In the *Fifth Symposium on Educational Advances in Artificial Intelligence* (EAAI), Quebec City, Canada, July 28-29, 2014.
(co-located with AAAI 2014)

DOROTHY: Enhancing Bidirectional Communication between a 3D Programming Interface and Mobile Robots

Emilie Featherston
Department of Electrical and Computer Engineering
Texas Tech University, USA
emilie.featherston@ttu.edu

Mohan Sridharan
Department of Computer Science
Texas Tech University, USA
mohan.sridharan@ttu.edu

Susan Urban
Department of Industrial Engineering
Texas Tech University, USA
susan.urban@ttu.edu

Joseph Urban
Department of Industrial Engineering
Texas Tech University, USA
joseph.urban@ttu.edu

Abstract

Dorothy is an integrated 3D/robotics educational tool created by augmenting the Alice programming environment for teaching core computing skills to students without prior programming experience. The tool provides a drag and drop interface to create graphical routines in virtual worlds; these routines are automatically translated into code to provide a real-time or offline enactment on mobile robots in the real world. This paper summarizes the key capabilities of Dorothy, and describes the contributions made to: (a) enhance the bidirectional communication between the virtual interface and robots; and (b) support multirobot collaboration. Specifically, we describe the ability to automatically revise the virtual world based on sensor data obtained from robots, creating or deleting objects in the virtual world based on their observed presence or absence in the real world. Furthermore, we describe the use of visually observed behavior of teammates for collaboration between robots when they cannot communicate with each other. Dorothy thus helps illustrate sophisticated algorithms for fundamental challenges in robotics and AI to teach advanced computing concepts, and to emphasize the importance of computing in real world applications, to beginning programmers.

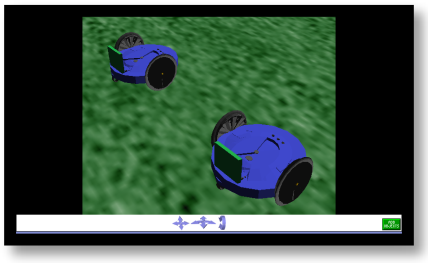
1 Introduction

There has been much emphasis over the last decade on the need to increase the participation of communities with longstanding underrepresentation in computing disciplines (NSF-BPC 2014; CPGE21 2007; PCAST 2007). Research shows that fostering interest in computing requires early intervention and long-term efforts. Researchers and educators have explored the design and use of 3D (i.e., graphical) programming environments and robotics to engage students with little or no prior programming experience. Graphical programming environments such as Alice (Dann, Cooper, and Pausch 2008), Scratch (Scratch 2009) and Greenfoot (Kolling 2009) have been used as an effective way of teaching students to program at the K-12 level. Many such efforts have used storytelling to learn programming through animations generated in simple pseudocode based on virtual characters, objects, and scenarios. These environments, however, do not support the enactment of graphical routines in the real world, and students may fail to make the transition to advanced computing concepts and their application to practical problems. Robots, on the other hand, are well-suited for illustrating practical applications of computing concepts (IPRE 2010; Lauwers, Nourbakhsh, and Hamner 2009; McWhorter and O'Connor 2009; RCJ 2010; Summet et al. 2009). However, it can be difficult to learn the complex syntax required to program robots. Many existing robot kits also limit the students' ability to obtain a deeper understanding of computing; such understanding can be achieved by programming robots capable of autonomous operation using sensor inputs.

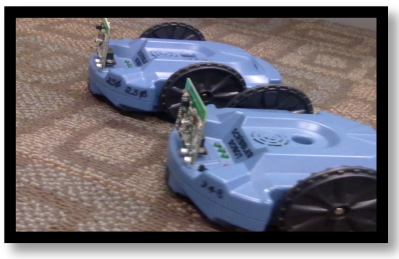
In recognition of these challenges, we developed the Dorothy (Design of Robot Oriented Thinking to Help Youth) 3D/robotics programming environment (South et al. 2013). Dorothy significantly extends the Alice programming environment to exploit the complementary features of graphical programming and robotics. Dorothy was initially developed through the efforts of seven undergraduate students from three universities who were supervised in research projects and summer internships, with partial support from a U.S. National Science Foundation Research Experiences for Undergraduates site project award.

Dorothy allows students to program in virtual worlds that include a robot character as a virtual representation of a physical robot (cohort) in the real world; one robot that has been integrated is the IPRE Fluke robot (IPRE 2010) shown in Figure 1. Students learn core computing skills when they use a drag and drop interface to create graphical routines comprising a set of instructions provided to the robot in the virtual world, and see the physical enactment of the corresponding program on one or more robots in the real world. We have also initiated the development of a curriculum for using Dorothy to teach the core concepts of computing, concurrent execution, and sensor input processing to middle school and high school students.

This paper describes the capabilities of Dorothy, focusing on recent contributions made to: (a) enhance the bidi-



(a) Virtual world.



(b) Real world.

Figure 1: Representation of the Fluke robots in the virtual world, and the physical robots in the real world.

rectional communication between the virtual interface and the robots; and (b) enable multirobot collaboration in the absence of communication between robots in the team. Specifically, we enable the addition (deletion) of objects to (from) the virtual world when they are detected (not detected) by processing sensor inputs on robots in the real world. Furthermore, we enable robots to use the visually observed behavior of teammates for collaboration when they cannot directly communicate with each other. Dorothy thus helps us use core robotics and AI concepts such as planning, perception, teammate modeling and multiagent collaboration to teach advanced computing concepts to beginning programmers. Dorothy also helps us emphasize to beginning programmers the importance of computing in real world applications, e.g., a team of mobile robots collaborating to investigate previously unexplored (and possibly dangerous) regions.

The remainder of this paper is structured as follows. First, Section 2 describes related work with the Alice 3D programming environment, robots for computing education, and the integration of Alice with robotics. Next, Section 3 summarizes the initial development of Dorothy and the associated curriculum. Section 4 describes the new contributions made to enhance the bidirectional communication and enable multirobot collaboration. Finally, Section 5 concludes with a summary and a discussion of future research.

2 Related Work

This section discusses previous work that has motivated the development of Dorothy, providing an overview of 3D programming environments, the use of robots for computing education, and research on integrating Alice with robots.

2.1 3D Programming Environments

Three-dimensional programming environments such as Alice (Alice 2014), Greenfoot (Greenfoot 2009), and Scratch (Scratch 2009) have been explored as an effective means for teaching computational thinking (Wing 2006). Using 3D tools, users create visual representations of virtual worlds by adding virtual objects to a scene with the programming environment. Students learn and use computational thinking concepts as objects are manipulated through drag and drop programming techniques as a means to get virtual objects to achieve a specific task. Dorothy builds specifically on the Alice 3D programming environment. Alice was developed at Carnegie Mellon University and uses storytelling to maintain user interest while teaching introductory programming concepts (Kelleher, Pausch, and Kiesler 2007). Alice provides multiple virtual objects in a large preexisting gallery, along with the ability to create new 3D virtual objects. Users can create a story in a virtual world with these objects using pseudo Java code, which illustrates data structures, functions, control structures, and the correlation between stories and programming concepts.

2.2 Robots for Computing Education

Robotics has also been used as a way to stimulate interest in computing, using tools such as the Fluke robots (IPRE 2010), LEGO NXT Mind-storms (LEGO 2014), and the iRobot create platform (IRobot 2014). For instance, the Advancing Robotics Technology for Societal Impact (ARTSI) alliance led by Spelman College focused on using robots to increase the number of African American students in computer science. The INSPIRED (INcreasing Student Participation in Research Development) program at Lamar University used Scratch and robots to motivate under-represented middle school students to pursue computing education (Doerschuk, Liu, and Mann 2009). The Hispanic Computer Brigade at San Jose State University used Alice and picocrickets to inspire Hispanic students to pursue computing (HCB 2010). Dorothy makes use of the Fluke robots created for the Myro (My Robotics) project by the Institute for Personal Robots in Education (IPRE 2010), a joint venture between Georgia Institute of Technology, Bryn Mawr College, and Microsoft Research, for promoting the use of robots in computer science education. These robots consist of an add-on board mounted on a scribbler wheeled robot base with a micro-controller (Parallax 2011). The wheeled base can move forward, backward, and turn. The base can also be fitted with a marker to draw patterns. The add-on board provides a suite of sensors, which includes a camera, infrared sensors, and light sensors, and supports communication between the robot and a computer using Bluetooth technology. The Myro project provides a framework with many functions that can be used for a variety of tasks such as I/O, movement, obstacle detection, sensor data acquisition, and image processing on the Fluke robots.

2.3 Alice/Robotics Integration

Dorothy represents one of the few instances of fully utilizing the complementary strengths of graphical programming and robotics by integrating Alice and the Fluke robots.

Another instance of using the Alice interface and robotics is the PREOP (Providing Robotic Experiences Through Object-Based Programming) program at the University of Alabama (Wellman, Anderson, and Vrbsky 2009; Wellman, Davis, and Anderson 2009). Dorothy and PREOP differ in the capabilities they provide and the target audience. PREOP uses the iRobot Create robot, which is represented by a custom virtual version in Alice and can be commanded to execute motion commands, play a song, or beep to a chosen frequency. Studies show that the use of PREOP significantly increased the students' self-ratings of confidence and computing abilities (Davis et al. 2009; Wellman, Davis, and Anderson 2009). However, PREOP does not support real-time bidirectional communication with the robots; it cannot simulate sensing or motion based on data from the robot's sensors and actuators. As a result, users cannot create and visualize graphical routines in virtual worlds that correspond to adaptive behavior in a physical world. Instead, the robot separately uses iRobot Create's bump detection to avoid obstacles. The PREOP program is also limited to operate on one instance of a specific type of robot. Dorothy, on the other hand, uses the Fluke robots, supports bidirectional communication, and enables interaction with multiple instances of the same type of robot, as well as multiple types of robots. Furthermore, Dorothy was developed primarily to engage students in grades K-12, whereas PREOP has been used in a college introductory course.

3 Overview of Dorothy

The design decisions made while creating Dorothy were motivated by the desire to enable: (1) bidirectional communication of sensor data and commands between robots in virtual worlds and their cohorts in the real world; (2) ease of interfacing with multiple robots and different types of robots; and (3) ease of use by students with little or no prior programming experience. Figure 2 is a diagram of the architecture that shows the flow of data and commands.

A robot handler is created to control the bidirectional flow of sensor data and commands between each robot in the virtual world and its cohort in the real world, enabling synchronous (i.e., real-time) or asynchronous (i.e., offline) execution of graphical routines in the virtual world and by the physical robot. Execution of a routine in a virtual world creates events that ask the robot to perform specific actions or answer questions. An action event is intercepted when the action is converted into a "runtime response" that uses all relevant information to act on a virtual object. The robot handler (for the robot under consideration) receives the information about the class/type of action, parameters needed for the action, and object/subject that will perform the action. The handler's *data receiver* passes this datagram to a *translator*. A command dictionary and a socket stream enable real-time translation of the action sequence to corresponding sensing and actuation commands for the physical robot. The translator first reads and compares the class-name of the action with a dictionary data type containing the class-names of all actions that the robot can perform. If a matching class-name is found in the dictionary, the corresponding action template contains a basic layout of the action and variables that need

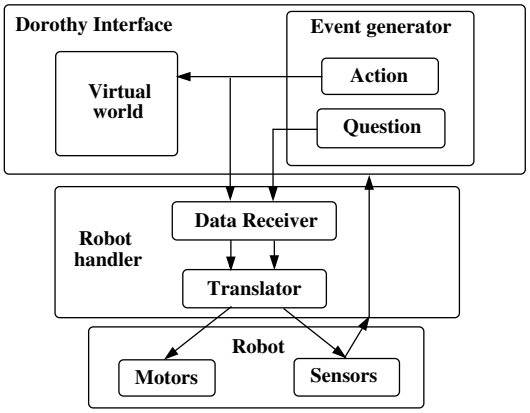


Figure 2: Overview of the flow of data and commands in Dorothy's architecture.

to be parsed from the datagram. Data extracted from the datagram is passed to the commands created for execution on the corresponding robot.

A question event is created when a user wants to check the condition of the virtual world, such as the presence of obstacles in front of the robot. The robot handler first decides whether the question can be answered by the robot; if so, the handler queries the robot's sensors for the relevant data, e.g., infrared sensors to detect obstacles. This data is communicated to the graphical environment, effectively overriding the virtual robot's observations with the real-world observations. Dorothy's architecture thus makes it possible for the user to see virtual robots and their physical cohorts moving and acting together.

The communication between a specific robot handler and the corresponding robot is achieved using socket communicators. The modular architecture allows these socket communicators to be written in any programming language. Dorothy can thus be used with different types of robots by creating and using a dictionary of appropriate commands/functions, e.g., for sensing and actuation, for each type of robot. These definitions of a robot's capabilities can be edited without making changes to the graphical environment. In our work, we have primarily experimented with the Fluke robot, but have also communicated between virtual worlds in Dorothy and the Erratic wheeled robot platform (Erratic-ROS 2012). Typical scenarios involved the creation of programs that command robots to move forward, backward, or turn. Bidirectional communication was demonstrated by enabling robots in the virtual world to respond to information from the infrared sensor on the physical robot cohorts. Further details about the initial development of Dorothy can be found in (South et al. 2013).

We have also developed a curriculum that uses Dorothy in individual lessons to teach core computing concepts. The curriculum consists of six lessons, each of 1.5-2 hour duration. The first lesson illustrates the importance of computing by discussing recent technological developments and logic puzzles. Instructors also demonstrate Dorothy and its ability

to control one or more robots. The second lesson has student teams interacting with robots through the virtual interface. Instructors demonstrate the need for sensor inputs from robots, and illustrate the use of basic variable assignment for generating motion patterns defined by user-specified parameter values. The third lesson introduces Boolean conditions and control structures using scenarios such as obstacle avoidance. Students create appropriate graphical routines and execute them on real robots. The focus of the fourth lesson is on iteration using repetitive pattern generation and obstacle avoidance, with student teams creating programs for similar tasks. Lesson five motivates the need for functions for program reuse and introduces students to basic image processing concepts, with tasks such as locating bright spots in a room and visual object recognition. The final lesson discusses concurrent execution concepts and poses challenging tasks as projects. Student investigate projects such as coordinating activities among multiple robots, with student teams participating in a competition to complete the projects. Different versions of the curriculum have been used for middle school and high school students. The workshops have been well-received by the students and their teachers, with feedback that has helped improve the tool and the curriculum. The lesson plans can be viewed online: `http://www.cs.ttu.edu/~smohan/Outreach/Dorothy.html`

4 Enhancing Communication and Collaboration

This section describes our more recent contributions that: (a) enhance the bidirectional communication capabilities of Dorothy; and (b) support multirobot collaboration in the absence of communication. Specifically, we describe the revisions made to Dorothy to visualize (in real-time) in a virtual world the information extracted by processing inputs from sensors mounted on a robot in the real world. We also describe the use of visual input, especially color information, to model the behavior of teammates and support multirobot collaboration when robots cannot communicate with each other. Dorothy can thus be used to introduce AI and robotics algorithms and thereby teach advanced computational thinking concepts.

4.1 Visual Objects in Virtual Worlds

To visualize objects in a virtual world based on sensor observations from a robot in the real world, we need to create and delete objects dynamically. Towards this objective, we create certain additional *squares*, the basic building blocks of objects in the virtual world, but hide them. These blocks are moved under the ground in the virtual world and are set to be invisible to the user. On the Fluke robot, we implement functions that use the basic functions provided by the Myro framework to detect obstacles and specific objects. For instance, obstacles within a certain region in front of the robot are detected using the infrared sensors, while objects with unique colors are identified by processing the camera images; inputs from the infrared sensors are processed in each cycle, while camera images are processed periodically. The function to process infrared inputs on the robot uses a function in the Myro framework that provides a number representing the proximity of an obstacle to the robot; these numbers are translated into a measure of distance, and numbers above a threshold are considered to represent the presence of an obstacle with high certainty. This function also extracts information about the relative bearing of the obstacle (e.g., left, right, and/or center). When a new obstacle is detected, i.e., the current location of the obstacle was previously unoccupied, one or more of the invisible squares are moved to an appropriate location and set to be visible. The squares are also suitably oriented relative to the robot; the orientation is also revised if there is already an obstacle at this location.

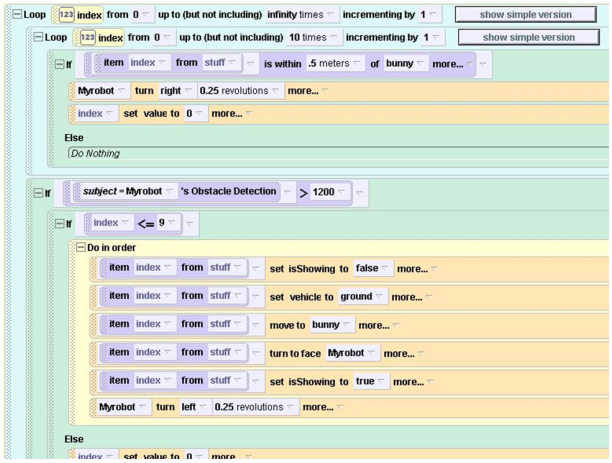


Figure 3: Snapshot of Dorothy code to add an object to the virtual world when it is detected in the real world.

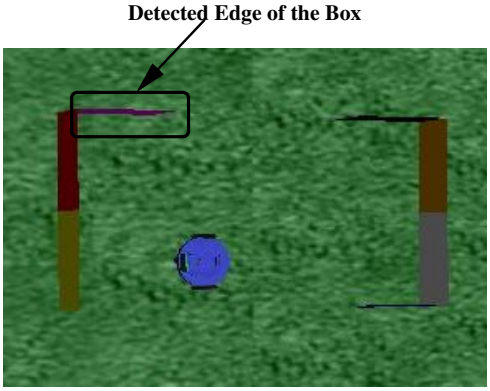


Figure 4: Visualization of a box that the robot was traveling around in the real world.

Since the information obtained by processing sensor inputs on robots may be erroneous, the virtual world provides an action that can be used to revise the newly added information, e.g., clicking on a square that was made visible based on sensor inputs will make the square invisible and push it underground (once again). Figure 3 provides a snapshot of a subset of the code implemented in Dorothy to add an object to the virtual world if it is detected in the real world.

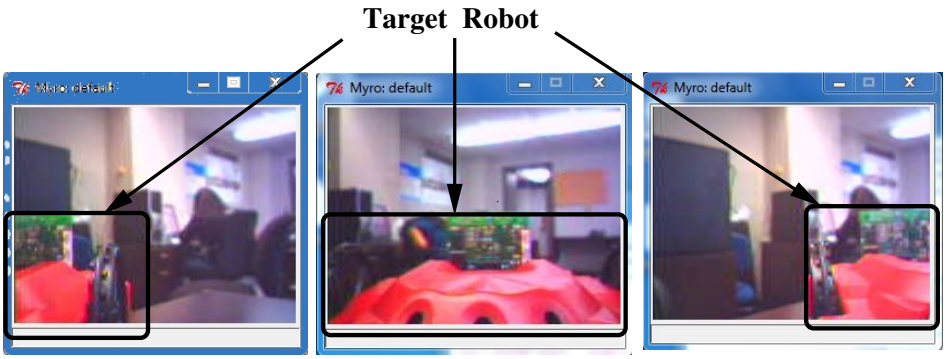


Figure 5: Illustrative example of using color information for multirobot collaboration; the robot has to detect another red-colored robot and follow it. The robot can detect three different states of the teammate: on-left, straight-ahead; and on-right, and select actions using an appropriate state transition diagram.

Figure 4 shows an illustrative example; the robot's virtual world initially does not have any information about the box that the robot is navigating in the real world, but this box is incrementally visualized in the virtual world as its corners and walls are observed by the mobile robot as obstacles in the real world.

Similar functions were created to process inputs from other sensors such as the light sensor and the camera, providing information that is used to visualize the corresponding objects in the virtual world. For instance, we implemented a function on the Fluke robot that uses the built-in functions of the Myro framework to segment and identify image regions of a desired color. The desired color is defined by setting the limits of the color's pixel values in the three-dimensional color space. The distance and relative bearing of any object detected in an image by this segmentation function are computed using the size of the segmented region in the image, (known) size of the object in the real world, image offsets, and known characteristics of the camera. The objects detected in images are visualized in the virtual world by (as before) making an appropriate number of squares visible at relevant locations and orienting them relative to the robot. The Dorothy interface is also augmented to allow users to select the sensors from which data needs to be acquired and visualized in the virtual world.

One significant outcome of the enhanced bidirectional communication in Dorothy is that teachers can design realistic projects that introduce beginning programmers to the sophisticated algorithms for challenges in image processing, planning, mapping, and autonomy, thus teaching advanced computational thinking concepts. For instance, students can design and visualize graphical routines for detecting and tracking a desired target (characterized by color, or other visual or non-visual properties) in the presence of previously unknown obstacles.

4.2 Behavior Modeling for Collaboration

To enable multirobot collaboration in the absence of communication between the robots, we explored the use of vision-based modeling of teammate behavior. As an illustrative example, we designed a task that required a Fluke robot to follow a teammate based (only) on the teammate's observed behavior. In this example, the teammate was characterized by a specific color (red); other characterizations of the teammate can also be used. We implemented a function on the Fluke robot, which uses the built-in Myro functions to define the color of interest. As described above, the desired color is defined by specifying the limits of this color in the color space; it is possible to incrementally compute these limits by considering the values of some image pixels corresponding to objects of the desired color. The robot is then able to segment and compute the coordinates of image regions corresponding to this color; as described above, this information can be communicated to, and visualized in, the virtual world.

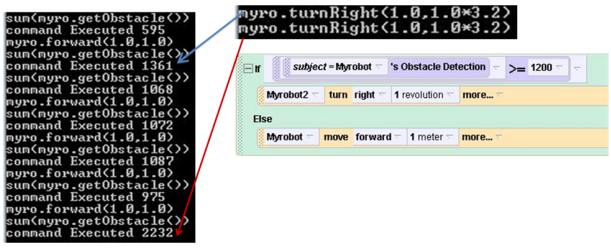


Figure 6: Illustrative example of multirobot collaboration.

A function created in the virtual interface uses the color information communicated by the robot to generate suitable movement commands for the robot; the objective is to follow the teammate assumed to exist in the segmented image region. This function currently assumes that the largest segmented region (of the desired color) is the target (i.e., the teammate) and executes actions such that this region stays close to the center of its field of view (i.e., center of image). Figure 5 illustrates how a robot identifies a teammate and chooses actions based on a simple state transition diagram to keep the teammate in view. Unknown to the robot, the teammate's actions are governed by an action policy that causes it to move around and avoid obstacles by randomly

turning left or right. In our experiments, the robot observing the teammate's motion was able to always keep the teammate in its view. The coupling between robots in the virtual world and their physical cohorts allows us to concurrently visualize this adaptive behavior in the virtual world and execute it the real world. Figure 6 shows a snapshot of this coupling between the virtual world and the real world.

Other examples can be designed for multirobot collaboration using different visual cues, other sensors, and other robot platforms. The key outcome is that this form of (indirect) communication presents new possibilities for illustrating sophisticated algorithms for challenges in planning and adaptive (or ad hoc) multiagent collaboration. Beginning programmers will thus have the opportunity to acquire sophisticated computing skills by solving complex problems.

5 Summary and Future Work

This paper described Dorothy, an integrated 3D/robotics educational tool created by augmenting the Alice programming environment for teaching core computing skills to beginning programmers. Graphical routines created in the virtual world are automatically translated into code to provide a real-time or offline enactment on one or more robots in the real world. One new contribution of this paper is the enhanced bidirectional communication between the virtual world and the real world, which allows us to visualize in a virtual world the information extracted by processing inputs from sensors mounted on a robot in the real world. Another key contribution is the use of visual input, especially color information, to model the behavior of teammates and support multirobot collaboration when the robots cannot communicate with each other. We have also developed a curriculum that can be used with Dorothy to illustrate sophisticated algorithms for fundamental challenges in robotics and AI, thus teaching core computing concepts and emphasizing the practical applications of computing to students. Dorothy has been used in pilot workshops organized for middle school and high school students. We plan to conduct qualitative and quantitative studies to evaluate the learning benefits for students using Dorothy. We will also extend Dorothy to illustrate other robotics and AI concepts, providing a more engaging learning environment for students and encouraging them to pursue advanced degrees and careers in computing.

Acknowledgments This research was supported in part by the U.S. National Science Foundation and the U.S. Department of Defense (CNS-1263183). Opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the NSF or the DoD.

References

Alice. 2014. Graphical Programming using Alice. `http://www.alice.org/`.

CPGE21. 2007. Committee on Prospering in the Global Economy of the 21st Century. Rising Above the Gathering Storm: Energizing and Employing America for Brighter Economic Future. The National Academies Press.

Dann, W.; Cooper, S.; and Pausch, R. 2008. *Learning to Program with Alice, 2nd Edition*. Prentice Hall.

Davis, J.; Wellman, B.; Anderson, M.; and Raines, M. 2009. Providing Robotic Experiences through Object-Based Programming (PREOP). In *Alice Symposium*.

Doerschuk, P.; Liu, J.; and Mann, J. 2009. INSPIRED Computing Academies for Middle School Students: Lessons Learned. In *Richard Tapia Celebration of Diversity in Computing Conference*.

Erratic-ROS. 2012. ROS package for the Erratic Robot Platform from Videre Design. `http://wiki.ros.org/Robots/Erratic`.

Greenfoot. 2009. Greenfoot: Teach and Learn Java Programming. `http://www.greenfoot.org`.

HCB. 2010. Hispanic Computer Brigade. `http://www.engr.sjsu.edu/hcb/`.

IPRE. 2010. The Institute for Personal Robots in Education. `http://www.roboteducation.org/`.

IRobot. 2014. IRobot: Robots that Make a Difference. `http://www.irobot.com/us/`.

Kelleher, C.; Pausch, R.; and Kiesler, S. 2007. Storytelling Alice Motivates Middle School Girls to Learn Computer Programming. In *SIGCHI conference on Human Factors in Computing Systems*, 1455–1464.

Kolling, M. 2009. *Introduction to Programming with Greenfoot: Object-Oriented Programming in Java with Games and Simulations*. Prentice Hall.

Lauwers, T.; Nourbakhsh, I.; and Hamner, E. 2009. CSBots: Design and Deployment of a Robot Designed for the CS1 Classroom. In *ACM Technical Symposium on Computer Science Education (SIGCSE)*, 428–432.

LEGO. 2014. LEGO Mindstorms. `http://mindstorms.lego.com/en-us/default.aspx`.

McWhorter, W., and O'Connor, B. 2009. Do Lego Mindstorms Motivate Students in CS1? In *ACM Technical Symposium on Computer Science Education (SIGCSE)*, 438–442.

NSF-BPC. 2014. NSF Broadening Participation in Computing Web Portal. `http://www.bpcportal.org/`.

Parallax. 2011. Parallax Inc.: Equip Your Genius. `http://www.parallax.com`.

PCAST. 2007. President's Council of Advisors on Science and Technology. Leadership Under Challenge: Information Technology R&D in a Competitive World. Executive Office of the President of the United States.

RCJ. 2010. The Junior League RoboCup Competitions. `http://www.robocup2010.org/competition_Category.php\?c=4`.

Scratch. 2009. Scratch. `http://www.scratch.mit.edu`.

South, D.; Shuman, M.; Thomas, K.; Ray, A.; Graham, S.; Huff, S.; Peeler, S.; Rainge, S.; Sridharan, M.; Urban, S.; and Urban, J. 2013. DOROTHY: Integrating Graphical Programming with Robotics to Stimulate Interest in Computing Careers. In *Alice Symposium*.

Summet, J.; Kumar, D.; O'Hara, K.; Walker, D.; Ni, L.; Banks, D.; and Balch, T. 2009. Personalizing CS1 with Robots. In *ACM Technical Symposium on Computer Science Education (SIGCSE)*, 433–437.

Wellman, B.; Anderson, M.; and Vrbsky, S. 2009. PREOP as a Tool to Increase Student Retention in CS. *Journal of Computing Sciences in Colleges*.

Wellman, B.; Davis, J.; and Anderson, M. 2009. Alice and Robotics in Introductory CS Courses. In *Richard Tapia Celebration of Diversity in Computing Conference*.

Wing, J. M. 2006. Computational Thinking. *Communications of the ACM* 49(3):33–35.