11-2022

Gesture Controlled Collaborative Robot Arm and Lab Kit

Abel A. Reyes
Michigan Technological University, areyesan@mtu.edu

Skylar Reinhardt
Michigan Technological University, srreinhar@mtu.edu

Tony Wise
Michigan Technological University, ajwise@mtu.edu

Nathir Rawashdeh
Michigan Technological University, narawash@mtu.edu

Sidike Paheding
Michigan Technological University, spahedin@mtu.edu

Recommended Citation
A. Reyes, S. Reinhardt, T. Wise, N. Rawashdeh, S. Paheding, "Gesture Controlled Collaborative Robot Arm and Lab Kit", 2022 American Society for Engineering Education (ASEE) Conference for Industry and Education Collaboration (CIEC), Tempe, AZ Feb. 4-11 2022, ASEE. Retrieved from: https://digitalcommons.mtu.edu/michigantech-p/16520

Follow this and additional works at: https://digitalcommons.mtu.edu/michigantech-p
Part of the Computer Engineering Commons, Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Gesture Controlled Collaborative Robot Arm and Lab Kit

Abel Reyes, Skyler Reinhardt, Tony Wise, Nathir Rawashdeh, and Sidike Paheding

Michigan Technological University

Abstract

In this paper, a mechatronics system was designed and implemented to include the subjects of artificial intelligence, control algorithms, robot servo motor control, and human-machine interface (HMI). The goal was to create an inexpensive, multi-functional robotics lab kit to promote students' interest in STEM fields including computing and mechtronics. Industrial robotic systems have become vastly popular in manufacturing and other industries, and the demand for individuals with related skills is rapidly increasing. Robots can complete jobs that are dangerous, dull, or dirty for humans to perform. Recently, more and more collaborative robotic systems have been developed and implemented in the industry. Collaborative robots utilize artificial intelligence to become aware of and capable of interacting with a human operator in progressively natural ways. The work created a computer vision-based collaborative robotic system that can be controlled via several different methods including a touch screen HMI, hand gestures, and hard coding via the microcontroller integrated development environment (IDE). The flexibility provided in the framework resulted in an educational lab kit with varying levels of difficulty across several topics such as C and Python programming, machine learning, HMI design, and robotics. The hardware being used in this project includes a Raspberry Pi 4, an Arduino Due, a Braccio Robotics Kit, a Raspberry Pi 4 compatible vision module, and a 5-inch touchscreen display. We anticipate this education lab kit will improve the effectiveness of student learning in the field of mechatronics.

Introduction

Within the context of the fourth industrial revolution, i.e. Industry 4.0, the concept of smart factories with connected intelligent systems has become the standard in recent years, with an increasing trend of adoption by leading companies in different regions around the world. For instance, the adoption rate of Industry 4.0 was reported at about 36% in North America, 27% in Europe, and 20% in Asia in 2020 [1]. The advance in technology has aided in automatization of processes, increase productivity and quality of final products within different industries. The use of robots within the manufacturing process has set a precedent in the industry in terms of efficient usage of resources, connectivity and mass production. Since the implementation of the first industrial robot in the 70s, robotics has witnessed a rapid increase in use. This is also seen in manufacturing and in replacing jobs that are currently human-held to improve safety or workers or to reduce cost. As such, more demand is made for people who can design, create, build, and maintain robots. Industry 4.0 is benefiting from advancements in robotics because it could match the goal to have zero downtime and maximize efficiency. Artificial Intelligence (AI) helps

achieve those goals providing the knowledge to automate processes with the use of machines [2].

According to Grischke et al. [3], autonomous robots depend on a knowledge base to reliably perform the assigned tasks, and AI comprises methods that are used to autonomously plan a sequence of actions to achieve the desired task. Pan et al. [4] mentioned that AI applications in robotics have experienced exponential growth in recent years, since it plays a leading role in modeling and intelligent control of robots, in addition to the fact of the industrial changing environment, and automated warehouses, where robots are expected to collaborate side by side with humans. Within human and robot interactions, AI plays a crucial role, and the safe achievement of this is known as human-robot-interaction (HRI)[5]. Du et al. [6] demonstrated the capabilities of AI in the context of achieving active collision avoidance. In addition, other developments in AI are related to collaborative robots to assist humans, Ewerton et al. [7] presented a reinforcement learning approach to achieve an optimal solution in co-manipulation and teleoperation tasks for collaborative robots.

On the other hand, a similar increase in job market is seen for AI and robotics related skills. This spans a wide area of fields, especially in the digital era. More products are gradually requiring computer programming skills, especially for smart devices. With all of these potential job opportunities, there is a demand for workers who can perform all these task, thus students' interest in these skills is vital. By equipping students with those skills in school, it will be beneficial for them to be better prepared for the future work environments and challenges; however, not all schools have the budget needed to keep such expensive programs as FIRST[8] robotics operational year-round. This is where our proposed learning kit in this paper can be deployed.

The goal of this work is to design a robotic training kit that would help teach students about certain topics relating to the field of robotics. The topics covered include computer programming, serial communication, HMI development, and AI technology. To complete this task, we developed the lab kit, develop programs to test the functionality of the components, and created a handbook to guide users through developing their own experiments.

The remainder of this paper is structured as follows. First, related works corresponding to collaborative robotics implementation and AI in robotics are briefly described. Then, a general overview of the project proposed is introduced, including the hardware and software portions. Later, the hardware architecture is presented and described. Then, all software-related topics in the project are described. Finally, the lab manual, implemented to be used along with the Gesture Controlled Collaborative Robot kit, is described. Projet outcomes and comclusions and recommendations are also reported.

Related Work

Unlike most robots that are isolated from human workers for stafety reasons, collaborative robots are designed to be able to operate alongside human operators by using cutting-edge technologies. For instance, Kragic et al. [9] highlights the challenges related to interactive and collaborative robots as well as suggesting a framework for which utilizes the aforementioned ideas of using explicit communication, exploration learning, and reactive control-based approaches to motion generation. The reactive motion generation architecture relies on three parts: perception, skills,

and behaviors. Also, the authors are aware that collaborative robotics has shown great promise to bring potentially complex tasks infrequently changing settings closer to automation, which in some ways is aligned with the premise of this project proposal. Veloso et al. [10] illustrated the progress and contributions made with mobile collaborative robots, summarizing the capabilities of the task as follows: a single destination task, an item transport task, a task to escort a person, and a semi-autonomous telepresence task. All of these tasks are equivalent from a navigational point of view; however, they also conducted other research, including speech interaction, interaction among multiple robots, learning from human demonstration, human observation, and human correction. Maurice et al. [11] presented a generic method for performing detailed ergonomic assessments of co-manipulation activities and its application to the optimal design of collaborative robots. Multiple ergonomic indicators were defined to estimate the different biomechanical demands which occur during manual activities. The proposed method consists of four components: the list of ergonomic indicators, a dynamic simulation framework in which a digital human model can interact with a controlled collaborative robot, a sensitive analysis framework, and a framework for optimizing design parameters of a collaborative robot with respect to relevant ergonomic indicators. Zahavi et al. [12] stated the importance of the collaborative robots industry and its continuous improvement to enhance the safety of human operators to work side-by-side carrying out complex tasks which could not be done before. In their work, the improvement of a collaborative robot was made with an external machine vision system to sort objects, this module was implemented using LabView on an ABB YuMi Robot, with the success rate of 95% in object detection. In addition, Salameen et al. [13] presented topics on digital automation and robot simulations. In this paper, a communication interface was developed with the use of open-source libraries, with the goal to provide an education tool by using an industrial robot.

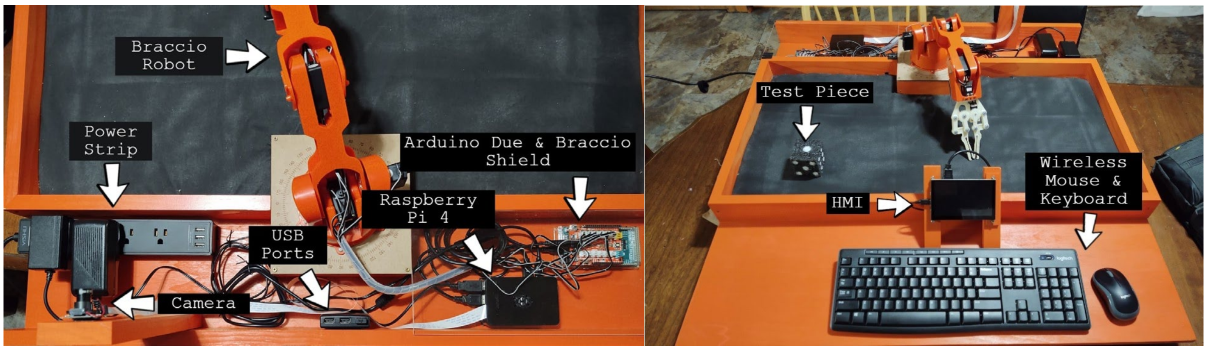
AI techniques have been frequently applied to the domain of robotics. Hadidi et al. [14] proposed to utilize the aggregated computational power of robots in a distributed robot system to perform a deep neural network based object recognition in real-time, such collaboration enables robots to take advantage of the collective computing power of the group in an environment to understand the collected raw data, while none of the robots would experience energy shortage. On the other hand, Galin et al. [15] showed the high demand of the introduction of robots in intelligent manufacturing, and state that collaborative robots should be smart enough to conduct social behavior during safe interaction with humans. Kim et al. [16] mentioned that the success of collaborative robots also relies on the use of soft sensors and actuators; however, there are limitations related to their usage. Machine learning algorithms attempt to overcome the related issues. Nontheless, machine learning also has limitations, for instance, the state-of-the-art machine learning methods are data-driven; therefore, requires a large amount of data for training, in which, data collection is time-consuming and it adds a considerable amount of computational load. Our proposed study in this work also describes possible approaches to alleviate those limitations.

System Design & Development

The proposed system is designed for educational purposes, and to stimulate and encourage the interest of students in the STEM field by providing the tool to get involved in subjects such as programming, AI, HMI design, and robotics. In our proposed framework, all of those topics are

placed single educational lab kit with the corresponding lab assignments. In addition, our proposed educational lab kit consists of the implementation of an inexpensive educational lab kit for schools using pre-existing hardware to teach more complex topics. Within the scope of this work, it is considered a robotic system with several functions, included but not limited to:

- Direct control of the robot using Arduino IDE
- Simple interactions using HMI that allows basic control of the robot
- Facial recognition for login authentication serves as an intro to computer vision and artificial intelligence
- Gesture-based collaborative interactions allow the use of gestures to relay instructions to the robot



(a) Top view of the educational robot kit (b) Frontal view of the educational robot kit

Figure 1. Gesture Controlled Collaborative Robot and the list of hardware components.

In terms of system configuration, several devices are involved and interconnected to guarantee the functionality of several use cases. Figure 1 shows the interconnection of all of the hardware components used for the implementation of the system, including the inputs and outputs devices and the controller boards. In addition, the Arduino IDE is used to program the Arduino Due along with the PyQt5 designer to elaborate the GUI of the HMI touchscreen.

Hardware Architecture

The implementation of the proposed system involves the use of several hardware components interconnected in such a way to accomplish the objectives described at the beginning of this paper. Figure 2 shows the architecture of hardware components involved in the Gesture Controlled Collaborative Robot and their corresponding connections.

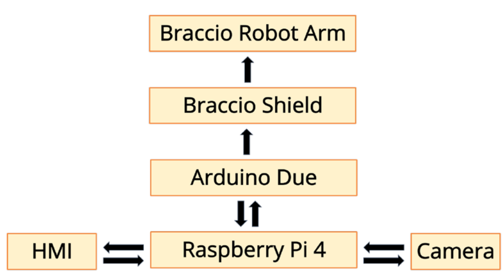


Figure 2. Hardware architecture diagram of the collaborative robot

The followings key worlds explain different hardware components:

- Braccio Robot Arm: a fully operational robot arm which is intended to be controlled via Arduino. It can perform several tasks such as moving objects.
- Braccio Shield: allows the user to hook up the servos directly to the Arduino board, working as an onboard voltage regulator.
- Arduino Due: a microcontroller board based on a 32-bit ARM core controller and is used to receive the serialized data from the Raspberry Pi and forward it to the Braccio Robot Arm through the Braccio Shield.
- HMI Screen: HMI stands for Human Machine Interface. This touch screen interface allows users to interact with premade or user-made screens that send information to the Raspberry Pi 4 and receive information from the Raspberry Pi 4 and the camera. Inputs via the HMI include features such as the buttons for jogging in the jog tab, or the buttons that run pre-installed programs.
- Raspberry Pi 4: acts as the brains of this board, giving commands to the Arduino Due and running the OS displayed on the HMI.

```python
import sys
from PyQt5.QtCore import pyqtSlot
from PyQt5.QtWidgets import QApplication,QDialog,QMainWindow
from PyQt5.uic import loadUi
import serial
import time

ser = serial.Serial('/dev/ttyACM0',9600,timeout=0.1)
ser.flush()




class HMI(QMainWindow):
    def __init__(self):
        super(HMI, self).__init__()
        loadUi('SD_HMI.ui',self)

        self.setWindowTitle('HMI System')
        self.compb.clicked.connect(self.initComs)
        self.pb1.clicked.connect(self.rb1)
        self.pb2.clicked.connect(self.rb2)


    def initComs(self):
        y = '<3>'
        ser.write(y.encode('utf-8'))
        time.sleep(0.1)
        waituntil = 1
        while waituntil :
            initstatus = ser.readline().decode('utf-8').rstrip()
            if initstatus != '':
                self.label.setText(initstatus)
                waituntil = 0
            else :
                initstatus = ser.readline().decode('utf-8').rstrip()
```

(a) Sample PyQ5t's

```c
void setup() {
    Serial.begin(9600);
    Robotsetup();
}

void loop() {
    recvWithStartEndMarkers();
    selectRoutine();
}

void recvWithStartEndMarkers() {
    static boolean recvInProgress = false;
    static byte ndx = 0;
    char startMarker = '<';
    char endMarker = '>';
    char rc;

    while (Serial.available() > 0 && newData == false) {
        rc = Serial.read();

        if (recvInProgress == true) {
            if (rc != endMarker) {
                receivedChars[ndx] = rc;
                ndx++;
                if (ndx >= numChars) {
                    ndx = numChars - 1;
                }
            }
            else {
                receivedChars[ndx] = '\0'; // terminate the string
                recvInProgress = false;
                ndx = 0;
                newData = true;
            }
        }

        else if (rc == startMarker) {
            recvInProgress = true;
        }
    }
}
```

(b) Sample of Arduino's code

Figure 3. Code samples for serial comunication

Software Development

For the software side in the system, three main components were implemented and are described as follows.

Human Machine Interface

The HMI is the graphical user interface that allows the user to interact with the machine. It was programmed using PyQt5. Qt is a set of cross-platform C++ libraries that implement high-level application programming interfaces (API) for accessing many aspects of modern desktop and mobile systems, including traditional user interfaces (UI) development services [17]. PyQt5 is a comprehensive set of Python bindings for QT v5 [18].

PyQt5 allows users to use the designer environment for “drag and drop” HMI design. This is useful for the implementations of buttons and their placement. This tool is easier for users not used to traditional object-oriented programming (OOP), allowing them to get familiar with this programming paradigm. By using PyQt5 it is just needed to connect objects to related functions, connections that could be done with a single line of code, and functions that may be defined later within a declared class.

```
void Robotsetup() {

  //Update these lines with the calibration code outputted by the calibration program.
  arm.setJointCenter(WRIST_ROT, 90);
  arm.setJointCenter(WRIST, 97);
  arm.setJointCenter(ELBOW, 85);
  arm.setJointCenter(SHOULDER, 90);
  arm.setJointCenter(BASE_ROT, 90);
  arm.setJointCenter(GRIPPER, 40);//Rough center of gripper, default opening position

  //Set max/min values for joints as needed. Default is min: 0, max: 180
  //The only two joints that should need this set are gripper and shoulder.
  arm.setJointMax(GRIPPER, 75);//Gripper closed, can go further, but risks damage to servos
  arm.setJointMin(GRIPPER, 20);//Gripper open, can't open further

  //2. Start to custom position.
  arm.begin(false);
  //Set each joint immediately to a set position
  arm.setAllNow(90, 97, 45, 45, 90, 40); //BASE_ROT, SHOULDER, ELBOW, WRIST, WRIST_ROT, GRIPPER
  //This method allows a custom start position to be set, but the setAllNow method MUST be run
  //immediately after the begin method and before any other movement commands are issued.


  //NOTE: The begin method takes approximately 8 seconds to start, due to the time required
  //to initialize the power circuitry.
  Serial.println("Robot Initialization Complete");

  //Set the base rotation joint to move at 3x speed. This may cause the arm to jerk.
  arm.setDelta(BASE_ROT, 3);
}
```

Figure 4. Initialize function of the robor arm in Arduino.

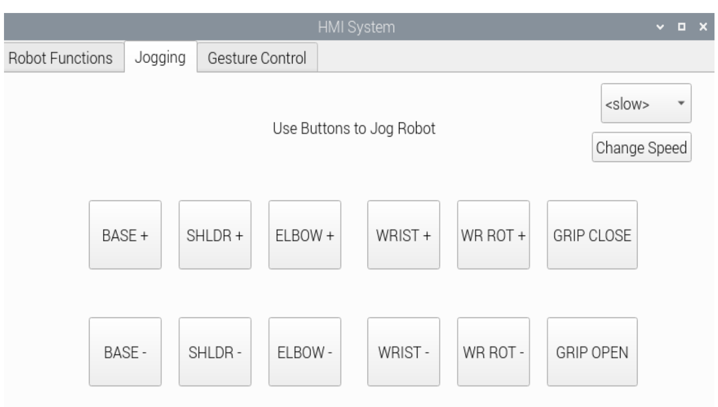


Figure 5. Snapshot of the HMI of the Gesture Controlled Collaborative Robot.

Figure 3 (a) showns a sample portion of the code used for the implementation of the HMI with PyQt5, in which the HMI class is defined along with the functions to initialize of the componets of the User Interface(UI), and the serial communication with the Arduino board. Figue 3 (b) shows the code for the Arduino board, in which the robot is initializes throug a setup function call and the serial communication is handled through the reception of serial data. Figure 4 shows the implementation of the setup function of the arm robot through arduino. The code shows the default values of the servo motors located on each of the joints of the arm robot.

Once the program is finalized, the following commenad can be launched through the terminal as a python program: "python3 NameOfProgram.py". Figure 5 shows an example of one of the HMIs implemented in the Gesture Controlled Collaborative Robot by using PyQt5 designer environment. Here the use can use touch to actuate all robot joints, and change the servo speed.

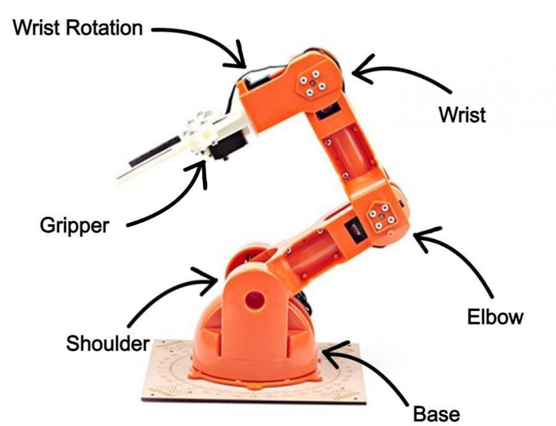


Figure 6. Graphic of the TinkerKit Braccio Robot arm servo locations and names.

In the UI presented in Figure 5, each of the UI buttons has a specific movement/rotation or action. Since the robot arm has 5 degrees of freedom, the 5 different axes where the robot can perform a movement are controlled on the screen. For instance, the + sign means a clockwise or a forward rotation, while the - means a counterclockwise or backward rotation. Figure 6 shows the robot arm servo locations, i.e. degrees of freedom, and their names. The UI buttons labeled as BASE ± control the servos in the Base of the robot arm (clockwise or counterclockwise rotation). The UI buttons labeled as SHLDR ± control the servos located in the shoulder (forward or backward rotation) of the robot arm. The buttons labeled as ELBOW ± control the servos located in the elbow of the robot arm (forward or backward rotation). Similarly for WRIST ± and WR ROT±. Finaaly, the UI buttons labeled as GRIP CLOSE/OPEN controls the servos located in the gripper of the robot arm, making them open or close the grip.

Facial Detection Login

The facial detection HMI module is intended to be used as a biometric login function to enable robot control. Also, this acts as an introduction to AI programming for users with the purpose to learn how to use OpenCV. Within this module, the user must log in with the face detection, and

once it detects and recognizes a face it unlocks the robot controls. Figure 7 shows the UI button (lower-left corner) for face detection based login.

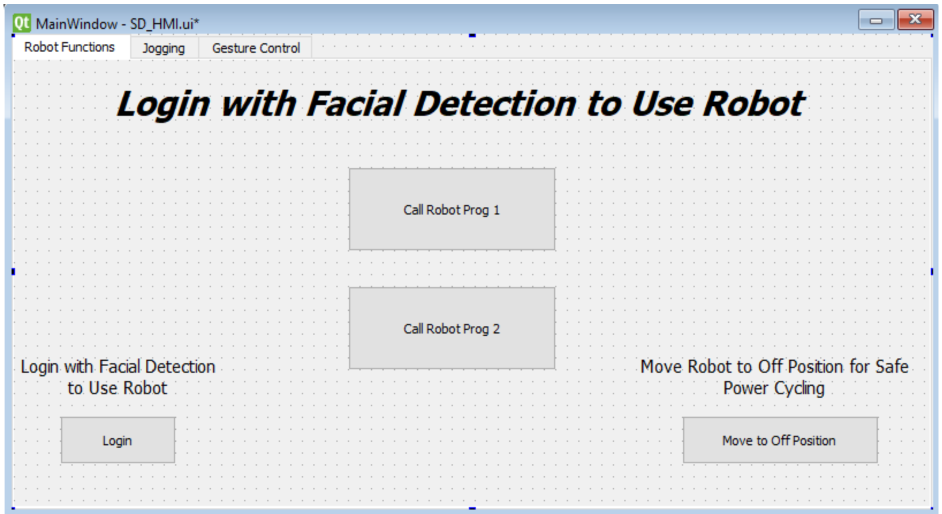


Figure 7. Snapshot of the login with facial detection module interface.

This module uses a predefined Haar cascade, an efficient object detection algorithm that was proposed by Viola and Jone [19], for face detection. This method consists of extracting features (edges, lines, etc.) and a machine learning-based approach where a cascade of weak classifiers are trained on many positive and negative examples. Figure 8 shows a schematic description of the cascade detection, in which a positive result from each weak classifier triggers the evaluation of the following one, which also has been adjusted to achieve relatively high detection rates. A negative result from any classifier leads to rejection of the sub-window immediately. The final result will be drawing a bounding box around the detected face by using coordinates from the prediction.

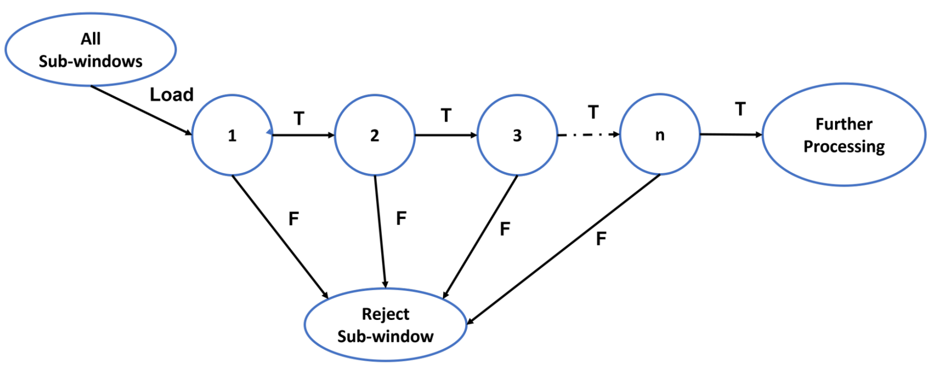


Figure 8. Schematic depiction of the detection cascade [20]. T: True; F: False.

Gesture Recognition Control (GRC)

The implementation of this module was intended to allow the user to use hand signs to step through the pick/place process. This module can be edited to move to another position by changing only a few lines of code.

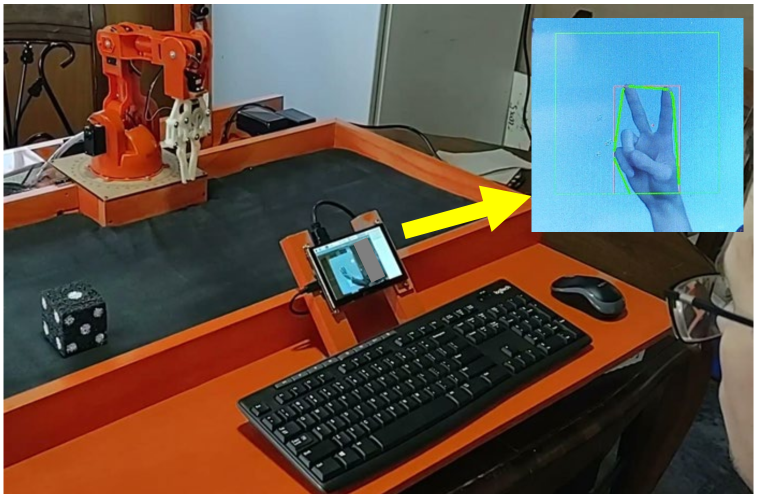


Figure 9. Demonstration of the gesture recognition control module.

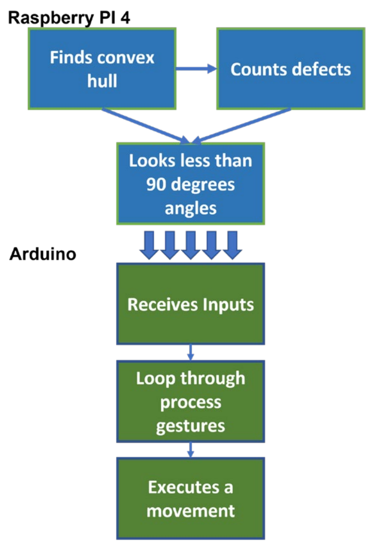


Figure 10. Flow diagram of the gesture-controlled process in the system.

The gesture control functionality was implemented using a convex hull defect counting mechanism that uses the cosine rule to count the number of fingers the user is holding up. The program developed to show this runs the program through a predetermined set of movements based on the number of fingers it detects if they are presented in the correct order. This was designed to be used as a base for the users to build off in the future to implement different functions using the same gestures.

Figure 9 shows a demonstration of this module where an user is interacting with the robot and HMI using hand gesture control. In this example, the two-finger gesture is detected. The use of hand signs allows the user to control the robot to step through the pick/place process. For instance, one finger sign will position the gripper of the arm to the predefined pick position and two fingers sign close the gripper.

Figure 10 shows a flow diagram of the data in this module, the top portion involves the process computed through the Raspberry Pi 4, in which the system first finds the convex hull of the contoured area within regions of interest, then it counts the defects within the convex hull. The following step consists of looking for angles within convex hull defects less than 90 degrees. Having this serial information is sent to Arduino based on the number of defects found. The bottom part is related to the serial information handled by Arduino, in which it receives input to enter an associated case statement, then starts a loop to count through process gestures and once it receives the correct gesture at the correct time it executes a movement. A sample of the code for this implementation is shown in Fig. 11.

```
 case 9:
   newData = false;
   gestureControl();
   break;

void gestureControl(){
  gesture_tot = 0;
  while (gesture_tot < 4){
  while(newData == false){
    recvWithStartEndMarkers();
    }

  String gesture = receivedChars;
  Serial.println(gesture);
  if(gesture_tot == 0){
  if (gesture == "pickmov"){
  //Set all the arm joints to relative positions
        arm.setAllAbsolute(90, 45, 45, 45, 90, 40);
  //Delay for 3000ms while still updating movement
        arm.safeDelay(1000);
        gesture_tot = gesture_tot + 1;
   }
```

Figure 11. Arduino code for the GRC implementation.

Learning Kit

The implementation of the gesture controller collaborative robotic system is complemented with the design of programming laboratory procedures to use the robotic system as an educational robot learning kit. The set of labs were designed to familiarize students with the available programs on the Raspberry Pi and Arduino Due, in addition to creating a sense of the importance and potential that different topics could have out there in the industry. Each of the labs has a set of well-marked objectives and a set of key terms related to the topics that students are expected to learn. The set of lab assignments were listed as follow:

- Lab 0: Familiarizing Yourself With the Lab Kit
- Lab 1: Basic of Coding the TinkerKit Braccio Robot
- Lab 2: Creating a Basic Pick and Place Program
- Lab 3: Programming the HMI
- Lab 4: Programming Gesture Control

Lab 0 and lab 1 have the goal to explore the hardware components in the teaching kit. In Lab 0, the student is expected to familiarize with the Gesture Controlled Collaborative Robot Lab Kit, understanding its capabilities and more details about usage components. In Lab 1, the student is

expected to learn some basics of coding in the Arduino board and learn to use Tinkerkit Library. In Lab 2, students will conduct experiments on a pick and place program using the TinkerKit Braccio robot. At the end of this lab, students will know about how to command the robot to pick up an object and drop it in set locations. The objective of Lab 3 is to provide knowledge to the students on how to make changes to the HMI design using PyQt5 Designer as well as connecting buttons to their appropriate function in the associated Python program. Finally, in Lab 4, the students will be learning how to use the gesture control functions with the OpenCV [21] documentation. This will be helpful to develop a robot control program using different methods. At the end of this lab series, the students are expected to have a basic understanding of different techniques to manipulate the Collaborative Robot Lab kit and its potential applications in the industry.

Performance

Two undergraduate students were able to complete all lab experiments following the guideline of the Gesture Controlled Collaborative Robot kit, according to the requirement listed in the initial proposal. The Robot kit allows users to log into the system by face recognition, then the user can interact with the GUI interface implemented as part of the HMI module. The user can press buttons on the touchscreen to control the servos of the Robot arm, moving the base, the elbow, or the wrist back and forth, right to left, and even rotate in such a way to cover more space to interact with objects. As a choice to control the Robot, the system was able to allow the user to control the Robot arm by using gestures with the hand.

Conclusion

The project was proposed as an inexpensive educational solution to promote different STEM topics to young scholar students. the gesture controlled collaborative robot kit was set using pre-existing hardware and keeping a relatively low cost of around $500-$600, covering a variety of interesting, complex, and industry highly demanding topics such as facial detection, gesture recognition, human-machine interaction, artificial intelligence, and computer vision. a fully functional prototype of the gesture controlled collaborative robot kit was developed and tested by the research team, validating the functionality of the login through facial detection and the operation of the robot through hmi or gesture recognition. in addition, a set of lab assignments and reference handbooks were documented, containing simple labs to familiarize students with the available programs on the different components that are part of the robot kit, such as Raspberry Pi and Arduino. It is anticipated this education lab kit will improve the effectiveness of student learning in the field of mechatronics, programming, artificial intelligence, and robotics.

References

[1] IOT Analytics: Industry 4.0 Adoption 2020 – who is ahead?, https://iot-analytics.com/industry-4-0-adoption-2020-who-is-ahead/

[2] Assembly: Robotics in the Age of Industry 4.0, https://www.assemblymag.com/articles/95694-robotics-in-the-age-of-industry-40

[3] Grischke, J., Johannsmeier, L., Eich, L., Griga, L., & Haddadin, S. (2020). Dentronics: Towards robotics and artificial intelligence in dentistry. *Dental Materials*, *36*(6), 765-778.

[4] Pan, Y., Al-Hadithi, B. M., & Yang, C. (2020). AI for Robot Modeling, Path Planning, and Intelligent Control. *Frontiers in Robotics and AI*, *7*, 19.

[5] Brewka, G. (1996). Artificial intelligence—a modern approach by Stuart Russell and Peter Norvig, Prentice Hall. Series in Artificial Intelligence, Englewood Cliffs, NJ. *The Knowledge Engineering Review, 11*(1), 78-79. doi:10.1017/S0269888900007724

[6] Du, G., Long, S., Li, F., & Huang, X. (2018). Active collision avoidance for human-robot interaction with ukf, expert system, and artificial potential field method. *Frontiers in Robotics and AI*, *5*, 125.

[7] Ewerton, Marco, et al. "Learning trajectory distributions for assisted teleoperation and path planning." Frontiers in Robotics and AI 6 (2019): 89.

[8] FIRST: For Inspiration and Recognition of Science and Technology, https://www.firstinspires.org/

[9] Kragic, D., Gustafson, J., Karaoguz, H., Jensfelt, P., & Krug, R. (2018, July). Interactive, Collaborative Robots: Challenges and Opportunities. In *IJCAI* (pp. 18-25).

[10] Veloso, M., Biswas, J., Coltin, B., Rosenthal, S., Kollar, T., Mericli, C., ... & Ventura, R. (2012, October). Cobots: Collaborative robots servicing multi-floor buildings. In *2012 IEEE/RSJ international conference on intelligent robots and systems* (pp. 5446-5447). IEEE.

[11] Maurice, P., Padois, V., Measson, Y., & Bidaud, P. (2017). Human-oriented design of collaborative robots. *International Journal of Industrial Ergonomics*, *57*, 88-102.

[12] Zahavi, A., Haeri, S. N., Liyanage, D. C., & Tamre, M. (2020, October). A Dual-Arm Robot for Collaborative Vision-Based Object Classification. In *2020 17th Biennial Baltic Electronics Conference (BEC)* (pp. 1-5). IEEE.

[13] Salameen, L., Estatieh, A., Darbisi, S., Tutunji, T. A., & Rawashdeh, N. A. (2020, December). Interfacing Computing Platforms for Dynamic Control and Identification of an Industrial KUKA Robot Arm. In *2020 21st International Conference on Research and Education in Mechatronics (REM)* (pp. 1-5). IEEE.

[14] Hadidi, R., Cao, J., Woodward, M., Ryoo, M. S., & Kim, H. (2018). Distributed perception by collaborative robots. *IEEE Robotics and Automation Letters*, *3*(4), 3709-3716.

[15] Galin, R., Meshcheryakov, R., Kamesheva, S., & Samoshina, A. (2020, May). Cobots and the benefits of their implementation in intelligent manufacturing. In *IOP Conference Series: Materials Science and Engineering* (Vol. 862, No. 3, p. 032075). IOP Publishing.

[16] Kim, D., Kim, S. H., Kim, T., Kang, B. B., Lee, M., Park, W., ... & Jo, S. (2021). Review of machine learning methods in soft robotics. *Plos one*, *16*(2), e0246102.

[17] Python Software foundation (2021, March), Python bindings for the Qt cross platform application toolkit, https://pypi.org/project/PyQt5/.

[18] Meier, B. (2019). *Python GUI Programming Cookbook: Develop functional and responsive user interfaces with tkinter and PyQt5*. Packt Publishing Ltd.

[19] Viola, P., & Jones, M. (2001, December). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (Vol. 1, pp. I-I). Ieee

[20] Kim, M., Lee, D., & Kim, K. Y. (2015). System architecture for real-time face detection on analog video camera. *International Journal of Distributed Sensor Networks*, *11*(5), 251386.

[21] OpenCV. (2015). *Open Source Computer Vision Library: http://opencv.org*.

Biographies

ABEL A. REYES is currently a Ph.D.student in Computer Sciences and Engineering at Michigan Technological University. He received a M.S. in electrical and computer engineering from Purdue University Northwest in 2021. His research interests are included, but not limited to machine learning, deep learning, artificial intelligence, computer vision, computer graphics, big data and virtual reality.

SKYLER REINHARDT received the bachelor’s degree in electrical engineering technology from Michigan Technological University in May 2021. After which, he joined Nemak as a control engineer where he is helping troubleshoot and design ladder logic PLC programs using Rockwell Automation's Studio 5000, and updating/developing HMI systems with WonderWare's InTouch HMI software for various machines throughout the Nemak - WIS Gateway plant.

TONY WISE received the bachelor’s degree in electrical engineering technology from Michigan Technological University in May 2021. He is currently enrolled in the Mechatronics master’s program, which focuses on computing, industrial automation, and robotics.

NATHIR RAWASHDEH is an assistant professor at the department of applied computing at Michigan Tech. since August 2019. Prior to this appointment, he was an associate professor in the Mechatronics Engineering Department at the German Jordanian University, where he spent 10 years. His industrial experience includes 5 years with Lexmark International, Inc. Lexington-Kentucky and MathWorks, Inc. in Natick-Massachusetts. Dr. Rawashdeh is a Senior Member of the IEEE and has experience with European-funded research capacity building projects. His research interests include mobile robots, autonomous driving, image processing and sensor fusion.

SIDIKE PAHEDING received a Ph.D. in Electrical Engineering from the University of Dayton, and currently, he is an assistant professor in the Department of Applied Computing at Michigan Technological University. Prior to joining Michigan Tech in 2020, Dr. Paheding was Visiting Assistant Professor at Purdue University Northwest. His research interests cover a variety of topics in image/video processing, machine learning, deep learning, computer vision, and remote sensing. Dr. Paheding is an associate editor of the Springer journal Signal, Image, and Video Processing, and ASPRS Journal Photogrammetric Engineering & Remote Sensing, and serves as a guest editor/reviewer for several reputed journals.