ASEE CoED Journal

Gonzalez FG. A Hybrid Physical-Virtual Educational Robotic Arm. *ASEE COED.* 2024;14(3).

RESEARCH

A Hybrid Physical-Virtual Educational Robotic Arm

Fernando G. Gonzalez, Ph.D.[1]

[1] Computing and Software Engineering, Florida Gulf Coast University

Keywords: Adaptive Systems, Dynamics, Modeling, Robotics, Simulation-Based Learning, Software-Hardware Systems Engineering

ASEE Computers in Education

Vol. 14, Issue 3, 2025

In the field of robotics education, introductory courses would ideally utilize heavy industrial arms for hands-on learning. This would provide students with valuable experience in joint programming, which involves direct control of each joint motor in the robotic arm to accomplish desired path planning and differential movements. This practice requires consideration of the physical properties of the large arm such as its large mass. However, the use of heavy industrial robotic arms has several drawbacks. They are large and expensive, require specialized maintenance, can pose safety risks, and they typically do not allow for direct control of the joint motors. The use of a small, lightweight, toy-like arm is not suitable since their lightweight construction means they do not exhibit behavior associated with heavier arms. Robotic arm simulators using a virtual arm doesn't offer the same level of hands-on engagement and excitement as a physical arm. In this paper, a hybrid solution that combines a small physical arm with a virtual arm is proposed. The simulations controlling the virtual arm is used to dictate the behavior of the small physical arm making the small arm behave as though it is a large heavy industrial arm. This approach provides students with the experience of working with a large industrial arm, but without the associated difficulties. The hybrid approach was implemented and used in our Introduction to Robotics course where the completion rate of the two relevant homework assignments was increased from 57.0% to 78.1% and a survey of the use of the physical arm indicates students overall agree it helped them get motivated to complete the assignments and enriched their learning experience. This approach offers a promising solution for practical robotics education.

1. Introduction

This paper presents a feature that was added to an existing education robotics software tool. The tool is presented in previous works.[1-4] The work presented only adds the integration of a physical arm to motivate the students to complete their assignments. The tool simulates a virtual arm that can be set up to model an arm's movements. The students can use the platform to develop and test their joint programming software code. This virtual arm allows the students to move the arm by programming the movements of each of its joint motors. In this activity, called joint programming,[3] the students move the arm along a specified trajectory by directly controlling the instantaneous velocity of each or the arm's joint motors. Differential movements[4] is a type of joint programming where the arm is moved with

a specified constant velocity along a specified path. It is used in painting, welding and applications requiring precise velocities. Both of these types of movements require the consideration of the arm's physical behaviors. For example, a heavier arm must take more time to accelerate and must begin to reduce its velocity earlier in the path to avoid overshooting the target. Joint programming involves forward and inverse robot kinematics as well as trajectory planning, however it is an exercise in developing software code to use these concepts to program the movement of the arm in real time.

The tool developed by the author is not aimed for use in institutions with established robotics programs but rather for programs that simply are offering the Introduction to Robotics course as a technical elective. This course is a very popular technical elective since it is multidisciplinary and covers many different areas of engineering and computer science. The tool the author developed is designed to focus on software engineering and computer science programs or programs where the student's primary education is in software development. This is the rationale for supporting joint programming software development activities.

The educational software tool has been designed to allow users to input the specifications of the robotic arm, including its physical properties. The user can provide the Denavit and Hartenberg (DH) parameters for each link to specify the arm's kinematic characteristics. The same input interface allows the user to input the maximum acceleration for each joint motor in both increasing and decreasing velocity scenarios, as well as the maximum speed. These values define the quickest rate at which a joint motor can alter its velocity and its highest attainable speed.

Currently, the virtual arm is represented using a stick figure format because it must adapt to various kinematic configurations, as defined by the user-provided DH parameters. However, the graphical representation of the virtual arm could stand to benefit from enhancements.

Joint programming is a technique used to maneuver a robotic arm along a smooth path, with the goal of positioning its end-effector to a specific location and orientation while avoiding obstacles or exceeding physical or imposed limits. This programming task entails calculating velocity polynomials for each joint motor such that the collective set of polynomials governs the arm's trajectory. If a joint motor is unable to adhere to its prescribed velocity schedule, the end-effector's velocity and trajectory will be altered, potentially putting the arm at risk of colliding with obstacles. Thus, it is essential that the velocity schedule for each joint motor is achievable. Failing to consider, or inadequately accounting for, the robotic arm's limitations can lead to unattainable velocity schedules. For instance, a joint of the arm might not be able to accelerate as rapidly as the schedule demands.

Each velocity polynomial is computed by solving a set of simultaneous equations to determine all the polynomial's unknowns. These equations are formulated based on constraints such as the initial and final angle and velocity the joint should have, along with a maximum acceleration limit. The number of constraints determines the order of the polynomial. A common approach involves using blended polynomials that have an acceleration period, a cruising period, and a deceleration period, much akin to driving a car. Each blend is a separate polynomial that needs to be computed. If the arm's limitations are not considered, the blends may be overlooked, simplifying the problem to a trivial state.

Small toy robotic arms, while cost-effective and generally work well, are lightweight and thus exhibit behaviors that allow for trivial joint programming. Their rapid acceleration capabilities mean that their acceleration limits can often be overlooked while still achieving a satisfactory trajectory. However, this can present a problem if a student, trained in joint programming using a small toy arm, then attempts to program an industrial arm; their methods might not be applicable. The ideal solution is to make the small toy arm emulate the behavior of a large, heavy industrial arm, thereby necessitating proper consideration of its limits to successfully maneuver the arm.

This paper presents a solution involving the integration of a small physical arm, namely the DOBOT, with an educational virtual arm platform, to provide the experience of programming a heavy industrial arm. The tool was also integrated with an arm that utilizes Hitec servo motors,[5] commonly used in small educational arms or robotic kits.

The work presented in this paper presents the integration of a physical but small robotic arm to our existing robotics educational tool that uses a virtual simulated robotic arm. The existing tool is designed to support teaching and learning concepts that are typically found in introductory robotics courses. The textbook titled "Introduction to Robotics,"[6] was used as a guide in the development of this tool. The tool supports topics including the DH parameter and frame placement convention, forward and inverse kinematics, trajectory planning and differential movements, both part of a general topic of joint programming, and robotic vision. The tool offers the flexibility to model any robotic arm that can be specified by its DH parameters. The virtual arm is created by entering its DH parameters, its range of movement, the types of joints either prismatic or revolute and the limits on velocity and acceleration. These limits are used to model the characteristics of the arm or the constraints of the movement problem.

The purpose of this tool is to motivate the students in completing their trajectory planning programming assignments. It does this by allowing a physical robotic arm to track the movements of the virtual arm thereby giving the students a sense that they are programming the movements of

a real arm. And since the movements are governed by the characteristics of the virtual arm, the small toy arm can be made to move in a way that resembles the movements of a much larger industrial arm see.[1,2,7] Although the physical arm may be small in comparison to an industrial arm, it still has linkages, actuators, sensors, gears, and many moving parts that are common in larger industrial arms. Watching this mechanism perform as the arm moves is motivating to the students who are generally curious as to how a real robotic arm works. This is the power of this work, motivating the students by providing a real physical arm that, while small and economical, still behaves as a large industrial arm.

2. Literature Review

There are many robotic simulators[8-12] that are both commercial and free. These can be used for educational purposes however their arms are controlled through higher level commands that do not lend themselves to joint programming. Our tool is specifically designed for this introductory robotics course and the activities that it can support are aligned with the course's textbook.

The following review of educational robotic platforms,[13] presents 27 simulation-based platforms referred to as virtual robotic platforms and 15 real platforms or platforms that involve physical robots. The survey did not present any hybrid platforms or platforms that use a real physical robotic arm and a simulated virtual arm at the same time, the configuration we are presenting. Of the 27 virtual platforms, 10 support forward and inverse kinematics and the other 17 support robot dynamic and control. The survey presents 15 platforms based on real physical arms. They ranged in price from $29.00 to $48,403.00. Some of these are small arms sometimes referred to as toy arms while other are heavier industrial arm. While this survey presented a total of 42 educational robotics platforms, it did not present any hybrid platforms. That is, the platforms either implement a virtual arm or a physical arm but no combination of the two. In contrast our tool supports the control of a physical arm while also using simulations to enforce the physical arm to exhibit specific behaviors such as that of a larger arm.

There are software tools that support the integration of physical arms however they are generally focused on K-12 and do not support learning advanced concepts. In Zhong, Zheng, and Zhan[14] they present the IRobotQ3D robot simulator where students use the Lego Mindstorms kits to build the robot first then program the steps that will make it move. In[15] they present a tool that uses a physical and virtual robot combination. However, the design of the arm is not by its DH parameters but rather by adding wheels and other physical parts and the programming is not at the joint motor level but higher. Their tool is aimed at the K-12 group. In Tijani et al.[16] and Cheluszka[17] they use small inexpensive educational robots but their implementation is also aimed at K-12 and does not allow for joint level

programming. In Nutakki et al.[18] they present a small robotic arm that can be controlled remotely via a web server however it also does not allow joint level programming. In addition, a reliable and high-speed internet connection must be available to the students.

There are also tools aimed at higher education but they are generally not considered to be a full integrated development environment (IDE) as is the tool describes in this paper. In Robinette and Manseur[19] they focus on the DH parameters and robot kinematics. It allows the student to specify the DH parameters and the web-based application draws the corresponding arm. This tool does not support joint programming. In Corke[20] they developed a MATLAB Toolbox for Robotics made freely available. This library is popular but is not an IDE. It requires the student to write programs in MATLAB. The library supports the implementation of a virtual arm but the students need to link the virtual arm into their program. While this tool is well suited for students pursuing an education in a robotics discipline, it is not well suited for a student taking a simple intro to robotics course as an elective. The learning curve to learn MATLAB and implement their virtual arm is much higher than what is needed for the presented tool. This tool like others also perform much of the work that the students should be implementing instead. For example, Corke's tool has functions to compute a trajectory whereas in the presented tool the student must implement a program to compute the trajectory.

Small light weight arms are also commonly used. In Indri, Lazzero, and Bona[21] they present one such implementation that uses standard LEGO Mindstorm Kits. This platform is intended for K-12. In Manzoor et al.[22] they present a robotic platform consisting of a physical 6 DOF arm with several sensors attached. The sensors include a camera and gripper pressure sensor. The sensor setup allows students to program the arm to see, feel, and react. Like many of the platforms employing a physical arm, there arm is light weight. Since our tool is focused on joint programming, the heavy mass associated with large industrial arms is necessary to allow the student to move the arm along a smooth trajectory considering the limits on acceleration and velocity. Therefore, in our hybrid approach, we also use a small lightweight arm but simulate the mass of the larger arms.

Large physical arms are not designed for educational purposes and generally come with a controller that performs all of the joint programming. They generally do not allow the student to bypass the controller to gain direct access to the joint motors due to liability and safety reasons. However, one industrial arm does allow for joint programming, the Franka Emika robot[23] the controller does not perform the joint programming but does supervise the movements to ensure safety. While this may seem like the perfect solution, being a large industrial arm, it comes with a high initial and maintenance cost, requires dedicated lab space, and poses dangers to humans who get in the way

of the arm. Furthermore, while they support trajectory planning, they do not support differential movements which is also part of joint programming. To be useful in education, differential movements must leave evidence of correct movement that cannot be observed by simply looking at the movement. For example, the presented tool can paint on a virtual canvas and leave evidence of correct movement by looking at the resulting painted canvas. This can be performed while a physical arm is tracking the movements of the virtual arm. The canvas will still be virtual.

The work in[22] shows how educational robotics can impact learning interests and attitude toward learning. They found a 21% increase in interest and a 9.8% increase in the student's attitude towards their physical education course. This is similar to the 21% increase in completion rate related to the corresponding homework in the work presented in this paper.

3. Methods and Context

We chose the DOBOT robotic arm,[24] for this application since it's small and inexpensive yet it's of high quality having good accuracy, repeatability, durability, and reliability. Our educational software tool however can integrate with many small robotic arms. Some customization may require some new code to be added to the software tool as was the case using the DOBOT.

The solution is to simply have the physical arm track the movements of the virtual arm. The simulation engine that models the virtual arm runs in a cycle. In each cycle the virtual arm sends messages to the physical arm updating its current location. Depending on the arm and its controller, the location update may consist of a set of updated joint angles or an updated location for the end-effector. Figure 1, Figure 2, and Figure 3 show examples of the DOBOT arm tracking the virtual arm for different angles.

The simulation engine determines the location of the virtual arm at every cycle. The cycle executes 10 times per second. In each cycle the simulation engine determines the new location of each joint in the arm and renders the arm in its new location. It then communicates with the physical arm and provides the updated arm location. Figure 4 shows the algorithm that is executed in every simulation cycle and Figure 5 shows the computation of the velocity and position considering the modeled characteristics of the arm.

If the physical arm can be controlled at the joint level, the simulation engine will send it the set of newly computed joint angles. The physical arm can then move its joint motors using these angles. However, if the physical arm has a controller that does not allow direct access to its joint motors, then the location of the end-effector is given to the controller. The controller then computes all of its joint angles using its inverse kinematic equations and moves the arm accordingly. This works behind the scenes and is transparent

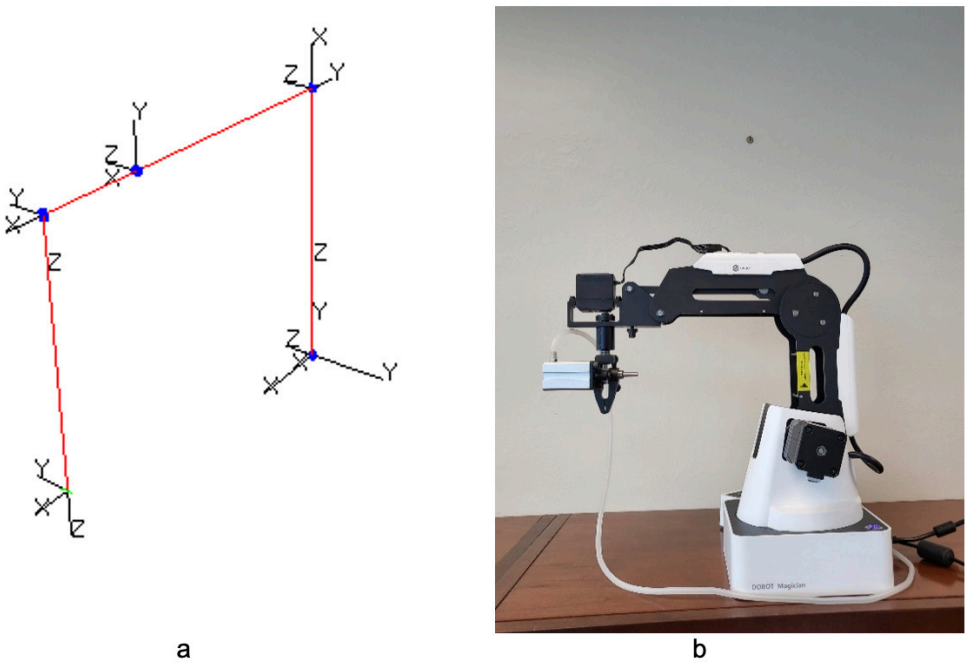


Figure 1. Example of the DOBOT tracking the virtual arm. (a) the virtual arm. (b) the DOBOT arm. The current positions for both arms are $\theta_1 = 0°, \theta_2 = 0°, \theta_3 = 0°$.

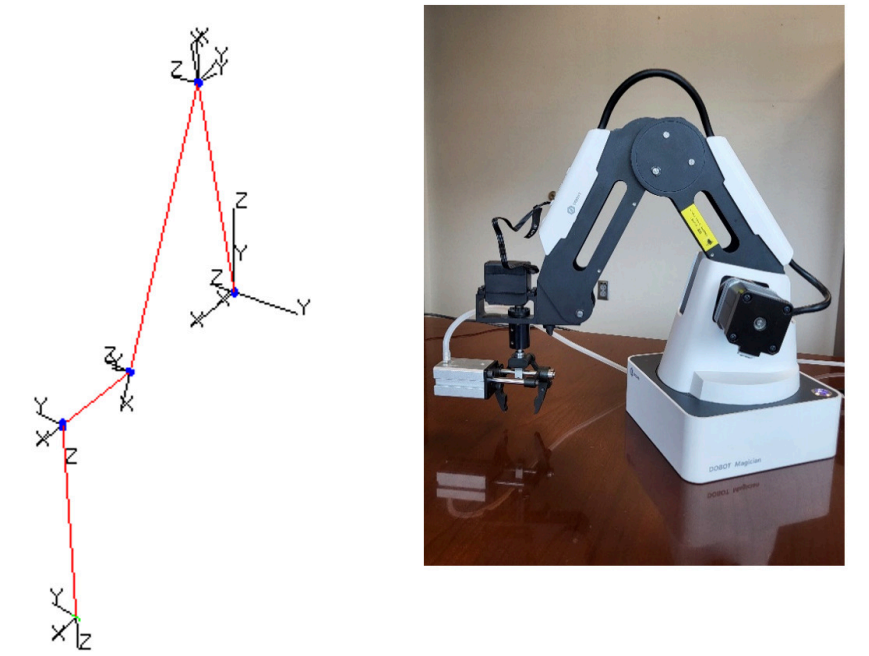


Figure 2. Example of the DOBOT tracking the virtual arm. (a) the virtual arm. (b) the DOBOT arm. The current positions for both arms are $\theta_1 = 0°, \theta_2 = 17°, \theta_3 = 60°$.

to student's view. The student programs each joint individually using joint programming and the physical as well as the virtual arm moves according to their program.

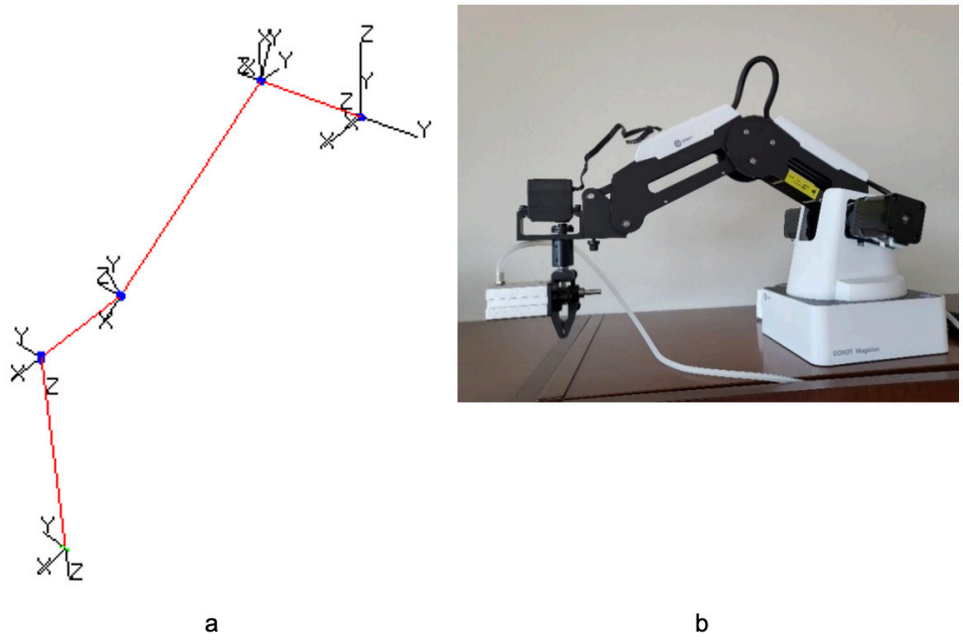


Figure 3. Example of the DOBOT tracking the virtual arm. (a) the virtual arm. (b) the DOBOT arm. The current positions for both arms are $\theta_1 = 0°, \theta_2 = 60°, \theta_3 = 25°$.

```
Update the current time, TNOW, to be TNOW + 0.01 seconds.

for each joint
     Compute the desired instantaneous velocity by evaluating the velocity
          polynomial at time TNOW.
     Compute the actual instantaneous velocity by considering the desired
          instantaneous velocity and the joint's physical limits.
     Update the joint angle based on the new instantaneous velocity.

Render the arm using the newly computed joint angles.

Pass the new joint angles or the new location of the end-effector to the physical
     arm.
```

Figure 4. Algorithm to update the location of the end-effector. Its executed every 10th of a second.

```
Acceleration = (DesiredVelocity - Velocity) / DeltaTime;

     if (Acceleration > MaxAccelaration)
          Acceleration = MaxAccelaration;
     else if (Acceleration < -MaxAccelaration)
          Acceleration = -MaxAccelaration;

       PrevVelocity = Velocity;

       Velocity = Velocity + Acceleration * DeltaTime;

       Position = Position + ((Velocity + PrevVelocity) / 2) * DeltaTime;
```

Figure 5. Algorithm to compute the position of a joint considering its maximum acceleration. DeltaTime is 0.01 seconds since the algorithm executes 10 times per second.

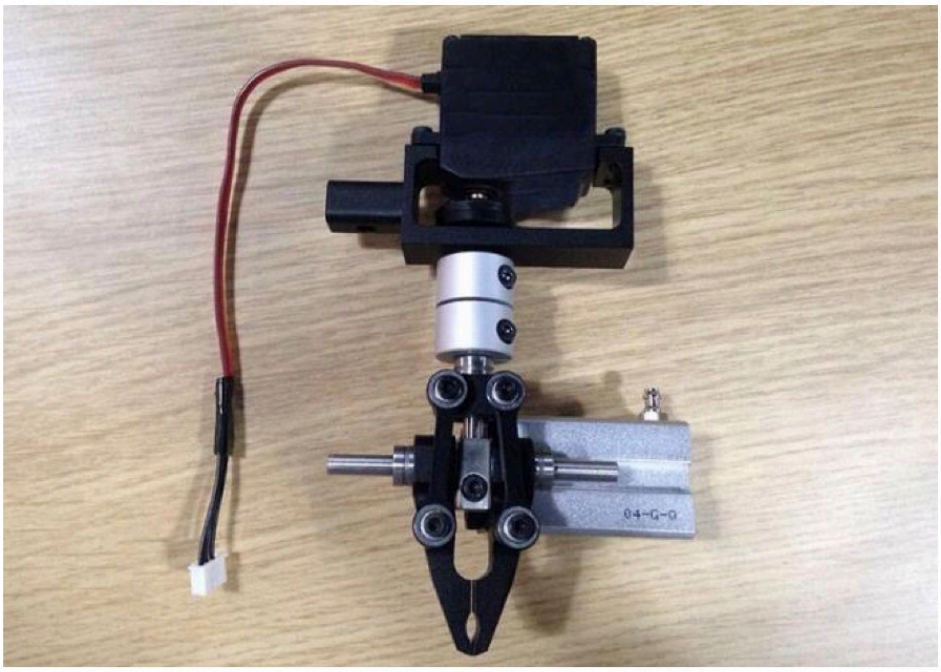


Figure 6. The DOBOT gripper assembly. The top black box is a Hitec servomotor that rotates the gripper along the approach axis and the bottom silver box is the air cylinder that opens and closes the gripper using an air pump also included in the DOBOT kit.

The DOBOT Arm

In our application we used the DOBOT Magician robotic arm, see Figure 1, because it is small and inexpensive yet relatively strong and precise. The arm is made of steel and its joint motors are implemented using relatively large stepper motors. It appears to be of high quality will endure the use in the lab. It is designed mostly for educational purposes.

The kit includes a gripper and a suction cup for its end-effector. In our application the gripper was chosen since it can open and close as well as rotate see Figure 6. The basic arm has 3 degrees of freedom (DOF), a rotating base, a rear arm and a fore arm. The rotating gripper is considered a 4^{th} DOF if its installed and since it opens and closes that adds half more resulting in a 4.5 DOF in the configuration used in our application.

The DOBOT comes with a controller that is implemented using an Arduino microcontroller that can get connected to a personal computer (PC) via a USB cable. It includes an Application Programming Interface (API) that runs on the PC and contains function calls to pass commands to the controller. The commands are generally used to move the arm.

The process of integrating the DOBOT arm is first to model the arm. That is, determine its DH parameters, the direction of rotation for each joint, and any characteristic that does not follow the DH convention. Then in the second step the communications between the virtual and the DOBOT arm is established.

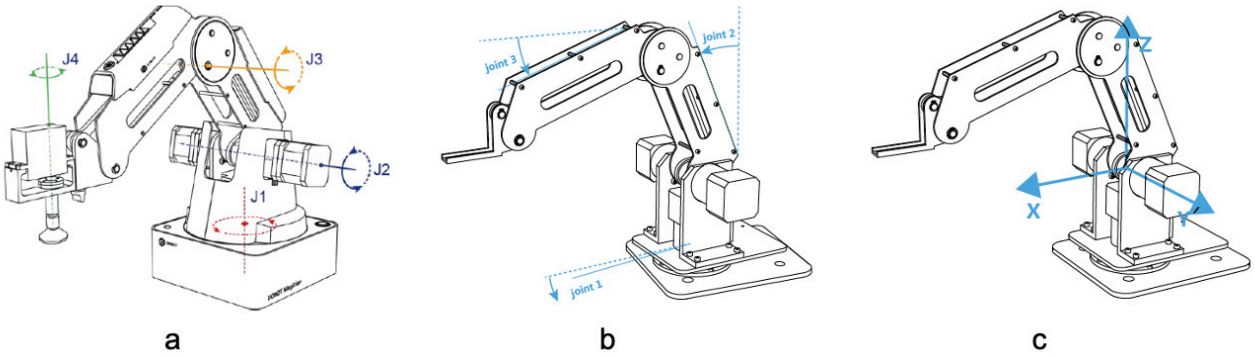


Figure 7. (a) the joint coordinate system of the DOBOT, (b) the direction of the joints, and (c) the origin of the coordinate system.

The first step; modeling the DOBOT.

The first step in integrating the DOBOT arm to the software tool is to determine its DH parameters so we can create the virtual model. The three joints of the DOBOT are shown in Figure 7 (a). The direction of rotation is shown in Figure 7 (b) and the origin of its coordinate system is shown in Figure 7 (c). Note the origin is not at the base of the arm but rather at the intersection of the axis or rotation of joints one and two. The floor the arm is resting on is at a negative Z position.

The DOBOT arm has two mechanically moving joints. The rear arm rotates the fore arm in such a way that rotations of the rear arm does not change the orientation the fore arm has with respect to the world frame. The fore arm also has its own joint motor and can rotate independently. The resulting orientation of the fore arm is a function of its own joint angle and that of the rear arm. The wrist is also mechanically linked to the fore arm in such a way that the wrist is always horizontal. It does not have a separate joint like the fore arm so rotating it to change its pitch is not possible. Figure 8 shows how the forearm rotates with rotations of the rear arm such as to stay horizontal. Without considering the mechanically linked joint, the DH convention will place the fore arm rotating by the same angle as the rear arm since its orientation with respect to the rear arm will stay the same.

To accommodate the mechanical links to conform to the DH parameter convention, the fore arm joint is represented by two joints, one controllable and the other uncontrollable. Imagine that the fore arm's joint motor is not mounted to the rear arm directly but rather to a plate. This plate is mechanically linked to the rear arm and rotates with rotations of the rear arm. The wrist is modeled as a joint that is not controllable but rather rotates with rotations of the fore arm. The DH parameters determined are represented in Table 1 below.

Figure 8. (a), the fore arm is horizontal. (b), the rear arm is rotated and the fore arm rotates to maintain its horizontal orientation with the horizon.

Table 1. DH parameters for the DOBOT Magician arm. Units are in mm and degrees.

Link name	θ	d	a	α
Base	θ_1	0	0	90
Rear arm	$90 + \theta_2$	0	135	0
Fore Arm Mech		0	0	0
Fore arm	$-90 + \theta_3$	0	147	0
Wrist Mech		0	59	90
Gripper	θ_4	110	0	0

The 90° added to θ_2 is to make the home position, Figure 8 left, the position with all thetas equal to zero, point up as opposed to straight out horizontal. The -90° added to θ_3 is to position the forearm horizontal as opposed to straight up in the home position. The DH convention using the DH parameters shown in Table 1 will have the rear arm and forearm rotating in the opposite direction as the DOBOT so their angles must be negated. The software tool allows the user to enter the following information to model an arm, see Figure 9:

1. The joint name
2. The joint type either revolute or prismatic
3. The direction or movement
4. Whether its controllable or uncontrollable
5. The four DH parameters
6. The range of motion for the joint
7. The maximum acceleration the joint can have and

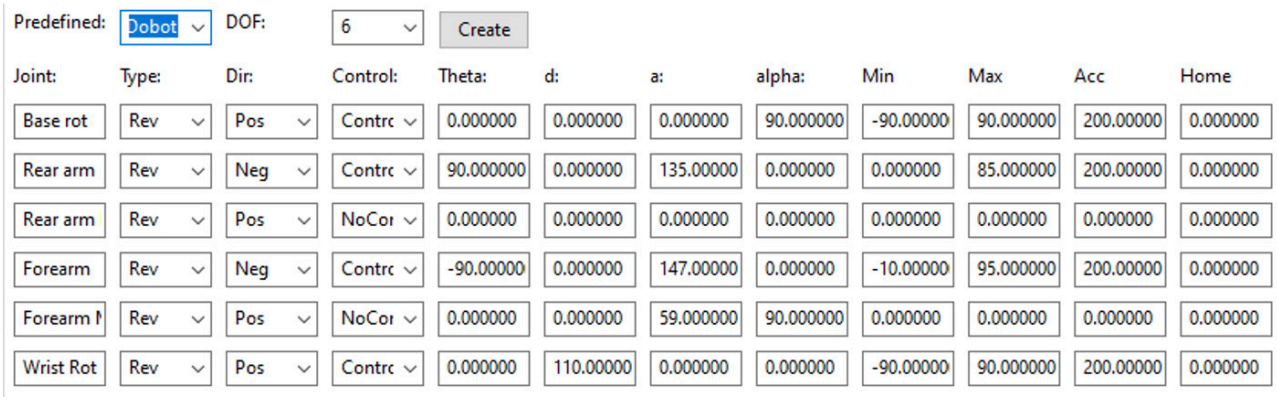


Figure 9. The user input screen used to specify the kinematics and movement limits of the arm.

8. The home position of the arm.

The DOBOT arm was added to the list of predefined arms so students only need to select the proper arm. Note the DOBOT is modeled as having 6 DOF as opposed to 4 since two uncontrollable mechanical joints were added to represent the mechanical linkages between some joints.

Figure : The user input screen used to specify the kinematics and movement limits of the arm.

The only software enhancement required to model the DOBOT was the inclusion of a function to manage the uncontrollable joints. This function, executed in each simulation cycle, calculates the values of the uncontrollable joints based on the values of all the controllable joints. In the case of the DOBOT, it simply assigns the value of the controllable rear arm to the uncontrollable rear arm, and the value of the controllable fore arm to the uncontrollable fore arm.

The second step, establishing the communication protocol with the physical arm

The next stage of integrating the robotic arm into the software tool involved researching the various methods the controller can use to manipulate the arm. The arm comes with a built-in controller and an API that can be linked into the software tool. Control of the arm is achieved through API function calls. Three API functions were utilized for this purpose: Jog, Point-to-Point (PTP), and Continuous Path (CP). The API has a command queue, which ensures that existing commands are completed before new ones are initiated. Function calls are available to clear this queue. Three types of movement function calls were explored, each one implemented into the tool and made to track the virtual arm.

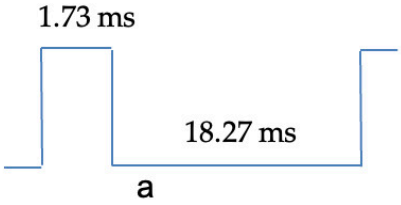
Jog: This function allows direct control of each joint, even allowing the user to input the desired velocity for the joint. However, only one joint can be controlled at a time by the user. Given that the real-world movement of an arm involves several joints moving at once, this method isn't effective as it stands. An attempt was made to make the joints move in sequence, albeit one at a time, with each clock cycle instructing a different joint motor to move. However, this resulted in a very jerky movement, accompanied by loud noise and significant vibrations. This is not how actual industrial arms operate, and such movement could cause damage to the arm over time. This method was therefore ruled out.

PTP: This method allows all joints to move concurrently and lets the user specify movements at the joint level. This method enables selection between the Cartesian space and joint space, with the latter being more suitable for this application. The input here is the desired new location, rather than the desired velocity. Every time a PTP movement is requested, the DOBOT's controller creates a 2-1-2 type blend path, starting and ending with a velocity of zero and cruising at the specified velocity. However, the jerkiness issue persisted with the PTP method due to the requirement for each path to start and end with zero velocity and our tool operating in a cycle executed 10 times per second. Various remedies were tried, such as flushing the queue with every new PTP movement, and providing new movement commands before the completion of the last one. However, these fixes only resulted in intermittent improvements and the arm still had periods of sudden stops and starts.

CP: Similar to the PTP method, this is designed for longer continuous movements. The API doesn't stop the arm when a movement is completed unless the command queue is empty. This resolves many issues the PTP method had, but it required movements to be specified in Cartesian space rather than joint space. Using the forward kinematic equations, the desired location of the end-effector in Cartesian space is computed and given to the API. This was the method used in our tool. For smooth arm movement, the velocity had to be greater than 100 degrees per second. Movement at a lower speed resulted in jerky motion, and movement at a higher speed made it hard for the API to keep up and the DOBOT would lag in movement. The main limitation was the USB communication that had to occur 10 times per second. However, in the comfortable range of 100 to 200 degrees per second, the DOBOT arm moved with smooth trajectories and was able to track the virtual arm effectively.

The Hitec servomotors

In[1] we used a robotic kit from Pitsco,[25] called the Tetrix Prime[26] in a similar fashion. The students in this summer camp were of middle school age and they programmed the arm using a higher-level robotic language that does not involve joint programming. However joint programming can be used with these arms. In this application the students design and built their own



$$7.5\% + 35° \frac{10.5\% - 4.5\%}{90° - (-90°)} = 7.50\% + 1.167\% = 8.67\%$$

b

Figure 10. (a) the PWM signal used to move the servomotor's position to 35 degrees, (b) the calculation for computing the required duty cycle for a 35-degree angle.

arm. They measured the DH parameters and created the virtual arm that corresponds. Then they programmed the virtual arm using a dedicated robot language and their physical arm moved tracking the movements of the virtual arm.

In order for this integration to function effectively, a dedicated microcontroller was required. This microcontroller received information regarding the desired position of each joint and generated electrical signals to the Hitec servomotors. For this task, we employed an XPlained board by Atmel, which utilizes the Atmel XMEGA-A3BU microcontroller. However, microcontrollers used in Arduino or Raspberry Pi could be more suitable choices. The Arduino, for instance, also employs an Atmel microcontroller, albeit a smaller one, and is complemented by an operating system and programming environment that is generally easier for users to learn.

Many small robotic arms used for educational purpose and toys use the Hitec servomotors. They are cheap and can be easily controlled. The Hitec servomotor has three wires, a positive input voltage of 5V, a ground and the control signal. The control wire takes a pulse width modulated (PWM) wave as input where the duty cycle tells the angle the motor is to turn to. The PWM signal must be between 3 to 5 volts and cycle at 50 Hz. The motors we used has a 180° range, (-90° to 90°). The duty cycle range is from 4.5% to -90° to 10.5% for +90°. Figure 10 shows an example of the input PWM wave needed to move the servomotor to 35°. The duty cycle must be 8.67% in this case or 1.73 ms high and 18.27 ms low. Note a frequency of 50 Hz corresponds to a period of 20 ms, $\frac{1}{50} Hz = 20ms$.

4. Results

The objective of this project is not to provide a platform for students to execute their joint programming projects, but rather to motivate them to complete these assignments. The educational software tool's virtual arm already serves as an adequate platform for students to develop and test their programs, effectively simulating a real arm, including large ones where their large mass must be considered. The integration with the DOBOT arm serves only to encourage students to complete their assignments by allowing them to interact with a real physical arm that performs actual physical movements. It is assumed that allowing the students to interact with a real physical arm as

Table 2. The homework grades for the Fall 2021 and the Fall 2022 in Introduction to Robotics.

Semester	HW 1	HW 2	HW 3	HW 4	HW 5	HW 6	HW 7	Average All other HW	Average HW 4 & 5
Fall 2021	92.59	88.46	80.77	55.77	58.27	79.81	65.38	81.40	57.02
Fall 2022	81.25	81.25	81.25	75.00	81.25	81.25	81.25	81.25	78.13

opposed to using computer simulations, tends to excite the students. It is also assumed that this excitement will encourage the students to complete their related homework assignments.

Our university does not allow students within a course to be treated differently. Therefore, separating the students into groups where one group uses the DOBOT and the other does not is prohibited. Instead, a comparison was made between two separate classes. The first class did not use the physical arm and served as the control group while the second class offered the year after did use the physical arm. Since our university only offers this class once per year, producing a sample set with statistical significance may take an unreasonable number or years. While a sample set consisting of only two classes is not statistically significant, given the assumptions made above, it is felt that showing feasibility and promise in the technology developed is sufficient at this level of implementation.

The course, CAP 4662 Introduction to Robotics, was offered in the Fall semesters of 2021 and 2022. In Fall 2021, the added feature of the DOBOT arm was not yet implemented, so students could only utilize the virtual arm. However, in Fall 2022, the new feature was added, integrating the DOBOT robotic arm into the tool. This allowed students to use the DOBOT arm in conjunction with this tool for their assignments. Homework 4 focused on joint programming, and homework 5 dealt with differential movements, both of which required the use of the DOBOT robotic arm. The remaining assignments did not involve programming or the use of the DOBOT arm. As shown in Table 2, the completion rates for both Homework 4 and 5 increased by about 21% after the integration.

The student group consists of Junior and Senior students in our Software Engineering undergraduate program. The program consists of about 18% female and 25% Hispanic. About 24% are first generation students.

A survey was also conducted and given to the class that used the DOBOT arms. The survey asked four questions shown in Table 3 along with the results. Nine of the 16 students responded to the survey.

5. Discussion

Five DOBOT arms were made accessible to students in a supervised, open lab. Students were required to bring their own laptops equipped with the installed software tool and connect the DOBOT via a USB cable. The

Table 3. The survey questions and their average response, strongly agree = 4, agree = 3, neutral = 2, disagree = 1, and strongly disagree = 0.

Number	Question	Average reply
1.	Using the DOBOT physical arm helped motivate me to complete the assignment.	3.2
2.	Using the DOBOT physical arm made the assignment feel more like programming an actual industrial arm.	3.6
3.	The use of the DOBOT physical arm enriched the robot joint programming experience.	3.2
4.	I will like to see more assignments use the hybrid virtual/physical arm technology.	3.1

software's connection to the DOBOT is straightforward and reliable. To establish a connection, the student simply needs to switch on the DOBOT and click the 'Connect' button on the connection screen. Upon initiating the simulation of the virtual arm, it synchronizes with the DOBOT, which subsequently mirrors the movements of the virtual arm. Two of the 7 homework assignments required the use of the virtual arm in the software connected to the DOBOT arm. In these assignments the students program the movement of the virtual arm which then moves the physical arm. Historically a sizable percentage of the students decide to skip these two assignments. In the first course that did not use the physical arm, 57.02% of the students chose not to complete the two assignments. In the following year when the student used the physical DOBOT arm, 78.13% of the students completed the assignments. While this data consisting of only two classes is not statistically significant, an improvement of 21% does show promise.

While the two courses had the same modality, set of homework assignments and policies, it is possible that other factors contributed to the change in motivation. To estimate the impact of other uncontrollable factors influencing motivation, the average completion rate for the other assignments, excluding Homework 4 and 5, were looked at. It was noted that the other homework assignment's participation rate remained relatively unchanged from Fall 2021 to Fall 2022 suggesting that the student cohort, environment, and other variables were similar across these two semesters. Therefore, the increased participation in Homework 4 and 5 can likely be attributed to the incorporation of the DOBOT arm.

A survey was conducted and the results shown in Table 3 indicates that on average the students agree that the DOBOT helped motivate them to complete the two relevant homework assignments and enriched their learning experience.

Overall, the project was successful in motivating the students to complete their relevant homework assignments. The percent of completion was increased by 21% and the students indicated the use of the DOBOT enriched their learning experience.

6. Conclusion

The application of a robotic arm simulator that manifests a virtual arm possesses numerous benefits. These include providing a learning platform without the expenses and complications associated with operating a large industrial arm. The employment of industrial arms is often not feasible, as they typically prohibit direct manipulation of their joint motors, thus rendering joint programming infeasible. Small, cost-effective arms, while accessible, do not capture the characteristics of an industrial arm. Their lack of significant mass renders them ill-suited for joint programming exercises. However, using a physical arm, whether industrial or smaller educational models, delivers an authentic experience that can engage students, motivating them to complete assignments and enriching their overall course experience.

This paper introduces a hybrid solution where a small, inexpensive educational robotic arm is incorporated into our simulation-based educational tool, replicating the behavior of a heavy industrial arm. This offers the tangible experience of programming a real physical arm, bypassing the difficulties tied to utilizing a heavier industrial counterpart. This feat was achieved by synchronizing the small arm with the virtual arm, creating an illusion of the student programming the physical arm directly. The virtual arm simulates the characteristics of a larger arm and thereby imposes those characteristics onto the smaller physical counterpart.

The Introduction to Robotics class was conducted in Fall 2021 using only the virtual arm, and later in Fall 2022 with the integrated physical arm. We noted that the class using the physical arm showed a completion rate of 78.13% on two joint programming assignments, which is a 21% increase from the 57.02% completion rate observed in the class that used only the virtual arm. No other assignment registered an increased completion rate, suggesting a comparable cohort of students.

Our future objectives include adapting the DOBOT's controller to support direct joint programming, which is anticipated to enhance the tracking process. Furthermore, we intend to improve the visualization of the virtual arm. The ultimate aim is to enable the virtual arm to emulate the physical arm with high fidelity, moving beyond its current simplistic representation

Submitted: December 05, 2023 EST. Accepted: July 15, 2024 EST. Published: December 31, 2024 EST.

REFERENCES

1. Gonzalez F, Zalewski J. An Educational Tool to Support Learning Robot Vision. *Trans Tech STEM Educ*. 2016;1:60-82.

2. Gonzalez F, Zalewski J. A New Robotics Educational System for Teaching Advanced Engineering Concepts to K-12 students. *Computers in Education Journal*. 2016;26:61-73.

3. Gonzalez F, Zalewski J. Teaching Joint-Level Robot Programming with a New Robotics Software Tool. *Robotics*. 2017;6(4):41. doi:10.3390/robotics6040041

4. Gonzalez F. Learning Robot Differential Movements Using a New Educational Robotics Software Tool. *Education and Information Technologies*. Published online 2022.

5. Hitec Servos. Accessed July 12, 2023. https://www.hitecservos.com/

6. Niku S. *Introduction to Robotics: Analysis, Control, Applications*. 2nd ed. Wiley; 2011.

7. Gonzalez F, Zalewski J, Pinzon G. An Educational Tool to Support Introductory Robotics Courses. In: *Proceedings of the 122nd ASEE Annual Conference*. ; 2015:14-17.

8. GAZEBO Robot Simulation Software. Accessed July 14, 2023. https://gazebosim.org/home

9. Robologix Logic Design Inc. Accessed July 14, 2023. http://www.robologix.com

10. Webots 7 Professional Mobile Robot Simulator. Accessed July 14, 2023. http://www.cyberbotics.com

11. Robotics Developer Studio. Accessed July 14, 2023. http://msdn.microsoft.com/en-us/library/bb648760.aspx

12. Schluse M, Priggemeyer M, Rossmann J. The Virtual Robotics Lab in education: Hands-on experiments with virtual robotic systems in the Industry 4.0 era. In: *52th International Symposium on Robotics*. ; 2020:1-8.

13. Ajwad SA, Asim N, Islam RU, Iqbal J. Role and review of educational robotic platforms in preparing engineers for industry. *Maejo International Journal of Science and Technology*. 2017;11(1):17-34.

14. Zhan Z, Zhong B, Shi X, Si Q, Zheng J. The Design and Application of IRobotQ3D for Simulating Robotics Experiments in K-12 Education. *Computer applications in engineering education*. 2021;30(2):532-549.

15. Zhong B, Zheng J, Zhan Z. An Exploration of Combining Virtual and Physical Robots in Robotics Education. *Interactive learning environments*. Published online 2020:1-13. doi:10.1080/10494820.2020.1786409

16. Tijani F, Callaghan R, de Villers R. An Investigation into Pre-Service Teachers' Experiences While Transitioning from Scratch Programming to Procedural Programming. *African journal of research in mathematics, science and technology education*. 2020;24(2):266-278.

17. Cheluszka P. The Use of Low-Budget Self-Assembly Sets for Research and Robotics Education. *Management systems in production engineering*. 2019;27(1):55-62. doi:10.1515/mspe-2019-0010

18. Nutakki C, Vijayan A, Sasidharakurup H, Nair B, Achuthan K, Diwakar S. Low-Cost Robotic Articulator as an Online Education Tool: Design, Deployment and Usage. In: *2016 International Conference on Robotics and Automation for Humanitarian Applications (RAHA)*. IEEE; 2016:1-5. doi:10.1109/RAHA.2016.7931888

19. Robinette MF, Manseur R. ROBOT-DRAW, an Internet-Based Visualization Tool for Robotics Education. *IEEE Trans Educ*. 2001;44:29-34.

20. Corke P. MATLAB toolboxes: robotics and vision for students and teachers. *IEEE Robotic and Automation Magazine*. 2007;14. doi:10.1109/M-RA.2007.912004

21. Indri M, Lazzero I, Bona B. Robotics education: Proposals for laboratory practices about manipulators. In: *IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*. ; 2013:1-8. doi:10.1109/ETFA.2013.6648018

22. Manzoor S, Ul Islam R, Khalid A, Samad A, Iqbal J. An open-source multi-DOF articulated robotic educational platform for autonomous object manipulation. *Robotics and Computer-Integrated Manufacturing*. 2014;30(3):351-362. doi:10.1016/j.rcim.2013.11.003

23. Haddadin S et al. The Franka Emika Robot: A Reference Platform for Robotics Research and Education. *IEEE robotics & automation magazine*. 2022;29(2):2-20. doi:10.1109/MRA.2021.3138382

24. DOBOT Magician. Accessed July 12, 2023. https://www.DOBOT.us/

25. Pitsco Education. Accessed July 12, 2023. https://www.pitsco.com/

26. Tetrix Prime. Accessed July 12, 2023. https://www.pitsco.com/Shop/Robotics