Int J Comput Math Learning (2010) 15:21–43
DOI 10.1007/s10758-010-9159-5

Approaching Complexity Through Planful Play: Kindergarten Children's Strategies in Constructing an Autonomous Robot's Behavior

S. T. Levy · D. Mioduser

Abstract This study investigates how young children master, construct and understand intelligent rule-based robot behaviors, focusing on their strategies in gradually meeting the tasks' complexity. The wider aim is to provide a comprehensive map of the kinds of transitions and learning that take place in constructing simple emergent behaviors, particularly for young children. Six kindergarten children participated individually in the study along five sessions. Regarding modes of engagement, it was found that the children conducted intensive and extended playful investigations of the robot's behaviors, interacting with it in a variety of ways; it was also found that their constructions were planful and anticipatory, as they could simulate how the behaviors play out even prior to running their programs. Three kinds of transitions were found in the children's comprehension of the system: one involved adaptation to the formal language; the second, coordination of multiple spatial perspectives; and the third involved a shift from viewing rules as one-time events to their view as recurring and continual descriptions of a process. Finally, it was found that the children employed two strategies to reduce the amount of information in the system: "pruning" involved ignoring part of the logical structure and focusing on another; "fusing" involved coalescing several rules or functions into one. These results are discussed with respect to previous literature on children's programming and with regards to understanding and supporting young children's learning through their construction of adaptive autonomous behaviors.

Keywords Robotics concepts · Cybernetics · Preschool education · Programming · Adaptation · Emergence

---

S. T. Levy (✉)
Faculty of Education, University of Haifa, Mount Carmel, 31905 Haifa, Israel
e-mail: stlevy@construct.haifa.ac.il

D. Mioduser
School of Education, Tel-Aviv University, Ramat-Aviv, 69978 Tel-Aviv, Israel
e-mail: miodu@tau.ac.il

1 Introduction

The aim of this paper is to provide a comprehensive map of the kinds of transitions and learning that take place while constructing simple emergent behaviors, particularly for young children. This paper is one of two in which we report on young children's ability to plan, implement and explain the adaptive behavior of a behaving artifact—a Lego-made robot with sensors. In the other paper (Mioduser and Levy 2010) we present our findings concerning two main questions: what do children *do*, and what do they *say* (explain) about what they do, while engaged in a series of behavior construction tasks of increasing complexity. In this paper we focus on *how* they do what they do, i.e., on characteristics of task engagement, on the strategies used and developed by the children to construct the robot's behavior, to debug faulty programs and cope with the increasing complexity of the tasks. The previous studies into children's abilities in analyzing and describing emergent behaviors with rules have demonstrated that unsupported, children are able to construct with a greater number of rules than they can analyze or describe (Mioduser et al. 2009). This gap is examined in the current study, as we attempt to understand how children's actions go beyond their articulations as they adapt to the task and succeed at constructing such behaviors. This study is unique in providing a broad view along several dimensions of children's boot-strapping strategies and transitions that take place during such activities.

2 Background: Children and Programmable Artifacts

Research on young children's programming has been published as early as the Logo Memos or research reports by the MIT-AI Lab in the early 1970's. Since then, published work on children's programming strategies sheds light on significant characteristics of the process by which they construct programs. Among the many issues studied were the distinction between top–down and bottom–up programming strategies (see Singh 1992); methods and strategies developed by children at different learning stages as they gain programming expertise (e.g., Kull 1986); children's ability to decompose problems or to cope with problems of increasing complexity (e.g., Linn and Clancy 1992; Kurland and Pea 1985); children's acquisition of programming strategies embedded in novel-approach programming languages, e.g., the ToonTalk animated programming language (Morgado et al. 2003). In general, findings are not consistent and even controversial. Research reports describe the difficulties faced by children in programming tasks, and emphasize the need to devise developmentally appropriate environments and pedagogical solutions to help them overcome these difficulties (Pea 1987; Mayer 2004).

More specific to this study, reports on children's work with programmable artifacts is also found in the literature since the early 1970's, such as in the pioneer work of Perlman (1974). Perlman developed the Tortis programming system for controlling the Logo turtle (Papert 1980/1993). Tortis was designed as a tangible programming system, allowing children to move physical objects (either as physical buttons or cards in a slot-machine) to express programs, rather than type commands in a computer. A row of cards represented a procedure or sequence of commands, which could be "edited" (e.g., replaced, moved), and executed by pushing a button at the end of the row (Morgado et al. 2006). Observations were conducted with pre-school children, as young as 31/2 years old (Perlman 1976).

Since then, several systems for controlling programmable artifacts were developed and studied. In some cases the system integrates between a computer component, i.e. a programming language or interface, and a physical component—an artifact or robot. In other cases the system leaves out the computer and the physical components themselves are programmable, or combined to construct the system's behavior. Examples of the first are the evolving set of systems designed to control Lego-made artifacts, e.g., the Lego-logo and Control-lab systems (Resnick et al. 1998), or the RoboGan system (Levy and Mioduser 2008). Examples of the programmable artifacts category are "Topobo", an assembly system embedded with kinetic memory (Raffle et al. 2004), or the "Electronic Blocks", in which electronics was put inside Duplo blocks for performing various functions (e.g., as sensor, action or logic blocks)—a program is actually what is embedded in a configuration of blocks (Wyeth and Purchase 2000). Another interesting example is "Story Rooms" (Montemayor et al. 2001), comprising a set of tools allowing "the creation of computer programs by physically manipulating computationally augmented (or aware) objects in a ubiquitous computing environment" (p. 300). In this approach, children build physical objects using a wide range of materials, and use "physical icons" (referring to sensors and actuators) to program—actually to make up a story with the physical components. An expansion of the idea of electronic manipulatives are kits in which the programs built with the blocks serve as an environment for learning about disciplinary content, as in the "System Blocks" designed as hands-on simulation tool to explore systems concepts (Zuckerman et al. 2005), or "roBlocks"—a robotic construction kit for math and science education (Scweikardt and Gross 2006).

Survey of the literature reporting on the implementation of environments for programming physical artifacts shows that most of it is descriptive, focusing primarily on the presentation of the tools, and their characteristics and learning potential (for example, see the historical overview of MIT's work on physical environments for children in McNerney 2004). Systematic report on research concerning young children's interaction with programmable physical devices is sparse. Several studies have investigated the processes by which children plan, create and relate to physical world objects they have designed and built (Fleer 1999, 2000; Carr 2000). However, little research has been conducted aiming to understand the process by which children construct the behaviors of smart devices, such as Lego-made robots or programmable toys.

In another study that analyzes the same dataset (Mioduser and Levy 2010), we have found that young children were mainly successful in constructing robot behaviors with rules, solving such tasks within a relatively small number of debugging cycles that increased slightly with task complexity up to an average of two. As for scaffolds, the children required more adult support in the more complex tasks involving four condition-action pairs (or two rules[1]) and displayed developmentally advanced problem-solving behaviors that correspond with the interface affordances: when debugging their programs, they backtracked to partial solutions, and constructed the more complex behaviors in a modular fashion. The level of complexity of their explanations increased with the complexity of the tasks; however, children were able to do more than what they were able to say about what they did—a gap of about one rule (two condition/action pairs) was observed between the complexity of the behaviors they constructed and the explanations following the construction. Finally, as task complexity increased, the children tended to use a technological rather than an intentional-psychological perspective. In the current study, we delve deeper into the processes of learning, the transitions taking place as the children grapple with the task at hand, increasing their ability to create emergent robot behaviors.

---

[1] The operational definition of *rule-base configuration* is the number of pairs of condition-action couples (If… Then… couples). One *robot control rule* consists of a pair of complementary condition-action couples (If true … Then…; If false… Then …).

We believe that children's engagement in constructing robots' behaviors affords and stipulates unique understandings of the device and of the behavior-construction process. We also hypothesize that an environment, which is both symbolic and physical (the computer interface and the physical robot and its environment), abstract and concrete, supports and requires unique strategies in facing behavior-construction tasks, debugging faulty or undesired behaviors, and for mentally modeling the increasing complexity of the emergent robot behaviors.

Upon these hypotheses the main research question we approach in this paper is: *What characterizes the process by which young children construct the adaptive behavior of a robot?*

We approach the question by analyzing three main aspects or sets of issues of the children's work:

Issue 1: Children's modes of engagement while solving the tasks.
Issue 2: Evolving abilities and understandings along the process.
Issue 3: Strategies for coping with complexity.

3 The Study

Six children participated in the study, three boys and three girls, selected randomly out of 60 children in an urban public school in the central area of Israel (socioeconomic status defined as mid-high). Their ages spanned from 5 years 6 months to 6 years 3 months, with a mean age of 5 years 9 months and a standard deviation of 3 months. At the time of the study, these children were mainly pre-literate: recognizing some of the letters, but not reading; counting but not adding or subtracting to ten.

Two sets of instruments have been developed for the study: a computerized control environment and a sequence of tasks. The computerized control environment was designed to scaffold the children's learning process. This environment includes a computer iconic behavior construction interface (Fig. 1), a physical robot (made with the Lego system) and modifiable "landscapes" for the robot's navigation (Fig. 2). A key component of the environment is an iconic interface for defining the behavior-control rules in a simple and intuitive fashion (Talis et al. 1998). The left panel shows the inputs to the system, the information the sensors can collect and transmit. The right panel presents the possible actions the robot can perform. The central section is devoted to the "behavior construction board" in matrix form. The configuration of this section changes with advancing tasks: starting with one condition-action couple and ending with that seen in Fig. 1: two complete rules or four condition-action couples. Each square shows an action to be performed when the two conditions (row and column) are met.

The subjects in our study participated in a sequence braided of two strands of tasks: Description and Construction. In this paper we focus on the Construction tasks, in which the child constructs specific robot behaviors (for papers relating to the Description tasks see: Mioduser et al. 2009; Levy and Mioduser 2008). The full set of description and construction tasks is presented in the "Appendix". Figure 3 presents the study design.

Prior to each construction task, the children were presented with a robot operating in an environment, such as circling the perimeter of an island. The child was interviewed regarding this behavior.

A Construction task began with explicating the program controlling the robot's behavior in the (previously observed) Description task. The child was then presented with a new

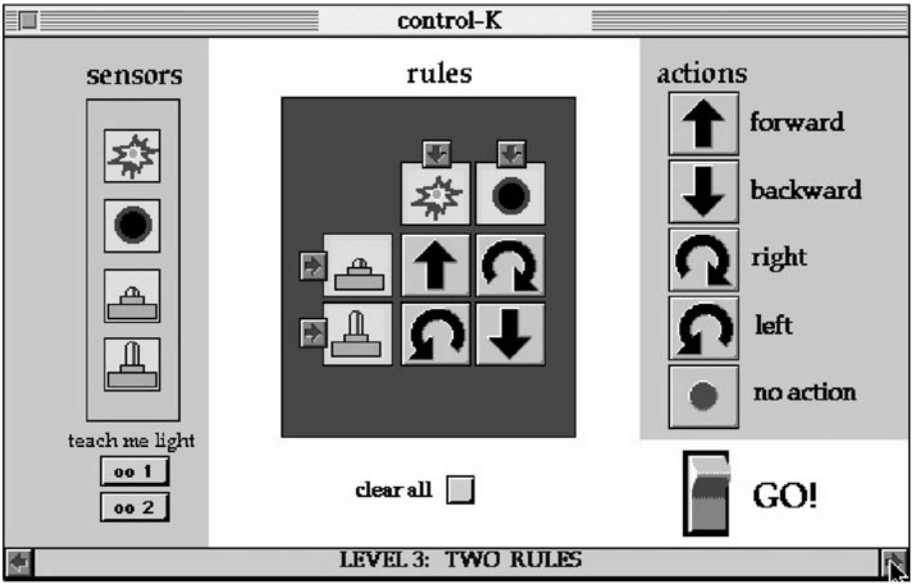

Fig. 1 Sample screen of the computer control environment for two interrelated rules

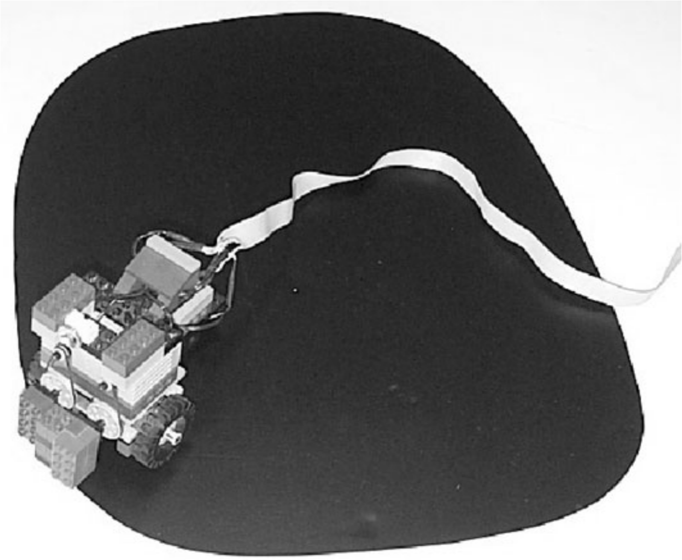

Fig. 2 Setting for 'guarding the island' task

goal, such as "teach the robot to cross a bridge over water" and proceeded to construct and test this behavior.

An example of a construction task takes place in an obstacle field, through which the robot is required to navigate without getting stuck at the barriers (Fig. 4). The robot has a front-protruding touch sensor that gets pressed when the robot runs into an object. The

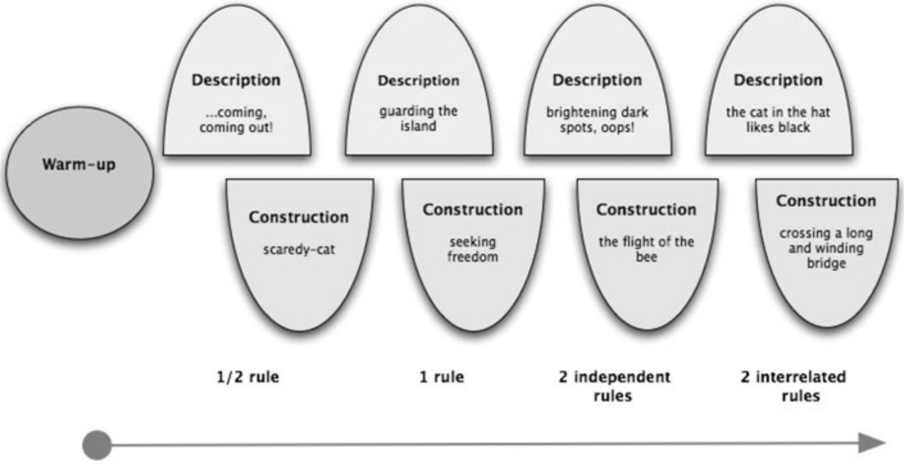

Fig. 3 Study design

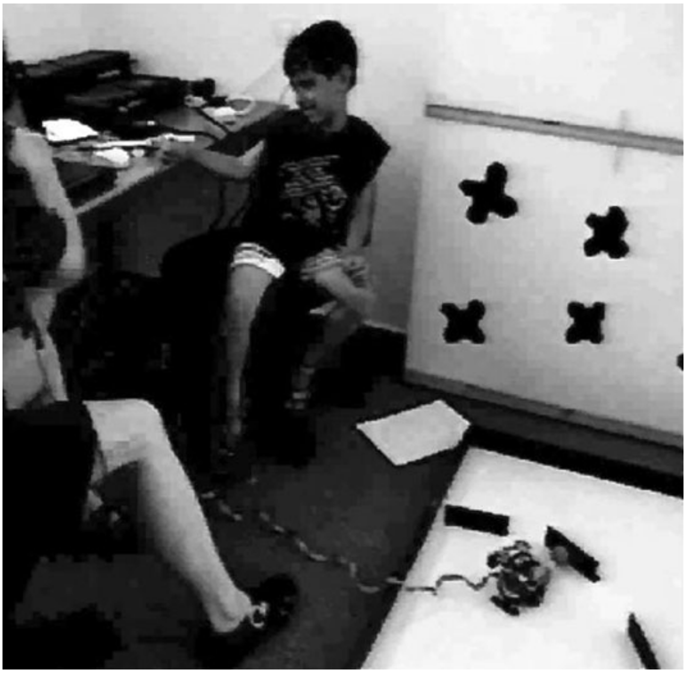

Fig. 4 Omer at the computer programming the 'avoid the obstacles' task

behavior construction board (Fig. 5) displays the two conditions: having the touch sensor pressed or unpressed. The child pulls the robot's navigation arrows (forward, backward, turning left and right) into the behavior construction board, aligning them with the appropriate conditions and then presses the Go! Button to run the program. In this example, Omer has constructed the robot's behavior at the computer, so that upon hitting an object, the sensor gets pressed and the robot turns in one place to the left. When it is free of

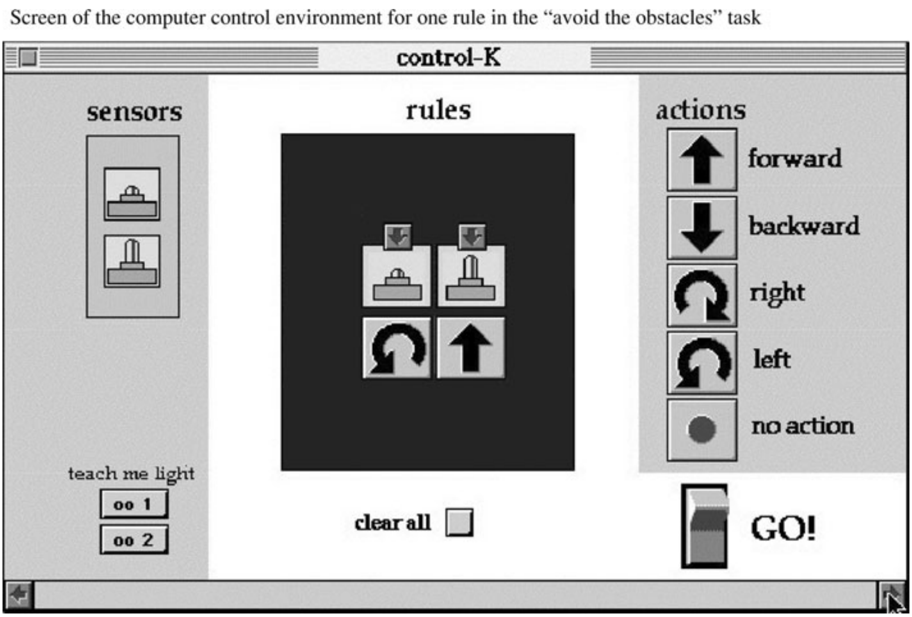

Fig. 5 Screen of the computer control environment for one rule in the "avoid the obstacles" task

obstacles, the sensor is not pressed, and the robot moves forward on the board. Putting this all together results in the robot roaming about the field, dodging barriers once hitting upon them and changing direction.

The tasks make use of the same robot in a variety of physical landscapes, and were designed as a progression of rule-base configurations. The operational definition of *rule-base configuration* is based on the number of pairs of condition-action couples ("If… Then…" couples). One *robot control rule* consists of a pair of related condition-action couples (If true … Then…; If false… Then…). The conditions are complementary, i.e. if one condition is "darkness", then the other is "light". The tasks progress through a range of increasing difficulty: half a rule (one condition-action couple), complete rule (two complementary condition-action couples), two independent rules and two interrelated rules, both made up of two pairs of condition-action couples.

The study lasted five 30–45 min sessions, spaced about one week apart. The children worked and were interviewed individually in a small room off the teachers' lounge. All sessions were videotaped. Adult intervention was offered in the form of prompts and decomposing interventions, e.g., asking about conditions and actions, which are not noted in the child's explanation.

Additional information can be found in previous publications (Levy and Mioduser 2008; Mioduser et al. 2009).

4 Findings and Discussion

To answer our main question on how children approach the adaptive robots' behavior-construction tasks, we analyzed and collected data concerning three main issues. These issues and their subtopics are presented.

Issue 1	Children's modes of engagement while solving the tasks.
	Participatory investigations.
	Anticipatory construction.
Issue 2	Evolving abilities and understandings along the process.
	Transition to the use of formal language.
	Coordination of spatial perspectives.
	Relating particular rules to general behaviors.
Issue 3	Strategies for coping with complexity.
	Pruning.
	Fusing.

Different from the reports in previous papers in which we have used more quantifiable signifiers of the children's activity and learning, in this paper several vignettes are used to illustrate the characteristics of the construction and learning processes.

4.1 Issue 1: Children's Modes of Engagement While Solving the Tasks

The focus in the first issue is on two main aspects of the children's work: their investigations of the characteristics of the tasks and their ability to plan the solutions (the appropriate robots' behavior rules).

4.1.1 Participatory Investigations

One unique affordance of the physical robot with respect to computational objects "living" only within the computer screen is the possibility to *participate* in the simulated system—through direct bodily interactions with the behaving object. For example, the children block the robot's way with their hands and shine a light on it from various angles along its route. In a sense, both child and robot can be viewed as two agents in a multi-agent system. In such interactions, the child's role shifts from designer and observer to that of participant. One may view the child, in Ackermann's words as both "diving in" into the system and "stepping out, being both part of it and standing apart" (Ackermann 1996). This shift in roles affords multiple views of the system and possibly leads to the construction of meaningful relations between these distinct perspectives.

These playful investigations bridge the concrete and abstract aspects of the system. In a highly interesting proposal of a programming environment for young children, Montemayor and other researchers' (e.g., in Montemayor et al. 2001; Wyeth and Purchase 2000) left out the computer (and the screen as programming environment) to focus completely on the physical reality. Children's work started by telling a story, then they made physical objects—the "actors" in the story—with attached sensors and actuators, and finally programmed the interaction rules among them. The complete programming experience takes place by manipulating and interacting with the physical components of the story being told. In our experience in contrast, we emphasize the co-occurrence of the programming experience at both the physical and the symbolic levels, or both the symbolic and the bodily interactions with the programmed system. Children's activity oscillates between understanding-by-interacting with the physical robot and reflecting-and-making with the symbolic representation of the rules governing its behavior. Physical participatory explorations—such as "scaring" the robot with a flashlight or blocking its path and watching it swerve off (see Fig. 6)—pull the child's attention to the components of the robot's actions and the relevant features of the environment. Their association in turn, helps focus on the relationship

Fig. 6 Playful investigations of the robot's behavior

between the changes in the environment and the robot's actions—in short, its *rules*. When programming the robot, the child's thinking bridges from rules to behaviors. When playing with the robot, these same behaviors are explored and decomposed back to their rules. An excerpt from Mali's first session demonstrates such participatory investigations.

A typical construction session is presented through Mali's playful explorations of her succession of constructions. Once constructing the requested behavior, she explores several variations on a theme. As we will show, constructions and their subsequent explorations involve physically moving between the computer and the robot's arena, from an observer's view to that interacting with the robot. This episode (5 min and 29 s) takes place at the end of a session that involved two prior activities:

> At first, Mali learned the robot's actions, by clicking arrows to operate it as with a remote-controlled car. Then, she participated in an interview regarding a demonstrated robot's behavior. During this interview, she used a flashlight to lead the robot out of a dark cave (by the "if-light → then-forward" rule). This same task was then used to explain the behavior construction interface mode, which is now set up for the first level: half a rule, or one condition-action pair. This is her first construction session with rules.
>
> Mali constructs the robot's behavior three times during this episode, each time more independently. In the third round, she quickly programs the robot on her own while the interviewer has her back turned for a moment, and proceeds to play with it.
>
> Mali is now constructing the following robot behavior: "keep away from the light". "Light" is a flashlight moving in at the robot's protruding light sensor. She designs three different programs. The first construction relates to the task, the second and third constructions are further explorations into the workings of the system.
>
> The focal part begins with the interviewer demonstrating and explaining the program used for the robot in the cave:
>
> **Interviewer**: *Now you know what I did here? There's a box here* [points to behavior construction board on interface] *where you can put one of these arrows* [points to arrow icons that can be dragged into the box].
>
> The task is then presented:
>
> **Interviewer**: *You can to teach him how to run away from the light. You will need to decide which of the four arrows here you need to put here, so that it will run away from the light. You will decide for the robot, you will teach it what to do.*

Mali: *Yes.*

Mali is now at the computer. After a reminder prompt (she has done this before), Mali clicks on an icon that teaches the robot "what is light". The interviewer then reminds Mali:

Interviewer: *When it sees the light, what should it do?*

Mali proceeds to pull a "turn" action icon into the box. Her first program is the following: "If you see light in front of you, turn". She runs the program and moves down to the floor with a flashlight. She circles the robot, placing the flashlight at different positions. She provides the "environment trigger", and the resulting behavior is "shyness". When the flashlight is placed in front of the robot; it slightly turns to the side, keeping away from the light and then stops. The motion is incremental, as the robot makes small interrupted steps. As soon as the flashlight has been avoided, the robot stops. Mali places the flashlight in front of the robot 17 times during 42 s, each one resulting in a small turn. Mali is pleased; she likes this result. While the robot is still moving, she proposes another idea.

Interviewer: *What is the robot doing? Is that what you had planned?*

Mali: *Or we can continue like this, forward* [refers to the robot's motion]

Interviewer: *Do you want to try something else?* [Mali stands up and moves to the computer]

Mali: *Yes, that he [the robot] will go forward, straight.* [Holding the flashlight, she uses it to point at the behavior construction interface and gestures a big motion forth away from her body towards the interface]

"Going forward, straight" is both the robot's behavior as seen in her articulation and the program she will program on the interface as seen by her gesture pointing to the computer. She is using her own body as referent for the robot's motion, gesturing its direction of motion—but targeting it at the computer screen, possibly creating a mental bridge between representation and the represented.

Mali proceeds to construct the robot's behavior and then operates it; then goes back down to the robot. She sees that its behavior has transformed. From a shy robot it has turned into an aggressor! Each time the flashlight is placed in front of it, the robot flies towards the light. This motion is continuous—the "if light → go forth" rule is activated continuously, until the flashlight is removed. She spends about a minute exploring its behavior, moving the flashlight in and out of the sensor's range. Mali smiles to herself; she is satisfied.

Later on, the interviewer latches onto Mali's explorative spirit and asks: "*What do you say, what does it do if you put a "backwards" arrow*?" Mali simulates the robot's behavior before programming it and says: "*He all the time goes backwards*". She describes the motion as continuous. Contrary to the first "light-turn" rule, and similar to the second "light-forward" rule—the robot moves away from the flashlight in continuous motion, stopping once the light is too dim to be above the programmed threshold. We later harbor on this point—single versus recurring events. As we will show, transitioning from the first to the latter was not always so fluid. With no assistance, she proceeds to construct the robot's behavior, operates it and uses the flashlight to interact with it for 40 s. The session ends at this point and Mali goes back to class.

In this episode, we see Mali's fluid motion between testing out new ideas, predicting the robot's behavior, and exploring it through participatory interaction. Within this short episode, she programmed the robot with rules for the first time and tried out three programs. Almost half the time was spent playing with the robot, manipulating its interactions with the environment by moving the flashlight in and out of its sensor's range. An important feature of any robot behavior construction environment is the possibility to move back and forth, and relate between the abstract a-temporal control rules on the computer, and their spatiotemporal enaction in the physical robot's behavior. This aspect of the learning environment supports the creation of connections between the underlying program and the system's emergent behavior.

We can see the formation of such associations in Mali's prediction of the robot's behavior for the third program. After experiencing the short single-event enactment of her first program and the continuous recurring actions resulting from her second program, she readily predicts the continuous action that will take place with her last program. We can see the facility with which Mali moves between the two worlds: the computer rule-based program and the robot's behaviors. We also note the ease with which Mali learns to work with the RoboGan interface. She quickly focuses on the robot and interacts with its underlying logic and behaviors, rather than upon learning the behavior construction language. Later activities will add complexity to the tasks and interface representations, but will preserve their structure. Mali's playfulness and interactive approach to the robot and its mind are maintained as well.

4.1.2 Anticipatory Construction

In this section, the children's construction activity is portrayed as preceded by mental simulation of the system. Rather than forming arbitrary programs, testing and then refining, their very first programs reflect anticipation of particular outcomes. From the very first, the children construct the robot's behavior with reasonable rules. While not always task-appropriate, they are based on sensible relations between the target behavior and the component rules. In the previous vignette, we have seen that Mali predicted the robot's behavior before running the program. Even in a more complex task, after programming the robot with two rules (four condition-action pairs: a bee-robot that buzzes when it finds flowers and avoid rocks), she responds to the interviewer's question about a situation she has not encountered with a detailed simulation of a string of events:

> **Interviewer**: *What happens if it also goes into a rock and also finds a flower?*
> **Mali**: *So she will also buzz, and will also press* [the touch sensor]. *And then she will turn and then again she'll buzz and then she'll go straight.*

In the following vignette we describe a short excerpt from Noga's first rule-programming session, in which she imagines the robot's behavior, creating a fully detailed simulation, even before running the program.

> Noga is engaged in teaching the robot to run away from the flashlight. She has just programmed the robot to turn when it sees light. Before operating the robot, Noga describes its behavior: "*… it turns first, and that he won't want to see the light all the time… and he will always look to the side that he has dark* [singsong intonation], *dark all the time! Dark, dark, dark.*" The richness of this description stands in stark contrast to the simple "if-light → then-turn" rule. The bridge between the two is her imagination of the interaction between the robot and the light. When she runs her program, she is delighted, moves the flashlight around the robot, watching its operation as it edges away from her hand, facing the dark as she had predicted.

Noga has a clear vision of the robot's operation and how it evolves from a simple rule. Based on this observation and others, as well as the small number of debugging loops, we propose that the children run a mental simulation of the rules as they interact with the robot and its environment. They can envision how these interact and emerge into robot behaviors, and how they play out as a temporal sequence of actions and interaction in space, even prior to its operation. Later, when they engage with the more complex tasks, they begin with partial programs, reflecting the limits of what they can mentally simulate.

4.2 Issue 2: Evolving Abilities and Understandings

The RoboGan interface and the navigating robot, as does any external manipulative environment, requires the user to adapt to its own way of perceiving the world and deciding how to act upon it. Each representation of a system structures in particular ways a selection of objects and their features, the kinds of actions it allows and how they relate to each other. This section points to three transitions the children went through during the series of behavior construction tasks, transitions that importantly impacted their growing understanding of the system: evolving abilities to use the *behavior construction language*, to coordinate among *spatial perspectives* and to understand how *rules* and *behaviors* relate.

4.2.1 Transition to Using Formal Language

In the study, the children were presented with a new language and a new form of notation that they were asked to understand and then use to construct with in the world. We examine these *language adaptations* among the children as a reflection of their growing understanding of the environment. These shifts may be central to comprehending the robot's mind, and how the language it knows can be used to manipulate and design an emergent system.

However, we begin with a word of caution with regards to the extent to which what a child *says* reflects her *understanding-in-action*. Shiri's case has shown us how verbal articulations may not always signify mindful reasoning by the children, reasoning that can be seen in the form of action upon and interaction with computational and physical objects. Shiri learned the behavior construction language quickly and used it to design the robot's behaviors, despite a very limited vocabulary and language proficiency. In words, she cannot show us her unique competence.

Shiri seems rather young for her age; she is slight and small, very shy and speaks softly in sentences that contain no more than two words, usually one. The words she uses are generic with meanings that do not mirror nor provide access to her precise and able reasoning: in most interviews, when asked what the robot is doing she says "going" or "walking". Only upon asking her pointedly about specific actions or environmental features does she answer in one or two words: "to turn", "straight", "gets pressed", "it gets stuck", and "needs to go forward". Many of her communications are in gesture—nodding, pointing or moving her hands to imitate the robot's actions. In fact, in coding for the complexity in her use of rules, she was consistently at the bottom of the group, describing no more than one condition-action pair.

In contrast to her hesitant and halted communications, her approach to the computer is immediate and confident, as she independently programs even the most complex of tasks. In our coding for an adult's support, she usually did not need more than very light prompting—framing the question, sometimes rewording; yet she did not require assistance in decomposing the problem. Noteworthy in her programming is that she laid out a full program from the start and then proceeded to debug it by replacing parts, rather than gradually building up from a partial program. Together with one other girl, Noga, they are the only ones that could conceive of the whole program from the start and independently decompose it in themselves, backtracking on the solution path to replace the problematic components. While Noga's exuberant articulations reflect a deep understanding of how the rules play out as the robot moves about the environment (see Noga's vignette in the "anticipatory constructions" section above), Shiri may not be recognized for her unique reasoning abilities as a result of her lower proficiency in language and communication. Indeed, it is possible that providing children like Shiri with such action-based environments that stimulate high-level reasoning may be more suitable for drawing upon her strengths and helping her develop the language tools to express and perhaps, in turn, transform, these existent understandings.

We now turn to transitions we have seen in how the children come to understand key language components of the robot behavior construction environment. In some cases, this transition is central as we can see in the following interaction.

> Tim is asked to program the robot with rules for the first time and create a light-escaping robot.
> **Interviewer:** *How do you tell the robot what to do?*
> **Tim**: *In Hebrew*
> This is the first step towards understanding the interface as a language in itself. Later, Tim programs the robot to move forward upon seeing light, resulting in the robot following the flashlight he holds out in front of the sensor.
> **Tim** (is troubled and complains): *He doesn't run away from the light*
> **Interviewer** (guides him back to the behavior construction board): *What did you write on the computer*?
> **Tim**: *Forward* (meaning the arrow language for describing the robot's component actions in navigating space).
> However, when the interviewer prompts and guides him in reformulating the rule "If light…?" then Tim replies "*An escape arrow*." Here we see a mélange of function and action, as the navigation arrow (turn) is interpreted in terms of its function in the particular environment (escape).

The robot's navigation through space is made up of increments of action that are defined as geometrical motions: moving forward or backwards in a straight line; turning on the spot to the left or to the right (similar to primitives in other such environments, e.g. Logo, Papert 1980/1993). One of the first challenges the children met was transitioning from everyday language such as "going about" and "escaping" to the formal terms. We demonstrate this transition when Noga shifts in her understanding from "turn" in everyday language to its geometrical meaning in the controlled system. The term "Lehistovev" in Hebrew has several meanings. One colloquial and frequent use of the word describes meandering, moving about with no particular aim. For example, one could "lehistovev" in the market, viewing the produce but not necessarily buying anything. The second meaning is that used in the behavior construction environment—the geometrical turn on one spot (think of a top). We will demonstrate how these two meanings are first conflated and then disentangled from each other.

In programming the robot to roam a field and avoid obstacles (touch sensor facing forward), Noga creates two programs. The first of these is: "if (the touch sensor is) unpressed, turn; if pressed, go backwards". When Noga operates the robot, it turns like a top, never exploring the terrain. Noga observes the robot for a few moments, turns to the interviewer, smiles and then slowly says: “It’s all the time turning. He does not know where.” This description combines actions (turning) and an intentional state (not knowing where to go). These meanings are seemingly contradictory—on one hand, “turning” should have gotten him roaming about, but it’s actually “all the time turning”. Resolving the two comes from thwarted intentions—perhaps if the robot knew where to go, it would not remain in place spinning like a top. Thus, her explanation bridges between the two kinds of turning. Right after this, Noga goes to the computer, and independently changes the program—replacing the “turn” arrows with a “straight” one. Upon operating the robot, she calls out “How much fun… it’s working! I succeeded!”

Noga re-described the robot’s “moving about” behavior as “straight” and “turn”. This followed her observation of running the first program and an attempt at resolving the two meanings of “turn” by humorously explaining the gap as the robot’s frustrated state of mind. Once the two meanings are aligned they are also compared and distinguished, and Noga quickly shifts into the interface geometry language.

From an additional perspective, we may refer to the transition from natural to interface language as transition among levels of description of the target behavior: from telling the story (using natural language) of what should happen, to defining (in interface language) how this should happen. In other words: from a description of the target behavior, to a definition of specifications or requirements, to their translation into the interface language. From the above vignettes we learn about children’s growing capability to traverse back and forth across the levels while solving a programming task.

4.2.2 Coordination of Spatial Perspectives

In order to program the robot with working behaviors, the children need to be able to see the world through its eyes as it moves through this world, a “robocentric” view. They also need to interpret the vertical layout of the arrow icons on the screen as horizontal. These multiple coordinations pose a significant challenge. Body syntonic learning has been described by Papert as learning that engages a child’s knowledge of his own body and experiences in the world (Papert 1980/1993, p. 63). In the following episode, we show how body syntonic learning aids Tim in making this coordination.

> Tim is programming the robot to roam the field and escape obstacles. We focus on the shift from his second to third program, and show how “playing robot” helps him relate between his program and his own observer view, disentangling the problem at hand. So far, Tim has programmed the robot to move backwards upon hitting an obstacle—this has left the robot in oscillatory motion at the barriers moving backward and forward (sensor pressed >>> backwards; sensor released >>> forward; and so on, back and forth). The robot has entered an endless “loop” until the interviewer halts the program.
>
> An important shift in Tim’s understanding of the task takes place when the interviewer suggests that he “play robot” (see Fig. 7). He goes into the field and acts out the robot in two stages. First, he positions himself, head down, in front of a row of obstacles and crashes his head in, knocking them all down. This clarifies the meaning of “hitting something” or ramming into an object. Next, when the interviewer suggests he try to avoid the obstacles, he comes up with a solution: circle the whole set of obstacles.
>
> **Interviewer**: *Ah, what are you doing? Are you turning?*
>
> **Tim**: *Yes* [gestures a broad circle around the obstacles] *Like this*

Fig. 7 Tim is playing robot, ramming his head into the obstacles in the *left figure* and *circling* the obstacles on the *right*

> This is the critical moment. In moving from action to gesture and articulation, a new idea forms. The previous idea of circling the barriers triggers another kind of turning—the geometrical turning like a top. Tim jumps up, goes to the computer and immediately changes the program to create this idea—deflect the robot from the obstacles by turning like a top. Upon reaching an obstacle, the robot turns away and changes direction, heading for new horizons in the field.

Tim's third program resolves the robot's endless loop, when it gets stuck upon the first barrier. These changes were supported via observing the robot's behaviors, by the interviewer's assistance in decomposing the problem, and more specifically by taking the robot's perspective and coordinating it with the program and the observer perspective.

4.2.3 Relating Particular Rules to General Behaviors

One of the difficulties in simulating emergent behaviors resides in the capability to relate a perceived one-time event to general recurring behaviors. It is easier to imagine an event as something that begins, happens and ends, rather than one that "happens forever" (such as gas particles moving and bouncing about in a container), or "is a recurrent part of a general behavior pattern" (such as avoiding obstacles). The rules that the children created pertained to different states the robot could be in, with some states being more salient than others. Falling off a bridge or hitting an obstacle was more interesting than mundane actions such as "moving". As a result, an event that was related to these salient states was thought of as separate from the preceding and ensuing events, especially in the earlier sessions. In some cases, this does not necessarily matter. However, in this study's construction tasks, such separation could lead to curious and unexpected behaviors.

For example, moving backwards upon hitting a barrier is a successful behavior for avoiding obstacles if it were a one-time event. However, repeated activation of this behavior results in the robot hanging in perpetual oscillation, repeatedly knocking at the wall, stuck at the obstacle (as in the above vignette on Tim's first program). We have seen four of the six children program the robot with a rule that emerged into this behavior in the second and third programming tasks, both of which involved escaping obstacles.

Gradually, the ability to relate single and self-contained behavioral units (e.g., a rule) to the expected/desired overall behavior evolved. In a previous paper (Mioduser et al. 2009) we described the observed process: “the children abstract rules from the robot’s behavior in the following way: (a) by observing the robot’s sequence of moves and actions in the landscape (episodes), with a primary focus on the robot’s actions, rather than on the environmental conditions, in a ‘‘robo-centric’’ approach; (b) by seeking repeating routines in the robot’s actions set off by particular features/props of the terrain (scripts), with the spatial conditions gaining some importance, partially decentering from the robot’s actions; and (c) distilling atemporal relationships between the environmental conditions and the robot actions (rules), when comparable importance is attributed to both conditions and actions in explaining the robot’s behavior, completing the decentering from the robot” (pp. 30). Thus, recognizing recurring regularities, facing the need to “clean” localized routines to make these applicable in diverse situations and eventually more generic/general, detaching the formal description of the behaviors from a specific chronology or time dependence and finally formulating the behavior in terms of condition/action couples are the understandings leading from specific behaviors to general rules of behavior.

These evolving abilities are of special importance for coping with the increasing complexity of the tasks, an aspect discussed in the next section.

4.3 Issue 3: Evolving Strategies for Coping with Complexity

How do the children conceptualize the robot’s behavior when its mind consists of several component rules? More specifically, how do they reframe the rules, when their complexity is beyond what they can describe?

There is a vast research literature reporting on students’ (including young students) difficulties while constructing computer programs (e.g., Papert 1980, 1993; Sleeman et al. 1988; Morgado 2005). Several categories of difficulties are relevant to our study, specifically concerning the complexity of the behaviors to be programmed. Although a detailed presentation is beyond the scope of this paper, we will briefly refer to these difficulties, and their connection with this study’s findings.

A key issue are the challenges faced in two closely interrelated stages of the process: “decomposing the problem” and “composing the solution” for the problem. Difficulties raise when either there is no systematic decomposition/composition process (the problem solution proceeds haphazardly), or the two requirements are treated separately—first a problem is analyzed and decomposed then pieces of code or program segments are assembled for each problem-part. Furthermore, another source of difficulties derive from the potential conflict between the perception of the problem and its logical solution in the physical world, and the way this solution ought to be implemented (often times in counter intuitive formalization) in the symbolic realm. Closely connected is the ability to conceive and envision the “conceptual machine” behind the program, how it controls behavior: how the program is run, what’s first, for how long, when an action will stop, what to do about situations not defined in the program (but intuitively obvious for a human), and many similar questions.

The behavior construction environment in this study, by enabling the continuous shift back and forth from enacted to symbolic problem-analysis and solution (programming), supported children’s development of strategies integrating both decomposition and composition process in gradual progression until the solution is reached (the two main strategies are presented below). In addition, as the programming entities are close in their representation to the represented physical actions, the distance between the physical/concrete behavior and its formalization is drastically reduced. Finally, the above properties of the environment (shifts-allowance and closeness between the represented and its representation) situates the “conceptual machine” or control mechanisms in a very concrete plane making evident its connection with the actual behavior of the physical system.

What specific strategies did the children develop to solve the tasks within this supporting environment? How did they overcome difficulties inherent in the decomposing/composing process, in bridging between the physical behavior of the robot and its symbolic representation, or in understanding the structure and flow of the problem solution? We will focus in the following on one particular problematic situation: coping with the tasks’ complexity. We found that the children programmed the robots with up to four condition-action pairs, yet their articulations usually capped at two. What is the meaning of this gap? Do the children simply reduce the number of rules in the problem and then gradually add onto them? Or, do they reformulate the rule structure to fit into their existing cognitive structures? We have found both.

The children reduced the amount of information when describing their designed systems to fit their cognitive boundaries using two strategies, strategies that we name “pruning” and “fusing”. The “pruning” strategy focuses on only some of the rules and ignores others, metaphorically cutting off some of the logical branches. The “fusing” strategy takes some of the rules and merges them into one entity—joins a number of conditions into one and associates them with a single action, necessitating another form of parsing the system. These two strategies are demonstrated in the following examples.

4.3.1 Pruning

We have captured “pruning” events in seven sessions out of the total of 17 sessions (these sessions relate to the last three tasks, in which two to four condition-action pairs are necessary to describe the robot’s behavior; in one session, information was lost). Five of the six children used this strategy.

As an example we discuss Mali’s work on the third task: making the robot act like a bee, moving about the field, avoiding rocks and calling out (buzzing) when it finds flowers (black cutouts on the white background of the field). The task was very easy for some children and most difficult for others.

> After successfully constructing this behavior in two rounds of programming, Mali is asked to describe what the robot is doing. Without further prompts Mali describes one rule of the two: “*If she* [the bee] *sees a flower, buzz to everybody*.” Two prompts were necessary to get her talking of how the robot avoided the “rocks” or obstacles: “*What did we define here… something related to rocks?*” after which she describes the rest of the robot’s behavior: “*If she goes in* [into the rock], *to turn.*” This is still only one section of the rule, as we see her prune the rule leaving only its more salient part. To help her complete the rule, a further prompt is necessary: “*If not?*” upon which she adds: “*To continue straight.*” One may view the structure of her first description as a doubly truncated tree (Fig. 8), where only part of the space is described even when scaffolded by an adult. It is interesting to note that while the first rule is described in functional terms (call your friends when you find a flower) the second rule, which was first ignored, is then described in mechanistic language—“to turn”—rather than “escaping” obstacles or rocks, the component action is described. It is possible that one reason for the full design being outside of her grasp is that she has not fully integrated its ulterior motives, or functions.

In the next section we describe the “fusing” strategy, in which a reformulation of the functions helps the children reduce complexity while retaining a complete map.

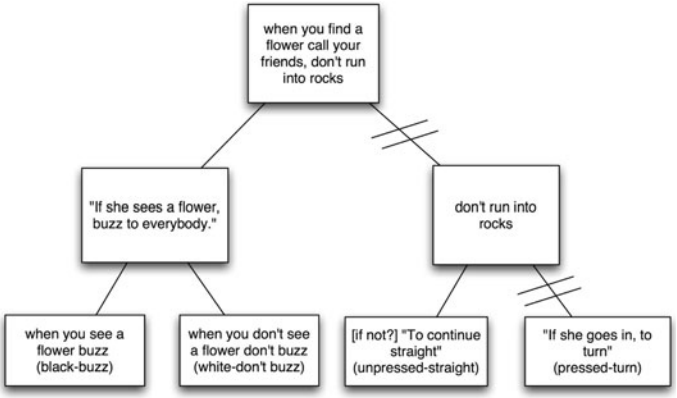

Fig. 8 Mali's "pruning" strategy for the "flight of the bee" task

4.3.2 Fusing

Fusing keeps a full picture of the problem but reduces its resolution, while reformulating how the system is segmented in terms of intermediate functions. Several conditions are compounded into one condition to simplify the problem. "Fusing" events were observed in five sessions out of the possible 17 sessions. For example, Homer describes the four rules he has programmed in terms of two functions; Mali describes them in terms of both functions and mechanistic rules.

Homer has constructed the behavior for a bridge-crossing robot. The bridge is a black curving path over a white background "pond", and the robot has two light sensors pointing downwards. He defined the rules: if the sensors' inputs are 'black-black' => go forward; if 'black-white' >>> turn right; if 'white-black' >>> turn left; if 'white-white' >>> go back. Upon successfully completing the task, he describes the robot's behavior: "*To cross the bridge, not to fall in the pond*". His description is in terms of functions, rather than actions and relates to two conditions rather than the four he had programmed with—bridge or pond. In fact, what are missing are the edge conditions—when the robot is about to leave the bridge on each side. When asked how he did it, he responds: "*I did the arrows well*" and no further explanation could be elicited.

When Mali describes the bridge-crossing robot, she says: "*That he's… when he's in the white he goes to the black, and when he's on the black he continues going*." Similar to Homer, she is referring to the bridge and the water, but in terms of their colors, the feature the robot can detect with its sensors. She uses black to refer to the bridge and white to the water around it. Rather than using a functional explanation (cross, not fall) she describes actions "go to black" and "continues going", even though they are quite different from the rules she has defined to keep the robot on the bridge (keep from falling off the right and left sides, upon reaching the end stop). Figure 9 describes the structure of Mali's explanation of the robot's behavior with respect to the rules she had used to design it. She has had to reframe the problem and invent new functions. For example, there is no single action of "getting back on the bridge" (or onto black) in the behavior construction environment. She aggregates these three rules by introducing an intermediate level between crossing the bridge safely, and the four detailed rules tying conditions and actions. Homer uses a

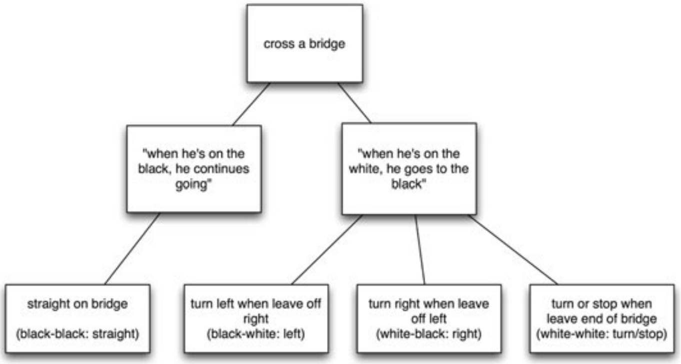

Fig. 9 Mali's "fusing" strategy for the bridge-crossing task

breakdown of functions as an intermediate level to describe the system. Both inventions (a more detailed functional description or a new set of rules) have a similar structure—one full rule, or two condition-action pairs, the same limit we have found regarding children's spontaneous reasoning about events, systems and phenomena (Mioduser et al. 2009).

5 Conclusion

In this study, we set out seeking an understanding of how young children master, build sense and make sense of the "intelligent" rule-based behaviors of an autonomous robot. Young children are surprisingly able to construct such behaviors with rules and can articulate their knowledge up to a certain level of complexity. We have focused on the processes by which they come to these understandings. In exploring these processes we focused on three issues: their modes of engagement, how their abilities and understandings evolved and strategies they used to cope with complexity that is beyond what they could explain in words. In this study, we take a broad view of such learning, attempting to provide a comprehensive map of the variety of paths and strategies in overcoming hurdles along the way.

One of the vital findings highlights the specific importance of the learning environment—it's being both physical and computational. This affordance encourages children to engage with the robot in playful explorations of its behavior, as their role shifts from constructor and observer to participant in the system. We have noted the intense explorations children conduct in this play and its prominence in supporting their understanding of how the rules are related to behaviors. Moreover, these child-robot investigations were found to help children separate and coordinate spatial perspectives, that of the robot and that of observer.

Another important feature is the planful nature of the children's constructions. Rather than arbitrarily programming the robot and testing out the resultant behaviors, we have found that they could simulate the robot's behavior even before the first enactment or "run".

In mastering the robot's behavior, several transitions were noted. We have highlighted the process of learning a new language, the formal language used in the behavior constructions system. The children's coming to coordinate the three spatial perspectives—robot-based, observer and interface inscriptions—was evident in their debugging cycles as well as their body-based explorations of the system. A third transition involved relating particular locally activated rules to a general pattern of behavior over time.

Finally, we were curious as to the one-rule gap between what the children could do and what they could say. In analyzing their articulations, we came to realize how the children reparsed the system, using one or two strategies—pruning and fusing—to reduce the amount of information in the system. We observed the children as they created an intermediate level of functions or rules. In pruning, these functions relate to part of the robot's behavior or rule structure. In fusing, a number of rules are coalesced into one rule.

We have characterized the process by which young children come to understand sophisticated robot behaviors via construction. In this endeavor, we wish to contribute both to theoretical and practical knowledge regarding learning processes geared at understanding intelligent devices. As such systems proliferate in our everyday surroundings, and as the "innards" of such systems have become available for inspection, exploration and manipulation for younger and younger ages—we hope to advance the support of such learning in educational settings.

Future research is suggested into several aspects of these findings. The small sample in the study begs testing with additional groups of children, possibly discovering additional aspects of children's grappling with and learning through constructing emergent behaviors with physical devices. A greater focus on emergence as a central concept could prove beneficial for early and initial learning about complex systems. The study suggests directions for the development of several forms of support for learning, that may be designed and tested: the role of the mediator in assisting the variety of transitions we have observed, as well as scaffolding within the design of the computation environment.

Acknowledgments We gratefully thank Dr. Vadim Talis, who collaborated with us in designing the RoboGan environment and in conducting the research with the children.

Appendix

See Table 1.

Table 1 Description and construction tasks

Rule-base configuration	Task	Description	Construction
Half a rule	Behavior	*…coming, coming out!* The robot is cowering inside a dark cave. A flashlight is placed above its nose and it gingerly follows it out of the cave. Once reaching the entrance, it struts out independently, disregarding the flashlight, its path tracing a straight line	*Scaredy-cat* Teach the robot to be afraid of the flashlight The children may choose to have the robot avert its "face" when a flashlight is placed in front of it Alternatively, they can have the robot retreat upon confronting the flashlight
	Environment	Dark cave, lighted surroundings, a flashlight	A flashlight
	Robot structure	A light sensor facing upwards, distinguishes light from dark	A light sensor is facing upwards, distinguishes the luminosity of the flashlight, from that of the environment

Table 1 continued

Rule-base configuration	Task	Description	Construction
	Rules	When the light sensor sees light, go forward When the light sensor sees dark, do not move	When the light sensor sees dark, stay put (automatically programmed) When the light sensor sees light, either turn (avert) or go backwards (retreat)
One rule	Behavior	*Guarding an island* The robot is placed upon an island. The robot moves across the island until it reaches its edge. It then travels around the perimeter of the island, its "nose" sniffing and following the island's rim	*Seeking freedom* Program the robot so it can move freely in an obstacles field The robot roams about the field, ramming into obstacles and extricating itself, while changing its heading
	Environment	A light colored island (white paper) on the background of a dark-colored rug	A walled board, with several barriers scattered throughout
	Robot structure	A light sensor facing down, distinguishes light from dark	A touch sensor facing forwards, it is un-pressed until it reaches a wall and then becomes pressed
	Rules	When the light sensor sees light, go forward When the light sensor sees dark, turn to the left	When the touch sensor is pressed, turn to the left or to the right When the touch sensor is un-pressed, go forward
Two independent rules	Behavior	*Brightening dark holes, oops! trapped by a hat…* A hatless robot travels through a landscape splattered with dark spots, flashing its light when it reaches a dark spot. However, when a hat is placed on its head, it turns like a top	*The flight of the flower-seeking bee* The robot is now a bee. Teach the robot-bee fly through a field without getting trapped in the rocks. Help it find flowers and notify its friends of the discovery, so they can come along and enjoy them as well The bee-robot navigates a field, extracting itself when it hits a rock. When it finds flowers it calls out to its friends
	Environment	Dark spots are scattered through a light-colored terrain A hat	A light colored board is "planted" with dark flowers and several barriers/rocks are scattered about
	Robot structure	A touch sensor faces upwards, is depressed when a hat is placed on top of the robot A light sensor faces downwards, distinguishing dark from light	A touch sensor faces forward, and is depressed when the robot hits a barrier A light sensor faces downwards, distinguishing dark from light
	Rules	When the touch sensor is pressed, turn left When it is un-pressed, go straight When the light sensor sees dark, flash. When the light sensor sees light, do not flash	When the touch sensor is pressed, turn left or right. When it is un-pressed, go straight When the light sensor sees dark, buzz. When the light sensor sees light, do not buzz

Table 1 continued

Rule-base configuration	Task	Description	Construction
Two interrelated rules	Behavior	*The cat in the hat likes black* The robot navigates across a large checkerboard. When the robot wears a hat, it searches for the black squares, homing in on them. It quickly moves across the white squares, turning for a while on a black square, before leaving it and homing in on the next black square When the robot is not wearing a hat, it moves across the board in a straight line, irrespective of the colors below	*Crossing a long and winding bridge* Program the robot to traverse a winding bridge, without falling off into the turbulent water flowing below. The robot starts out at one end of the bridge, tracing a jagged route as it heads forward, reaches the edges of the bridge and turns away. When it reaches the end of the bridge, it can stop, continue straight or turn around
	Environment	Large checkerboard made up of black and white squares. A hat	A black winding strip against a white background
	Robot structure	A touch sensor faces upwards, and is depressed when a hat is placed on top of the robot A light sensor faces downwards, distinguishing dark from light	Two light sensors are facing down, side-by-side They distinguish light from dark
	Rules	When the touch sensor is depressed and the light sensor sees dark or light, move forward When the touch sensor is un-pressed, and when the light sensor sees black, move backwards When the touch sensor is un-pressed and the light sensor sees light, turn to the right	When both light sensors see black, go forward When the right light sensor sees black and the left light sensor sees white, turn to the right When the right light sensor sees white and the left light sensor sees white, turn to the left When both light sensors see white, then either stop, go straight, turn right or left

References

Ackermann, E. (1996). Perspective-taking and object construction. In Y. Kafai & M. Resnick (Eds.), *Constuctionism in practice: Designing, thinking, and learning in a digital world* (pp. 25–37). Mahwah, New Jersey: Lawrence Erlbaum Associates.

Carr, M. (2000). Technological affordance, social practice and learning narratives in early childhood setting. *International Journal of Technology and Design Education, 10*, 61–79.

Fleer, M. (1999). The science of technology: Young children working technologically. *International Journal of Technology and Design Education, 9*, 269–291.

Kull, J. (1986). Learning and logo. In P. Campbell & G. Fein (Eds.), *Young children and microcomputers* (pp. 103–130). Englewood Cliffs, NJ: Prentice-Hall.

Kurland, D., & Pea, R. (1985). Children's mental models of recursive logo programs. *Journal of Educational Computing Research, 1*(2), 235–243.

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.

Linn, M., & Clancy, M. (1992). The case for case studies of programming problems. *Communications of the ACM, 35*(3), 121–132.

Mayer, R. (2004). Should there be a three-strikes rule against pure discovery learning?—The case for guided methods of instruction. *American Psychologist, 59*(1), 14–19.

McNerney, T. (2004). From turtles to tangible programming bricks. *Personal and Ubiquitous Computing, 8*(5), 326–337.

Mioduser, D., & Levy, S. T. (2010). Making sense by building sense (I): Kindergarten children's construction and understanding of adaptive robot behaviors (submitted).

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

Montemayor, J., Druin, A., Simms, S., Churaman, W., & D'Armour, A. (2001). Physical programming: designing tools for children to create physical interactive environments. CHI 2002, ACM conference on human factors in computing systems. *CHI Letters, 4*(1), 299–306.

Morgado, L. (2005). Framework for computer programming in preschool and kindergarten. Doctoral dissertation, Universidade de Trans-os-Montes e Alto Duro, retrieved at: http://www.scribd.com/doc/24041133/Framework-for-Computer-Programming-in-Preschool-and-Kindergarten#stats.

Morgado, L., Cruz, M., Bulas, G., & Kahn, K. (2003). Taking programming into kindergartens: Exploratory research activities using toontalk. In Cnotinfor Lda (Ed.), *Proceedings of the 9th European Logo Conference Eurologo 2003*, August 27–30, Porto, Portugal.

Morgado, L., Cruz, M., & Kahn, K. (2006). Radia Perlman—A pioneer of young children computer programming. In A. Méndez-Vilas, A. Solano Martín, J. A. Mesa González, & J. Mesa González (Eds.), *Current developments in technology-assisted education* (Vol. III). Badajoz, Spain: Formatex.

Papert, S. (1980, 1993). *Mindstorms: Children, computers, and powerful ideas, 1st edn*. Cambridge, MA: Basic Books.

Pea, R. (1987). Programming and problem-solving: Children's experiences with Logo. In T. O'Shea & E. Scanlon (Eds.), *Educational computing (an Open University reader)*. London: Wiley.

Perlman, R. (1974). *TORTIS—Toddler's own recursive turtle interpreter system. MIT AI Memo No. 311/ Logo Memo No. 9*. Cambridge, MA: MIT AI Lab.

Perlman, R. (1976). *Using computer technology to provide a creative learning environment for preschool children*. MIT AI Lab Memo No. 360/Logo Memo No. 24, Cambridge, MA: MIT AI Lab.

Raffle, H., Parkes, A., & Ishii, H. (2004). Topobo: A constructive assembly system with kinetic memory. In *Proceedings of the SIGCHI conference on human factors in computer systems*. Vienna, Austria: ACM.

Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V, Kramer, K., et al. (1998). Digital manipulatives: New toys to think with. In *Proceedings of the CHI'98 conference on human factors in computing systems*. Los Angeles, California: ACM.

Scweikardt, E., & Gross, M. (2006). roBlocks: A robotic construction kit for mathematics and science education. In *Proceedings of ICMI 06*. Alberta, Canada: ACM.

Singh, J. (1992). Cognitive effects of programming in Logo: A review of literature and synthesis of strategies for research. *Journal of Research on Computing in Education, 25*(1), 88–104.

Sleeman, D., Putnam, R., Baxter, J., & Kuspa, L. (1988). An introductory pascal class: A case study of students' errors. In R. Mayer (Ed.), *Teaching and learning computer programming* (pp. 237–256). Hilssdale, NJ: Erlbaum.

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

Wyeth, P., & Purchase, H. (2000). Programming without a computer: A new interface for children under eight. In *Proceedings of first australasian user interface conference*.

Zuckerman, O., Arida, S., & Resnick, M. (2005). Extending tangible interfaces for education: Digital montessori-inspired manipulatives. In *Proceedings of CHI 2005*. Portland, OR: ACM.