ToonTalk in Kindergartens: Field Notes

Leonel Morgado, Maria Gabriel Bulas Cruz, Ken Kahn

ToonTalk IN KINDERGARTENS: FIELD NOTES

LEONEL MORGADO, MARIA GABRIEL CRUZ, KEN KAHN

Department of Engineering, University of Trás-os-Montes e Alto Douro
Engenharias II, Quinta de Prados, 5000-199 Vila Real, PORTUGAL.
E-mail: leonelm@utad.pt

The emergence of a fully graphical, animated programming language, such as ToonTalk, prompted the question: is computer programming a viable educational activity with kindergarten-aged children? Since May 2000, we have been conducting ToonTalk sessions with 3- to 5-year old children, trying to determine what approaches are more successful, what activities are viable, what issues may arise in the process. From November 2001 to February 2002, the sessions took place regularly in a normal kindergarten room, where during each session 2 children would be playing with ToonTalk, the others would be conducting other kindergarten activities. This paper aims to present the experience acquired during these sessions.

1 Introduction

Computers have a fundamental presence on our everyday lives, not only as automated elements, reacting to sensors or situations, but as tools. Tools with a major ability: they can be adapted to completely different tasks, suiting the needs of their users. This adaptation, by which we refer to the common term "programming", allows the same tool to do tasks with as little in common as word processing and number crunching, information collection and entertainment, and so on.

But how many users can use this ability? Most simply use "applications", programs made by other people. Many applications allow for customization by the end users, but such skills elude most of them.

This amounts to a huge wasted potential. Among other motives, we may quote Smith & Cypher [5], referring to the gap between human communication and computer programming languages as the "Grand Canyon". Norman [3], quoted by them, proposed two ways of closing this gap: moving the users closer to the system (teaching traditional programming to humans – the traditional way to do things) or moving the system closer to the user (making computer programming resemble human communication). Smith & Cypher recall that traditional computer programming education is hardly a discipline for every user: it requires effort, dedication and persistence, just like learning a foreign tongue. And they conclude that computer programming should seek out ways of getting closer to human communication, making it possible for every human being to benefit from the computer tool's ability to be adjusted to different needs.

This approach is mentioned in an ACM paper from 1996 [2], where end-user programming is one of the strategic research and development areas pointed.

Seeing computer programming as a communication skill made us launch this consideration: in what ways could this skill be introduced in pre-school contexts? Modern pre-school education, rather than focusing on a curriculum, aims to give children basic skills, allowing them to more easily and fully develop a personal worldview and self-learning abilities. It seemed to us perfect sense to give children the possibility to discover and develop programming skills. Skills centred on learning how to state rules, how others (e.g., a computer) would follow them, and how different rules would interact. All these things seem to be useful, education-wise, even out of a computer context. Closer to the computing context, giving children the ability to understand that a computer can be tailored, rather than simply used, is an obvious advantage.

2 Preliminary issues

The preschoolers' limited ability to conceptualize rules and abstractions may be considered, from a broader perspective, the main issue at stake. But the most obvious hurdle for the introduction of computer programming concepts in preschool may be the absence of reading and writing skills. This is a technical computing hurdle (icons, logographic writing, are means of expressing the child programmer's intentions), but we felt that to reach better conclusions we needed a way to allow children to better express their desires.

Luckily however, we are not alone in the desire to bridge this gap. Albeit works dealing with preschool children and programming are limited [4], some methods and tools being used with older children seemed suitable for the initial research. Two tools drew our attention: StageCast Creator and ToonTalk [1, 5].

From the very start, ToonTalk seemed to us to be the most adequate tool for the target age group (3-5 year olds), for it employs larger control elements and requires a smaller number of mouse skills. Also, it's the only one of the two available in Portuguese. A more detailed discussion of this decision was presented at the seminar "Playground – novos ambientes de aprendizagem" [6].

ToonTalk is an implementation of a concurrent constraint programming language [7] as an animated cartoon, where children control an avatar moving in a city (metaphor for the entire computation). The city has houses (agents, actors, processes or objects), where robots (methods) can be programmed. The programming employs objects such as boxes (tuples, arrays, vectors or messages), scales (comparison tests), trucks (agent spawning), bombs (agent termination), notebooks (program storage), text and number pads (constants), and birds and nests (channel communication). Robots are programmed moving them with the mouse and performing the required actions (method actions), inside their thought bubbles. After training a robot the method preconditions are visible as the thought bubble of the robot.

3 The initial sessions: preparation

In May 2000, upon the release of the European Portuguese version of ToonTalk, three Vila Real (Portugal) kindergartens were willing to cooperate in our research: "As Árvores", "S. Pedro Parque" and "Araucária". In each kindergarten, two children were selected by the teachers, for weekly ToonTalk sessions, until the end of the school year, in June 2000 (three children in "As Árvores"). There were four children aged 4, three aged 5 (five boys and two girls).

These sessions focused mainly on: -

- determining if only 10- to 15-minute sessions were viable, or if the media allowed longer sessions;
- testing if children could use ToonTalk's generalization, given their limited abstraction skills;
- allowing for comparison of results yielded by a directed approach, against a coached-style approach.

The first issue was mainly for our organization and definition of goals: longer sessions allow for more complicated or elaborated activities than shorter ones.

The second issue was key for further ToonTalk experiments: children's inability to use generalization would severely compromise any options of rule programming for generic circumstances. Two different approaches on the generalization concept were used: **explanation** and **usefulness**.

The third issue was aimed at determining the strategy for the following year's sessions: **directed** (i.e., proposing activities and conducting children on how they might achieve them) or **coached** (letting children choose what they want to do and help them achieve it).

4 Initial activities and results: summary

On the aforementioned paper [6], we presented the activities, data collected and perceived results.

Briefly, these can be summarized as: -

- 20-minute, 30-minute and even 40-minute sessions were viable: the children weren't bored at all, as long as they could explore and experiment with new things;
- some use of generalization was achieved within ToonTalk, employing a "usefulness" approach;
- with the coached approach, children seemed to gain greater control over the ToonTalk objects and environment, but the programming activities were simpler, compared to the directed approach.

Directed sessions were based upon a simple activity: programming an "image-swapping" robot, i.e., one that takes two images and then exchanges their places, and some variations of this.

This is achieved by placing two boxes in the ground, thus combined into a two-hole box; different images are placed in each hole (we used a tree and a flower). A robot, upon receiving this box, floats into its thought bubble, where we can command it with the mouse. The robot then is made to pick the flower and drop it outside the box, then pick up the tree and drop it in the hole previously occupied by the flower. We conclude by making the robot pick up the flower and drop it in the hole previously occupied by the tree.

In the ***coached sessions***, the children could decide what they wanted to do, only sporadic suggestions were presented. This caused a larger focus on the interface tools, the most visible elements in ToonTalk. Birds and nests were also prime attention targets, since they provide a highly animated, amusing activity.

After the initial session, where these basic activities were explored, we introduced the concept of robot programming as "robot teaching". Their program choices were related to the tool manipulation activities: a 4-year boy wanted to make a robot to clean up the room, for instance (we suggested that instead he programmed a box-cleaning robot), although a 4-year girl did want to make a tree-chopping robot!

5 Attempts at using generalization: summary

In ToonTalk, if a robot is thinking of a specific picture, number, etc., its thoughts can be "vacuumed" with an animated vacuuming tool [1, 6], making it accept any object. A more restricted generalization, limited to objects of the same type, can be specified by using the same tool to "erase" instead of vacuum.

Two approaches were used to present this concept: **explanation-based**, under which this behaviour was both explained and demonstrated to children, and then further practiced by means of a theatre play; and **usefulness-based**, under which the children would program robots that were similar, but worked on different objects, such as one to swap trees and flowers, another to swap trucks and bombs, etc. The generalization behaviour was then introduced as a way to make robots less "picky".

Each approach was tried out with two children. The explanation approach didn't achieve much, not even after a theatre play was set up (a child as the robot, the other as the programmer). But the usefulness approach yielded better results, with two children being able to understand and explain the method [6].

6 Second year sessions: environment

For the second year, the coached approached was selected: given that children were enthusiastic about ToonTalk, we assumed this approach might provide greater insights on their progress and hurdles.

A room at the University campus was allocated for the weekly sessions with each group of children.

The project was presented to parents at two kindergartens, involved on the previous year. Afterwards, the parents organized themselves to drive the children from the kindergartens to the campus and back. In all, 22 children were involved: 19 were aged 4 or 5, and the activities were conducted in mixed groups of both ages and genders. Three of the children were 3-year olds, and those were in two specific groups.

The sessions were performed with 3 laptops, one for each pair of children, but they were connected to regular desktop keyboards, monitors and mice.

7 Sessions with the 3-year olds

These sessions were conducted using the directed approach: we felt it would be best to provide extra support for younger children, which have also the physical hurdle to overcome, not just the conceptual one.

Originally, one of the groups for the sessions was intended to include only two children at this age, which we will call C and R (both were girls). However, R didn't enjoy getting out of the kindergarten at all; she missed the first session and the third, and the kindergarten teacher considered that she should be replaced. She was replaced by a 4-year old, which we will call P. Due to calendar difficulties, only 5 sessions were conducted, from February 12th to April 2nd 2001, and a later session later on June 4th, 2001.

The other group was composed by a single 3-year old girl, which we will call J. She took part in 7 sessions, from March 29th to June 7th 2001.

The first group, with C and R, performed very simple activities, mainly due to the instability of the group composition: by the 4th session, a new girl was involved, and introductory activities were resumed. Even so, it should be noted this group managed to program and correctly use a robot on the second session, although no further work was possible along this line, to see how well they had understood the concept.

However, the second group, with J, allowed a more consistent set of activities, thus summarized:

1. Flying and moving around, playing with toys, letters and numbers.
2. Combining images to create a painting, place it on the wall (by using a wall sensor and control).
3. Playing with the vacuum cleaner and the wall sensor; programming a robot to write her name.
4. Building houses; programming a robot to put gifts on the wall. Launching the robot to new houses.
5. Playing sounds, teach robots how to play them. Use it on the back pictures, turn pictures around.
6. Playing around with birds. Exploring their operation, by hiding and copying nests.
7. Experiences programming robots to play sounds, using birds to clean up rooms.

It should be pointed out that on both groups, 3-year olds managed to program robots very early on, albeit with assistance: the first group on second session, the second group on the third session. All children had previous computer experience, which rendered easier such actions as mouse and button control. On the second group, references were made by the child, during later sessions, to previous aspects of the earlier sessions, showing that details of the activity were not alien to her. For instance, on the 7th session, she tried to give a nest full of things to a robot, and complained about it not being accepted because "things were properly tidied up" – this required explanation that things had to be tidied up inside boxes, not nests, but it is revealing that in fact some of the previous concepts on programming were grasped. It should be noted that the variety of robots programmed by J were suggested to her, and help was supplied, but this took the form of explanation, not specific instructions. She was able to take decisions on her own, requiring quite an understanding of the environment, e.g., turn around a picture to stop the robot playing sounds on its back.

8 Sessions with the 4- and 5-year olds

These sessions were conducted from February 14th 2001 to May 31st 2001. A group had 10 sessions, another 11 and the last one had 8.

Due to space limitations, a full account of the sessions is not possible here, but key points can be given.

In terms of the children's appreciation of the activity, except for one boy, all the others enjoyed playing in ToonTalk, from the very start; however, a 4-year old girl later on would usually prefer typing aimlessly in a word processor (she couldn't write) – she considered it "serious work" (*sic*). She would only enjoy ToonTalk if a more "serious" activity was chosen, e.g., programming a robot to write her name endlessly.

Due to the fact that there were 6 children at each session, two at each computer, and only one adult, activities were geared towards giving them some ability to use ToonTalk unattended for some time.

Overall, the children were able to program several robots, but that wouldn't be their main activity: they would copy things, enlarge them, put pictures side by side to construct scenarios for some story, etc.

Choosing the adequate situation or proposal for robot programming proved crucial. "Teaching" a robot could be interesting if the result was considered "funny". For instance, creating a huge repetition of their name; building lots of houses, blowing up houses, producing many birds.

Since only short amounts of time were possible, to each group in turn, we didn't manage to work higher-order concepts such as generalization within these groups. Several children could follow instructions or execute ideas regarding robot programming, but lower-order skills such as manipulation and control of objects, or orientation in the city and notebook were necessary, so that all children would enjoy the activity.

On another hand, this limited time frame of attention given to each pair of children allowed them more room for failure, and several events made us became aware of specific skills, or specific moments in which the ability to complete programming activities could be compromised:

- the initiation of the programming process (giving a box to the robot) may not be understood as an example or starting situation, but instead as an "interrupt" or "switch", like turning on a toy, therefore lacking the association between preset conditions and an intended action;
- consequentially, the concept of an activity being taught as something with "in" and "out" conditions, or as something with a "before" and "after" state, using a box as a container for acting upon, is not absolutely clear: quite often a child will start teaching a robot by giving it an empty box (using it like an "on" switch), and then "teach" with objects taken from the tool box or notebook;
- most children find it hard to grasp the "end of program" concept: after teaching a robot, they quite often proceed playing around with the objects in the tool box, regardless of the fact that they are moving them with the robot's hands, not the programmer's hand;
- the concept of sequencing activities taught to robots can be confusing: quite often, children expect robots to start "playing" the moment they finished teaching them, forgetting that they had to give them a starting box; but, they wouldn't make that assumption for robots intended to be used in a specific situation, such as, "when it reaches the new house", or "when the picture is turned around";
- "debugging" is an extremely alien concept. Most children assume that a robot learned exactly what they meant to teach him, to the point of sometimes being satisfied with the "teaching" per se, not

bothering to check if the robot worked; also, they would be puzzled upon seeing a robot repeat the redundant mistakes they made while programming it (like dropping a box on the wrong place and picking it up again), for they would forget about such deviations from the intended path, remembering only the intended path or goal.

9 Conclusions

Very young children can successfully use ToonTalk to program simple robots and use them, as long as the path to the discovery of the necessary techniques is supported, encouraging children to explore and try out new situations or approaches. Logical explanations for abstractions or programming constructs are also helpful in order to allow them to construct more complex programming situations, in which they combine more than one robot, combine an expected event with the execution of a robot, or combine a newly-programmed robot with a previously used one.

However, programming requires the acquisition of specific skills, which in turn require the development of specific teacher strategies and planning. Given only the rudiments of the operation of a programming system, children will quite often employ it as a simple non-programmable playground, or use programming as a way to produce a somewhat chaotic or unpredictable (from their point of view) chain of events. Again, adequate support is required, so that children can acquire the power of combining simple pieces (robots) and learn to guess at the likely result, or be able to decompose a problem into simpler pieces and set about to create them.

The results from the original generalization experiences seem to indicate that an approach based on simple, real-world, everyday explanations and usefulness can be more fruitful than one based on abstract explanation of a specific behaviour, be it before or after its observation.

We feel that the support that children require in order to learn how to program should be given as several small incentives to trial, experimentation and interpretation by the child, rather than explanations or sequences of instructions to follow.

References

1. Kahn, Ken ToonTalk™ – An animated programming environment for children. *Journal of Visual Languages and Computing* 7 (1996) pp. 197-217.
2. Myers, B., Hollan, J. and Cruz, I., eds. Strategic Directions in Human Computer Interaction. *ACM Computer Surveys 28* (1996).
3. Norman, D. Cognitive Engineering. In *User centered system design: New perspectives on human computer interaction*. (Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1986).
4. Perlman, R. Using computer technology to provide a creative learning environment for preschool children. MIT AI Lab Memo 360. Logo Memo 24 (1976).
5. Smith, D. C. and Cypher, A. Making Programming Easier for Children. In DRUIN, A., ed. *The Design of Children's Technology*. (Morgan Kaufmann Publishers, San Francisco, California, USA, 1999).
6. Morgado, Leonel, Cruz, Maria and Kahn, Ken, Working in ToonTalk with 4- and 5-year olds. Paper presented at the "*Playground - novos ambientes de aprendizagem*" Seminar at Casa de Vilar, Porto, Portugal, organized by CnotInfor (2001). Available on-line at the following web address: http://www.utad.pt/~leonelm/TTon4-5.htm.
7. Saraswat, Vijay, Concurrent Constraint Programming, MIT Press, Cambridge, MA, USA (1993).