A Low-cost, Low-energy Tangible Programming System for Computer Illiterates in Developing Regions

Andrew Cyrus Smith
African Advanced Institute for Information & Communications Technology
acsmith@csir.co.za

Abstract

We present a low-cost, low-energy technology design that addresses the lack of readily available functional computers for the vast number of computer-illiterate people in developing countries. The tangible programming language presented is an alternative entry point into the field of Information Technology. We conclude with a list of further work needed.

1. Introduction

The majority of residents in developing countries do not have access to a working computer. Neither is a large number of this majority fully literate, having limited reading and writing skills. On-going attempts by local governments and funding agencies have not yet been able to fully overcome the computer illiteracy situation in developing countries. In an attempt to improve this situation, relief programmes supply computing infrastructure to schools, or community centres, and provide training. But this model is not always effective. As an example of this, consider the trained personnel who soon leave the training centre for more profitable employment in private practice, leaving no expertise behind to assist the local population. Another potential problem is the lack of technical support for repair of the sophisticated computers. The well-meaning suppliers of the equipment are not always aware of the lack of resident technical support when they return to their industrialised countries. In both of these examples, these once well-equipped training centres go into disrepair and are eventually no longer functional.

The goal of these training centres is to increase the number of information-technology- (IT) literate citizens as it is believed that the modern economy is based on IT.

In contrast to the provision of high-end technologies to these training centres, we propose a novel, alternative means of introducing illiterates to IT. Other methods have previously been reported [2]. If we acknowledge the high level of illiteracy in developing countries, it serves little purpose to introduce large numbers of sophisticated personal computers to these areas. An alternative approach is to first develop, in the illiterate population, the cognitive process of logical thinking required in the IT field. Having developed this ability, the illiterate person has a tool for potentially controlling a number of objects in the immediate surroundings. If so desired, this person is ready to receive training using the traditional personal computer and subsequently become integrated in the IT mainstream.

Our approach makes use of symbols and physical artefacts (figure 1) to compile a sequence of actions, and is an extension of GameBlocks [3]. Two groups of the general population who will benefit the most from this technology are the young children and the elderly. It can be argued that both of these groups are typically illiterate and do not possess the fine motor skills of a healthy young adult, making the use of large and tangible input devices a strong contender as a replacement for a keyboard. The population group that fits between these age extremes typically does not have these problems and is not the target group of the research reported on in this paper.

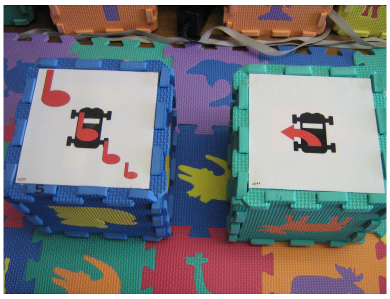

Figure 1. A minimalist coding example which is read and interpreted from left to right ("*play tune number two and then turn left*").

The technology we present is simple, requires little energy to operate, can be modified, is open-sourced, and can be interfaced to various output devices.

Our research contribution with this paper is our motivation for using a tangible programming environment in developing regions. We also give a description of one implementation of a tangible programming environment.

2. Motivation for using tangible interfaces

In many programming environments, programming code consists of keywords designed by the original programming language architect. In the modern context this is usually a vocabulary familiar to people in the western world (English), ignoring the ethnic background and vocabulary of the users of the system. Physical programming languages have recently seen much development, but the associated tangibles are fixed in their physical appearance and maintain the look-and-feel as envisaged by the original designer. It can be argued that an item crafted by the user herself would better represent the user's intensions than something made by someone geographically distant and from a very different culture.

We therefore hypothesise that the entry to the world of programming can be made less traumatic by using objects crafted by the end-users themselves.

We propose that physical syntax objects should rather be shaped at the point of usage, not centrally by a team of designers who might be located far from the point where the technology is being used, or having done their design research using user representation from the local population.

We also propose the use of a physical material that can be sculpted to conform and represent the coder's own interpretation of the action encoded by the object.

Prior research [1, 4, 5, 6] in the field of tangible interfaces points to increased physical activity by the end-user. Instead of the end-user being stationary in a seated position in front of a computer screen and keyboard, extended movements are possible and indeed required to manipulate the tangible input devices. If we accept that children in developing regions are physically more active that children in developed regions, then we can argue that the first group could possibly be at an advantage when it comes to using tangible interfaces.

We therefore hypothesise that using increased physical space and 3-dimensional input objects will reduce the divide which exists between the computer-literate and those who have never had computer exposure.

Our system removes all text from the user interface. In addition, the tangible interface objects in our system have properties of texture, weight, and colour. There is also a strong correlation between the shape and orientation of the input object, and the object's functionality. Because of these tangible and visual properties, the cognitive burden on the user is reduced as compared with text-only input systems.

We therefore hypothesise that our input devices are well suited for computer-illiterate people.

3. The tangible programming system description

3.1 The language

The current version of the tangible language consists of six instructions. These are move forwards, move backwards, turn left, turn right, play tune number one, and play tune number two (figure 2). It is a simple, single procedure language that allows for sequential execution.

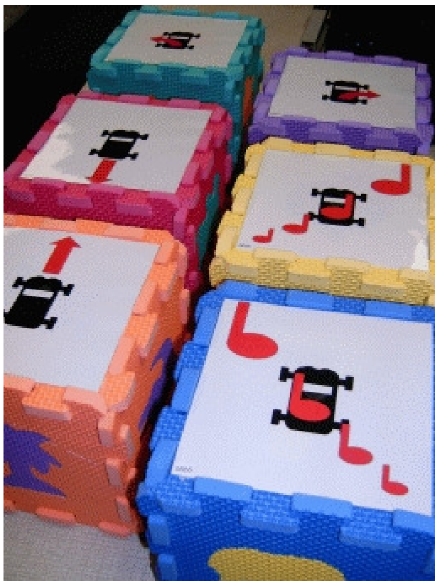

Figure 2. The six instructions used in the tangible programming system (clockwise from bottom left): move forwards, move back, turn left, turn right, play tune number one, and play tune number two.

3.3 The system

The tangible programming system consists of foam cubes (figure 2), a programming mat, controlling electronics (figure 3), and an output device. The cubes contain low-cost magnets to encode the functionality that each cube represents.

Embedded into the programming-mat are low-cost magnetic sensors. These sensors are wired to the controlling electronic circuit. The controlling circuit is a very simple design containing a single 8-bit microprocessor plus some discreet logic circuitry.

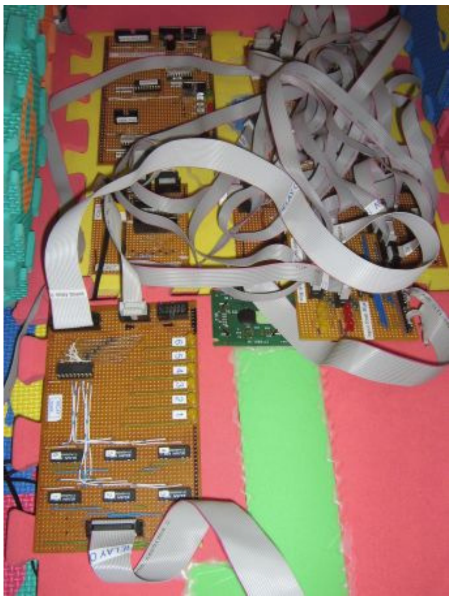

Figure 3. The proof-of-concept electronic controller circuitry.

Program output is made tangible by the motions and sounds emitted from a remote-controlled toy car. The toy car executes a number of actions such as moving and playing musical tunes These movements are based on the sequence of the cubes placed onto the programming mat.

4. Overview of the evaluation methodology

The described system was evaluated with children in a number of environments in South Africa. These include two science shows held nearly 900 kilometres apart, and a few sessions held in our laboratories. In all cases where tests were conducted away from our laboratories we did not have any a-priory information on the testees. Although not formally established in the tests it can be argued that the social, cultural, educational and financial disposition of the testees differed between these two geographic regions away from our laboratories.

Children from both primary- and secondary- schools participated in these tests. In total, approximately 200 children have been given the opportunity to experience the system. In all cases the evaluation took place in a controlled environment with the number of testees being limited to groups no being larger than 20.

The following is a description of the testing process: Initially the testees were welcomed and given a questionnaire relating to their background and knowledge of IT. They were then introduced to the system and given a challenge which had to be solved using the system. This stage was iterated a number of times in order for the testees to become comfortable in using the system. The sessions concluded with the testees completing the second part of the questionnaire. This part solicited testees' comment on the system which will inform future research.

We have not yet formally analysed the information gathered during the user tests.

5. Future work

5.1. Component costs

The current concept demonstrator has been implemented using dual-in-line integrated circuits (DIL IC's) and other discreet through-hole components. This approach makes for easy construction and fault finding which is ideal for regions where only basic manufacturing technology is available. However, system component cost can be reduced by combining the active components into a single IC. Field Programmable Gate Arrays (FPGA's) or Application Specific Integrated Circuits (ASIC's) are examples of high-density, low component cost options.

5.2. Customisable and recyclable tangibles

The bright and light-weight foam cubes used in the concept demonstrator are attractive and robust, but they are manufactured from man-made materials and not easily recyclable. Future research can include the use of natural material such as wood, cardboard, and soft stone as tangibles. These materials can be sculpted at the point-of-use to meet the exact requirements of the end-user. The natural materials have the additional advantage of being of very low cost, easily recyclable, and readily available.

6. Conclusions

We have presented an alternative programming environment which addresses a number of problems in the developing world. The problems include the low level of IT literacy, the lack of IT maintenance infrastructure, and limited motor skills in some cases. The presented system can provide an alternative entry point into the field of IT, aimed at the computer illiterates of developing countries. We concluded by briefly introducing two areas in which the system may be improved to make it better suited for developing regions.

7. Acknowledgements

This research was financially supported by the South African Department of Science and Technology.

8. References

[1] Horn, M.S. and Jacob, R.J.K. Tangible programming in the classroom with tern. In *CHI '07 extended abstracts on Human factors in computing systems*, ACM Press (2007), 1965-1970.

[2] Kelleher, C. and Pausch, R. Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers, *ACM Comput. Surv.*, 2005, vol 37,83-137, no 2.

[3] Smith, A. C. Using magnets in physical blocks that behave as programming objects, Proc. *Tangible and embedded interaction,* 147-150, ACM Press (2007).

[4] Suzuki, H. and Kato, H. Interaction-level support for collaborative learning: AlgoBlock-an open programming language, *CSCL '95*: Computer support for collaborative learning, 1995, 349-355.

[5] Tang, T. S. Storytelling Cube: A tangible interface for playing a story.

[6] Wyeth, P. and Wyeth, G. Electronic Blocks: Tangible Programming Elements for Preschoolers, In *Proc INTERACT 2001*, 2001.