Enabling Affordable Industrial Robotics Education through Simulation

Prof. Scott A Kuhl, Michigan Technological University

Scott Kuhl is an Assistant Professor of Computer Science and an Adjunct Assistant Professor of Cognitive & Learning Sciences at Michigan Technological University. He received his Ph.D. in Computer Science from the University of Utah in 2009. His research interests include immersive virtual environments, head-mounted displays, and spatial perception. A link to his web page can be found at http://www.cs.mtu.edu/.

Prof. Aleksandr Sergeyev, Michigan Technological University

Aleksandr Sergeyev is currently an Associate Professor in the Electrical Engineering Technology program in the School of Technology at Michigan Technological University. Dr. Aleksandr Sergeyev earned his bachelor degree in Electrical Engineering at Moscow University of Electronics and Automation in 1995. He obtained the Master degree in Physics from Michigan Technological University in 2004 and the PhD degree in Electrical Engineering from Michigan Technological University in 2007. Dr. Aleksandr Sergeyev's research interests include high energy laser propagation through the turbulent atmosphere, developing advanced control algorithms for wavefront sensing and mitigating effects of the turbulent atmosphere, digital inline holography, digital signal processing, and laser spectroscopy. Dr. Sergeyev is a member of ASEE, IEEE, SPIE and is actively involved in promoting engineering education.

Mr. James Walker, Michigan Technological University

James Walker holds an M.S. in computer science from Michigan Technological University, where he currently performs virtual reality research in pursuit of his Ph.D. He was the lead software developer for the robotics simulator described in this paper.

Shashank Barkur Lakshmikanth, Michigan Technological University
Mr. Mark Highum, Bay de Noc Community College

Mark Highum is currently the Division Chair for Technology at Bay College. He is the Lead Instructor for Mechatronics and Robotics Systems and also teaches courses in the Computer Network Systems and Security degree. Mark holds a Master's in Career and Technical Education (Highest Distinction) from Ferris State University, and a Bachelor's in Workforce Education and Development (Summa Cum Laude) from Southern Illinois University. Mark is a retired Chief Electronics Technician (Submarines) and served and taught as part of the Navy's Nuclear Power Program. Mark is active with SkillsUSA and has been on the National Education Team for Mechatronics since 2004.

Dr. Nasser Alaraje, Michigan Technological University

Dr. Alaraje is an Associate Professor and Program Chair of Electrical Engineering Technology in the School of Technology at Michigan Tech. Prior to his faculty appointment, he was employed by Lucent Technologies as a hardware design engineer, from 1997- 2002, and by vLogix as chief hardware design engineer, from 2002-2004. Dr. Alaraje's research interests focus on processor architecture, System-on-Chip design methodology, Field-Programmable Logic Array (FPGA) architecture and design methodology, Engineering Technology Education, and hardware description language modeling. Dr. Alaraje is a 2013-2014 Fulbright scholarship recipient at Qatar University, where he taught courses on Embedded Systems. Additionally, Dr. Alaraje is a recipient of an NSF award for a digital logic design curriculum revision in collaboration with the College of Lake County in Illinois, and a NSF award in collaboration with the University of New Mexico, Drake State Technical College, and Chandler-Gilbert Community College. The award focused on expanding outreach activities to increase the awareness of potential college students about career opportunities in electronics technologies. Dr. Alaraje is a member of the American Society for Engineering Education (ASEE), a member of the ASEE Electrical and Computer Engineering Division, a member of the ASEE Engineering Technology Division, a senior member of the Institute of Electrical & Electronic Engineers (IEEE), and a member of the Electrical and Computer Engineering Technology Department Heads Association (ECETDHA).

Ruimin zhang, Michigan Technological University

I am a PhD student in Computer Science Department in Michigan Technological University. My interest is in Virtual Reality.

Mr. Mark Bradley Kinney, Bay de Noc Community College

Mark Kinney became the Executive Dean for Business, Technology, and Workforce Development in July of 2012, but first came to Bay College as the Executive Director of Institutional Research and Effectiveness in February 2009. Prior to that, Mark served as the Dean for Computer Information Systems and Technology at Baker College of Cadillac and as the Chief Operating Officer and network administrator at Forest Area Federal Credit Union. He has taught a wide range of courses in the computer information systems discipline and holds certifications in both Microsoft Excel and Microsoft Access. Mark has a Master's in Business Administration with a concentration in Computer Information Systems from Baker College, as well as a Bachelor's in Business Leadership and an Associate's of Business from Baker College. Currently, Mark is completing his dissertation in fulfillment of the requirements of a Doctorate in Educational Leadership from Central Michigan University.

Enabling affordable industrial robotics education through simulation

Abstract

Existing industrial robot training software is often too expensive for schools to provide for students or for students to acquire on their own. For example, high schools and community colleges may want to provide students with a basic level of experience with programming industrial robots. If the software is accessible and free, such training software could provide a platform for anyone to learn more about industrial robotics. In this paper, we describe the development of “RobotRun”, a software package that simulates an industrial robot and teach pendant controller. The software allows students to practice basic programming tasks which control the movement and function of the robot. When completed, this open-source program will be suitable for use in high-school outreach activities and in any degree program which focuses on industrial robotics such as two- or four-year Electrical Engineering Technology programs. RobotRun was written in the Java programming language by two students over the course of a summer. It provides a 3D view of a robotic arm, allows the use of different end effectors, and will eventually be used to simulate different factory environments and processes. In addition, the system allows students to learn about controlling the end effector in different coordinate frames and programming paths that the robotic arm should follow. The teach pendant controller resembles real teach pendants and therefore provides students with a learning experience that can be transferred to real-world industrial robotics applications. This project is a part of a larger collaboration between Michigan Technological University and Bay de Noc Community College which aims to develop curricula and training materials to supplement the RobotRun software.

Background & Introduction

Providing K-12 students and others in the workforce the tools necessary to easily learn robotics can help support the goal of improving student access to STEM-related fields and careers. There have been many publications, initiatives, and projects which have aimed to increase robotics education among students. Many of these research projects are aimed at applications such as mobile or personal robots[1–5] instead of the practical skills necessary to operate and program industrial robots. Educational experiences with industrial robotics can help train students to develop skills necessary for some jobs, and also give them a wider knowledge of robotics, programming, problem-solving techniques, mathematical principles, and cooperative learning.

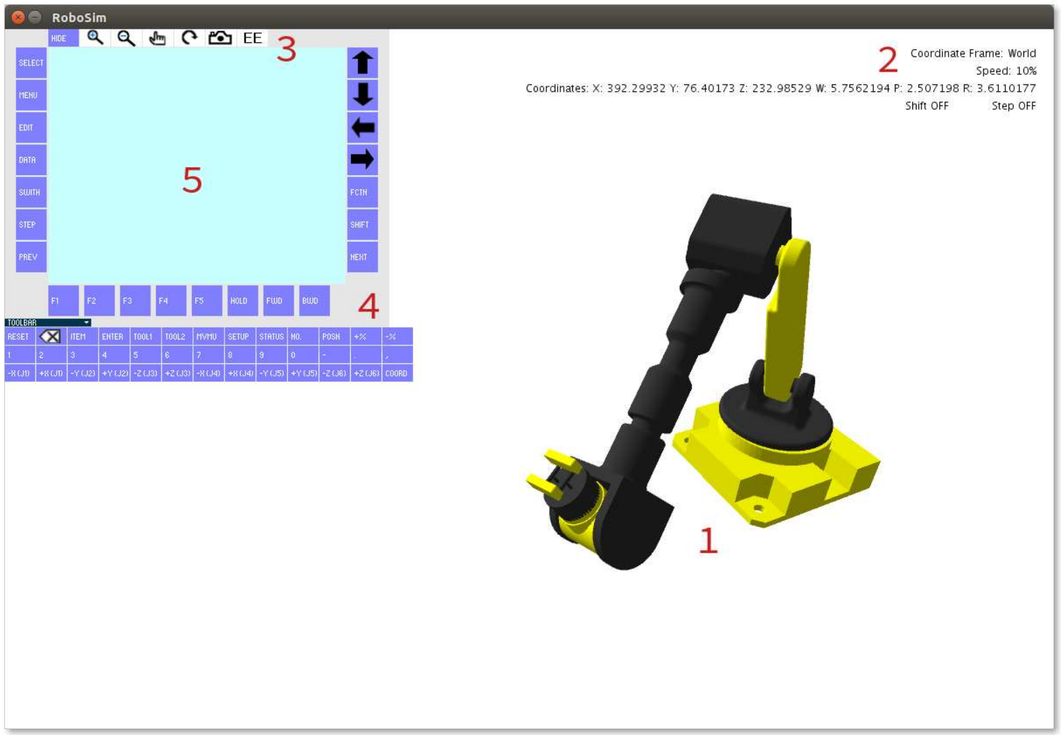
This paper describes the development of a free, open-source tool, called "RobotRun" which aims to help meet this need for industrial robotics training. This project is funded by the National Science Foundation and is a collaboration between Michigan Technological University and Bay de Noc Community College. Other parts of the project, outside the scope of this paper, include the development of training and curriculum materials that can be used by teachers to learn how to effectively teach robotics concepts using our software.

Existing software for industrial robotics remains expensive and cumbersome for many educators to use. For example, a high school teacher who wants to expose their students to basic concepts related to industrial robotics control might have to buy software that costs hundreds or thousands of dollars for a small number of licenses such as RoboLogix (Logic Design Inc.),[6] ROBOGUIDE (FANUC),[7] and RobotStudio (ABB).[8] Other free and open-source educational robotics software packages focus on providing a platform for simulating mobile or personal robots such as Gazebo.[9] All of these existing tools are either too expensive, complex, or have a focus outside of training people for industrial robotics. Our RobotRun software is functional, but is still under development. Currently, it provides a 3D view of a robotic arm, allows selection of different end-effectors, and allows the user to control the robot through a realistic teach pendant which is analogous to teach pendants used in industrial robotics. In the future, we plan to fine-tune the software based on feedback, add features, and add some common industrial scenarios which would be useful for training.

Overview of the Software

We began development on the RobotRun software in the Summer of 2015. Two Computer Science graduate students collaborated with other students and faculty in the Department of Computer Science and the School of Technology to identify the required features for the software. Although the initial version of the software is complete, additional software development is planned. The software is written in Processing (which builds upon the Java programming language) and uses OpenGL to display a real-time 3D visualization of a robotic arm. It has been tested on Windows and Linux. In the future, we hope to add support for Mac OS X.

An overview of the graphical user interface is shown in Figure 1. The figure shows (1) the robotic arm which was modeled using CAD software and imported into our program with custom software. The arm was designed to resemble the appearance of typical industrial robots. Figure 1 also shows (2) a small status display which shows the current active coordinate frame (joint or world), speed, joint rotation, and whether the "shift" and "step" buttons are on or off. A (3) utility toolbar has buttons for hiding the interface, moving the camera, making a video recording of the scene, and changing the tool/end effector. When the camera movement button is activated, dragging the mouse on the scene will change the camera's view on the robot. A (4) teach pendant is displayed on the left side of the screen which resembles real-world teach pendants. The teach pendant includes a (5) text display and all of the buttons needed to program the robot and control other functions. Figure 2 shows a close-up view of the teach pendant with a menu being displayed. The user will interact with the robot primarily through this teach pendant to allow users to practice the skills that they need to efficiently program a real-world robot.



```
Coordinate Frame: World
Speed: 10%
Coordinates: X: 392.29932 Y: 76.40173 Z: 232.98529 W: 5.7562194 P: 2.507198 R: 3.6110177
Shift OFF        Step OFF
```

Figure 1: An overview of RobotRun interface showing a robotic showing a robotic arm with an end effector (right), a teach pendant (upper left), and overlaid status information (upper right). See the text for a description of the individually labeled items in this figure.

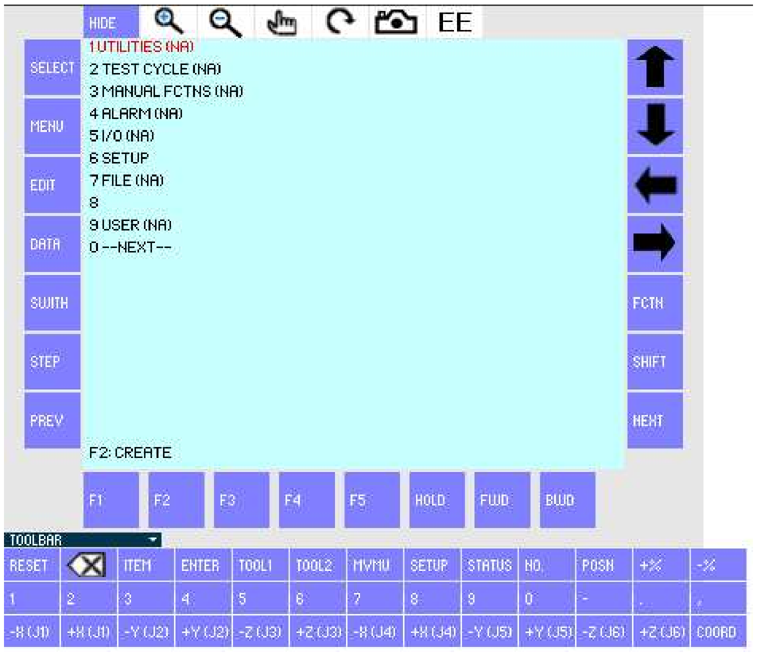


Figure 2: A closeup view of the teach pendant displaying a menu.

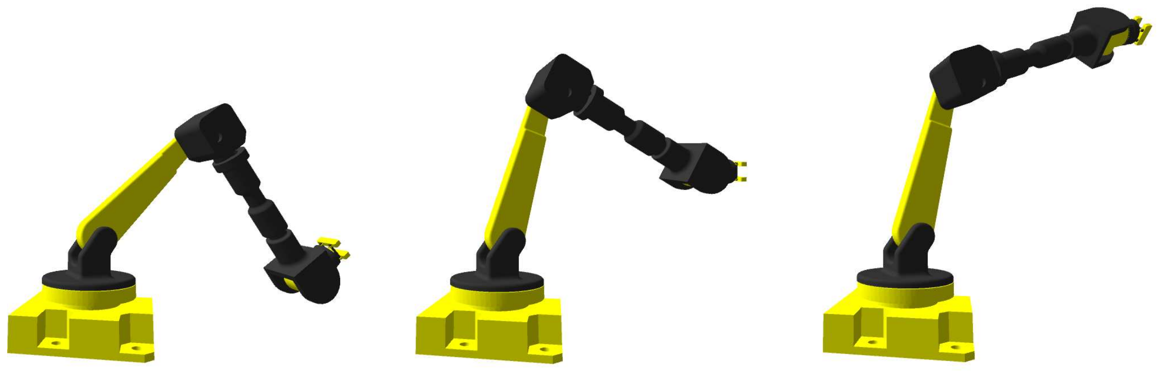


Figure 3: A series of images showing the robotic arm moving the end effector up in world coordinates using our IK implementation.

Implementation Details

In order to simulate the robotic arm, our software needed an inverse kinematics (IK) solver which enables the software to calculate appropriate joint angles which cause the end effector to be at the appropriate position. Since the primary goal of the software is to serve as an educational tool, we have implemented a basic, but functional, IK algorithm. The algorithm is a variation of Cyclic Coordinate Descent (CCD). For every movement of the robot, a series of points along the path are calculated and then CCD is performed to move the end effector through all of the points on the path. Although the existing implementation is likely adequate in most circumstances, we hope to either improve the existing IK algorithm or replace it with a different approach so that the robot's movement appears smoother, more natural, and less computationally expensive. For example, we plan to implement a Jacobian Transpose solution and compare the results. The series of screenshots shown in Figures 3 through illustrate the movement of the end effector vertically in world coordinates.

Each of the robot's joints can be jogged using the buttons at the bottom of the teach pendant. When one of the buttons is clicked with the mouse, the corresponding joint starts to move until the button is clicked again. There are also buttons to adjust the speed at which the joint changes. When set to operate in the world coordinate frame, jogging the robot changes the position of the end effector and our IK algorithm calculates appropriate joint angles. Initial support has also been added to create and save custom frames as an alternative to the world coordinate frame. This is done by having the user select three points that, together with the starting point, form approximately orthogonal line segments, which are then converted into perfectly orthonormal line segments to serve as the axes of a custom coordinate frame. The software internally converts between world coordinates and custom coordinates when performing calculations.

Since this software is intended to be an educational tool, we included a feature which allows users to record videos of their screen along with audio from the user's microphone. This allows a teacher to create videos which demonstrate how to do common tasks or allows a student to record a video of what they have completed and send it to a teacher. This feature is available if ffmpeg is installed—a free and open-source video/audio encoder and decoder.

Programs for the robot can be created, edited, and run. To create a program, the user enters a name using the keys on the teach pendant. Next, a series of points are recorded by jogging the robot and pressing the appropriate keys on the teach pendant to either record a new point or overwrite an existing point. Motion instructions can be fine (i.e., effectively linear), continuous (the end effector follows a curving path between points), or circular (the end effector traces a path along an arc). The software executes fine motion instructions by linearly interpolating between the current and next point.

For continuous instructions, assuming that the current point is A and the next two points are B and C, the software creates a curved path by linearly interpolating toward a target point that moves from B to point C as the end effector draws nearer to point B. The instruction can specify the amount of curving which controls the speed of the "moving target" from B to C relative to the movement from A to B. Our tests indicate that this approach does as good a job of creating curved paths as, e.g., splines while being less computationally intensive due to the lack of expensive trigonometric function calls.

Circular motion instructions require the user to specify three points, A, B, and C; when executed, the end effector will then trace an arc from A, through B, and to C. The path is calculated by creating a 2D coordinate system from the supplied points, finding the center of a circle whose circumference intersects those three points, calculating intermediate points with the parametric equation $P = r * cos(t) * u + r * sin(t) * n \times u + center$ where t sweeps from 0 to 2π, removing points that do not lie along the desired arc, and finally converting back into the native 3D coordinate frame.

Tool instructions can also be recorded in the program to control the tool on the end effector. We have added initial support for different end effectors; however, more work is necessary to complete this feature. The "step" and "shift" keys are used when the robot is programmed. They act as toggles where the user presses them once with the mouse to toggle the switch on or off. This simulates holding down the keys—something that can't be done via the mouse since it is impossible to click and hold multiple buttons onscreen simultaneously. As we continue development of the software, we will consider alternative methods of simulating the operation of these keys. Initial and incomplete support for user-specified frames in the programs is available.

Future Work

Although the initial version of the RobotRun software can create and execute complete programs with multiple types of instructions, we are not finished developing the software. There are many additional tasks that we hope to complete. As discussed, we still need to complete the creation of custom coordinate frames and explore possible improvements to our IK algorithm to make movement appear smoother. The software also needs additional testing from users with a variety of backgrounds to ensure that there aren't problems that consistently confuse users. We also hope to add additional end effectors and create some scenarios and objects for the robot to interact with.

Lastly, we plan to release the completed software as a well-documented, open-source project. By making the source code and the completed program freely available, we believe that we can significantly improve the accessibility of industrial robotics among a wide range of students and displaced workers. The open-source nature of the project will also make it possible for knowledgeable programmers across the world to continue to fix bugs, propose changes, and add features.

Conclusions

In this paper, we described the development of RobotRun. The software seeks to provide industrial robotics education opportunities in a way that is useful and accessible to students and teachers alike. Although work on the software is ongoing, the current system allows a user to control and program the robotic arm and end effector. The software features a teach pendant system that resembles real-world robots. By creating a realistic learning environment and providing features that will help educators, such as built-in screen-recording and audio recording features, we hope that this system will help increase the amount of robotics education opportunities in K-12 and higher education settings.

Acknowledgements

This project is supported by National Science Foundation grant DUE-1501335.

References

[1] Illah Reza Nourbakhsh. The educational impact of the robotic autonomy mobile robotics course. Technical report, Carnegie Mellon University, 2003.

[2] Namin Shin and Sangah Kim. Learning about, from, and with robots: Students' perspectives. In *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on*, pages 1040–1045. IEEE, 2007.

[3] Stefano Carpin, Mike Lewis, Jijun Wang, Stephen Balakirsky, and Chris Scrapper. Usarsim: a robot simulator for research and education. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1400–1405. IEEE, 2007.

[4] Francesco Mondada, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Adam Klaptocz, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco, 2009.

[5] Seul Jung. Experiences in developing an experimental robotics course program for undergraduate education. *Education, IEEE Transactions on*, 56(1):129–136, 2013.

[6] Robologix, logic design inc. `https://www.robologix.com/`. Accessed: 2016-01-31.

[7] Roboguide, fanuc america corporation. `http://robot.fanucamerica.com/products/vision-software/roboguide-simulation-software.aspx`. Accessed: 2016-01-31.

[8] Robotstudio, abb. `http://new.abb.com/products/robotics/robotstudio`. Accessed: 2016-01-31.

[9] Gazebo, open source robotics foundation. `http://gazebosim.org/`. Accessed: 2016-01-31.