ISBN: 978-81-941567-9-6
January 3-6, 2020
epiEME 8
Germia and Panorkou
pp. 121-129

EXPLORING ANGLES IN A PROGRAMMING ENVIRONMENT

Erell Germia and Nicole Panorkou
Montclair State University
germiae1@montclair.edu, panorkoun@montclair.edu

In this paper, we describe the results from a whole-class design experiment in a sixth-grade classroom where students explored angles through activity in Scratch programming. The retrospective data analysis shows that through this programming activity students were able to form advanced generalizations about angles that are not accessible with the static representations of angles on paper. These findings illustrate the power of dynamic programming environments for transforming students' reasoning about angle measurement.

INTRODUCTION

The understanding of angles is foundational for working with other geometric concepts such as polygons, symmetry, transformations, and for developing arguments in geometric proofs. Even though angles are essential in understanding many aspects of mathematics, students continue to struggle in understanding this concept. Previous research has found that young students develop a variety of misconceptions with regards to angles. Examples include students reasoning that an angle measure depends on the length of its sides (Smith, King, & Hoyte, 2014), or that an angle only goes counter-clockwise (Mitchelmore, 1998), or that the word 'angle' evokes a 'right angle' prototype (Devichi & Munier, 2013).

Smith et al. (2014) argued that students develop these misconceptions because angles are introduced using static representations on paper. Students are often expected to reason about the figural aspects of geometric objects when they are asked to work with the appearance of a static drawing (Hollebrands, 2003). Mitchelmore (1998) differentiated between *dynamic angles* as illustrating the motion of opening or rotation from *static angles* as the result of that motion. Research has shown that when students work with dynamic angles, for example, by modelling angles using physical body rotations (Smith et al., 2014), they can abandon their misconceptions about angles (Devichi & Munier, 2013). In addition to physical motion, students can experience dynamic angles through dynamic geometry environments (DGEs). Hardison (2018) found that when angles are presented in a DGE (e.g., Geometer's Sketchpad [GSP]) they can reason about angles as an amount of rotation. In learning geometry, it is important for students to work with figures rather with a drawing (Parzysz, 1988). When students work with digital environments, they can control or manipulate mathematical objects and identify invariant features and mathematical relationships (Hollebrands, 2003). We believe that this is the kind of reasoning that would help develop students' figural understanding of angles.

Research on technological tools shows that students adapt their understanding of a geometric object consis-

tent with the functionalities afforded by the computer environments (Hollebrands, 2003). Studies using GSP treat angle as a property of geometric shapes (e.g., Hollebrands, 2003) or as a concept to be quantified (e.g., Hardison, 2018). For example, in GSP, a student can create a parallelogram by constructing two pairs of congruent parallel sides and verify this parallelism using angle measure. On the other hand, research on early programming environments, such as Logo programming (Papert, 1980), provided evidence that the programming environment helped students to pay attention on the direction of a turn and its measurement in degrees (e.g., Clements & Battista, 1989). For instance, students constructed a parallelogram using commands such as FORWARD motion and RIGHT turn (Clements & Battista, 1989) and reasoned about angles as the amount of turn. Additionally, the role of the Logo environment was significant for students to understand turns as conceptual objects involving iterations and directionality and construct dynamic rotations imagery (Clements, Battista, Sarama, & Swaminathan, 1996). Although exploring angles in a programming environment was found to be beneficial for students to actively construct a dynamic conception of angles, the Logo programming language was perceived as "too difficult to impact mathematics learning" (Hoyles & Noss, 1992) and discouraged further research on using Logo.

Scratch (www.scratch.mit.edu), a recent development on programming environments built based on the constructionist perspective of Logo (Papert, 1980), allows students to program interactive projects using a drag-and-drop, and snapping blocks system that encourages young students to program even without any prior programming experience (Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010). Previous research on exploring students' activity in Scratch programming tasks showed that Scratch makes learning engaging and meaningful to students (Maloney, et. al., 2010) and can support students' mathematical reasoning (Benton, Hoyles, Kalas, & Noss, 2016). For instance, Calao, Moreno-León, Correa, & Robles, 2015) found that students who engaged with Scratch activities in their mathematics class have increased their understanding of mathematical concepts and processes. Considering the above, Scratch programming opens up a new opportunity for studying students' angle reasoning. Although some of these studies incorporated angles in their task design, angles are used as input values for commands on the amount of turn (e.g., Benton et al., 2016), but not focusing on students' reasoning. Consequently, our goal was to examine students' reasoning about angles as in a Scratch programming task and that this task is relevant to students' daily experiences. More specifically, we explored the following research question: How do students reason about angles as a result of their engagement with the programming task on Scratch?

METHODS

In this paper, we present a whole class design experiment (Cobb, Confrey, DiSessa, Lehrer, & Schauble, 2003) with sixth-grade students working on a Scratch task. We used a design experiment methodology to engineer particular forms of reasoning within the context of angles and investigate how these forms of reasoning developed through students' engagement with our designed task on Scratch.

DESIGN AND CONJECTURES

The students involved in this study did not have any prior experience with Scratch. Therefore, we introduced

them to some basic elements of the Scratch interface (Figure 1). First, students learned about the *Sprites*. The sprites are the programmable objects that perform the actions on the *Stage*. For instance, we used an image of an artificial satellite as one of the sprites to be programmed using the *Blocks*. These blocks, organized in the *Blocks Palette*, contain programming syntax and are shaped into puzzle-pieces that can be dragged to the Scripts Area. Also, Scratch blocks can be vertically snapped together to form a script. Part of the task is for the students to observe how the arrangements of blocks matter to the intended output. Scratch executes the codes following the order of blocks snapped vertically or wrapped by other blocks (e.g., Forever, Repeat) that repeat the commands inside the loop. We explain the difference between these two arrangements in the following paragraphs.

Figure 1. The Scratch interface

We designed the task "Satellite orbits the earth" (https://scratch.mit.edu/projects/196121316/) after the students have completed a module on Orbit in their science class, aiming to relate the programming task to the content they have been exploring in science. In the task, the students were asked to make the artificial satellite sprite move around another sprite, the earth, in orbit. For the satellite sprite to orbit the earth sprite, students need to create a script that moves the satellite for a short distance (e.g., 15 steps) and turns it for one degree using 360 repetitions, defining in that way a circle as a polygon with 360 sides. Our goal was to provide opportunities for students to see the purpose and utility of mathematics (Ainley, Pratt, & Hansen, 2006), specifically of angle measurement, to complete the programming task.

We asked the students to use our pre-selected blocks in Scratch to direct their attention on a particular mathematical idea embedded in the blocks (see Scripts Area in Figure 1). Similar to Logo's FORWARD or BACK (steps) and RIGHT or LEFT (turn) commands (Clements & Battista, 1989), the Scratch blocks also offer students to focus on the translation and direction of turn. Specifically, the syntax of the "move_steps" block moves the sprite a specific value for a translation (positive or negative) while the "turn_degrees" block changes the direction on which the sprite is facing. Scratch executes the value in "turn_degrees" block using the external angle. Hence, a combination of these two blocks can make the sprite move and turn. For example,

the script "move 50 steps, turn 90 degrees, move 50 steps, turn 90 degrees, move 50 steps, turn 90 degrees, move 50 steps, turn 90 degrees" will create a square movement. The "Repeat _" block can be used to make the same square using a shorter script (Repeat 4[move 50, turn 90]). The "Forever" block infinitely repeats the script within the loop until the stop sign is clicked. The block "when (flag) clicked" starts the simulation, while "pen down" and "clear" tracks the path of the sprite and erases it, respectively.

Analysis

At the end of the experiment, we conducted a retrospective analysis to identify episodes when students focused on a particular mathematical idea or discourse (Cobb et al., 2001) in the context of angles. Then, we reanalysed these episodes to identify potentially reproducible patterns (Cobb et al., 2003) on angle reasoning as a result of students' interaction with the task and their social sharing process in a programming activity (Papert, 1980). Specifically, our analysis was guided by themes related to turns, such as the iteration and directionality, and the role of the computer environment for developing turn concepts and dynamic rotations (Clements et al., 1996). In this paper, we present episodes of one pair of students, Paul and Laura, working on the task aiming to provide an example of the forms of reasoning that students' exhibited.

RESULTS

We describe some forms of reasoning about angles that students exhibited as they interacted with our task. We discuss those by giving examples from four episodes from our conversations with some students.

Episode 1: Exploring angles as turns

Students first explored the pre-selected blocks found in the Scripts Area, tried different values on the blocks and observed the change in the output. When students used only the "turn_degrees" block, they noticed that the satellite sprite was only turning in place. When they tried only the "move_steps" block, they observed that the sprite is only moving on a straight line. Therefore, they decided to combine the two blocks to observe a different output. For instance, Paul discussed his initial attempt on moving the satellite around the earth sprite:

Paul: I made it rotate!

Researcher: How did you do that?

Paul: I put it at 15 degrees that way and I put the rotation, say left to right. And then, I turned it another 15 degrees with 20 steps [Figure 2a].

(a) (b)

Figure 2. Paul's script (a) before the Repeat block, (b) after the Repeat or Forever blocks

Similar to Paul, students first observed the change in the sprite when the "move_steps" and "turn_degrees" blocks are clicked individually or snapped together (Figure 2a). Connecting the "turn_degrees" and "move_steps" blocks creates a syntax using the mathematical concepts of translation and angles to move and turn the sprite. By exploring different ways to rotate the satellite and identifying the constraints of their script (snapping the two blocks together only moves the sprite once), they started building more complex scripts. For instance, Paul identified that snapping the blocks together within the Repeat or Forever block would result in the iteration of the code (Figure 2b).

Researcher: How does it work?

Paul: When I covered it ["move_steps" and "turn_degrees" blocks] with all these [Repeat and Forever blocks], it made it [sprite] repeat each one several, several times.

Researcher: How many times?

Paul: Like ten, all of these ["move_steps" and "turn_degrees" blocks]. And then it [Forever block] made it [satellite] go on and on in this one [the blocks inside the Repeat block]. So, the special, it [Forever block] makes it go forever, these ten steps. So, when I got to this, it just kept going out.

Paul built a more complex script by experimenting with the control blocks while identifying the need (Papert & Harel, 1991) to iterate the turns and create a motion of a continuous dynamic rotation (Clements et al., 1996).

Episode 2: Expressing angle relationships

In multiple instances during the design experiment, students were asked to describe what they have learned from working with the task and articulate the reasons behind their approach. In the following excerpt, Paul was trying to make the satellite to rotate by experimenting with the "turn_degrees" and "move_steps" tools.

Paul: It would turn that way.

Researcher: Why is it turning that way?

Paul: It's turning this way. And it's slowly orbiting.

Researcher: Why is it rotating like that? What do you think?

Paul: Because the degrees are too small.

Similar to Paul, students were able to generalize that the smaller the degree angle in the "turn_degrees" block (if the value in the "move_steps" block stays the same), the more time it will take for the satellite to make a full turn (Figure 3).

Another mathematical relationship that the students noticed was that the smaller the value in the "move_steps" block (if the value on the "turn_degrees" block stays the same), the smaller the track the sprite creates and the faster it will make a full turn (Figure 4).

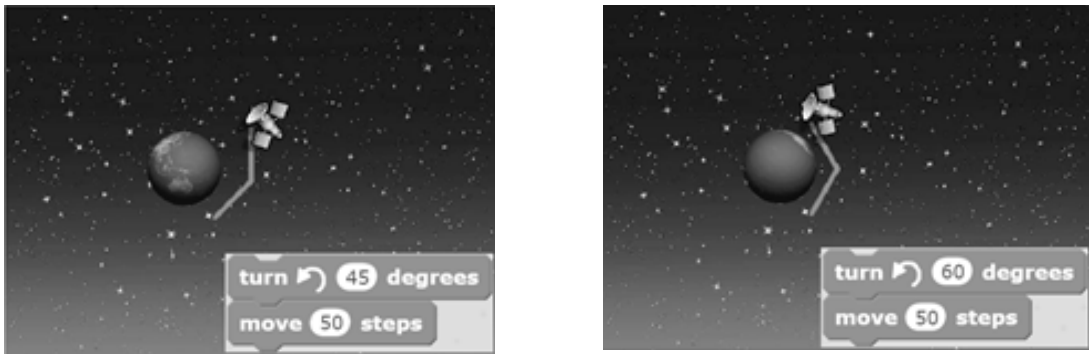


Figure 3. The same value for translation but different degrees of rotation

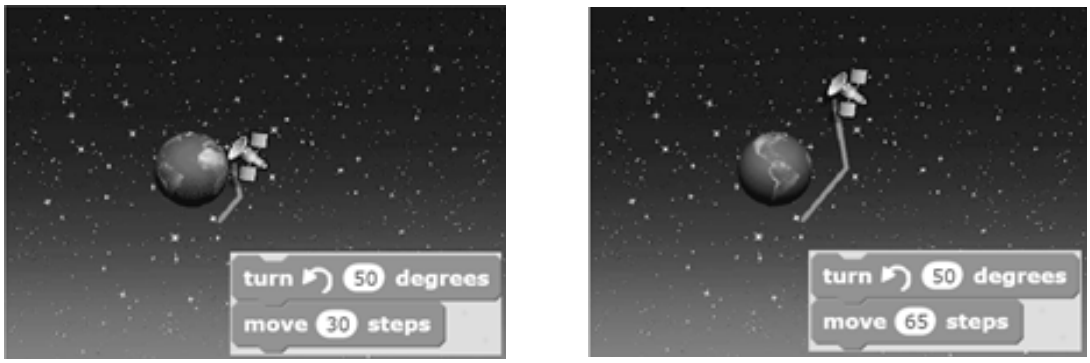
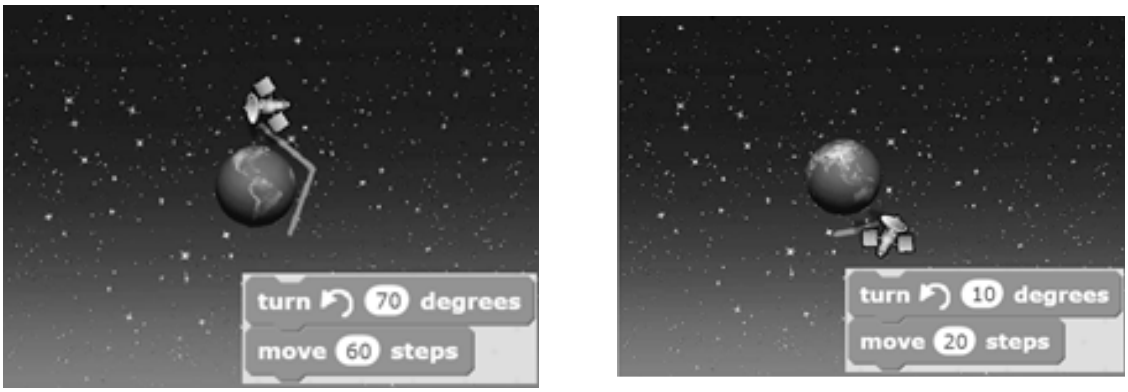


Figure 4. The same degrees but with different steps

As students discovered that changing the values in the "move_step" block can affect the sprite's turn, they explored different values for the "turn_degrees" block. These explorations helped students complete the task and make the satellite orbit the earth. By experimenting with different values for steps and degrees, they were able to generalize that a small number of steps and degrees will create a *smooth* circle-like movement around the earth compared to a large number of steps and degrees (Figure 5).



(a) (b)

Figure 5. (a) a large number of steps and degrees **(b)** a small number of steps and degrees

When students tried different degrees turn, they avoided the notion of a right angle prototype (Devichi & Munier, 2013). Also, when they tried different combinations of values for the degrees and steps, students exhibited that they do not consider angles as dependent on side length (Smith et al., 2014).

Episode 3: Constructing codes as formulas

Throughout the design experiment, students exchanged their ideas by collaborating and sharing what they have noticed while working on the task. We encouraged the students to interact with another student while developing their codes. The following conversation between Paul and Laura shows how students compared their work. Although both students successfully moved the satellite to orbit around the earth, their scripts were different.

Paul: Look, look. This is the formula!

Laura: I did it! [Raising right arm].

Paul: [Checking Laura's screen]. Look at mine. Mine [output] is not as fast as yours. But mine does the same job. It gets around the earth.

The excerpt above shows that Paul considered the codes he constructed as the formula to solve the task. Also, Paul compared his codes to Laura's and identified the similarities and differences between their constructed codes (Figure 6). For instance, Paul used a counter-clockwise "turn_degrees" block while Laura used a clockwise block exhibiting the bi-directionality of turns (Clements et al., 1996). By comparing their solutions, students realised that angles can turn both clockwise and counter-clockwise, avoiding the misconception that angles only go counter-clockwise (Mitchelmore, 1998). Paul also used 20 steps in the move block instead of 30 that Laura used, and as a result, his orbit was smoother and slower than Laura's as Paul expressed, "Mine is not as fast as yours." Students' independent explorations provided them opportunities to discover different ways to solve the task while closely attending to how they used the concept of angles in the programming activity.

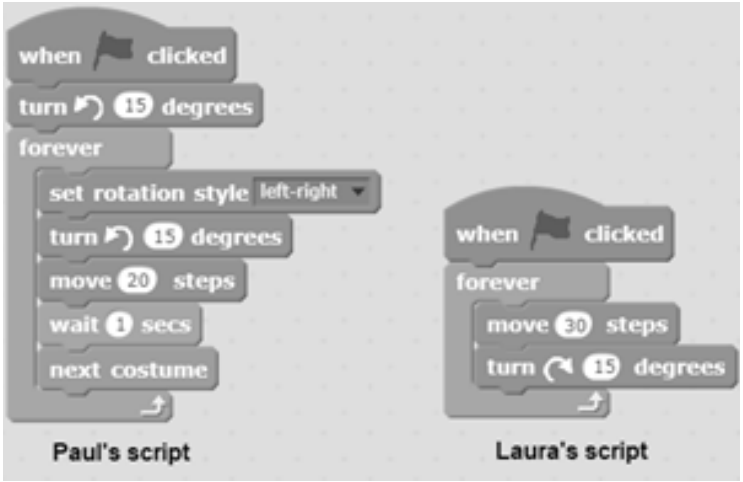


Figure 6. Paul's script (left) and Laura's script (right) for orbiting the satellite

We noticed that while working together to compare their work and answer questions of one another, students developed deeper understanding of mathematical ideas. They conceptualized multiple solutions to a problem by trying to understand how the solution of others work. This social sharing process is one of the benefits of undertaking constructionist activities in computational environments (Papert, 1980).

Episode 4: Connecting ideas about angles

By the end of the experiment, students were able to describe a full orbit as 360 degrees:

> Paul: It moves back from the earth. That's the code. And I'm going to make them like I'm going to stop it, it goes back where it was. There's 360 backflip, it goes back into space from earth. That is what I call science. But I want to do it better.

In addition to his math reasoning about orbit, Paul also made an explicit link to the science context of satellite sent into space from the earth. Similar to Paul, students were able to make connections between concepts used in the programming task with the ideas in mathematics, other disciplines, or even with their everyday lives. When students actively engaged with the designed task on Scratch, they creatively integrated scientific, mathematical, and technological ideas as a learning experience meaningful to them.

CONCLUDING REMARKS

This paper illustrated the potential of a Scratch programming task for developing students' mathematical understanding about angles. Students were able to reason about the effect of the angle measurement in the "turn_degrees" block and the number of steps in the "move_steps" block on the nature of an object's turn. They also utilized the clockwise-counterclockwise motion that an object can follow and identified that 360 degrees makes an object return to its original position. Moreover, the students learned that turns may be constructed differently but can form similar results. Therefore, we consider that integrating relevant programming activities in mathematics classrooms is important in providing opportunities for younger students to explore the dynamic nature of angles and advance their understanding of angles used in other disciplines.

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under grant #1742125. The views expressed do not necessarily reflect official positions of the Foundation.

REFERENCES

Ainley, J., Pratt, D., & Hansen, A. (2006). Connecting engagement and focus in pedagogic task design. *British Educational Research Journal, 32*(1), 23-38.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2016). *Building mathematical knowledge with programming: insights from the ScratchMaths project*. In: Proceedings of Constructionism 2016, pp. 25-32.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing mathematical thinking with

Scratch: An experiment with 6th grade students. In *Design for Teaching and Learning in a Networked World, 10th European Conference on Technology Enhanced Learning, EC-TEL 2015* (pp. 17-27). Toledo, Spain: Springer. doi: 10.1007/978-3-319-24258-3_2

Clements, D. & Battista, M. (1989). Learning of geometric concepts in a Logo environment. *Journal for Research in Mathematics Education, 20*(5), 450–467.

Clements, D. H., Battista, M. T., Sarama, J., & Swaminathan, S. (1996). Development of turn and turn measurement concepts in a computer-based instructional unit. *Educational Studies in Mathematics, 30*(4), 313-337.

Cobb, P., Confrey, J., DiSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. Educational researcher, 32(1), 9-13.

Cobb, P., Stephan, M., McClain, K., & Gravemeijer, K. (2001). Participating in classroom mathematical practices. The journal of the Learning Sciences, 10(1-2), 113-163.

Devichi, C. & Munier, V. (2013). About the concept of angle in elementary school: Misconceptions and teaching sequences. *Journal of Mathematical Behavior, 32*(1), 1–19.

Hardison, H. L. (2018). *Investigating high school students' understandings of angle measure* (Doctoral dissertation). Retrieved from https://getd.libs.uga.edu/pdfs/hardison_hamilton_l_201805_phd.pdf.

Hollebrands, K. F. (2003). High school students' understandings of geometric transformations in the context of a technological environment. *The Journal of Mathematical Behavior, 22*(1), 55-72.

Hoyles, C., & Noss, R. (1992). *Learning mathematics and Logo*. Cambridge, MA: MIT Press.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(4), 16.

Mitchelmore, M. (1998). Young Students' Concepts of Turning and Angle. *Cognition and Instruction, 16*(3), 265–284.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. NY: Basic Books, Inc.

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism, 36*(2), 1-11.

Smith, C. P., King, B., & Hoyte, J. (2014). Learning angles through movement: Critical actions for developing understanding in an embodied activity. *Journal of Mathematical Behavior, 36*, 95–108.