Learning Mathematics through Programming

Misfeldt, Morten; Ejsing-Duun, Stine

Published in:
CERME9

Publication date:
2015

Document Version
Early version, also known as pre-print

LEARNING MATHEMATICS THROUGH PROGRAMMING: AN INSTRUMENTAL APPROACH TO POTENTIALS AND PITFALLS

Morten Misfeldt & Stine Ejsing-Duun

Research Lab: ICT and Design for Learning, Aalborg University, Copenhagen

In this paper we explore the potentials for learning mathematics through programming by a combination of theoretically derived potentials and cases of practical pedagogical work. We propose a model with three interdependent learning potentials as programming which can: (1) help reframe the students as producers of knowledge and artifacts, (2) support abstraction and encapsulation, and (3) promote thinking in algorithms. Programming is a topic that has recently gained interest in primary and lower secondary education levels in various countries, and hence a specific analysis of the potentials in relation to mathematics is paramount. Analyzing two cases, we suggest a number of ways in which didactical attention to epistemic mediation can support learning mathematics.

INTRODUCTION

Programming and mathematics are often thought of as strongly connected activities. Partly because of their shared genes—the first computers were conceptualized and build by mathematicians—but also because programmers attend to logic, procedures, and functions in order to obtain their goals. Over the years a number of projects in mathematics education aimed at utilizing programming to obtain mathematical learning goals with the students. The earliest of these projects tended to collapse in mainstream implementation due a complex combination of lacking technological readiness of the school system, teacher competences, and more principal didactical difficulties with connecting programming activities to accepted mathematical curricular goals.

Recently, several countries have included basic programming in the national curriculum. In some of these countries (such as Estonia and France) programming is placed in direct curricular connection to mathematics, whereas in others (England, and Sweden) programming is related more to a design and engineering agenda. However, in all cases the focus is not on developing general "humanistic" skills with technology, rather it is on thinking in algorithms, writing programs, and developing technology. In other countries such curricular changes are being discussed and tested on a small scale. Hence, it makes sense to take a closer look at the arguments that have previously been proposed for utilising programming in mathematics education. In this paper we will modestly attempt to describe these arguments, however in order to compare and combine previous thoughts on this topic we will employ the instrumental approach to the use of Information and Communication Technology (ICT) in mathematics education. The instrumental approach was developed in a French didactical tradition to meet the challenge that computer algebra systems posed

to mathematics education and it has in the last decade become a European mainstream framework for addressing ICT in mathematics education.

In this paper we describe some of the main intellectual projects and frameworks in mathematics education that used programming as a means to obtain mathematical learning goals. We suggest classifying these projects in three clusters; (1) *viewing students as producers*, (2) *supporting abstract thinking,* and (3) *developing algorithmic thinking*. Using the instrumental approach as theoretical framework we describe two educational situations utilizing pupils' programming activities in order to learn mathematics.

WAYS OF THINKING ABOUT TEACHING MATHEMATICS WITH PROGRAMMING

The tools that we choose to bring to mathematics students do influence the learning of mathematics that becomes likely or possible (Guin et al., 2005, Ainley, Pratt, & Hansen, 2006). And in that sense bringing programming into mathematics teaching does support certain types of learning. Bringing programming into the classroom with the purpose of learning mathematics easily leads to a version of the *planning paradox*; the more detailed the teacher articulates the mathematical learning goal, the more difficult it can be for pupils to appropriate programming as a personal instrument (Ainley et al., 2006).

Students as producers: Constructionism and a different mathematics

Serious attempts to use programming in teaching mathematics in primary and lower secondary school started with Seymour Papert. Papert's idea was simple–to create an interactive universe (microworld) that children access through mathematics, which prompts them to think mathematically by embedding nuggets of mathematical knowledge into the microworld that the pupils playfully stumble upon while developing projects.

As a means to obtain this goal, Seymour Papert developed the programming language LOGO, where the child steers a small turtle around the screen with commands such as "forward 10" and "right 90". The turtle can leave a trace allowing the child to create various geometrical figures. Papert's pedagogical strategy, constructionism, suggests that children learn in a particularly efficient way when they are engaged in developing constructs such as beautiful patterns, interactive art, computer games, etc., and in his bestseller, Mindstorms (1980), he describes LOGO as a 'mathematical microworld' that allows children to engage in such projects. The teacher's role in such work is to connect the children's work and intentions to "powerful ideas" from our mathematical heritage (Papert, 2000).

During the 1980s there was great enthusiasm and confidence that LOGO and similar programming languages would radically reform mathematics teaching in primary schools, and the first ICMI study on technology in mathematics education was focussed on how technology influenced mathematics as a topic (Churchhouse &

International Commission on Mathematical Instruction, 1986). However, the results in mainstream implementation did not entirely live up to the expectations. There are a number of reasons for the disappointing results; for instance, students easily overlook the nuggets of mathematical knowledge (Noss & Hoyles, 1992, Ainley et al 2006), making their work in the microworld non-mathematical.

Abstraction and concept formation: APOS theory

The idea that programming could be helpful in mathematics education in the late 1980s also developed in the context of teaching mathematics in high school and college. Here the geometric and artistically framed LOGO program was less popular. On the contrary, teachers often utilized common programming languages such as BASIC, COMAL and PASCAL to support learning. One of the outspoken hopes was to create a process-oriented approach to abstract mathematics, basing abstract constructions in concrete numerical computations. The arguments for this approach were often based in constructivism and radical constructivism, which claims that all abstract learning has a concrete starting point, as well as in the and in the discussions of process-object duality (Sfard, 1991). Ed Dubinsky's work is probably the clearest description of the learning potential of programming (see Breidenbach et al., 1992). His theory is often referred to as APOS theory and it is situated in a radical constructivist framework (Glasersfeld, 1995). APOS is an acronym for action, process, object, and scheme. The theory describes mathematical concept formation as beginning with performing actions on well-understood mathematical objects; these actions can be organized in processes and encapsulated into objects. These objects can be related to one another in schemas. The encapsulation stage is, as famously described by Sfard (1991), crucial and hard. And the schematic aspects of concept formation is similar to Skemp's relational understanding (1971). This rather general learning theory of mathematical concept formation relates to the use of computers because they can significantly empower and enrich the concrete numerical calculations that are—in this conception—the necessary foundation for concept formation.

Process approach to mathematics: Algorithmic thinking

The ability to think in algorithms and procedures is promoted as an important learning goal in mathematics. Algorithmic thinking describes students' ability to work with algorithms understood as systematic descriptions of problem-solving and construction strategies, cause-effect relationships, and events. A recipe is a good example of an algorithm: (1) Add all dry ingredients together. (2) Stir. (3) Add 2/3 cup of the water and stir. (4) If the dough is steady, then stir for 2 minutes. Otherwise, go to step (3) and add more water. Algorithmic thinking is about being able to develop, execute, and make machines to perform such algorithms. Donald Knuth (1985) views algorithms as a crucial phenomenon constituting the intersection between computer science and mathematics. He traces the study of algorithms to the

mathematical masterpiece Al Kwarizm from the 9th century (Katz, 1993). Knuth defines algorithms as follows (Knuth, 1985, p. 170):

> I tend to think of algorithms as encompassing the whole range of concepts dealing with well-defined processes, including the structure of data that is being acted upon as well as the structure of the sequence of operations being performed; some other people think of algorithms merely as miscellaneous methods for the solution of particular problems, analogous to individual theorems in mathematics.

Hence algorithms, according to Knuth, consist of both a recipe and the actual objects dealt with by the recipe. Knuth analyzes the difference between mathematical thinking and algorithmic thinking. He finds that a first approximation algorithmic thinking relates to (1) representation, (2) reduction, (3) abstract reasoning, (4) information structures, and (5) algorithms. Mathematical thinking can, according to Knuth, relate to all of these, however other aspects are also present such as (a) formula manipulation, (b) behavior of functions, (c) dealing with infinity, and (d) generalization. Hence algorithmic thinking is strongly related to mathematical thinking but emphasizes specific and slightly different aspects than other types of mathematical thinking.

Before we introduce classroom examples exemplifying these learning potentials, we will introduce the instrumental approach that we use as a general theoretical framework for the use of ICT for mathematics teaching. This framework will be used to analyze the cases and create a connected description of the different learning potentials.

THE INSTRUMENTAL APPROACH

The instrumental approach (Guin, Ruthven, & Trouche, 2005) addresses students' use of technology when learning mathematics from the perspective of appropriating digital tools for solving mathematical tasks. It builds on a framework from computer science (Verillon & Rabardel, 1995), which is inspired by activity theory (Nardi, 1996; Rabardel & Bourmaud, 2003), and hence views computational artifacts as mediating between user and goal (Rabardel & Bourmaud, 2003). It is an important aspect of this conceptualization that humans have goals on various levels, and hence that the goal of smaller actions can feed into larger plans (Nardi, 1996). Furthermore the approach presupposes a continuation and dialectic between design and use, in the sense that a pupil's goal-directed activity is shaped by his use of a tool (this process is often referred to as *instrumentation*), and simultaneously the goal-directed activity of the pupil reshapes the tool (this process is often referred to as *instrumentalization*) (Rabardel & Bourmaud, 2003, p. 673). In students' work with technology the distinction between *epistemic mediations* and *pragmatic mediations* (Guin et al., 2005; Rabardel & Bourmaud, 2003) operationalize the difference between learning with technology and just using technology to solve tasks. Epistemic mediations relate to goals internal to the user—affecting his or her conception of, overview of, or knowledge about something Rabardel & Bourmaud (2003) use the example of a

microscope, and Lagrange (in Guin et al., 2005, ch. 5) refers to experimental uses of computers) and pragmatic mediations related to goals outside of the user—making a change in the world (Rabardel & Bourmaud use the example of a hammer, Lagrange (in Guin et al., 2005, ch. 5) refers to the mathematical technique of "pushing buttons"). Finally, Rabardel & Bourmaud (p. 669) introduce sensitivity to a broader conception of the *orientation* of the mediation. Instrumented mediations can be directed towards (a combination of) the objects of an activity (the solution of a task), other subjects (classmates, the teacher), and oneself (as a reflective or heuristic process). Hence the theoretical framework consists of the concepts: *instrumental genesis,* as consisting of *instrumentation and instrumentalization,* the concepts *epistemic* and *pragmatic mediations,* as well as a sensitivity towards the *orientation* of an instrumented mediation. The orientation of the mediation can be towards oneself, external objects, and other subjects.

EXAMPLES OF LEARNING MATHEMATICS WITH PROGRAMMING

These classroom observations are taken from the project *Children as Learning Designers in a Digital School.* The project is the realization of a research call from the Danish Ministry of Education. The research project is directed toward the area students own production and student involvement (Levinsen et al., 2014) and it explores:

1) How students' digital production impact on learning processes and the qualification of learning results regarding subjects and trans-disciplines; and
2) How ICT involves designs for learning that allow students to act as learning designers of their own learning practice in terms of form, framing, and content on their learning, engagement, and motivation.

The project comprises of a number of interventions in different schools. The examples in this paper come from a mathematics class where children in 5th grade (approximately 11 years old) program games for peer pupils to play and discuss using iPads and the software program Hopscotch. We present two activities that we suggest are related to the three different learning potentials described earlier.

Creating a good game: "It has to be fun"

The first example relates directly to the students' potential as artifacts producers. Oliver is trying to solve a problem–he wants to move his figure using tilt (i.e. by tilting the iPad). He asks the others for help. Instead of suggesting a solution, Ally asks him, "Why aren't you just tapping it?" Oliver answers, "Because it's a game, Ally. It should be fun."

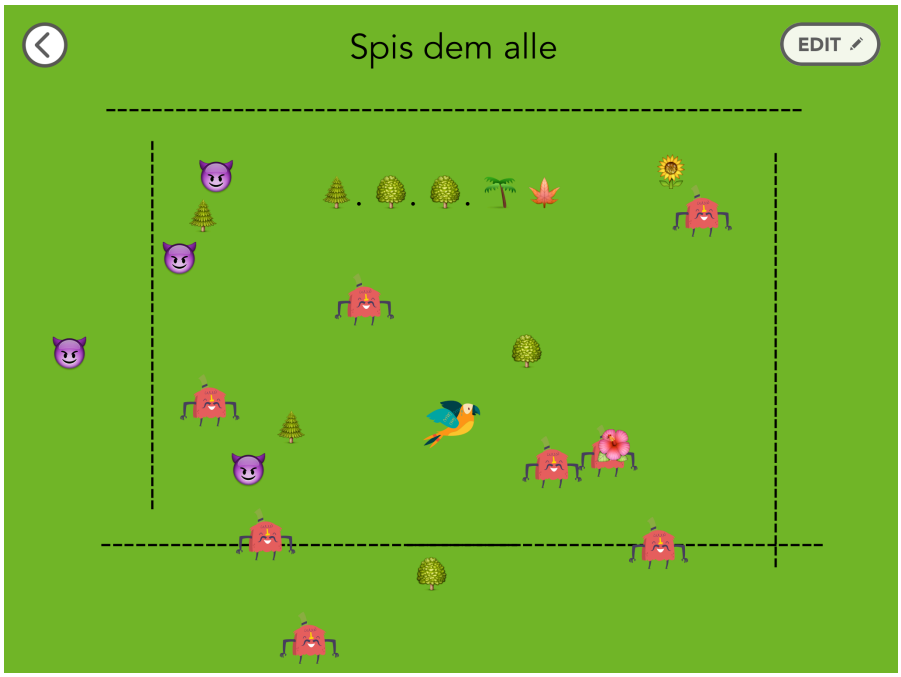

Figure 1: Oliver's first game, "Eat them all". The player controls the parrot by tilting the iPad. The goal is to eat the toasts and avoid the purple devils.

The motivation for Oliver is obviously that the game he creates should be fun. Programming is merely a means for obtaining that goal. Throughout the course of the intervention Oliver gets really far in the process of making games. He is very independent and on his own he examines other games in order to, e.g., make points.

For the same reason—wanting to develop good games—more pupils want to make countdowns, scoring systems, control with arrows, etc. They know the game genre well and what is needed to make a good game. These elements can only be done using variables or values related to algebra, a topic that they do not know particularly well. Despite the lack of algebra knowledge (algebra is considered "above their level" in the school), half the class voluntarily and with a high level of focus attends as the teacher demonstrates how to use algebraic concepts (variable and coordinate systems) to make an arrow control. In order to move one object (e.g., the avatar) by touching another object (e.g., an arrow) the pupils need to make a move-variable. Despite the emergence of this rapid algebra course, the pupils are not working with a task defined by the teacher. The teacher has merely defined a frame, "make a game", and the pupils themselves start defining tasks within it.

From an instrumental perspective the pupils' interest in understanding the mathematical concepts that the teacher are oriented towards can be viewed as a pragmatic end of creating a good game. Such an end is indirectly oriented towards peer students as players of their games. Obtaining the pragmatic goal of creating a game does require students to obtain epistemic goals—in this case about the coordinate system—as sub-goals along the way. But understanding and acknowledging that there are mathematical sub-goals might not be so easy. In this case, mathematical sub-goals are strongly supported by the teacher's choice to create

a “what kind of math do I need in order to make my game” crash course, deliberately focusing the pupils’ attention on the mathematical aspects of creating arrow control and scoring systems.

Thinking in algorithms

The second example we initially see as relating to algorithmic thinking, but also to using programming as a way to support later abstraction and reification. The activity is introductory (just after the teacher has introduced the course structure and learning objectives). In plenary, the pupils and the teacher program a small cardboard figure which the teacher has set up on the whiteboard. They decide to call him 'Puff'. The teacher challenges the pupils by asking how to make Puff do various things and the following dialogue happened (translated from field notes):

Teacher:	Puff can only speak mathematics. How can I make him go right?
Zack:	Go right.
Teacher:	He does not know how far he should go.
Marc:	Go 2 centimeters to the right.
Teacher:	Yes, but unfortunately he does not know centimeter on the screen.
Austin:	Displace two units to the right.
Teacher:	Yes, units he understands. But he does not know what right is.
Ann:	You must move to a coordinate.
Oliver:	If he should go to the left, then: Go -3.
Ann:	Could you get him to go to a coordinate?
Teacher:	Yes, he would go there–but he would then fly around.

The teacher demonstrates her point by moving the cardboard figure from one point to another instead of sliding between points. She then shows how moving with a positive number makes Puff go right and negative number, as suggested by Oliver, will make him move left. After some discussion the teacher raises another issue:

Teacher:	They [the sprites in Hopscotch] are ego–they see the world from their own noses. How do you think he can go downwards?
Girl:	I am just guessing...can he rotate degrees?
Teacher:	Yes he can.
Zack:	Rotate 90° clockwise.
Teacher:	Now, you have to try programming each other. Give each other a rule and a signal. Use rotation and units. Make a square by controlling the other orally.

Pupils work together in pairs. They try to control each other. Zack and his teammate come over to the teacher and are frustrated. Zack says that he does not know which way to turn when she just says “turn 90° degrees”. His teammate complains that he

“just lies down on the floor” instead of moving around. The teacher talks with them about being precise and setting an x- and y-axis on the floor.

This example shows how pupils struggle with translating programming the figure on the whiteboard, which has two dimensions, to programming each other in three dimensions. Zack understood that the language should be precise, but he also teases on purpose. Several pupils have this kind of negotiation with mathematical concepts (turning, coordinate system, etc.), but some are also getting away with just saying “turn 90 degrees right/clockwise” without their partners correcting them. Those who are being controlled sometimes find that if they follow the instructions they end up walking into things, especially as the units are not precisely specified. Most use one step as a unit, some are using one foot as a unit.

By having pupils translate the programming activity to a classroom situation the teacher might promote reflection on the relationship between spatiality and algorithms. Another consequence of having pupils mediate programs by playing roles is that they get an understanding of what precision means. After this introduction the pupils described programming as a mathematical language that you ‘speak’. From an instrumental perspective, the pupils here aim at affecting other subjects directly through programming and it has the effect that they negotiate what a good algorithm is and what it means to be precise in such instructions. It is a classical point that learning to program can benefit from attempting both to act as the creator of algorithms and as the performer (this is described as "playing turtle" by Papert, 1980). But it is interesting that the pupils’ negotiation of the instructions is resolved by the teacher through introducing mathematical concepts (the 2-dimensional coordinate system). The teacher consistently introduces the solutions to pupils’ problems in mathematical terms; this seems like a strong didactical strategy that supports the pupils talking and thinking about mathematics when they work.

PROGRAMMING TO LEARN MATHEMATICS–THE CHALLENGE OF MAINTAINING AN EPISTEMIC FOCUS

In this section, we will discuss whether the mathematics learning potentials that have previously been suggested in the literature about programming and mathematics education can be viewed as genuine mathematics learning potentials in the sense that they involve epistemic mediations towards mathematical concepts. It is obvious from the cases that the pupils need help with mathematical concepts when they try to appropriate a programming language to develop their games. The two examples show how such situations can facilitate the engagement with classical mathematical concepts such as numbers, the coordinate system, and orientation/angles. In both cases the pupils interact with the teacher, each other, and games developed by others in order to handle the challenges. But one can discuss whether the overall pragmatic purpose of improving their skills with Hopscotch and potentially making a better game support or hinder the pupils’ epistemic focus on mathematical concepts. The analysis shows that it is—in these specific cases—not reasonable to disregard this as

only a pragmatic mediation with little educational value. But this could very well have been the case if the teacher had not been so careful in attracting the pupils' attention to explicit and relevant mathematical ideas. However, the pupils also bring in mathematical ideas (for instance, about angles) without being prompted by the teacher. Hence it would be meaningful to investigate further how the classroom norms and shared ideas about mathematics (the sociomathematical norms discussed by Cobb, Stephan, McClain, & Gravemeijer, 2001) affect the mathematical value of introducing programming.

By using the instrumental approach it is apparent that pupils' epistemic relation to mathematics is necessary for programming to be successful in mathematics education. We see several ways that this epistemic relation can be strengthened or hindered. The teacher did acknowledge and talk about the different goal-levels of an activity; this allowed her to talk directly about mathematical goals, even though these where sub-goals of the larger goal of creating a good game. Constantly focusing attention on mathematical concepts as problem solvers and conflict settlers were also actively applied by the teacher, especially when the pupils programmed each other. When the pupils are either negotiating or in cognitive conflict, this teacher turns to mathematical concepts and principles as part of the way forward for the pupils. In that sense our analysis suggest that the potentials for learning mathematics through programming, as previously described in the literature, depends largely on the teacher's approach and didactical principles.

REFERENCES

Ainley, J., Pratt, D., & Hansen, A. (February 01, 2006). Connecting Engagement and Focus in Pedagogic Task Design. *British Educational Research Journal*, 32, 1, 23–38.

Buckingham, D. (2005). *The media literacy of children and young people: a review of the research literature*. London: OFCOM.

Churchhouse, R. F., & International Commission on Mathematical Instruction. (1986). The Influence of computers and informatics on mathematics and its teaching. Cambridge; New York, NY, USA: Cambridge University Press.

Cobb, P., Stephan, M., McClain, K., & Gravemeijer, K. (2001). Participating in Classroom Mathematical Practices. *Journal of the Learning Sciences*, *10*(2), 113–163.

Glasersfeld, E. von (1995). *Radical constructivism: a way of knowing and learning*. London; Washington, D.C.: Falmer Press.

Guin, D., Ruthven, K., & Trouche, L. (2005). The didactical challenge of symbolic calculators turning a computational device into a mathematical instrument. New York: Springer.

Holm Sørensen, B., Audon, L., & Levinsen, K. (2010). *Skole 2.0.* Århus: Klim.

Hoyles, C., & Noss, R. (1992). *Learning mathematics and logo. Exploring with Logo Series*. MIT Press, Cambridge, MA; MIT Press.

Katz, V. J. (1993). *A history of mathematics: an introduction*. New York: HarperCollins.

Knuth, D. E. (1985). Algorithmic Thinking and Mathematical Thinking. *Amermathmont The American Mathematical Monthly*, *92*(3), 170–181.

Levinsen, K., Sørensen, B. H., Tosca, S., Ejsing-Duun, S., & Karoff, H. S. (2014). Research and Development Projects with ICT and students as learning designers in Primary Schools: A methodological challenge. I*n Proceedings of the 4th International Conference on Design for Learning: Expanding the field.* Stockholm: Stockholm University.

Nardi, B. A. (1996). *Context and consciousness activity theory and human-computer interaction*. Cambridge, Mass.: MIT Press.

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

Papert, S. (2000). What's the big idea? Toward a pedagogy of idea power. *IBM Systems Journal*, *39*.

Rabardel, P., & Bourmaud, G. (2003). From computer to instrument system: a developmental perspective. *Interacting with Computers*, *15*(5), 665–691. doi:10.1016/S0953-5438(03)00058-4

Resnick, M. (2012). Point of View: Reviving Papert's Dream. *Educational Technology* -Saddle Brook Then Englewood Cliffs Nj-, *52*(4), 42–45.

Sfard, A. (1991). On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educ Stud Math Educational Studies in Mathematics*, *22*(1), 1–36.

Skemp, R. R. (1971). *The psychology of learning mathematics*. Harmondsworth, Eng.; Baltimore: Penguin Books.

Verillon, P., & Rabardel, P. (1995). Cognition and artifacts: a contribution to the study of though in relation to instrumented activity. *European Journal of Psychology of Education*, *10*(1).