Proceedings of the 9th RSI International Conference on Robotics and Mechatronics (ICRoM 2021), Nov. 17-19, 2021, Tehran, Iran

Design of an Intelligent Control System for Safe Collaboration between Human and a Robotic Manipulator

Arsalan Aeini[a], Erfan Droudian[a], Armin Ghanbarzadeh[a], Esmaeil Najafi[a,b]

[a]Faculty of Mechanical Engineering, K.N. Toosi University of Technology, Tehran, Iran
[b]Mechatronics Department, Fontys University of Applied Sciences, Eidhoven, The Netherlands
Emails: arsalan.aeini@gmail.com, erfan.dr77@gmail.com, ghanbarzadeh.armin@gmail.com, najafi.e@kntu.ac.ir

***Abstract*—Human Robot Collaboration (HRC) is a concept in which a human and a robot such as manipulators divide work tasks in a shared work area. In the environments where robots are employed, various safety systems such as fences and barriers are used to ensure the safety of common workspace. This paper presents a new intelligent control system for safe collaboration between human and a robotic manipulator. The designed control system has two main parts. The first part is object detection, where deep learning algorithm YOLOv3 has been implemented to detect various objects with different shapes. The images are partly captured by Intel® RealSense™ Depth Camera D435i and partly are taken from the Microsoft Common objects in context (COCO). The second part is a controller that enables the manipulator to move safely. To investigate the performance of the designed control system, a number of sample collaborative tasks have been simulated in the software Robot Operating System (ROS). The provided simulation results validate the proper performance of the object detection algorithm and the robot controller.**

***Index Terms*—Safe Human Robot Collaboration (HRC), Object Detection, Deep Learning, Robot Operating system (ROS)**

I. Introduction

Human-robot collaboration (HRC) has gotten a lot of attention as the industrial industry moves away from rigid routine methods of production and toward a much more adaptive and intelligent form of automation within the Industry 4.0 paradigm. Human-robot collaboration is when a human operator interacts with robots. By combining the execution of mechanical frameworks with the versatility and capacity of human laborers, human-robot interaction permits closer collaboration in gathering forms that request expanded effectiveness. In any scenario, ensuring human safety while working alongside robots is in top priority. As a result, the initial step is to acquire appropriate knowledge of various industrial robot safety systems.

A. Workspace Hazards

The possibility of contact between humans and robots is inevitable when they work in close proximity. In many circumstances, this interaction is both required and desirable. However, this raises the chance of accidents that must be addressed [1].

Engineering failures, human errors, and unfavorable environmental conditions are the three most common causes of robot-related incidents [2]. Faults in the robot's mechanics (weak connections between parts, faulty electronics) or controller errors are examples of engineering failures (programming bugs, faulty algorithms). Human errors, on the other hand, are more controllable and can be caused by a variety of circumstances such as inattention, exhaustion, failure to follow guarding rules, insufficient training programs, or erroneous initial robot activation procedures. Adverse environmental conditions including, severe temperatures, poor sensing in inclement weather, and bad illumination, all of which can cause the robot to respond incorrectly (see Fig. 1).

The purpose of this paper is to design a safe system for controlling a robotic manipulator that assists humans in sorting fruits. This manipulator shares its workspace with a human operator. There are two main challenges in the selected. First, the manipulator must correctly detect the fruits since it is essential for sorting them. Second, the safety of the human operator must be guaranteed in order to work in the proximity of the manipulator. This requires a suitable safety system.

The following is a description of the paper's structure. A brief overview of several safety systems for HRC is given in Section II. Section III investigates the object detection model and dataset. Section IV displays the proposed safety system. Section V demonstrates the results of the system and Section VI concludes the paper.

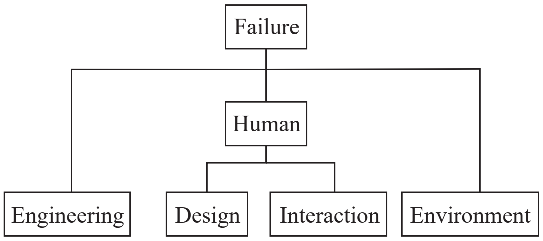


Fig. 1: Classification of failures. Failures can be caused by engineering or human worker errors, as well as environmental conditions.

978-1-6654-2094-5/21/$31.00 ©2021 IEEE

II. Related Works

There are two types of safety measures used in industrial settings: passive safety measures and active safety measures. Safety devices that improve robot safety without changing the robot's behavior are referred to as passive protection. Passive safety is static and relatively easy to implement. As a result, passive safety mechanisms are extremely reliable, yet they can be readily circumvented. Active safety refers to safety features that alter the behavior or environment of a robot in order to avoid potentially dangerous circumstances. Because they must perceive the environment as well as manage the robot, these devices are significantly more complicated.

In [3] a collision detection model based on the torque and position sensors on a Kuka LWR manipulator is proposed. After recording signals from unintentional accidents between the manipulator and operator, they trained a multi-layer neural network (MLP) on this data to output external torque values corresponding to impact status. In a similar paper [4], only joint position sensors were used for training. They found that competing results can be obtained using only joint position sensors. This is an important aspect since many Co-Bots do not have built-in torque sensors on links. By adding more logic to the output of the first network [5], they were able to detect collision directly with less delay. The methods stated used a model-based system to detect the difference between applied and necessary torque, and by processing this data the detection could be made if an impact occurred.

A collision detection system based on vibration sensors on the robot linkages and end-effector has been proposed [6]. A drawback of this methodology is that it requires the establishment of vibration sensors on the robot. An alternate solution that uses wearable sensors to deliver force feedback in the occurrence of an impact has been proposed in [7]. This method is robust in detecting impact but the device restricts worker movement.

The proposed methods stated before have a shared problem. They all detect impact after it has occurred. Therefore the robots maximum speed and torque must be further limited. To remedy this problem, another type of active protection is used which uses sensors to predict the probability of impact and prohibit injuries from occurring. The first and most widely adopted method in manufacturers today is laser and lidar sensors. These can detect the presence of a human with great accuracy, speed and reliability. After detection, the robots can operate at a slower speed and torque profile, or shut off completely depending on the risk and distance. Lidar and laser light sensors are considerably expensive and only used in situations where HRC is necessary. Due to this economic factor, they are unappealing to many smaller workshops and manufacturers who cannot afford the initial investment.

To remedy this problem, some researchers have adopted ways to use cheaper instruments for instance cameras and RGB-D sensors in junction with artificial intelligence software. Implementing a monitoring system for the shared workspace HRC, which supplements the robot, to find the human operator and guarantee that the robot maintains a minimum safe distance from its human partner at all times has been proposed [8]. The monitoring system is composed of four neural networks: an object detector, two neural networks for evaluating the detection, and a simple, customized speech recognizer for further voice instructions.

Implementation of depth space cameras (RGB-D) to capture workspace information and detect the different elements present such as Human hand, head, object positions, etc. using multiple sensors gives the system a clear 3D representation of the environment thus increasing the accuracy [9] & [10]. For monitoring and collision detection, these systems employ virtual three-dimensional (3D) robot models and real images of human operators captured by depth cameras. The system can alert an operator, halt a robot, relocate the robot away from an approaching operator, or change the robot's trajectory away from an approaching operator as a result of collision detection. These techniques can be engaged dependent on the presence and location of the operator in relation to the robot.

For human-robot collision detection, data-based techniques are also proposed. Data is utilized to train a system, which is then used to estimate and identify collisions. Researchers in [11] improved safety by employing a combination of visual perception to recognize human activities and tactile perception,termed Mixed perception, to analyze actual human-robot contact. Using different volunteers, two datasets containing contact and visual data are gathered.

When entering the shared workspace, the action recognition system classifies human activities using the skeletal model of the latter by a 3D-CNN with 99.7% accuracy, and the contact detection system distinguishes between intentional and un-intentional interactions. Physical contact detection between the operator and cobot occurs by a 1D-CNN network with accuracy of 96%.

III. Object Detection for Robot Operating System

The task at hand can be summarized into the following steps:

- A robotic manipulator must operate near a human in a shared workspace.
- The person is performing a task such as picking out the rotten fruits and etc.
- There are 3 fruit classes that the manipulator must sort into different collection boxes. They are apples, oranges, and bananas.
- The manipulator picks up the fruit and drops it into the respective box.

A. Dataset & transfer learning

In order to train a suitable object detection algorithm, a dataset is required. Transfer learning with weights from the COCO dataset [12] was used to lower training time. This model was already trained on pictures of the 4 classes focused in this paper, however more images were acquired and labeled. This helps to increase accuracy after training and customize the algorithm for this application.

Fig. 2: Intel® RealSense™ Depth Camera D435i.

B. Object detection algorithm

YOLO V3 [13], [14] by Darknet was used for object detection. YOLO has a few advantages over classifier-based systems. It looks at the whole image at test time so its predictions are informed by global context in the image. Moreover, it makes predictions with a single network assessment, in contrast to systems like R-CNN which require thousands for a single prediction. This makes it several orders of magnitude faster than the previous methods.

The YOLO V3 has been implemented and tested within Robot Operating System (ROS) Melodic and Ubuntu 18.04. Input video is obtained using Intel® RealSense™ D435i Depth Camera, as shown in Fig. 2.

IV. Designed Safety System

By using relatively inexpensive and available sensors such as Kinekt or Intel RealSense depth cameras, we can ensure that smart and active safety mechanisms are obtainable by smaller facilities that otherwise could not invest in expensive hardware. The sensors can also be used for other reasons except for safety, such as object detection and classification. This means that by installing one set of equipment the manufacturer can use it for multiple uses which makes this method an optimal investment.

A. Object detection implementation

As illustrated in Fig. 3, the model predicts bounding boxes around the objects of interest. More, precisely, the bounding box completely covers the whole body of the person, especially his hands. This information is sufficient to track the human operator. In the next step, we need to calculate the position of the manipulator end-effector.

B. Position of the manipulator

The robot dynamics can be used to obtain link and end-effector position. To simplify our procedure, we have assumed that our manipulator is stationary.

1) Manipulator kinematics: The robotic manipulator is considered to have three degrees of freedom, as shown in Fig. 4. The forward kinematic of the manipulator is used to determine the position of the end-effector [15]. The forward kinematics of a robot alludes to the calculation of the position and orientation of its end-effector outline from its joint coordinates. Summary of the interface parameters in terms of the link frames:

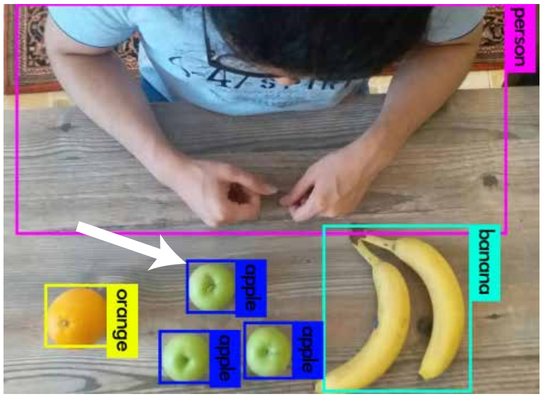


Fig. 3: The designed workspace, where the selected apple will be used for simulation of the manipulator, discussed in Section V.

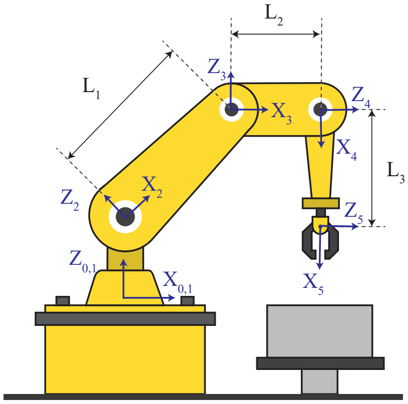


Fig. 4: Schematic of the 3-DOF manipulator.

$$\begin{aligned} a_i &= distance\ from\ \hat{Z}_i\ to\ \hat{Z}_{i+1}\ along\ \hat{X}_i; \\ \alpha_i &= angle\ from\ \hat{Z}_i\ to\ \hat{Z}_{i+1}\ about\ \hat{X}_i; \\ d_i &= distance\ from\ \hat{X}_{i-1}\ to\ \hat{X}_i\ along\ \hat{Z}_i; \\ \theta_i &= angle\ from\ \hat{X}_{i-1}\ to\ \hat{X}_i\ about\ \hat{Z}_i \end{aligned} \tag{1}$$

where the Denavit Hartenberg parameters are given in Table I.

The kinematic equations calculation is possible after the link frames have been identified and the relevant link parameters have been found. Individual link-transformation matrices can be computed using the link parameter values. After that, multiplying the link transformations together to obtain the single transformation that connects frame $\{N\}$ to frame $\{0\}$:

$$^0_5T =^0_1 T \times^1_2 T \times^2_3 T \times^3_4 T \times^4_5 T \tag{2}$$

TABLE I: Parameters of the Denavit Hartenberg method

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	+90	0	0	θ_2
3	0	L_1	0	θ_3
4	0	L_2	0	θ_4
5	+90	0	L_3	θ_5

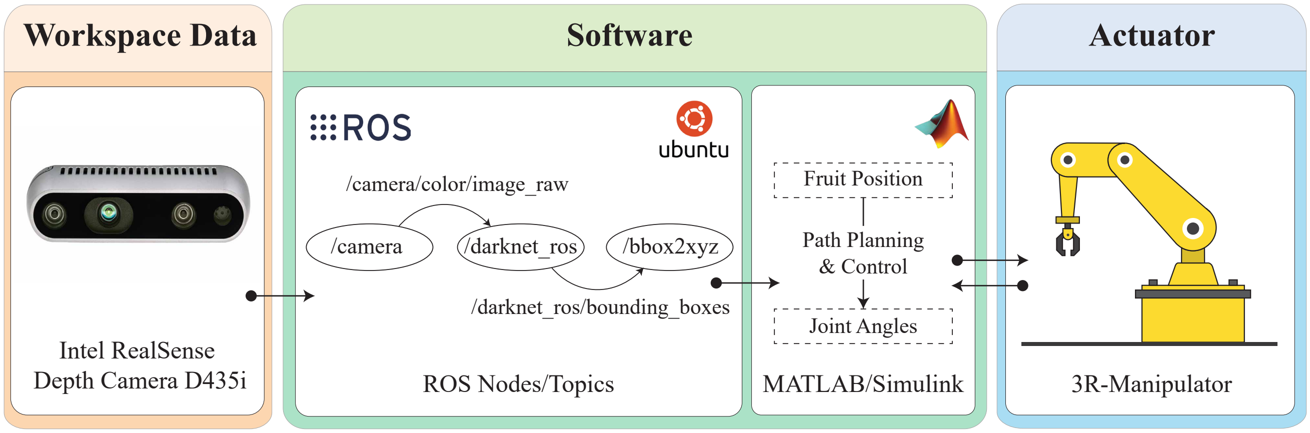


Fig. 5: Interaction between subsystems of the proposed safety method

Moreover, the transportation matrix is calculated as:

$$
{}^{i-1}_{\ \ i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}
$$

The first three rows in the last column of ${}^{0}_{5}T$ Matrix are of interest. They are X_E (the X position of the end-effector), Y_E, and Z_E respectfully.

$$
\begin{aligned}
X_E = {} & L3 \times (c\theta_4 \times (c\theta_1 \times c\theta_2 \times s\theta_3 + c\theta_1 \times c\theta_3 \times \\
& s\theta_2) - s\theta_4 \times (c\theta_1 \times s\theta_2 \times s\theta_3 - c\theta_1 \times c\theta_2 \times \\
& c\theta_3)) - L2 \times (c\theta_1 \times s\theta_2 \times s\theta_3 - c\theta_1 \times c\theta_2 \\
& \times c\theta_3) + L1 \times c\theta_1 \times c\theta_2 \\
Y_E = {} & L3 \times (c\theta_4 \times (c\theta_2 \times s\theta_1 \times s\theta_3 + c\theta_3 \times s\theta_1 \\
& \times s\theta_2) - s\theta_4 \times (s\theta_1 \times s\theta_2 \times s\theta_3 - c\theta_2 \times \\
& c\theta_3 \times s\theta_1)) - L2 \times (s\theta_1 \times s\theta_2 \times s\theta_3 - c\theta_3 \\
& \times c\theta_3 \times s\theta_1) + L1 \times c\theta_2 \times s\theta_1 \\
Z_E = {} & L2 \times (c\theta_2 \times s\theta_3 + c\theta_3 \times s\theta_2) + L1 \times s\theta_2 \\
& - L3 \times (c\theta_4 \times (c\theta_2 \times c\theta_3 - s\theta_2 \times s\theta_3) - \\
& s\theta_4 \times (c\theta_2 \times s\theta_3 + c\theta_3 \times s\theta_2))
\end{aligned} \tag{4}
$$

Thus, we have both the position of the end-effector and the position of the human operator in the workspace. For more accurate examination, we can consider a spherical space with a certain radius around the center of the end-effector. As a result, when this spherical area comes into contact with the bounding box of the human operator, we can define an algorithm for the robot which decides to change the robot's trajectory, alert the human or pause its procedure until the path as depicted in Fig. 5.

V. Experimental Data and Results

A. Object detection algorithm in ROS

The detection software has three main nodes, which are /camera, /darknet_ros and /bbox2xyz. The structure of the nodes and topics are shown in Fig. 5. In the following paragraphs, we will discuss the nodes in detail.

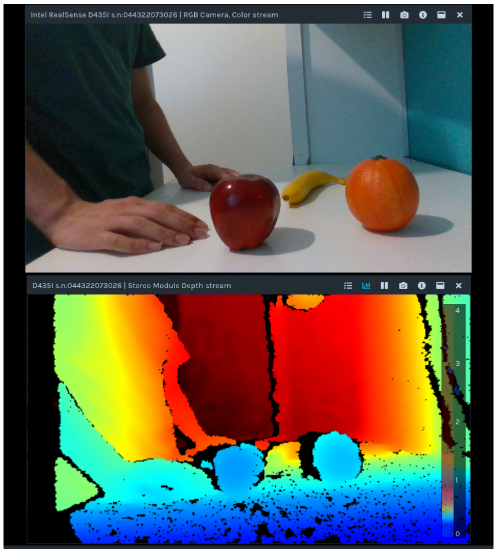

Intel RealSense D435I s.n:044322073026 | RGB Camera, Color stream
D435I s.n:044322073026 | Stereo Module Depth stream
4
3
2
1
0

Fig. 6: RGB and Depth image by Intel RealSense D435i.

1) /camera: We have used a depth camera which is Intel RealSense D435i. This camera, can provide us a series of RGB and also depth images of our workspace. The depth image will be useful in the next steps of the work as illustrated in Fig. 6. There are several topics that the node of the camera is publishing. We use the /camera/color/image_raw as our input for the object detection model. In other words, the darknet_ros node subscribes to /camera/colore/image_raw and performs the detection.

2) /darknet_ros: This node is the main node object detection (YOLO V3). To simplify the running section, we wrote a launch file to run different nodes simultaneously in the /darknet_ros package. This node contains several publishers and subscriber nodes such as bounding boxes of detected objects. The /darknet_ros node subscribes to the image that is published by the /camera node and publishes several topics including bounding boxes, detection images and, etc.

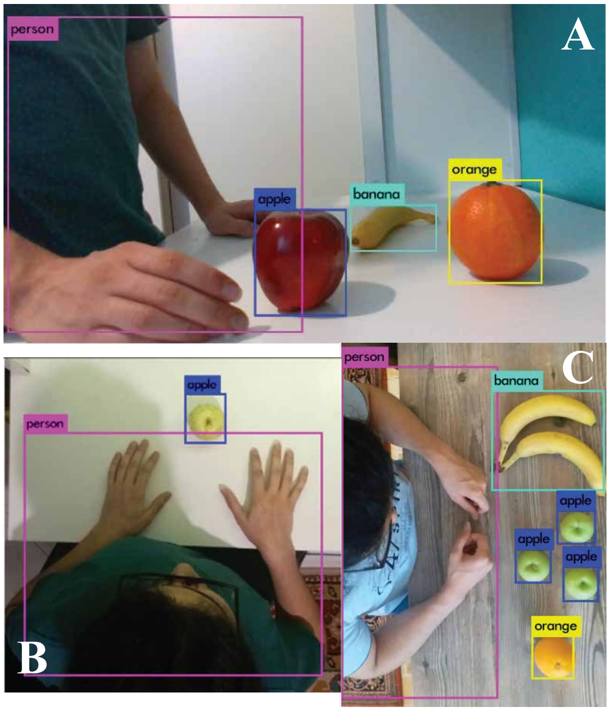


Fig. 7: Detection of objects in the workspace. (Prediction confidence: A. apple:99%, banana:96%, orange:93%, person:98% B. apple:98%, person:98% C. apple:77%;72%;63%, banana:99%, orange:78%, person: 100%)

3) /bbox2xyz: The third node is the /bbox2xyz which subscribes to the bounding box message of the darknet_ros node and publishes the position of every bounding box in the detection image. This node also calculates the X, Y, Z coordinates of objects relative to the workspace corner. The bounding boxes of the workspace are illustrated in Fig. 7.

B. Safety system model for HRC

Finally, we can address the main problem, which is designing a safety system for our manipulator to safely operate near the human operator. As it has been explained in the previous section, we have an object detection model that receives an image feed from the workspace and publishes predictions of objects.

After extracting the bounding box of different fruits, we can use them to control the manipulator. The bounding box coordinates are in pixels, so we must convert these measurements to real word coordinates. The following formulas are used:

$$\begin{aligned} X_{object} &= \frac{X_{max} + X_{min}}{2} \times \frac{workspace_{width}}{image_{width}} \\ Y_{object} &= \frac{Y_{max} + Y_{min}}{2} \times \frac{workspace_{hight}}{image_{hight}} \end{aligned} \tag{5}$$

For example, the dimension of the picture is 1080×788 [pixels], and the center of the bounding box (selected apple in Fig. 3) is 661×568 [pixels]. By scaling the picture, the dimension of the picture is 80×50 [cm]. Thus, by using (5), the obtained coordinate for the center of the apple are 49×36 [cm].

In the next step, we have used a Simulink system to simulate the manipulator for sorting the fruits. The system will get four points which are, the first position of the manipulator (home), position of the fruit to be picked up, position of the collection box where the robot put the fruit, and the final position of the robot (home). This system plans the trajectory according to a second degree parabolic equation, which determines the velocity and joint angles of the robot's links during movement. With a PID controller, the joint angles of the robot's links in a sequence of time will be obtained. The simulation results have been shown in Fig. 8.

Therefore, by having the bounding box of the person and having the position of the end-effector of the manipulator by using the dynamic of the robot, we can maneuver our robot to not enter the bounding box of the person or change its trajectory in order to avoid hitting the human operator. As a result, we can have a safety system that avoids hitting the people working in proximity of the manipulator.

VI. Conclusion

In this paper, different safety systems for human-robot collaboration have been investigated. Moreover, an object detection model has been implemented for the detection of objects in the workspace. The input images were partly captured by the Intel RealSense D435i camera. By using ROS nodes, important data were gained from the object detection and all the devices and nodes were connected in the ROS environment. Different ROS packages were modified to work in our system. Finally, a safety system is proposed, which has been defined based on the output of the object detection model. For future works, we are going to design an algorithm for the robot to avoid collision with the operator. Moreover, ROS has a built-in simulating environment, called GAZEBO, which can be used to simulate different control algorithms before deployment in reality.

References

[1] M. Vasic and A. Billard, "Safety issues in human-robot interactions," in *Proceedings of the IEEE international conference on robotics and automation*, 2013, pp. 197–204.

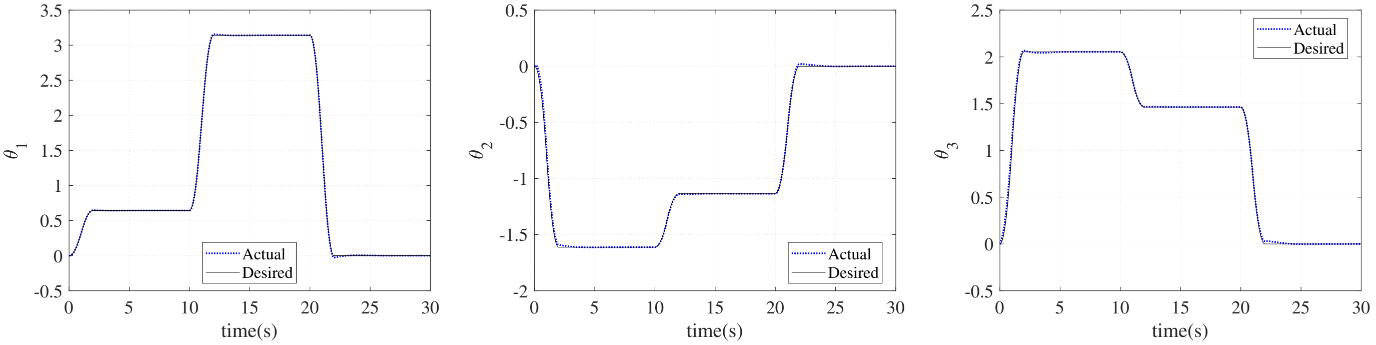
[2] O. Ogorodnikova, "Methodology of safety for a human robot interaction designing stage," in *Proceedings of the IEEE Conference on Human System Interactions*, 2008, pp. 452–457.

[3] A.-N. Sharkawy, P. N. Koustoumpardis, and N. A. Aspragathos, "Manipulator collision detection and collided link identification based on neural networks," in *Proceedings of the Springer International Conference on Robotics in Alpe-Adria Danube Region*. Springer, 2018, pp. 3–12.

[4] A.-N. Sharkawy, P. N. Koustoumpardis, and N. Aspragathos, "Neural network design for manipulator collision detection based only on the joint position sensors," *Robotica*, vol. 38, no. 10, pp. 1737–1755, 2020.

[5] Y. J. Heo, D. Kim, W. Lee, H. Kim, J. Park, and W. K. Chung, "Collision detection for industrial collaborative robots: A deep learning approach," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 740–746, 2019.

[6] F. Min, G. Wang, and N. Liu, "Collision detection and identification on robot manipulators based on vibration analysis," *Sensors*, vol. 19, no. 5, p. 1080, 2019.



(a)

(b)

(c)

Fig. 8: Workspace control of the robot manipulator.

[7] M. Anvaripour and M. Saif, “Collision detection for human-robot interaction in an industrial setting using force myography and a deep learning approach,” in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 2149–2154.

[8] H. Rajnathsing and C. Li, “A neural network based monitoring system for safety in shared work-space human-robot collaboration,” *Industrial Robot: An International Journal*, 2018.

[9] F. Flacco, T. Kröger, A. De Luca, and O. Khatib, “A depth space approach to human-robot collision avoidance,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2012, pp. 338–345.

[10] A. Mohammed, B. Schmidt, and L. Wang, “Active collision avoidance for human–robot collaboration driven by vision sensors,” *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 9, pp. 970–980, 2017.

[11] F. Mohammadi Amin, M. Rezayati, H. W. van de Venn, and H. Karimpour, “A mixed-perception approach for safe human–robot collaboration in industrial automation,” *Sensors*, vol. 20, no. 21, p. 6347, 2020.

[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Proceedings of the Springer European conference on computer vision*, 2014, pp. 740–755.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[14] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[15] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.