working alone and a 27% chance that she would be working collaboratively. This is contrasted to the dark blue team where no one took on a primary role, but each took turns sharing the roles during the day. For example, when Janet assumed the role of programmer, there was an 84% probability

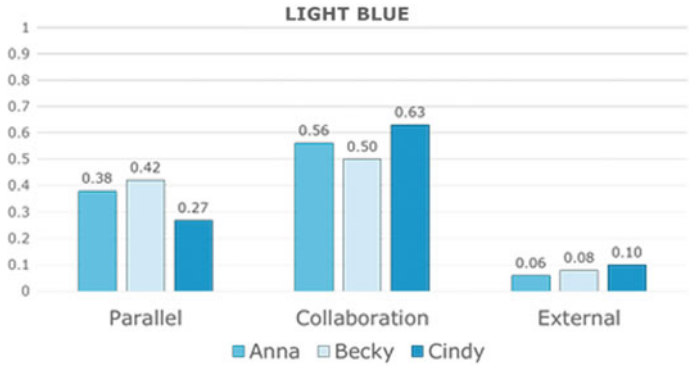

Fig. 13.7 Collaboration light blue

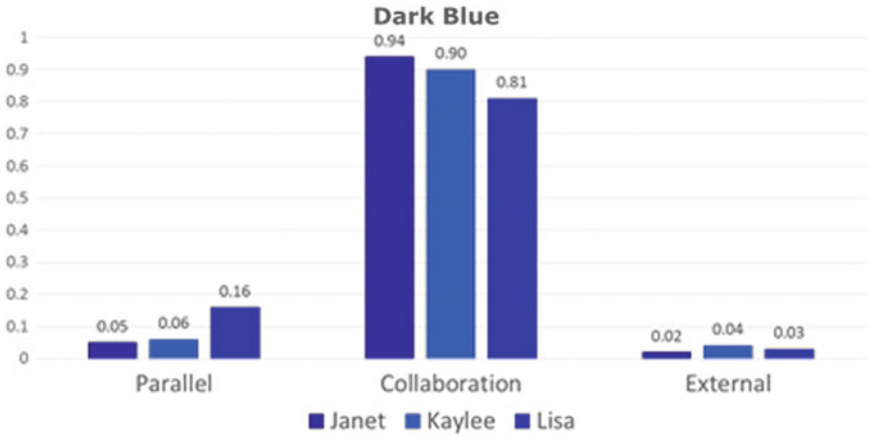

Fig. 13.8 Collaboration dark blue

that she would be working collaboratively. Overall, and for both groups, the two roles of tester and analyst do stand out as having a higher probability of being collaborative for all students.

13.3.3 Computational Thinking

In answer to research question #3—How do different types of collaboration in a robotics programming environment relate to different types of computational thinking?, we first present the frequency and relative frequency of the different computational thinking codes for each student in each group, across the afternoon (Table 13.3).

Table 13.3 Frequency and relative frequency of CT codes per student

	Analysis		Alg. thinking — variable		Alg. thinking— operation		Design		Debugging	
	Freq	Pct (%)	Freq	Pct (%)	Freq	Pct (%)	Freq	Pct (%)	Freq	Pct (%)
Anna	66	34	13	7	58	30	36	18	23	12
Becky	124	48	1	0	20	8	105	40	11	4
Cindy	81	29	5	2	34	12	123	45	34	12
Janet	97	26	133	35	62	16	31	8	53	14
Kaylee	61	19	105	33	70	22	36	11	50	16
Lisa	46	23	60	30	22	11	45	21	32	16

The idea of the variable is a foundational concept in computer programming (Soloway, Ehrlich, Bonar, & Greenspan, 1982). However, the total frequency of discourse concerning the variable is 19 for the light blue team as compared to 298 for the dark blue team. This pattern was set early in the process. When Anna was learning to use the software, she would bench test the robot. Through this process, she was learning how the values of the variables affected the movement of the robot. However, she did this in complete silence. The dark blue team completed the same process but did this collaboratively.

To shed further light on students' computational thinking, we present a short vignette of the dark blue team's discussion as they discuss changes to their algorithm (Table 13.4).

Next, we present boxplots (Fig. 13.9) for each of the two groups that depict the group's CT coded discourse frequency per troubleshooting cycle. The bold line represents the median and the size of the box the variance. The light blue team engaged in 92 complete troubleshooting cycles over the afternoon, the dark blue team completed 78 troubleshooting cycles.

Analysis of the box plots revealed differences across the two groups as regards CT talk. While the light blue team spent a lot of time discussing the physical design of the robot, the dark blue group spent more time discussing the algorithms needed to program the robot, and in particular, variable aspects of the algorithm. Given the nature of the programming environment and the structure of the programs, for computational thinking to be more evident, we would have expected to hear discourse about the values of the variables. The dark blue group discusses the variables more, and the value of the variables more. It may be reasonable to argue that they are learning more about the robotics environment by discussing the variables in the context of the gaming environment.

In answer to research question #4—How does engagement in specific types of computational thinking relate to student learning of robotics as measured by the difficulty of challenges undertaken?, and to further explore the impact of the differences in group CT-based conversations, we developed an analysis of the difficulty of the challenges each group sought to solve over the course of the afternoon. Table 13.5 presents missions attempted per group.

Table 13.4 Dark blue group CT discussion example

Speaker: Utterance	CT Code
J: That distance just needs to be a little longer. Yeah, we got it there.	DO
K: This one a little longer and then we add the thing. Because it goes waaa.	DO
J: We can have it go diagonal and then diagonal and then straighten itself out, but I…	A
K: Okay so the last turn is a little longer right?	ATO
J: Yeah should be like two.	ATV
K: So two point five?…wait I mean not two point five, two, one, one point…	ATV
J: Wait, was it the turn or the distance, oh that's the, oh that's moving it.	ATO
K: Distance was good the turn I think was a little short because…	ATO
J: No I think it, the distance should be longer when it goes.	ATO
K: Yeah, yeah that's right, sorry, so one point five do you think?	ATV
J: Um lets make it two,	ATV
K: Okay	O
J: Lets make it two.	ATV
K: Okay. Right, now…	O

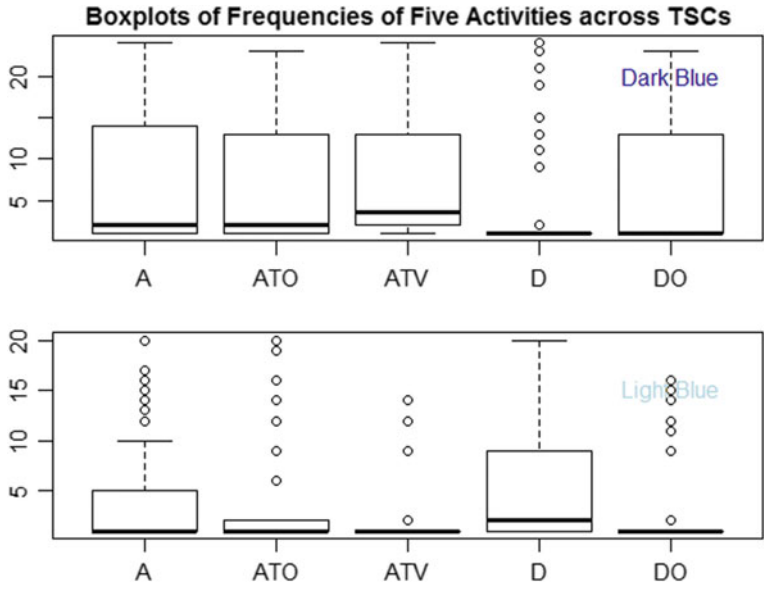

Fig. 13.9 Boxplots of for the two teams

Table 13.5 Missions attempted per group

	Score	Dark blue team	Light blue team
Blue Ball (pollution reversal)	2		
Green Bacteria (disinfect)	2		x
Yellow Ball (poll. reversal)	6		
Harvester (Corn)	10		x
Pink Bacteria (disinfect)	10	x	
Pizza and Ice Cream	10		
Red Bacteria (disinfect)	14	x	

As can be seen from this table, the dark blue team attempted two missions that were rated as more difficult, whereas the light blue team attempted two missions that were relatively simpler. We take this as evidence of the knowledge the dark blue team built, working collaboratively over the course of the day. With a greater level of knowledge shared among the group, the group chose to attempt more challenging missions.

The final analysis consists of a scatter plot, which visualizes the relationship of student scores on their final programs (y-axis) and the amount and type of CT talk each group engaged in during each TSC (x-axis). The scatter plot is presented in Fig. 13.10.

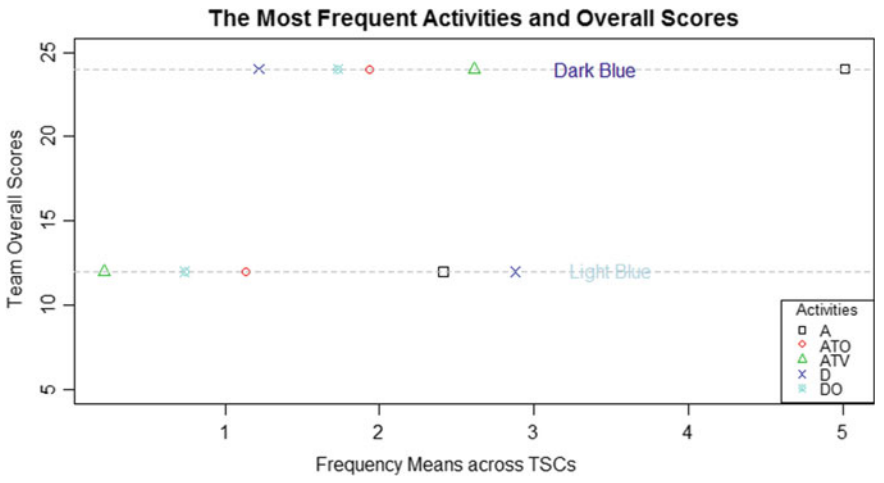

Fig. 13.10 Scatter plots for the most frequent CT codes and program scores

The scatter plot indicates that the dark blue team, who had the highest programming score, also had the highest mean for all of the computational thinking codes, except for the design code. This team spent a lot of time discussing the algorithm they were working to develop. Meanwhile, the light blue team spent a lot of time discussing the design of the robot. The design relates to the physical elements.

13.4 Discussion

This study investigated the relationship of roles to collaboration and computational thinking in the multidimensional problem space of robotics. Our analysis indicates that the emergent roles commonly adopted in robotics learning environments, emerged in this study. Moreover, the roles afforded by the environment: (a) influenced the type of discourse that was used to discuss the activity and (b) affected the common understanding of the different systems in a robotics environment. Indeed, these emergent roles played a part in the level of collaboration that occurred within the group. However, it is not the nature of the roles that mattered, but rather how the roles were negotiated within the groups themselves that influenced collaboration and student learning outcomes.

In the light blue group, the roles were adopted early on and adhered to, roles were not shared among the group members, and therefore, collaboration was hindered. Since Becky and Cindy had little chance to program, they were not able to engage in much algorithmic thinking at either the variable or operational level. For example, together, Becky and Cindy made 54 algorithmic thinking operations comments overall, whereas, Anna, the main programmer, made 58 such comments overall. Meanwhile, in the dark blue group, the three girls shared the main roles, each taking a turn as programmer, builder, tester, and analyst. This led to a common language and a common understanding that allowed for collaboration and, arguably, greater understanding of programming and robotics, as evidenced by the group's selection of more challenging tasks. Importantly, as noted above, this group also worked with the variable aspects of the programming icons. Through engaging with the variables of power and rotations, the group was able to develop more control over the movement of their robotic device and feel confident to attempt the more challenging tasks. It is probable that the joint engagement in programming supported the group's deeper engagement with the activity. Also, as shown in the scatter plot, the dark blue group spent more time discussing abstract elements of the activity (the algorithms), whereas the light blue group spent more time discussing the physical design of the robot, which is, arguably, less abstract. It is likely that the collaborative discussions had in the dark blue group supported the group's computational thinking.

Moreover, we note that the greatest probability for collaboration for both groups occurred when students were enacting the tester and analyst roles. This stands to reason, testing the robot is the activity where all group members can equally participate through observation and discussion. And the role of the analyst was enacted subsequently to testing. In other words, students would test the robot, jointly observe

the execution of the program, and then jointly enact the analyst role in discussing the executed program. Therefore, this result is not surprising, but expected. It is the external and manipulative nature of the robotic activity that supports robust cognitive engagement (Sullivan & Heffernan, 2016).

Collaboration in groups is difficult, but due to the expense of educational robotics, many teachers are forced to create groups. In order to increase the probability that groups will collaborate, scaffolding is necessary (Winne, Hadwin, & Perry, 2013). However, in order for scaffolding to be successful, we must understand the conditions for interactions and the interactions that are indicative of learning (Hoadley, 2010). This research illustrates the importance of scaffolding role rotation in robotics activity. While some may be tempted to interpret these findings as support for a scripted approach, we resist such an interpretation. Rather than providing students with scripts for how to interact, we believe that a simple enforcement of the idea of role rotation could support a group in building enough intersubjective understanding to develop high levels of coordination in the group. Robotics is a very creative activity (Sullivan, 2017). Groups should be supported to share roles, but also given freedom to explore problem solutions in an authentic way. Perhaps it is time to think about a middle ground between completely emergent roles and highly scripted roles. We would advocate for a middle ground in the case of robotics and other highly creative activities. For example, one way to structure this would be to have students intentionally switch roles after a certain amount of time. In this way, the roles would rotate among students and would support collaboration as students would gain knowledge and share it with one another. Another idea would be to introduce a wide-screen multi-touch display, in place of a laptop, for programming. It is possible that such a technology would create more access to the programming tool and allow for greater conversations among students.

Future research should focus on providing varying levels of scaffolded support to students working with robotics. Is it enough to enforce a role rotation within groups, or will students need more support to engage in collaboration? We believe that collaboration within a group is influenced by a multitude of factors. However, we also believe that setting certain conditions for participation may enable higher levels of coordination, for example enforcing role rotation. Future research studies could create conditions that include role rotation and those that support emergent roles. Engagement in collaboration and the development of computational thinking could then be measured in each of the conditions. Moreover, environments where students are developing computational thinking abilities often have a material technological component. Future research should examine how scaffolding student participation by prescribing shared control of the material artifacts affects student discussions and the ability to coordinate the work of the group. For example, as mentioned above, we think that a wide screen multi-touch display may shift collaborative interactions by virtue of allowing students to more closely observe programming activities. In this scenario, students who are not directly manipulating the software could still meaningfully participate through developing a deeper understanding of the programming blocks, thereby improving their ability to reason about the program and recommend possible changes to programs by virtue of close observation. Again, such research

may include conditions (widescreen, multi-touch display, vs. laptop). Such studies will improve our ability to provide robust collaborative robotics learning environments for students.

References

Ali, S. R., Brown, S. D., & Loh, Y. (2017). Project HOPE: Evaluation of health science career education programming for rural latino and european american youth. *The Career Development Quarterly, 65,* 57–71.

Bakerman, R., & Quera, V. (2011). *Sequential analysis and observational methods for the behavioral sciences*. New York: Cambridge University Press.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48–54.

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

Chi, M. T. H. (1997). Quantifying qualitative analysis of verbal data: A practical guide. *Journal of the Learning Sciences, 6*(3), 271–315.

CSTA. (2016). *Computational Thinking.* Retrieved from https://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf.

Dillenbourg, P. (1999). *Collaborative learning: Cognitive and computational approaches*. New York, NY: Elsevier Science.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In P. A. Kirschner (Ed.), *Three worlds of CSCL. Can we support CSCL* (pp. 61–91). Heerlen, NL: Open Universiteit Nederland.

Fischer, F., Kollar, I., Mandl, H., & Haake, J. (Eds.). (2007). *Scripting computer-supported communication of knowledge. Cognitive, computational, and educational perspectives*. New York, NY: Springer.

Forman, E. A., & Cazden, C. B. (1985). Exploring Vygotskian perspectives in education: The cognitive value of peer interaction. In J. V. Wertsch (Ed.), *Culture, communication, and cognition: Vygotskian perspectives* (pp. 323–347). New York: Cambridge University Press.

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(38), 38–43. https://doi.org/10.3102/0013189X12463051.

Hoadley, C. (2010). Roles, design, and the nature of CSCL. *Computers in Human Behavior, 26*(4), 551–555.

Ji, P. Y., Lapan, R., & Tate, K. (2004). Vocational interests and career efficacy expectations in relation to occupational sex-typing beliefs for eighth grade students [Electronic version]. *Journal of Career Development, 31*(2), 143–154.

Jones, A., & Issroff, K. (2005). Learning technologies: Affective and social issues in computer-supported collaborative learning. *Computers & Education, 44*(4), 395–408.

Krippendorff, K. (2004). *Content analysis, and introduction to its methodology* (2nd ed.). Thousand Oaks, CA: Sage Publications.

Lee, I., Martin, F. L., Denner, J., Coulter, R., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), 32–37. https://doi.org/10.1145/1929887.1929902.

National Instruments (2014). *LabVIEW object-oriented programming faq*. Retrieved online at http://www.ni.com/white-paper/3574/3n/.

National Science Foundation (2016). *Building a foundation for CS for all.* https://www.nsf.gov/news/news_summ.jsp?cntn_id=137529.

O'Donnell, A. M. (1999). Structuring dyadic interaction through scripted cooperation. In A. M. O'Donnell & A. King (Eds.), *Cognitive perspectives on peer learning* (pp. 179–196). Mahwah, NJ: Erlbaum.

Rieber, L. (2005). Multimedia learning in games, simulations, and microworlds. In R. E. Mayer (Ed.), *The Cambridge handbook of multimedia learning* (pp. 549–567). New York: Cambridge University Press.

Rummel, N., & Spada, H. (2007). Can people learn computer-mediated collaboration by following a script? In F. Fischer, I. Kollar, H. Mandl, & J. Haake (Eds.), *Scripting computer-supported communication of knowledge. Cognitive, computational, and educational perspectives* (pp. 47–63). New York, NY: Springer.

Rummel, N., & Spada, H. (2009). Learning to collaborate while being scripted or by observing a model. *Computer-Supported Collaborative Learning, 4,* 69–92. https://doi.org/10.1007/s11412-008-9054-4.

Soloway, E., Ehrlich, K., Bonar, J., & Greenspan, J. (1982). What do novices know about programming. *Directions in Human-Computer Interaction,* 87–122.

Stijbos, J. W., & De Laat, M. F. (2010). Developing the role of concept for computer-supported collaborative learning: An explorative synthesis. *Computers in Human Behavior, 26*(4), 495–505.

Sullivan, F. R. (2008). Robotics and science literacy: Thinking skills, science process skills, and systems understanding. *Journal of Research in Science Teaching, 45*(3), 373–394.

Sullivan, F. R. (2011). Serious and playful inquiry: Epistemological aspects of collaborative creativity. *Journal of Educational Technology and Society, 14*(1), 55–65.

Sullivan, F. R. (2017). The creative nature of robotics activity: Design and problem solving. In M. S. Khine (Ed.), *Robotics in STEM education: Redesigning the learning experience* (pp. 213–230). Springer: The Netherlands.

Sullivan, F. R., & Heffernan, J. (2016). Robotic construction kits as computational manipulatives for learning in the STEM disciplines. *Journal of Research in Technology Education, 49*(2), 105–128. https://doi.org/10.1080/15391523.2016.1146563.

Sullivan, F. R., & Wilson, N. C. (2015). Playful talk: Negotiating opportunities to learn in collaborative groups. *Journal of the Learning Sciences, 24*(1), 5–52. https://doi.org/10.1080/10508406.2013.839945.

United States Bureau of Labor Statistics (2017). Employment of women on non-farm payrolls by industry sector, seasonally adjusted. Retrieved from https://www.bls.gov/news.release/empsit.t21.htm.

Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes.* (M. Cole, V. John-Steiner, S. Scribner, & E. Souberman, Eds.) Cambridge, MA: Harvard University Press.

Wing, J. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Winne, P. H., Hadwin, A. F., & Perry, N. E. (2013). Metacognition and computer-supported collaborative learning. In C. Hmelo-Silver, A. O'Donnell, C. Chag, & C. Chinn (Eds.), *International handbook of collaborative learning* (pp. 462–479). New York, NY: Taylor & Francis.

Wise, A. F., Saghafian, M., & Padmanabhan, P. (2012). Towards more precise design guidance: Specifying and testing functions of assigned student roles in online discussions. *Educational Technology Research and Development, 60*(1), 55–82.