5-2019

The Effects of Incorporating Coding on Student Experience and Understanding of Middle School Mathematical Concepts

Ashley Tewes
St. Catherine University

The Effects of Incorporating Coding on Student Experience and Understanding of Middle School Mathematical Concepts

Submitted on May 12, 2019

in fulfillment of final requirements for the MAED degree

Ashley Tewes

Saint Catherine University

St. Paul, Minnesota

Advisor: Heather Benedict Date: 05-13-2019

Abstract

The purpose of this action research project was to study the effects of incorporating coding into a middle school math classroom on student experience with and understanding of mathematical concepts. The project used five data sources to examine these effects. Two of the data sources were used to examine the effects on student experience; 1) a survey to gauge student perception and 2) a chart to measure student engagement by tracking their time on task. Three of the data sources were used to examine the effects on student understanding; 1) a pre-assessment given before the coding project, 2) a post-assessment given after the coding project, these two data sources were used to determine if students grew in their understanding of the math concepts. Finally 3) a rubric was used to assess key learner outcomes, accuracy, application, coding efficiency, presentation, and creativity. After analyzing the data, it was found that incorporating coding into the middle school math classroom could have a positive impact on student math experiences and their understanding of math concepts.

Keywords: coding, math, engagement, Scratch, computer science, programming

Computer Science opportunities in K-12 schools have increased significantly in recent years. Not only is there a large demand for people to work in computing fields, but administrators and teachers are realizing that teaching computer science can foster important 21st Century skills for learners. Some of these skills include problem-solving, critical thinking, computational thinking, creativity, and collaboration.

Currently, computer science, and specifically, computer programming are offered as elective courses in many schools in the United States. Since students typically choose their elective classes, offering computer science as an elective means that not all students are getting this exposure. Instead, it is recommended to give all students programming (coding) experiences within their content area classes (i.e. math). Incorporating coding experiences into math classrooms enhances mathematical thinking and learning in many ways. A few of these include giving students a meaningful math experience, fostering 21st Century Skills, and increasing content knowledge. There are many studies reporting the positive impact coding has on student 21st Century skills. However, using coding as a vehicle to intentionally target and solidify math skills has yet to be determined. Without this kind of information, it is difficult for school districts to carve out time for coding during math class. If research showed that using coding to teach and learn math allows students a deeper understanding of the mathematical concepts, then schools will be more willing to adopt the idea of implementing coding experiences in the math curriculum.

Little is known about the benefits of incorporating computer programming into core content area classrooms, specifically math, to enhance content understanding. Therefore, there is

a need to gather and present information about whether or not using coding in the math classroom has any effect on student content learning. This paper examines how incorporating coding into middle school math classrooms affect student experiences with math and their understanding of mathematical concepts.

Literature Review

Computer Science opportunities in the K-12 setting have increased tremendously in recent years (Moreno-Leon, Robles, & Roman- Gonzalez, 2016). There are many advantages to teaching computer science to children, aside from the high demand for workers in computer programming related jobs and the 21st Century being heavily influenced by computing. (Sáez-López, Román-González, & Vázquez-Cano, 2016). Many students lack problem-solving and computational thinking skills (Kalelioglu & Gulbahar, 2014). Professor Seymour Papert claimed that the Logo programming experience (coding involving graphics) could enhance these intellectual thinking skills (Papert, 1980). Computer science is often an elective in much of the U.S. (Guzdial, 2016); however, many researchers recommend it to give all students these experiences in a cross-curricular type of way, especially in math and science (Batista, 2014; Calder, 2010; Sáez-López et al., 2016). Coding experiences, used in the math classroom, bring many contributions to mathematical thinking and learning (Batista & Baptista, 2014). The following review of literature will focus on one subdomain of computer science, programming otherwise referred to as coding, and how incorporating coding experiences into the math classroom affects student experiences with math and their understanding of mathematical concepts. More specifically, how incorporating coding in the middle school math classroom

makes math more meaningful, increases content knowledge, and nurtures learners' 21st Century skills.

Meaningful Math Experience

Students have positive attitudes and high motivation when visual coding languages are integrated into a core content area classroom (Sáez-López et al., 2016). When coding was initially taught in K-12 classrooms, Resnick et al. (2009) explained that three things made the implementation difficult; syntax errors with text-based coding, introducing coding through activities that were not intriguing to students, and the fact that coding was not taught in union with core classes (Batista & Baptista, 2014). Due to these problems, it has been suggested that visual programming language be used in schools (Sáez-López et al., 2016). Visual coding languages consist of preset blocks that can be dragged together based upon their shapes and desired outcomes. This block-based structure saves students a lot of frustration because they are not discouraged by syntax errors (Sáez-López et al., 2016). The blocks are often organized or labeled by color and can be connected to other blocks similar to the way puzzle pieces connect. This can make learning a visual programming language intriguing to students since they are able to tinker with the blocks, just as they would do with physical building blocks, which allows creative expression to happen faster.

The Massachusetts Institute of Technology (MIT) Media Lab research group Life Long Kindergarten lead by Mitch Resnik, aimed to create a visual programming language, Scratch, with which students could learn coding through tinkering within a meaningful social setting (Batista & Baptista, 2014). Scratch allows students to drag and drop blocks and to create coding projects that interest them. Through trial and error, students are immediately able to see if their

program works or not (Calder, 2010). Scratch allows for student choice and freedom as students are able to create many different types of projects. Students have the power to choose how they express their interests through coding (Resnick et al., 2009 as cited by Moreno-Leon et al., 2016). In a study by Sáez-López et al. (2016) centered on different coding platforms, children spent significant time working in Scratch. This visual programming language is hosted on a social platform that was able to nurture the creative ideas students brought to the classroom (Sáez-López et al., 2016).

Since Scratch is an engaging and intuitive programming language for students, it has proved to be an excellent platform for problem-solving through mathematical concepts (Kalelioglu & Gulbahar, 2014). Aspinall explained that Papert viewed math as a place to explore, make mistakes, and self-correct these mistakes (Aspinall, 2017). Coding allows students to do math in a way where they can see the connection to their own lives (Aspinall, 2017). Students expressed that this type of integration in the classroom motivates them to learn more as they enjoy the active approach it brings to learning (Sáez-López et al., 2016). In Ke's (2014) analysis of a mathematics classroom, students created games in Scratch to enhance their mathematical understanding. Students reflected after the experience that they had more confidence in mathematics and had positive attitudes towards the subject. In conclusion, by implementing a visual programming language, i.e. Scratch Programming, in a cross-curricular way, students find more value, meaning, and enjoyment in learning math.

Increased Content Knowledge

More often than not, visual programming languages like Scratch are used to develop programs and games of choice; however, these languages can also be used to teach specific

learning targets that align with the targets content area teachers are required to teach. When teaching coding in this explicit way, it can enhance math content knowledge and help learners form a more concrete understanding of concepts (Batista & Baptista, 2014). Coding is valuable for all age levels and integrating it into content area classes, specifically in math, helps develop problem-solving skills and logical reasoning (Kalelioglu & Gulbahar, 2014).

There are many ways to focus on intended math outcomes when teaching coding. One way is by having students create a program that performs a certain algorithm. Calder's (2010) study had students create a program that generated simple addition equations. The children playing the game needed to match the solution to the equation to the corresponding number of aliens. In Wherfel, Pearson, Shebab, and Tapia's (2015) study, students created animations for different polygons when learning geometry. Another way teachers can integrate coding into the math classroom is to allow students to play programs that are already created and developed. For example, Scratch has academic activities created by members of the Scratch Community like Triangle Classification, the Circle Theorem, and Prime Factorization (Batista & Baptista, 2014).

Teaching math through a computing lens contributes to the understanding of the mathematics content. A study by Yadav, Stephenson, and Hong (2017) found that when computational thinking experiences were integrated into a specific math classroom, the class had a much higher understanding of mathematics processes than a class that did not get the same experiences. Furthermore, Moreno-Leon et al. (2016) found when working with coding in the classroom that middle school-aged children showed improvement in academic performance, specifically math test scores related to a comparison of numbers, understanding of probability, and mathematical thinking structures. There is a lot of mathematical thinking involved when

creating a Scratch project (Batista & Baptista, 2014). Students use measurements of geometry, length, and time, coordinates, angles, relational thinking with input, repeated procedures, and algorithmic problem solving (Calder, 2010). Since students can physically experiment with math, (i.e., angle size), they form a deeper understanding that would not be possible without using computing (Calder, 2010). Additionally, students receive an object-based representation of different math concepts which helps build connections for them when learning new mathematical concepts (Ke, 2014). By being able to simulate problems visually, students can break down new mathematical language to something that is more intuitive for them (Ke, 2014).

21st Century Learning Skills

When students use coding in the math classroom they are learning essential skills for the 21st Century, such as how to problem solve, think computationally, and collaborate effectively (Resnick et al., 2009). Students are motivated to develop problems that interest them when they learn by doing. Visual programming languages encourage students to find solutions to their projects. (Kalelioglu & Gulbahar, 2014). When students are working to solve the problems they have chosen, they are mastering learning targets while solving real issues (Johnson et al., 2014 as cited by Sáez-López et al., 2016). Students must use critical thinking and problem-solving skills to create a program that works since computer programs tell a computer exactly what to do (Sáez-López et al., 2016). Therefore, this process enhances problem-solving skills (Kalelioglu & Gulbahar, 2014).

Embedding coding experiences into other content areas provides students to think computationally. Using these practices inside the classroom gives students the opportunity to gain understanding of how the world around them works and strategies to solve large problems

(Sáez-López et al., 2016). Computational thinking is a fundamental set of skills that everyone must know to function well in society (Aspinall, 2017). Although computational thinking is generally connected to computer scientists, it is used by all people and in many disciplines (Yadav, Stephenson, & Hong, 2017). These ideas have had a lot of influence in scientific fields but also the artistic and humanities fields as well.Yadav et al. (2017) argued that computational thinking should be a requirement for all K-12 learners.

Collaboration is another 21st Century skill that is nurtured by implementing coding in core content area classrooms. Since computing focuses on creativity, choice, and finding solutions to problems, it is collaborative by nature (Israel et al., 2015). Coding is essentially another language, learning to code gives students another way to communicate their thoughts and ideas to others (Aspinall, 2017). Additionally, certain coding enivronments are created to be collaborative. Through the Scratch Community, Scratch is naturally collaborative,allowing users to share their projects directly on the website. The Scratch community can comment on and critique each other's projects. This type of interaction creates opportunities for effective collaboration (Resnick et al., 2009).

Although implementing coding experiences into content area classrooms is usually viewed as a positive experience, there has been opposition. Graves (1990) argued that math teachers are assigned to teach computer science or computer programming courses and are not informed enough to explain it at the level needed for college readiness. Often this occurs because there are not enough teacher training opportunities for these teachers to be informed about what students currently need to be successful (Graves, 1990). Additionally, Graves (1990) claimed that math departments should be separate from computer science departments because math

teachers should focus on math curricula while computer science teachers should focus on computer science curricula. However, teacher professional development programs are continuing to be improved to include experiences around teaching computer science and computational thinking, unpacking standards, and using technology (Yadav et al., 2017). Another point of opposition comes from teachers finding that there is not enough time in the school day or school year to provide computer science experiences (Israel et al., 2015). If students are not taking a separate computer science class, then it becomes the teacher's job to teach these skills along with the context of their discipline (Israel et al., 2015).

Computer science experiences are becoming widespread in the K-12 setting (Moreno-Leon et al., 2016). The next step is to integrate coding opportunities into other content areas. To address this, the researcher asks; in what ways does incorporating coding into middle school math classrooms affect student experiences with and understanding of mathematical concepts?

Theoretical Framework

The theory of constructionism, developed by Massachusetts Institution of Technology (MIT) professor Seymour Papert, is based on building knowledge structures through student actions. This happens most frequently when students are constructing objects or ideas meaningful to them (Ackermann, 2019). The theory of constructionism focuses on an individual learner expressing their ideas and making their ideas shareable. This model explains how ideas get transformed when students are given the freedom to express their work by making their favorite representations (Ackermann, 2019). Papert viewed students projecting their feelings and expressing their ideas as a key to learning (Ackermann, 2019). This process sharpens and shapes

student ideas and knowledge around a concept. One of the representations Papert focused on was the use of computers and computer-based technology to teach children (Aspinall, 2017). He viewed math classrooms as places where students could explore and discover how math could be useful in their everyday lives through the interaction of math and computing. Papert suggested that when students see connections between math and their own interests, they learn to love math (Aspinall, 2017). Through the process of self-correction (in coding or math), students develop new ways to solve problems and persevere through uncertainty (Aspinall, 2017).

The theory of constructionism suggests that by letting students create artifacts around a concept, such as a computer program, they are able to develop and sharpen their ideas. Furthermore, students learn to love the subject they are studying (Ackermann, 2019). The theory of constructionism guided this action research study. In this study, coding was integrated into a math classroom. The effect this integration had on student understanding of a math concept and their overall math experiences were examined. The theory of constructionism suggests that students should have a deeper understanding of the math concept after participating in the coding experience. Additionally, their math experience should be enhanced after creating computational artifacts.

Methodology

In this study, students created a coding project that calculated distance and midpoint depending on input from a user. For example, if the user typed in the points (1,2) and (-3,5) the program would take those points and calculate the midpoint and distance between them. To answer the research questions, this study utilized a mixed-methods design. The researcher administered a pre-assessment that asked questions related to distance and midpoint, a post

assessment that asked the same questions from the pre-assessment, a rubric used for scoring the projects and a chart to track time on task. The qualitative survey asked a series of questions and students responded using a likert scale. The qualitative data was a student survey describing student attitudes towards using coding in the math classroom.

The population for this action research study was seventh grade students enrolled in a mid-sized middle school in the Midwestern US (n = 107). These seventh grade students were enrolled in Intermediate Algebra for both first and second semesters of the school year. Intermediate Algebra was a required high school math course offered at the middle school level. The sample features 54 females and 53 males and was representative of two-year accelerated math students in the middle school population.

The quantitative data collected before students created their coding project was a pre-assessment that was comprised of three questions on distance and midpoint, each worth one point. At the conclusion of the coding project, the same three questions were given to students as a post-assessment. When the students turned in their coding projects the researcher graded the projects based on a rubric that assessed proficiency towards key learner outcomes accuracy, application, coding efficiency, presentation, and creativity, and on task time in their project. Each of the learner outcomes was scored 0-3 points, for a total of 18 points. Additionally, while students worked on their projects, the researcher tracked the time on task of three students, chosen at random. The researcher observed the three students after a series of 15 minutes. At the end of the each 15 minutes the researcher recorded a plus sign if the student was on task and a minus sign if the student was not working. The plus and minus signs were recorded in a chart where each student had a column broken into 15 minute intervals. The qualitative data measure

collected was a survey the students filled out on Google Forms. The survey was five questions and asked students to reflect on their project experience. This survey was given to gauge student attitudes towards using coding in the math classroom.

Analysis of Data

The student received a point for each of the three questions in both the pre-assessment and post- assessment only if the question was completely correct. The researcher recorded the scores from the pre-assessment and post-assessment in a spreadsheet. Then, the researcher compared the pre-assessment with post-assessment scores and identified if there was an increase in score, no change, or a decrease in score. Next, the researcher used at rubric to score the student Scratch projects. The researcher recorded the score of each student in a spreadsheet and created a graph that identified the breakdown of the scores. In order to analyze the data from the time-on- task chart, the researcher totaled the number of "+" signs within each student column and calculated a percentage that described what percentage of time the student was on task.

To analyze the qualitative data measure, the student survey, the researcher totaled the number of students who strongly agreed, agreed, were neutral, disagreed, or strongly disagreed with the five survey statements. The totals were then converted into percentages and put into a graph and were used to analyze themes related to student perceptions of the implementation of a coding project in the math classroom.

Findings

The purpose of this study was to identify ways that incorporating coding into middle school math classrooms affects student experiences with math and their understanding of mathematical concepts. The research design was comprised of quantitative data sets, a pre- and

post-survey analyzing growth in understanding of a math concept, a project rubric used to analyze mastery of key concepts at the conclusion of the coding project, and a student on task time chart to analyze student engagement. The study also utilized a qualitative measure, a survey that explored student perceptions towards using coding in the math classroom.

Student Math Experiences

The first research question this study addressed was related to identifying prominent feelings and perceptions students had towards using coding in the math classroom. The results of the survey illustrated in Figure 1 show that 37.6% strongly agreed with the statement "I Enjoy Using Scratch in Math" while 33.7% agreed, 22.8% of students remained neutral, 4.0% of students disagreed, and 2.0% of students strongly disagreed. Based on these findings it can be concluded that the majority of students enjoyed using Scratch in the math classroom to demonstrate their understanding of the distance and midpoint formula.

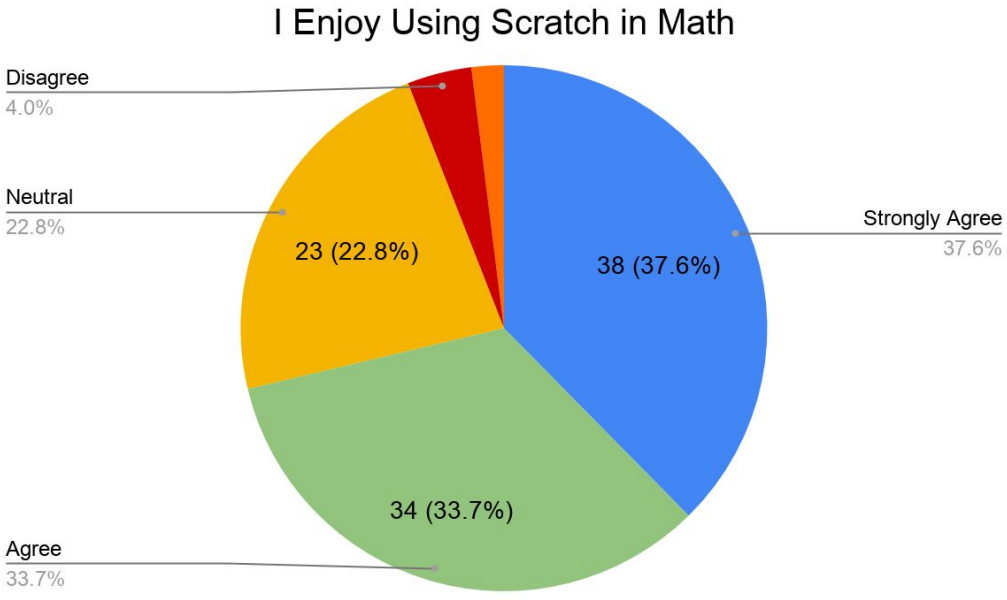

Figure 1. I enjoy using Scratch in math. This figure illustrates the responses to a question in the student survey.

In the same survey, students were asked if they would like to use Scratch again during the school year (Figure 2). The results of the survey illustrated in Figure 2 show that 44.6% strongly agreed with the statement "I would like to use Scratch again in another unit to demonstrate my understanding of a topic" while 22.8% agreed, 25.7% remained neutral, 5.0% disagreed, and 2.0% strongly disagreed. Based on these findings it can be concluded that most students would like to use Scratch again during the school year.

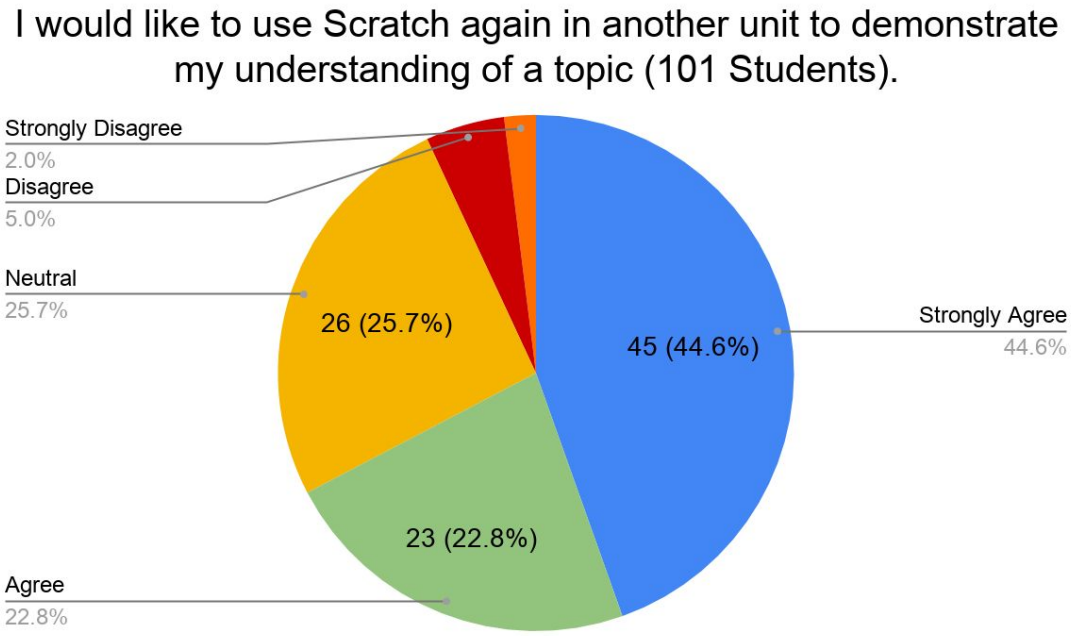

Figure 2. I would like to use Scratch again. This figure illustrates the responses to a question in the student survey.

Additionally, students were asked if the topic of distance and midpoint became more meaningful to them after creating their coding project (Figure 3). The results of the survey illustrated in Figure 3 show that 16.8% of students strongly agreed with that statement "The distance formula and/or midpoint formula feels more meaningful after creating this project"

while 42.6% agreed, 33.7% remained neutral, and 6.9% of students disagreed. Based on these findings it can be concluded that most students felt the concepts of distance and/or midpoint either became more meaningful or remained the same to them.

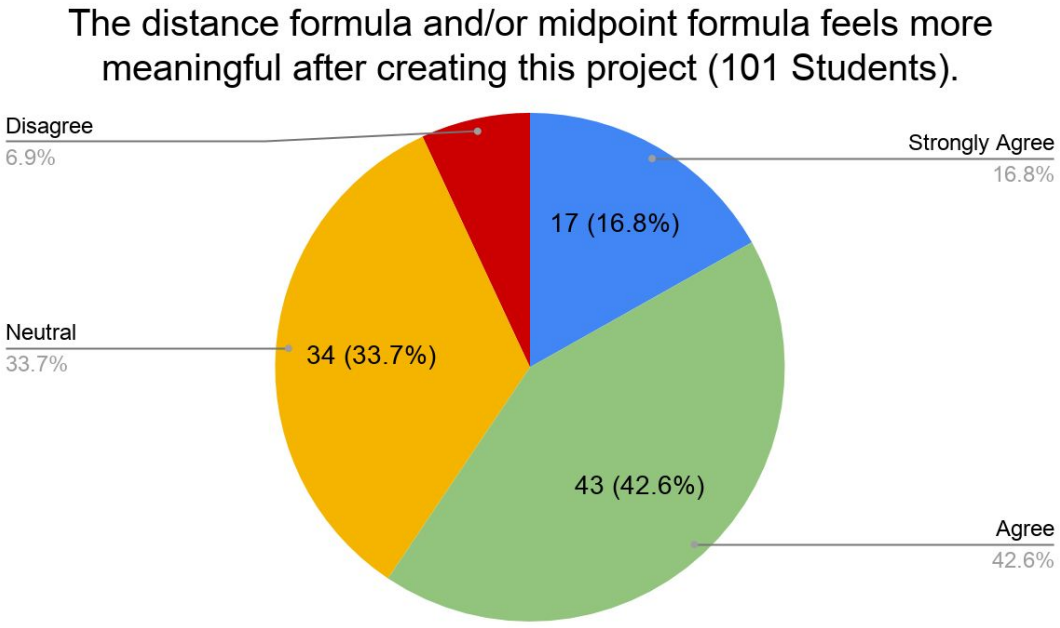

Figure 3. The topic became more meaningful. This figure illustrates the responses to a question in the student survey.

Another question students were asked within this survey was whether they collaborated during the creation of their project. The results of the survey illustrated in Figure 4 show that 79.2% of students collaborated during the creation of their project while 20.8% of students chose not to collaborate with others during the creation of their project. Some students chose to work alone and others chose to listen to music while creating their project.

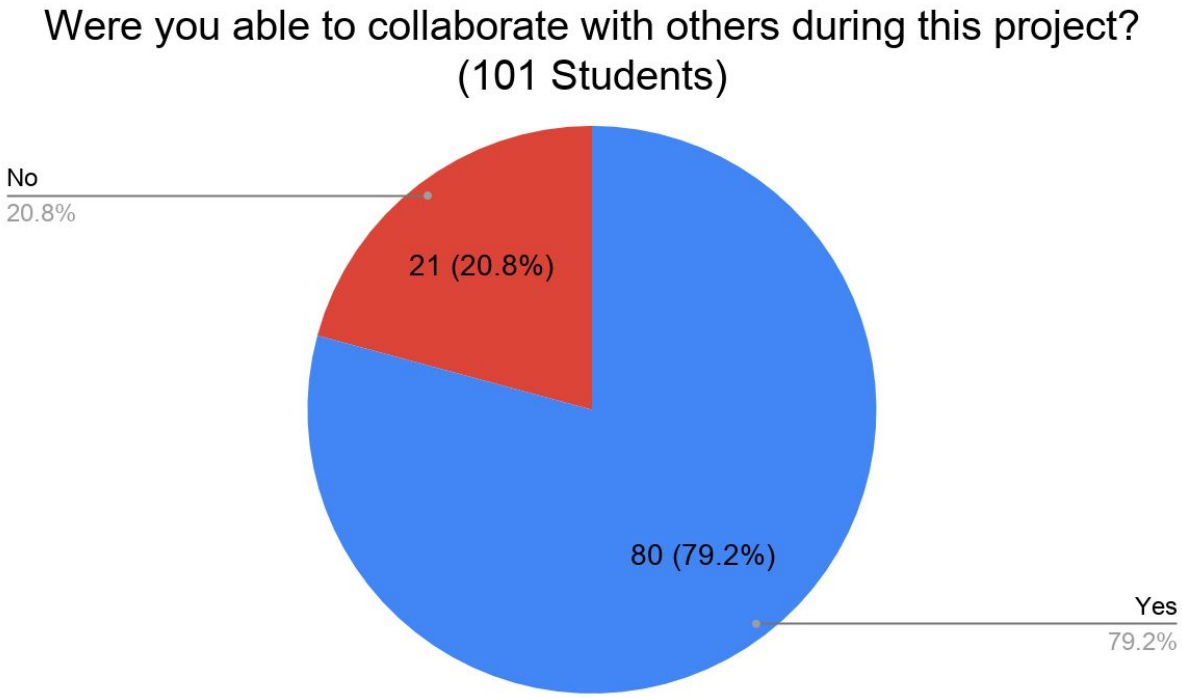

Figure 4. Student collaboration during the project. This figure illustrates the responses to a question in the student survey.

Another measure of student experience came from the time-on-task chart. The three students who were randomly selected for the study showed a high percentage of on task time when working on their projects. The on task time chart showed that two of the three selected students were on task for the entire 60 minutes of work time. The third student struggled to get started, but once the researcher helped the student start their program they were on task the rest of the project work time. This led to the student being on task 75% of the project work time.

Student Understanding of Mathematical Concepts

The second research question this study addressed was whether using coding in the math classroom affected student understanding of specific math concepts. To answer this question, the

researcher gave a three-question pre-assessment on the topics midpoint and distance before the coding experience and concluded the project with the same post-assessment. Findings from analysis of these data points are explained below.

There were 107 students who took the pre and post-assessment on midpoint and distance. When the results from the pre to the post-assessment were analyzed, 39% of students grew in their understanding of midpoint and distance, as their score from the pre to the post-assessment increased (Figure 5). Of the 107 students, 48% of students scored the same score from their pre to their pot-assessment. Very few students' scores decreased when comparing their scores from their pre to their post-assessment (13%).

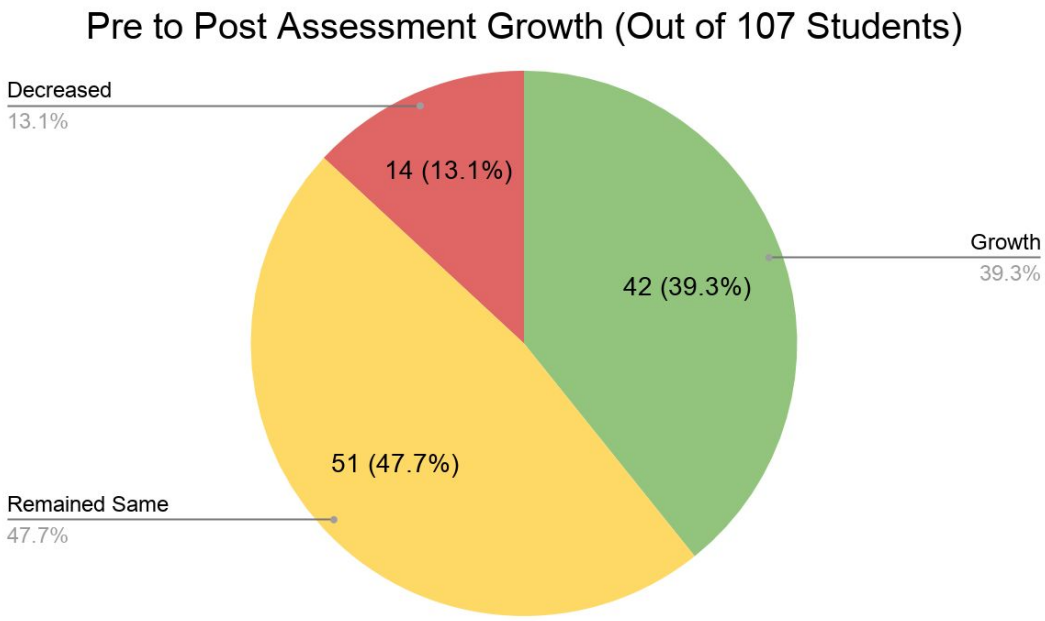

Figure 5. All student growth from the pre to post-assessment.

On the initial pre-assessment of midpoint and distance, 40 students of the 107 showed mastery in their understanding of midpoint and distance as they scored 3/3 on the pre-assessment. Therefore, the researcher analyzed the scores of the students who did not score

100% on the pre-assessment (n=67). Figure 6 demonstrates that of these students who did not score 100% on their pre-assessment, 62.7% increased in score from their pre to their post-assessment. A small number of students received the same score or decreased in score from the pre to the post-assessment (37.3%).

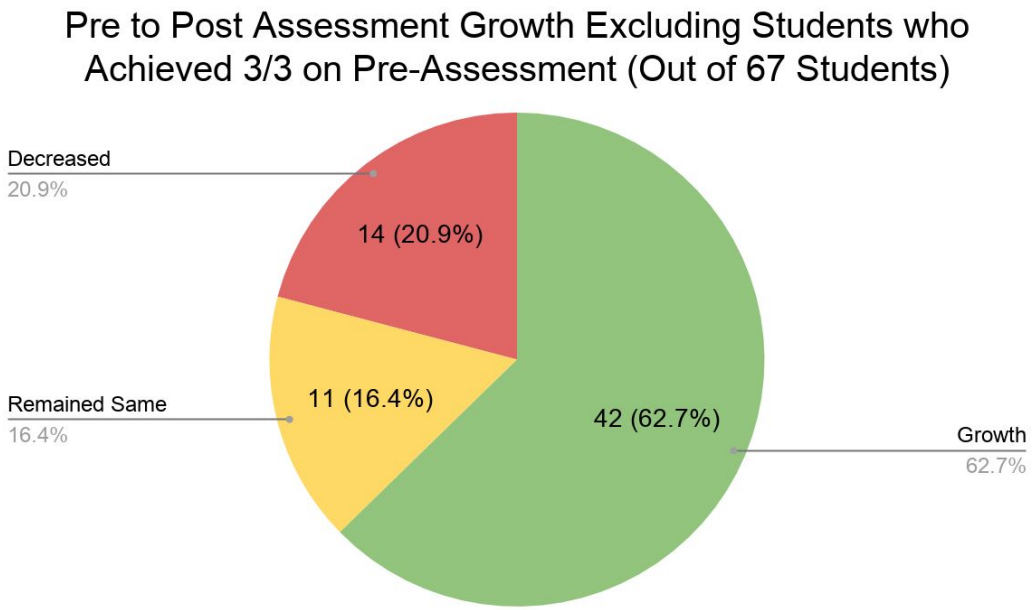

Figure 6. Specific student growth from the pre to post-assessment.

Additionally, the researcher used a project rubric to assess the coding projects and identify student mastery over key math and coding topics. Figure 7 shows the results of the project rubric that was used to assess the coding project and identify student mastery over key math and coding topics. Almost all students (95.3%) showed mastery over the assessed criteria (scored 18/18 on their coding project). One student scored a 16/18 on the project as his program did not run correctly. The five students who scored a zero did not turn it in their project.

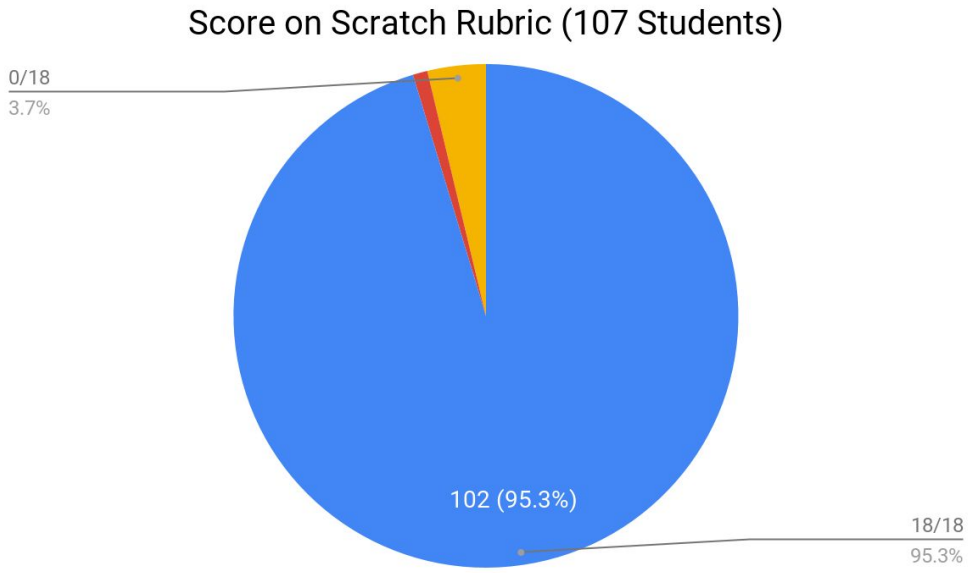

Figure 7. Score on Scratch project using a rubric.

Through triangulating the different data sources many themes were found. Overall, students enjoyed their experiences using code in the math classroom. Many shared that the math concepts of distance and midpoint became more meaningful to them when coding was integrated into their learning experience. Furthermore, the coding project allowed students to either increase in their understanding of distance and midpoint or remain the same after completing a coding project. Aside from the main math outcomes of understanding distance and midpoint, students were also able to demonstrate mastery over key learner outcomes such as, accuracy, application, coding efficiency, presentation, and creativity. Incorporating coding could be a way to keep students engaged in the math classroom, as students were on task the majority of the project work time.

Action Plan

The purpose of this study was to understand ways in which incorporating coding into the middle school math classroom affected student experiences with math and their understanding of mathematical concepts. The research question posed was: In what ways does incorporating coding into middle school math classrooms affect students' experiences with math and their understanding of mathematical concepts? Given the analysis of the data in this study, several conclusions can be drawn supporting the use of a visual programming language in the math classroom.

This study showed the incorporation of Scratch Programming in the math classroom led to math becoming more meaningful to students. Overall, the students enjoyed the experience and shared they would like to use Scratch again to demonstrate their understanding of math concepts. The use of code within their math classroom gave more meaning to their unit of study. Another conclusion that can be drawn is incorporating code into the math classroom can foster important 21st Century skills, such as collaboration and problem solving. Another conclusion that can be drawn is that incorporating code into the math classroom leads to high engagement. In the on-task-time survey, the students selected had high percentages of on-task-time.

When using code in the math classroom as a vehicle to help teach a specific math skill, student understanding of the math concept improves. In this study, the scores from student pre to post assessments either grew or remained the same. This demonstrated that overall, student understanding of midpoint and distance grew from completing a coding project within the math classroom. Very few students decreased in their score from the pre to the post-assessment. Furthermore, by incorporating code in the math classroom students will get practice around 21st

Century learner outcomes such as accuracy, application, coding efficiency, and creativity. Therefore, allowing students to use coding in the classroom to demonstrate their current knowledge of a skill or continue to grow in their understanding of a math concept leads to academic growth.

Based on this study, the researcher plans to incorporate code into the math classroom multiple times each school year. It is recommended that all math teachers aim to implement code into their math classroom to improve student experience and understanding of mathematical concepts. As long as the implementation of the coding project is aligned and centered around a key math concept, teachers are fostering crucial 21st Century Skills in students, all while improving understanding of key math concepts.

References

Ackermann, E. (2019). *Piaget's Constructivism, Papert's Constructionism: What's the difference?* Learning.media.mit.edu. Available at: http://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf [Accessed 16 Feb. 2019].

Aspinall, B. (2017). Code breaker: increase creativity, remix assessment, and develop a class of coder ninjas! San Diego, CA: Dave Burgess Consulting, Inc.

Batista, S., & Baptista, C. (2014). Learning object for linear systems: Scratch in mathematics.*International Journal on New Trends in Education and Their Implications, 5*(1), 71-81.

Calder, N. (2010). Using Scratch: An integrated problem solving approach to mathematical thinking. *Australian Primary Mathematics Classroom, 15*(4), 9-14.

Graves, D. (1990). Computer science and math: Estranged partners in a changing world. *Education, 110*(4), 486-489.

Guzdial, M. (2016). Bringing computer science to U.S. schools, state by state. *Communications of the ACM, 59*(5), 24-25.

Israel, M., Wherfel, Q., Pearson, J., Shehab, S., & Tapia, T. (2015). Empowering K-12 students with disabilities to learn computational thinking and computer programming. *Teaching Exceptional Children, 48*(1), 45-53.

Johnson, L., Adams Becker, S., Estrada, V., & Freeman, A. (2014). NMC Horizon Report: 2014 K-12 edition. *Austin, Texas: The New Media Consortium*. Retrieved from http://www.nmc.org/pdf/2014-nmc-horizon-report-he-EN.pdf.

Kalelioglu, F., & Gulbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33-50.

Ke, F. (2014) . An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing.*Computers and Education, 73*, 26-39

Lye, S., & Koh, J. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, *41*, 51-61.

Moreno-Leon, J., Robles, G., & Roman-Gonzalez, M. (2016). Code to learn: Where does it belong in the K-12 curriculum?. *Journal of Information Technology Education:Research, 15*, 283-303.

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

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., & Silverman, B. (2009). Scratch: programming for all. *Communications of the ACM, 52*(11), 60-67.

Sáez-López, JM., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education, 97(C)*, 129-141.

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM, 60*(4), 55-62.