# Kindergarten Programming goes Mobile

Asi Kuperman
*Tel Aviv University, Tel Aviv, Israel*

Ruti Aladjem
*Tel Aviv University, Tel Aviv, Israel*

Davis Mioduser
*Tel Aviv University, Tel Aviv, Israel*

The dyad “programming environment” and “programmable device (e.g., robot)” in the educational scene underwent several transformations in configuration over almost fifty years, since the first programmable floor turtles appeared. From the turtle’s migration to the screen, through its revival as physical device in the Lego-Logo system (and subsequent descendants), the loss of the robot’s “umbilical cord” when communication became wireless (adding spatial flexibility and freedom to the robot’s navigation but at the same time challenging children’s understanding of the communication and control process), and currently the transition to the mobile, “almost wearable” programming environment. Each transition brought about new questions about children’s understanding and learning. The interaction with technology allows kindergarten children to explore and be engaged in technological thinking and to construct a solid basis for the development of complex ideas, through direct manipulation of physical objects. The Kinderbot environment scaffolds this learning experience by providing tools to think and experiment with, as well as a structured pedagogical program. Originally designed as a desktop environment, recently a mobile version of Kinderbot has been developed and implemented. The findings from the initial comparison between the desktop and mobile experiences, reported in this paper, outline three key themes. The first concerns the **perspective of the child programmer**: In mobile programming separation and delays characterizing desktop program-first-then-run disappear and spatial perspectives become unified – the programmer is able to enact the robot’s behaviour while programing “in situ”, where the action is. The programming environment becomes “almost wearable” accompanying the programmer and the robot in the physical space while the actual commands are executed. The second focuses on differences among desktop and mobile programming in different **phases and modes** in the programing experience. The third theme relates to differences in aspects of **collaborative programing** processes.

*Key Words: Mobile programming, enacted programming, educational robotics*

## 1. RATIONAL AND BACKGROUND

Over the past 50 years, since the introduction of the programmable floor turtle, the dyad “programming environment” and “programmable device” (e.g., robot) in the educational scene underwent several transformations in configuration (Resnick & Silverman, 2005). These transformations included stages such as: The turtle’s migration to the screen (e.g., first as in Logo, and more recently in Scratch and Scratch Junior); its revival as a physical device in the Lego-Logo and subsequent systems (e.g. Control Lab, generations of Lego programmable bricks, and KIBO); the loss of the robot’s “umbilical cord” when communication became wireless (adding spatial flexibility and freedom to the robot’s navigation but at the same time challenging children’s understanding of the communication and control process); the introduction of tangible programming which allowed to replace the mouse and keyboard with tangible manipulatives (e.g., “smart” blocks, Belk, 2013; wooden puzzle pieces, Rave & Mioduser, 2014); and recently, the transition to mobile programming environments. Each transition brought about new questions about children’s understanding and learning.

Mobile devices have vast potential for supporting learning and learners. They have become a significant part of daily life, integrating seamlessly with a range of common activities to the point of being regarded as MindTools

E-mail: Asi Kuperman
asikuper@post.tau.ac.il

and as “extensions of the self" (Belk, 2013; Resnick & Silverman, 2005). The qualities of mobile devices in the context of learning and above all, the fact that unlike a desktop computer, they do not constrain the learner to a fixed location, may transform how the child programmer experiences physical behaving devices. Our current research aims to gain understanding of children’s programming experience with a *mobile* learning/programming environment used in kindergartens (the Kinderbot system) compared to their experience using a *desktop-based* version of the system. The preliminary findings reported here, unveil major themes and questions arising from the “migration” from desktop to mobile, and suggest directions for further research.

The main question addressed was: *What characterizes children’s programming (concerning, e.g., skills, strategies, collaborative work) in the transition from a desktop-based to a mobile programming environment?*

## 2. METHOD

### *2.1 The programming environment*

“Kinderbot”, the programming environment used by the children in this study, has been designed to support and scaffold the acquisition of basic programming concepts, and to advance the technological thinking of kindergarten children. Rooted in Papert’s constructionist vision, and his conception of the learning value of the combination of the physical programmable "Turtle" with the Logo programming language, the environment combines symbolic and physical components and allows for playful investigations of both abstract and concrete concepts (Papert, 1980; Mioduser, Levy & Talis, 2009). Kinderbot was originally designed as a desktop programming environment; recently, a mobile (tablet) version has been developed.

The system comprises a (virtual) programming environment, a (physical) robot (built using Lego’s EV3 programmable brick), and (physical) landscapes for the robot’s navigation. The visual user interface (Figure 1) allows the programmer to control the robot’s behavior by constructing programs using icons that represent inputs (e.g., incoming from a touch or distance sensor), outputs (e.g. movements based on activating motors), and configurations of inputs and outputs (e.g., a linear sequence of instructions, a conditional statement). The programming sequence is structured in modes of increasing difficulty (shown at the right of the Figure). The basic modes (1-3) focus on acquiring the basic competencies of controlling the robot and sequencing commands (from immediate mode activation up to creating a script for further running). The more advanced modes introduce the definition of routines (mode 4) and rules of increasing complexity linking between measured inputs and outputs (modes 5-7). The structure and use of the environment are similar on both the desktop and mobile versions, aside for the touch operation modality (e.g., tapping, dragging) in the mobile version. In this paper, we focus on the children’s experience using the first three modes - as they form the basis for acquiring the symbolic language for controlling the robot’s behavior (Mioduser & Levy, 2010).

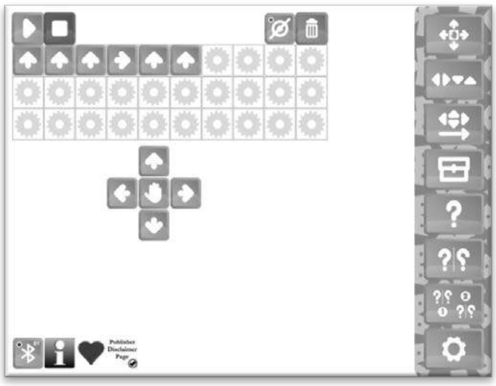

*Figure 1: the icon based Kinderbot UI*

### *2.2 Data collection*

Data collection was conducted by means of: (a) observations of children's performance; and (b) semi-structured interviews with four teachers who use the programming environment in their kindergartens. In the following preliminary data and its analysis are presented.

Naturalistic observations of ten children (age 5-6) in regular kindergarten setting engaged with the mobile and desktop environments were conducted. Each observation session was 20-30 minutes long, the children were observed while constructing routes and creating narrative contexts for the robots to travel through. Concerning ethical issues, the study was conducted in a kindergarten which takes part in a comprehensive project called "Developing Technological Thinking". Parents approved the participation of their children in the studies as well as the use of photography for research and academic purposes.

The teacher interviews focused on issues such as the way children manipulate the mobile device, their posture and position relative to the robot's navigation, their displacement in space, mistakes made, and communication issues. The mobile version of the environment has been initially implemented in two kindergartens. Hence, two teachers had experience with both the desktop and mobile environment while the other two had experience only with the desktop environment. The two teachers who had experience with both environments were also asked to describe their understanding and insights concerning the differences in the programming experience.

## 3. FINDINGS

The following three main themes arose from the analysis of the data and the comparison of the desktop and mobile experience: Changes in the programmer's perspective; changes in foci and learning patterns in programming; changes in patterns of collaboration among peers while programming.

### *3.1 Theme one - changes in programmers' perspectives*

In the desktop mode, there is a physical separation between the location of the computer and that of the robot. The programmer must face the computer screen for entering the commands, while the robot is in a distant position. It even may be pointing to a different direction than the indicated by the commands in the computer screen, meaning that, e.g., it's "ahead" and the "ahead" icon (pointing upwards in the screen) differ in spatial configuration. Hence, to control the robot and to provide commands for its movements, the programmer needs to be able to visualize the robot's perspective while sitting in front of the computer screen. A common strategy that children naturally apply, is to move back and forth between the two spatial worlds: the screen programming environment and the physical robot's scene - thus switching cyclically between the physical/concrete environment to the virtual/abstract environment.

In contrast, with the mobile interface, the programmers naturally tend to take position in the vicinity of the robot facing the same direction as the robot, thus taking the perspective of the robot and following its movements while the commands are performed. In other words, the physical distance and gaps in orientation between the programming environment and the programmed object becomes minimal, and the separation (spatial, and in some modes also temporal) between the symbolic creation of the action and the physical execution of the action is negligible.

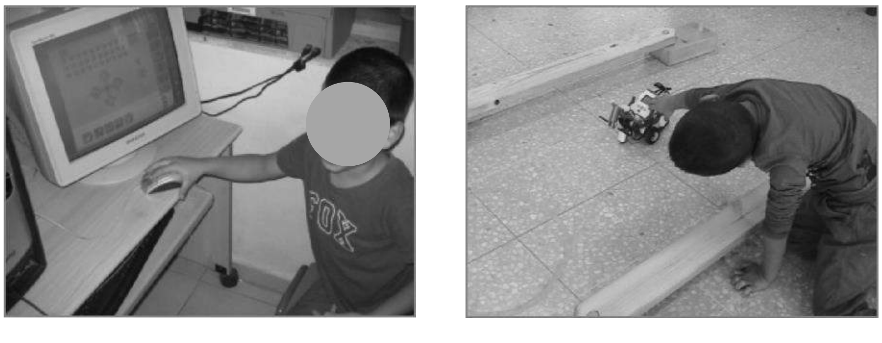

*Figure 2: Switching perspectives while programming using the desktop version*

In previous studies conducted with the desktop programming interface, away from the robot and its physical environment, the integration between the symbolic description of the expected behavior and its physical instantiation was constructed iteratively in the programmer's mind, in a process marked with spatial and temporal delays between the two. In mobile programming these delays disappear, and moreover, the spatial perspectives become unified - the programmer can enact the robot's behavior while programming "in situ", where the action takes place. This brings as close as possible the three components of the symbolic-concrete dialog: playing the robot, describing its desired behavior with symbols and following its actual behavior (Fig. 3).

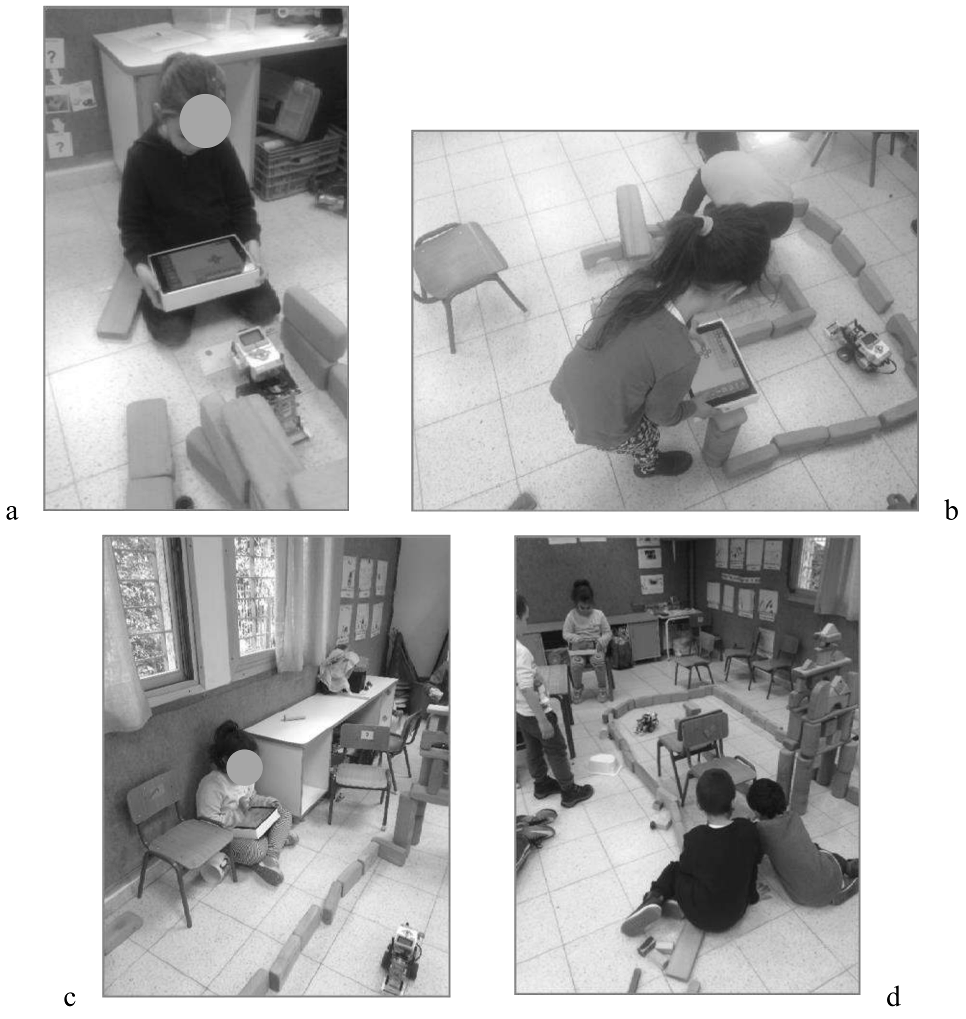

*Figure 3: Using Kinderbot mobile*

### ***3.2 Theme two - changes in foci and learning patterns in different programming modes***

The desktop and mobile modes of Kinderbot share the same logic and structure, yet several differences were noted in the ways in which the programming was carried out by programmers on the different platforms. Following is a comparison of children's performances in the three basic modes of programming between the platforms.

#### *3.2.1 Programming mode one: Immediate execution command-by-command*

The first programming mode allows the learner to get acquainted with the basic robot-controlling commands. The four controls introduced in this mode are simple manipulation commands, represented by four arrow-shaped icons for "forward", "backward", "turn right" and "turn left" actions (see Figure 1). Each command symbolizes a single step of the robot in the physical world. For example, in order to program the robot to take two steps forward and a step to the left - the commands would be: forward, forward, turn left, forward.

In the desktop platform, this phase is usually quite brief, ending once the programmer has grasped the use of the controllers. For example, the programmer needs to understand that selecting a vertical sign for "up" (↑) on the user interface, will translate into a horizontal step "forward" of the robot in the physical world.

In the mobile version, controlling the robot seems even simpler and more intuitive than in the desktop interface. Overall, the experience of the first mode is quite similar to that of using a remotely controlled vehicle (the tablet functioning as the remote control), an experience that many kindergarten children are already familiar with. The programmer is standing near the programmable robot and has eye contact with it and with the immediate result of running a command. Since the programmer is facing the same direction as the robot, she/he is not required to a change in perspective or a mental visualization of the robot's navigation - their perspectives are the same (Figure 3 a,b). However, some children capable of maintaining an appropriate mental model of the spatial behavior of the robot, chose to do the programming while sitting in one fixed spot (Figure 3c).

For all children using the mobile platform, the first mode's stage became extremely brief and was quickly exhausted as the programmers grasped almost immediately the concept of how to control the physical robot and is ready to move on.

### *3.2.2 Mode two: Immediate execution of a sequence of commands.*

During the second programming mode, the programmer is introduced to the notion of sequencing commands. In this phase, similarly to the previous mode, each command is carried out immediately, in real time, so the programmers can view the immediate outcome of their chosen command. In other words, there is no delay between the symbolic action and the physical action. However, unlike the previous mode, each command is stored in an instructions line, forming a visual representation of the sequence performed – this is a program which remains on the screen and can be activated repeatedly.

A sample task in this mode is the one in which the robot must be programmed to navigate a path, while avoiding various obstacles. The navigation sequence is executed immediately, step by step, while being constructed, but it is also being recorded. The program can then be executed, debugged and modified. Thus, the learners experience and experiment with the different commands and their immediate effect on the physical environment, as well as with the idea of constructing a program that can be further executed again and again.

On the desktop platform, the programmers need to continually adjust their perspective while working in the computer screen, to the robot's perspective. For example, at some point, the programmer's left may be the robot's right and vice versa. For a script requiring multiple steps, they also should be able to estimate the actual distances and the number of steps needed to cover it, given the length of a single step. In addition, given that the robot acts in a concrete physical environment, several factors (e.g., friction, delays) might affect its functioning, generating confusing gaps between the expected (programmed) and the actual behavior. As a result, the programmer using the desktop platform is extensively engaged in cycles of trial and error and debugging.

In the mobile version, since the execution is immediate, there is lesser need for prediction and planning. In addition, less debugging is required as the programmer is effectively following the robot in each step with lesser need to accommodate between the robot's point of view and her/his own, or to preplan the next move. The result is that this mode (as the previous) is exhausted in a few cycles and the programmers seem to arrive to the next mode sooner. However, the price is that children spend less time handling challenges or engaging in debugging procedures.

### *3.2.3 Mode three: planning and programming a script*

In this phase, the programmer is introduced to preplanning a sequence of commands (a script). Similar to mode two, the objective here is to program the robot to navigate a complex path. However, unlike the previous phase, a sequence of commands will be executed only after a fully program is created and saved. In order to construct the sequence, the programmer is required to use strategies that involve decomposing the problem, reconstructing it modularly, and planning several steps ahead (anticipation). Once the sequence is executed, it may need to be adjusted using different debugging strategies.

In the desktop platform, the progression to this phase is natural while in the mobile platform this mode seems to pose a substantial challenge. We observed that programmers in the mobile platform pass through the first two modes of programming (i.e., initial learning of the controls and initial sequencing of commands) very briefly, as they seem to pose less of a challenge than on the desktop platform. As a result, programmers arrive to the third phase (involving programming of a script) sooner but having spent less time experimenting and developing necessary knowledge. Although not systematically measured by the educational teams in the kindergartens, their observations were that this phase requires more scaffolding on the mobile programming experience in comparison to the desktop experience.

An interesting difference between the desktop and mobile experience relates to debugging. In desktop programming in which the programming environment is spatially detached from the behaving robot, children's main debugging strategy was "replacement": after observing a robot's undesired move, they replaced the instruction causing the (wrong) move by another one. In mobile programming, where the children followed closely the robot's movements, the main strategy was "counterbalancing": not replacing but adding an instruction aimed to correct the wrong move. The result was a sort of continuous "real time" debugging process.

### *3.3 Theme three - changes in patterns of collaboration among peers while programming*

Teamwork and collaboration processes implicated interesting differences between the two modalities of work. In desktop programming, collaboration between peers developed naturally and was essential, due to the different spatial positioning of the robot and the computer which require a dual perspective. As a result, team work evolves: while one child sits by the computer and enters commands, her/his peers stand by the robot and reports on the outcomes of the commands or programs suggesting further actions. A natural "division of labor" and configuration of complementary perspectives emerge, as one child takes the perspective of the robot while the other takes the one of the programmer. The children arrive to the desired results together, through discussion and collaboration. In addition to the dual perspective collaboration, in some cases two programmers sit jointly by the computer and discuss the programming scenario and issues (e.g., the number of steps or turns needed to complete the task). The navigation track is also often constructed collaboratively by peers discussing its path or the location of obstacles in it. Sometimes the track is built or adjusted dynamically, during the programming session, by several learners in a collaborative process.

Due to the nature of the mobile device, the programming process is more individual in nature. The mobile programming environment is in a handheld, personal device and it is not well suited for manipulation by more than one person. In addition, unlike the desktop scenario described above, the person holding the mobile programming environment stand by herself/himself just with the robot, following its behavior and grasping in real time the bugs to be fixed. However, some forms of collaboration emerge, as other children wish to be involved and actively look for ways to contribute. Most of the collaboration on the mobile mode revolves around the construction of the navigation track for the robot; children participate by adding obstacles, preparing signs (beginning, end etc.) and discussing potential scenarios for enhancing the track (Figure 3d).

## 4. CONCLUDING REMARKS

The interaction with technology allows kindergarten children to be engaged in technological thinking and to construct a solid basis for the development of complex ideas, through direct manipulation of physical objects (Levy & Mioduser, 2010). The programming environment scaffolds this learning experience by providing tools to think and experiment with. Originally designed as a desktop environment, a mobile version has been recently developed and implemented. The initial findings from the comparison between the desktop and mobile experiences outline key themes that should serve as basis for further systematic research.

The emotional aspects of using the mobile based programming environment should also be further explored. The use of mobile devices creates a new kind of intimacy (Belk, 2013; Turkle, 2008) and further research should explore the implications of the apparent closeness between child and robot. For example, past studies on the desktop environment, found that after some engagement with the robot, children develop a technological -as opposed to an intentional-psychological- perspective towards the robot and relate to the robot as a technical construct and not as an entity with personal will or wish (Ackermann, 1991; Kuperman & Mioduser, 2012).

The mobile, often dubbed as an extension of the self, minimizes the physical distance and separation between the child and robot and this may affect her/his perception of the robot.

Although beyond the scope of this paper, we are aware that the advanced phases of programming as well, dealing with rule and adaptive behavior based on incoming data (via the sensors) should also be examined in depth. Reaching a deeper understanding of the different aspects of the mobile programming experience, may contribute to the design of environments that make use of the unique affordances of the mobile device.

The results of this preliminary study have practical implications as well. The mobile version of the robotic environment is currently in use by several kindergartens in our project. Many of the reported observations will serve as basis for the further design of tasks emphasizing the unique opportunities afforded by the mobile modality.

## 5. REFERENCES

Ackermann, E., (1991). The agency model of transactions: Towards an understanding of children's theory of control. *In J. Montangero and A. Tryphon (Eds.). Psychologie genetique et sciences cognitives.*

Belk, R. (2013). Extended self in a digital world. *Journal of Consumer Research*, 40(3), 477–500

Bers, M.U. and Horn, M.S. (2010) 'Tangible programming in early childhood: revisiting developmental assumptions through new technologies', in Berson, I.R. and Berson, M.J. (Eds): *High-Tech Tots: Childhood in a Digital World*, Information Age Publishing, Greenwich, CT, pp.49–70.

Kuperman, A., & Mioduser, D. (2012). Kindergarten Children's Perceptions of "Anthropomorphic Artifacts" with Adaptive Behavior. *Interdisciplinary Journal of E-Learning and Learning Objects*, *8*, 137-148.

Levy, S., & Mioduser, D. (2008). Does it ''want'' or ''was it programmed to...''? Kindergarten children's explanations of an autonomous robot's adaptive functioning. International Journal of Technology and Design Education, 18(3), 337–359.

Levy, S. T., & Mioduser, D. (2010). Approaching complexity through planful play: Kindergarten children's strategies in constructing an autonomous robot's behavior. *International Journal of Computers for Mathematical Learning*, *15*(1), 21-43.

Mioduser, D., & Levy, S. T. (2010). Making sense by building sense: Kindergarten children's construction and understanding of adaptive robot behaviors. *International Journal of Computers for Mathematical Learning*, *15*(2), 99-127.

Mioduser, D., Levy, S. T., & Talis, V. (2009). Episodes to scripts to rules: Concrete-abstractions in kindergarten children's explanations of a robot's behavior. *International Journal of Technology and Design Education*, *19*(1), 15-36.

Papert, S. (1980). Mindstorms: Children, Computers, and Powerful Ideas. Cambridge, MA: Basic Books

Rave, G., and Mioduser, D. (2014). Young Children construct planning skills via programming the behavior of a virtual robot using tangible programming language. In *Proceedings of the 9th Chais Conference for the Study of Innovation and Learning Technologies* (Hebrew), 286-292.

Resnick, M. (1998). Technologies for lifelong kindergarten. *Educational technology research and development*, *46*(4), 43-55.

Resnick, M., & Silverman, B. (2005). Some reflections on designing construction kits for kids. *IDC '05 Proceedings of the conference on Interaction Design and Children*, 117-122.

Sharples, M., Taylor, J., & Vavoula, G. (2007). A theory of learning for the mobile age. *In R. Andrews & C. Haythornthwaite (Eds.), The Sage handbook of elearning research* (pp. 221-47). London: Sage.

Talis, V., Levy, S. T., & Mioduser, D. (1998). *RoboGAN: Interface for programming a robot with rules for young children*. Tel-Aviv: Tel-Aviv University

Turkle, S. (2008). Always-on/always-on-you: The tethered self. In Handbook of mobile communication studies, ed. James E. Katz, 121–137. Cambridge, MA: MIT Press.