
Programming on the Move: Design Lessons from IPRO

Matthew Berland

Department of Interdisciplinary Learning & Teaching
The University of Texas at San Antonio
1 UTSA Circle
San Antonio, TX 78249 USA
matthew.berland@utsa.edu

Taylor Martin

Department of Curriculum and Instruction
The University of Texas at Austin
1 University Station D5700
Austin, Texas 78712-0379 USA
taylormartin@mail.utexas.edu

Tom Benton

Department of Curriculum and Instruction
The University of Texas at Austin
tom@activelearninglab.org

Carmen Petrick

Department of Curriculum and Instruction
The University of Texas at Austin
carmenpetrick@gmail.com

Copyright is held by the author/owner(s).
CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.
ACM 978-1-4503-0268-5/11/05.

Abstract

Computer programming is often a stationary, solitary task; such tasks do not work well for most novices. This work describes the IPRO project that uses our 'Programming Standing Up' framework (PSU) to reframe programming as a mobile, social game. IPRO is a programming and simulation environment for iOS in which a learner programs a virtual robot to play soccer in a virtual space shared with her cohort. This work presents examples of secondary school students learning with IPRO. We then connect the examples to PSU design principles and evaluate those principles in terms of the examples.

Keywords

Handheld Devices and Mobile Computing; End-user Programming; Children; Empirical Methods, Qualitative; E-Learning and Education; Robots

ACM Classification Keywords

K.3.2 Computer and Information Science Education

Introduction

Computer programming remains a solitary, stationary practice despite some collaboration frameworks [1]. Novice programmers often find this paradigm daunting. Furthermore, there is significant evidence that collaboration [2], embodiment [3], and ludic practices [4] apply well to computational logic. To address this

gap, our work builds upon this research in the area of embodied cognition, which examines the role of the body and of gesture in complex reasoning. This paper reports on a project called 'Programming Standing Up' (PSU), which aims to open programming to embodiment by moving programming away from the computer and into the realm of a mobile, social activity. IPRO is a PSU application designed to teach computer programming in secondary schools. IPRO is a programming and simulation environment for iOS (iPhone, iPod Touch, iPad) in which a learner programs a virtual robot to play soccer in a virtual space shared with her cohort [5]. Because it is an iOS app, a classroom of students working with IPRO takes on a very different appearance from one engaged with a more traditional programming exercise. Students are able to model the behavior of their robots in the classroom space and are also able to collaborate with their peers, to troubleshoot common problems as well as develop heterogeneous but complementary designs. In addition to exploring alternative modes of programming interaction, this work attempts to engage students with computer science concepts with greater efficacy than traditional instruction [6, 7, 8].

IPRO is indebted to the rich history of robotics in STEM teaching and learning that traces a path backwards to Papert's original vision for educational robotics [9]. The Mindstorms robotics kit stem from that work and integrate basic construction pieces (e.g., LEGO blocks, gears) with a graphical programming environment [10]. Products such as RoboLab, RCX Code [11] and NXT [12] continue this theme, introducing a variety of physical components including sophisticated sensors. These kits have been primarily used to provide a physical context for programming, such as in the

RoboCup tournament [13]. However, for novices the act of programming often remains disconnected from the physical action it precipitates [14]; robotics-related projects such as Topobo [15] or Kinematics [16] can bring aspects of programming into the physical world, but they often do so eliminating formal programming from the interaction. Though the history of embodied programming is voluminous, the authors found no literature in which learners build, compile, and run working programs on a handheld device using a programming language designed for that device.

IPRO

The IPRO language is a graphical, functional language in which the primary structure is built around conditional statements and robot actions. The conditional logic is based on true or false facts about the state of the game world; there is a predicate logic of binary sensors, such as "is another player in front of me?" or "is the goal to my left?" Second, all possible IPRO robot actions have clear physical world equivalence (e.g., move left, turn right), and there are no possible syntax errors. This allows students to physically perform the programs they have written and to isolate the source of their troubles.

The Space

In IPRO, virtual robots move in a virtual hexagonal grid space (See Figure 1). In the IPRO classroom, students maneuver on a large tarp covered with a hexagonal grid on the floor (seen in Figure 2). A student can move forward or backward, turn, and see or detect the ball just as their robot can. This allows the learner to draw on physical experiences to make sense of how a robot moves and take advantage of the physical space for computational logic such as in [17, 18, 19].

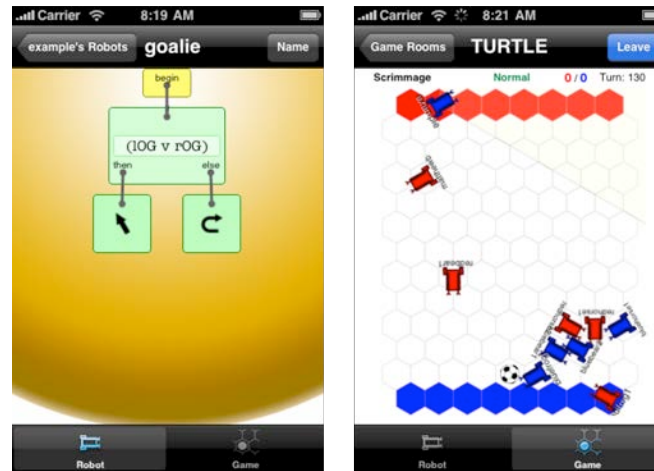


figure 1. IPRO programming (L) and game (R) spaces

IPRO in ACTION

Below, we present examples of how the PSU model affects users and learners. These examples are taken from 4 classroom settings in which different groups of students practiced programming in IPRO. Two classes of high school students from an urban, public school and two classes of middle school students from a summer robotics camp participated. After approximately 15 minutes of guided practice with the IPRO language, each student programmed a robot with a partner. Students could edit their code, share sections of code, enact their programs on the hexagonal grid on the floor, or test their robots in a solo play mode. Every 20 minutes the instructor called for all students to enter their robots in a match against their classmates' robots. The matches were projected on a large screen at the front of the room. After seeing how their robots fared in competition, students cycled back into another phase of design in which they could make improvements to their programs. The high

school students spent 1 hour designing their robots and competing in matches, while the middle school students spent 2 hours.

We administered pre-/post-tests, collected log data of every interaction with the IPRO system, videotaped the classes, and interviewed individual students about their actions as they were programming. The following examples from students' interactions with IPRO highlight features of the programming environment that we found facilitated embodied cognition, mobility, and collaboration using a game-like structure.

Robot, Do What I Do: Designing Code with Movement

In one situation, a high school student faced an impasse in which an otherwise capable robot could not score when it found itself located between the ball and the goal. To solve her conundrum, she moved into the middle of the room and performed several competing programs. Through these actions, the optimal program

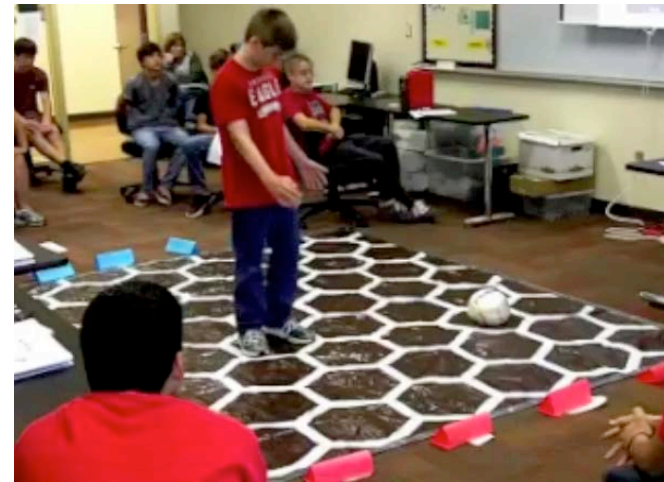


figure 2. A student writing an IPRO program with the device (hidden) in his right hand

became obvious. Without problem, she wrote the code that she had just enacted with her body.

To a student new to programming, code that directly represents perceived reality is easier to understand and easier to write. By enacting the program, she was the designer, compiler, and executable simultaneously; this is a powerful status for a learner.

Playing Robot: Collaborative Debugging

In another situation, two middle-school students working together on the same team had made changes to their programs, but they were unsure if the changes would be effective. They decided to act out their programs together to better understand their creations. Checking their iPods for reference often, they stepped through their programs; they discovered the programs were not working properly. One of the students had incorrectly predicted the movement of the ball, and the other student had misunderstood how the program loop was functioning. Their misconceptions became evident only through the cooperation and coordination of the two student-robots.

The programs proved easy to interpret through movement for our students. As such, they can better collaboratively debug their code when their robots act in inconsistent or nonsensical ways. One of the most challenging skills in computer programming is finding runtime errors, and learning how to find errors is easier when multiple people are conditionally acting on each other's behaviors.

Gooooaaaal!: Coding as Social Gaming

Looking much like a World Cup crowd, students in IPRO watched their matches while standing in front of the

large screen urging on their team. Cascading groans followed each change of possession, and the room cheered loudly after every goal. Immediately following a match, the students scurried back with their teammates to work on improving their robots for the next competition.

The investment and engagement are palpable; students are working because they choose to do so. Though it is more immediately applicable than most school content, programming suffers from an engagement problem. Authentic instructional tasks are important factors for student engagement, particularly at the middle and high school levels [20]. These include tasks that have relevance in the real world, include ill-defined problems, and allow students to investigate using different resources from multiple perspectives [21, 22]. As IPRO is centered on soccer, students begin their first programming experience with a concrete metaphor.

As students' robots play the game, the students realize that not every robot on the team should work the same. Just as a soccer team includes strikers, goalies, and midfielders, a successful IPRO team includes robots with different objectives: offense, defense, or ball control.

You're Just Standing There!: Social Feedback

During one high school match, several robots functioned poorly: one robot did not move; one robot went in a tight circle; and a poorly coded "striker" actively fled the ball. The mistakes were apparent to all players, but the responses varied widely. Some players suggested fixes to bad players, some players lightly mocked the code, and the designers occasionally asked for help with their code from their peers. The social

reinforcement of code correctness proved powerful, especially in a closed social situation. Most of the authors of the failed robots quickly fixed their robots or solicited help.

Conclusion

Programming is rarely described as a game, as collaborative, or as embodied, and open social feedback in programming is usually tied to open source projects rather than first-time learners. As we illustrate in this paper, these features can be incorporated into programming interfaces and beneficially impact programmers. These features may be particularly important for novice programmers and increasing the pipeline from novice to expert programmers. This work describes our pilot work and initial design findings. Future work with IPRO includes: a quantitative study of performance as it related to embodiment; case studies of embodied cognition; and a process understanding of how students came to learn more complex programming concepts.

Acknowledgements

We thank Gilbert Slade & Chadwick Wood for their hard work on this work. This work was supported by NSF Grant EEC-1025223.

References

- [1] Beck, K., & Andres, C. *Extreme Programming Explained: Embrace Change* (2nd Edition ed.): Addison-Wesley Professional, 2004.
- [2] Forte, A. & Guzdial, M. Computers for Communication, Not Calculation: Media as a Motivation and Context for Learning. In *Proceedings of 37th Hawaiian International Conference of Systems Sciences*. Big Island, Hawaii, 2004.
- [3] Fadjo, C. L., Lu, m., & Black, J. B. Instructional Embodiment and Video Game Programming in an After School Program. *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, Chesapeake, VA, 2009.
- [4] Berland, M. & Lee, V. Collaborative Strategic Board Games as a Site for Distributed Computational Thinking. *International Journal of Game-Based Learning*, (2010, in press).
- [5] Berland, M., Martin, T., and Benton, T. Programming Standing Up: Embodied Computing with Constructionist Robotics. *Proceedings of Constructionism 2010*.
- [6] Moher, T. Embedded Phenomena: Supporting Science Learning with Classroom-sized Distributed Simulations. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 691-700, 2006.
- [7] Price, S., and Rogers, Y. Let's get physical: The learning benefits of interacting in digitally augmented physical spaces. *Computers & Education* 43, 2004, 137-151.
- [8] Xie, L., Antle, A., and Motamedi, N. Are Tangibles More Fun? Comparing Children's Enjoyment and Engagement Using Physical, Graphical, and Tangible User Interfaces. *Proceedings of the Second International Conference on Tangible and Embedded Interaction*, 191-198, 2008.
- [9] Papert, S. *Mindstorms*. New York: Basic Books, 1980.
- [10] Baum, D. (2003). *Definitive Guide to LEGO Mindstorms*. New York: Apress.
- [11] Erwin, B., Cyr, M., and Rogers, C. Lego Engineering and Robolab: Teaching Engineering with LabView from Kindergarten to Graduate School. *International Journal of Engineering Education*, 16(2), 1-11, 2000.
- [12] Wolz, U. Teaching Design and Project Management with LEGO Robots. *Proceedings of the SIGCSE Technical*

Symposium on Computer Science Education, 11-15, 2001.

[13] Sklar, E., and Parsons, S. RoboCupJunior: A vehicle for enhancing technical literacy. *Proceedings of the AAAI-02 Mobile Robot Workshop*, 2002.

[14] diSessa, A. *Changing Minds: Computers, Learning, and Literacy*. Cambridge, MA: MIT Press, 2000.

[15] Raffle, H., Parkes, A., and Ishii, H. Topobo: A Constructive Assembly System with Kinetic Memory. *Proceedings of the ACM Conference for Computer-Human Interaction*, 2004.

[16] Oschuetz, L., Wessolek, D., and Sattler, W. Constructing with Movement Kinematics. *Proceedings of the ACM Conference for Tangible, Embedded, and Embodied Interaction*, 257-260, 2010.

[17] Fernaeus, Y., and Tholander, J. Finding Design Qualities in a Tangible Programming Space. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 447-456, 2006.

[18] Fernaeus, Y., and Tholander, J. "Looking at the Computer but Doing It On Land": Children's Interactions in a Tangible Programming Space. People and Computers XIX: the bigger picture: *Proceedings of Human Computer Interaction 2005*, 3-18, 2006.

[19] Jacob, R., Girouard, A., Hirshfield, L., Horn, M., Shaer, O., Solovey, E., and Zigelbaum, J. Reality-Based Interaction: Unifying the Next Generation of Interactions Styles. *CHI '07 extended abstracts on Human factors in computing systems*, 2465-2470, 2007.

[20] Marks, H. M. Student Engagement in Instructional Activity: Patterns in the Elementary, Middle, and High School Years. *American Educational Research Journal*, 37(1), 153-184, 2000.

[21] Jonassen, D. Designing Constructivist Learning Environments. In C. M. Reigeluth (Ed.), *Instructional Design Theories and Models: A new paradigm of*

instructional theory. Mahwah, NJ: Lawrence Erlbaum Associates, Inc, 1999.

[22] Herrington, J., Oliver, R., & Reeves, T. C. Patterns of Engagement in Authentic Online Learning Environments. *Australian Journal of Educational Technology*, 19, 1 (2003), 59-71