University of Wisconsin Milwaukee

UWM Digital Commons

Theses and Dissertations

June 2019

Trajectory Tracking of a Four Degree of Freedom Robotic Manipulator

Mohammad Rezwan Sheikh
University of Wisconsin-Milwaukee

Follow this and additional works at: https://dc.uwm.edu/etd

Part of the Mechanical Engineering Commons

Recommended Citation
Sheikh, Mohammad Rezwan, "Trajectory Tracking of a Four Degree of Freedom Robotic Manipulator" (2019). *Theses and Dissertations*. 2252.
https://dc.uwm.edu/etd/2252

This Thesis is brought to you for free and open access by UWM Digital Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UWM Digital Commons. For more information, please contact open-access@uwm.edu.

TRAJECTORY TRACKING OF A FOUR DEGREE OF FREEDOM ROBOTIC MANIPULATOR

by

Mohammad Rezwan Sheikh

A Thesis Submitted in
Partial Fulfillment of the
Requirements for the Degree of

Master of Science
in Engineering

at
The University of Wisconsin-Milwaukee
August 2019

ABSTRACT

TRAJECTORY TRACKING OF A FOUR DEGREE OF FREEDOM ROBOTIC MANIPULATOR

by

Mohammad Rezwan Sheikh

University of Wisconsin-Milwaukee, 2019
Under the Supervision of Professor Mohammad Habibur Rahman

A robotic manipulator can be utilized for multiple applications. As there are some expensive and bulky robotic manipulators with multi-functionalities are available in the market but not affordable for many people, a low-cost robotic manipulator, Dobot Magician, available in the market is used in this research to add more features in it. The forward kinematics and inverse kinematics are analyzed in this research besides studying about PID and computed torque control approaches. In this research, alphabets and numbers are coded using an object-oriented programming language (C#) to make the learning of alphabet, numbers, words, and sentence writing more fun to the children. Moreover, it has been found that the same robot and the same operation can be implemented in other applications. A speech recognizer is implemented to control the robot to execute some activities of daily living tasks which make it more accessible to the elderly individuals and the people with disabilities.

Keywords: Alphabet, Dobot, C+, Forward Kinematics, Kinect, Voice-Controlled, Inverse Kinematics, Object-Oriented programming, PID Control, Robotic manipulator

© Copyright by Mohammad Rezwan Sheikh, 2019
All Rights Reserved

With the name of Almighty Allah,
the All-Merciful, the Very-Merciful

Dedication

To

My Parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I should express my gratitude to Almighty Allah (SWT) that it would not have been possible without his blessings. I want to express my love, respect, and prayer for my parents because of their sacrifice for me. I would like to acknowledge my advisor and mentor Dr. Mohammad Habibur Rahman for his immense support and guidance. I am glad to work with Dr. Mohammad for my research in robotics as he has a vast knowledge in robotic technology, especially in rehabilitation robotics. It is always pleasure to work under the supervision of someone who not only gives research related support, but also gives positive counseling in life’s difficult situations, and I have found that kind of personality in him. I also want to thank Dr. Yongjin Sung and Dr. Mohammad Mashud for accepting my request to be in comprehensive exam committee even with their busy schedule. I am also grateful to all of my lab colleagues, especially Md Rasedul Islam, Md Assad-Uz-Zaman, Asif Al Zubayer, and Tanvir Ahmed for their support and I always found them well-disposed toward me even with their busy schedule.

CHAPTER 1

INTRODUCTION

1.1 Background

Robots are everywhere: in industries, scientific research, space, entertainment, homes, sports, and even in education to enhance the learning environment. It is not so far that people were familiar with the robot through science fiction novels and movies. Now almost all imaginations are becoming real in a way that people are closer to robots in their daily life.

Children are more inclined towards technological devices during their playtime (Beran et al. 2011). Application of robot always encourages interactive learning and engages children in learning activities (Highfield 2010; Wei et al. 2011). There are many robotic kits, from low-cost LEGO Mindstorms to humanoid robots NAO costs thousands of dollars, available in the market. The low-cost robots have very limited functionality compared to the high-cost humanoid robots which are not affordable to many parents (McComb 2008).

Another potential application of robots, as an assistive device, is to assist the elderly and/or individuals with disability in their activities of daily living (ADL). It is now an emerging field of research which is growing exponentially for more than the last 20 years (Foresi et al. 2018). In a qualitative study with thirty cognitively intact older adults, reveals that the participants of that study were willing to acquire robots as devices (Grossi et al. 2019). Assistive robotic devices can help in physical activities like cooking, and health monitoring, or can provide services like medicine reminder, event reminder, and offering suggestions (Broekens et al. 2009; Deutsch et al. 2019). People with disability due to stroke, traumatic brain injury, and spinal cord injury also require robotic devices not only for their rehabilitation purposes but also for assisting in their daily life. A large number of robotic devices for rehabilitation and assistive purposes are still confined in the research laboratories which are not commercially available (Islam et al. 2017). A commercially available robot ARMin, first marketed in 2011 (Riener et al. 2011), is only used for arm rehabilitation purposes, which is bulky, heavy, and quite expensive. Even with utmost necessity; many older people cannot imagine buying an assistive device because of the high price (Sarkar and Das 2019).

A robotic device with multifunctional abilities, commercially available, and affordable to a large population which can assist all members in a household cooperating with their daily activities is the most desirable one. Instead of building a new multifunctional robot, adding more functionalities in a commercially available one can save both time and money. Dobot Magician is such kind of commercially available lightweight, intelligent training robotic arm which is specially used for STEM education. We can add more functionalities to the Dobot, so the children can use it as a learning aided device, and the older people can use it as an assistive device. For instances, many children, with learning disabilities, are not interested in learning how to write alphabets, numbers, words, and sentences. Children are always fascinated with using technological devices as learning tools (Mousa, Ismail, and Salam 2017). Dobot robot has ability to draw and write in its workspace using its own software environment which is not suitable for teaching children alphabets, numbers, words, and sentences. Creating an environment giving Dobot some mobility will increase its workspace and developing a secondary software environment where user can change the input and output for getting different functionalities will increase its acceptability among all ages.

1.2 Specific Aims and Objectives

The specific aims of this research project are:

1. To teach children how to write alphabet, numbers, words, and sentences using robot, and to write a large signboard which cannot be printed using a printer;
2. To develop a robot guided environment to pick and place multiple colored blocks to teach children shapes and colors, and to develop a model of an industrial robot for picking and stacking boxes in different places;
3. To develop an environment for older and disabled people that the robot will pick their desired objects and carry it to them; and
4. To add a voice-controlled interface for individuals with disabilities.

To accomplish the above research aims the specific research tasks of this project includes-

i. To develop the forward kinematics model of Dobot Magician robot to find the end-effector tip's cartesian co-ordinate positions from given joint angle coordinates;

ii. To find the inverse kinematics of Dobot Magician robot to find the joint angle co-ordinates from the end-effector tip's cartesian co-ordinate positions;

iii. To develop motion planning of the Dobot Magician;

iv. To develop a graphical user interface using the C# programming language to provide the users of all ages a simple and easy to learn software environment to control the Dobot for multiple application;

v. To develop a method for writing alphabet, numbers, words, and sentences that kids with learning disabilities may become more interested in learning those; and

vi. To integrate the Kinect sensor, for voice sensing, with the Dobot to perform voice-controlled command execution by it.

1.3 Contribution

The key contributions of this research are

- A motion planning of writing alphabets and numbers is developed for Dobot, that can be utilized by other robotic manipulator, which imitates the human motion planning for writing alphabets and numbers; and
- Voice controlled command execution is introduced to the Dobot Magician.

CHAPTER 2

DOBOT DESIGN, SPECIFICATION, WORKSPACE, AND FEATURES

2.1 Background

This chapter outlines the design, specification, workspace, and features of the Dobot Magician robot. The Dobot Magician is a high precision multi-function robotic arm designed explicitly for Desktop use. It is designed in a way to add different end-effectors which perform writing, drawing, 3D printing, Laser engraving, and gripping blocks (pick and place).

2.2 Specifications

The Dobot Magician is a four-axis robotic arm with three stepper motors and one servo motor. Its end-effector, operated by the servo motor and a pneumatic pump, can carry up to 500 gram (payload) either using a gripper or a vacuum suction cup. Table 1 describes the specifications of a Dobot Magician.

Table 1.1 Specification of Dobot Robot (Shenzhen Yuejiang Technology Co. Ltd. 2019b)

Specifications	
Number of Axes	4
Payload	500g
Max. Reach	320mm
Position Repeatability(Control)	0.2 mm
Communication	USB/WIFI/Bluetooth
Power Supply	100 V-240 V, 50/60 Hz
Power In	12V / 7A DC
Consumption	60W Max
Working Temperature	-10°C – 60°C

2.3 Physical Properties of Dobot

The Dobot Magician robot is made up of Aluminium alloy 6061, and ABS engineering plastic, and it weighs around 3.6 kg (arms, controller, and end-effector attachment).

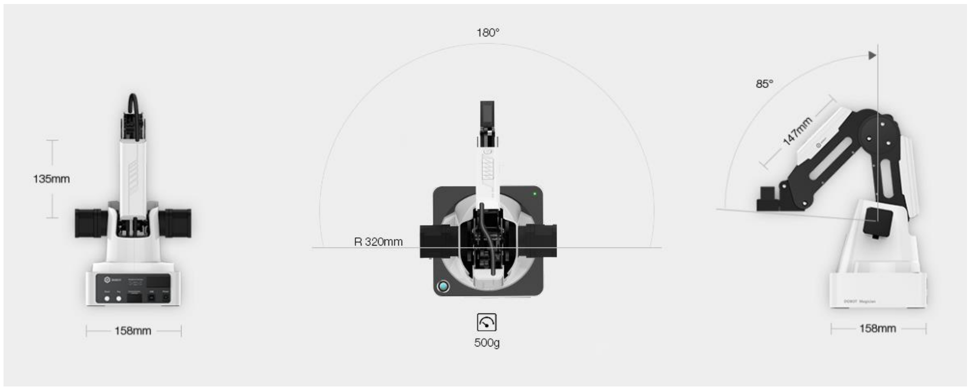
Table 1.2 Extensible I/O Interfaces

Extensible I/O Interfaces	
1	I/O*10 (Configurable as Analog Input or PWM Output)
2	Controllable 12 V Power output*4
3	Communication Interface (UART, Reset, Stop, 12 V, 5 V and two I/O included)
4	Stepper *2

Table 1.3 Physical Properties of End-Effectors' Attachment

End-effectors		
3D Printer Kit	Maximum Print Size (L*W*H)	150 mm *150 mm *150 mm (MAX)
	Material	PLA
	Resolution	0.1 mm
Laser	Power Consumption	500 mW
	Type	405 nm (Blue laser)
	Power	12 V
Pen Holder	Pen Diameter	10 mm
Vacuum Suction Cap	Cap Diameter	20 mm
	Pressure	-35 kPa
Gripper	Range	27.5 mm
	Drive Type	Pneumatic
	Force	8 N

2.4 Workspace of Dobot



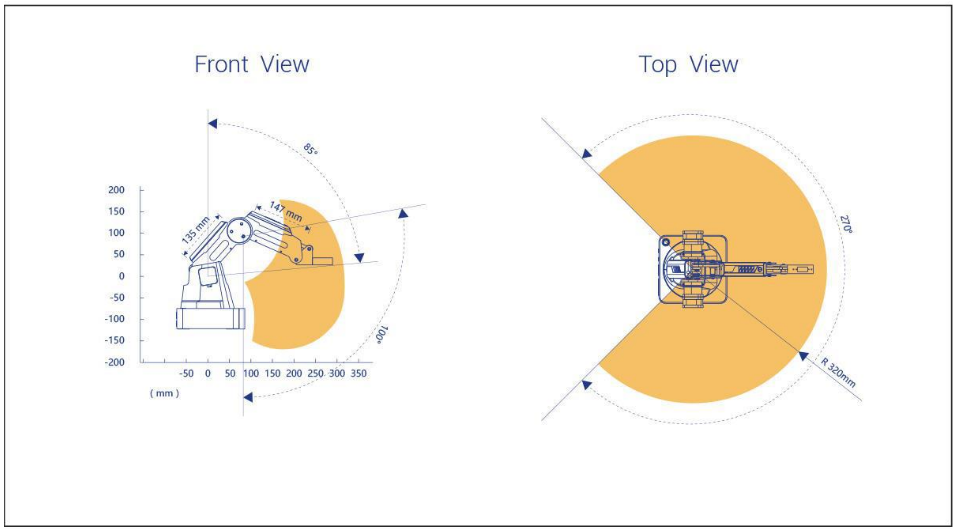


Figure 2.1 Workspace

Table 1.3. Joint Range of Motion and Speed

Axis Movement		
Axis	Range	Max Speed (250g workload)
Joint 1 base	-135° to +135°	320° /s
Joint 2 rear arm	0° to +85°	320° /s
Joint 3 forearm	-10° to +95°	320° /s
Joint 4 rotation servo	+90° to -90°	480° /s

2.5 Application Software and SDK

Dobot Magician uses a software named DobotStudio which is developed by the manufacturer of the Dobot. It runs on graphical user interface (GUI). It has features like giving command to the Dobot for performing operations like writing and drawing, laser engraving, 3D Printing, and controlling the robot using hand gesture (leapmotion). It has two more features, Blockly and Script for graphical and script programming respectively.

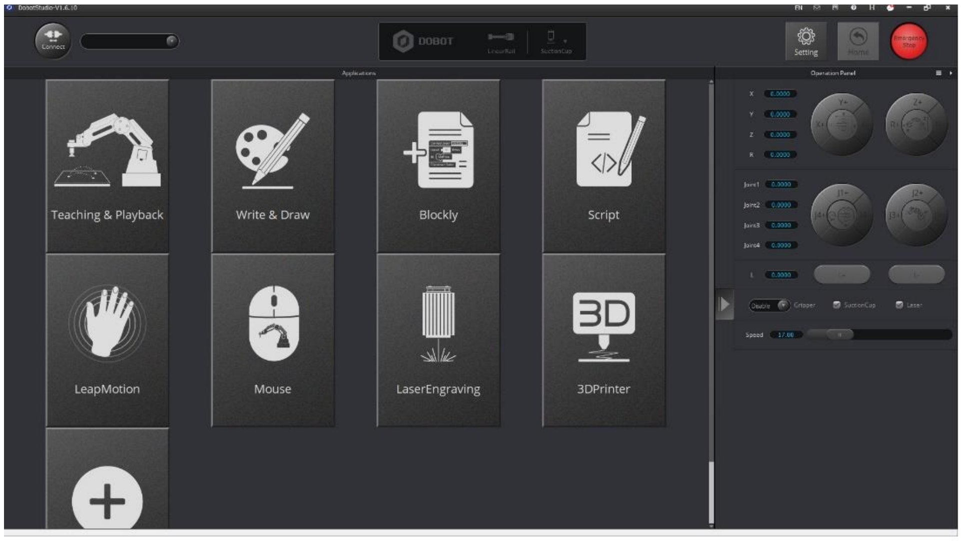


Figure 2.2 DobotStudio Software Interface

CHAPTER 3

DOBOT KINEMATICS

3.1 Background

In this chapter, we present the kinematic modeling of the Dobot Magician. This chapter describes the details of the kinematic modeling of the Dobot Magician. Modified Denavit-Hartenberg (DH) notations were used to develop the kinematic model of the Dobot robot.

3.2 Kinematics

Modified D-H conventions are used in developing the kinematic model of the Dobot Magician robot. The procedure of coordinate frame assignment and the definition of DH parameters are summarized in *section 3.2.1*.

3.2.1 Coordinate Frame Assignment Procedure

There are several ways to assign coordinate frames to the manipulator links. For the Dobot Magician we have followed the Denavit-Hartenberg method(Denavit 1955; Craig 2014). The steps are as follows(Hartenberg and Denavit 1964)

- assuming each joint is 1DoF revolute joint;
- identification and locating the axes of rotation;
- labeling the joint axes $Z_0, \dots \dots, Z_n$;

locating the origin of each link-frame (O_i) where the common perpendicular line between the successive joint axes (i.e., Z_{i-1} and Z_i) intersects. If the joint axes are not parallel, it is required to locate the link-frame origin at the point of intersection between the axes;

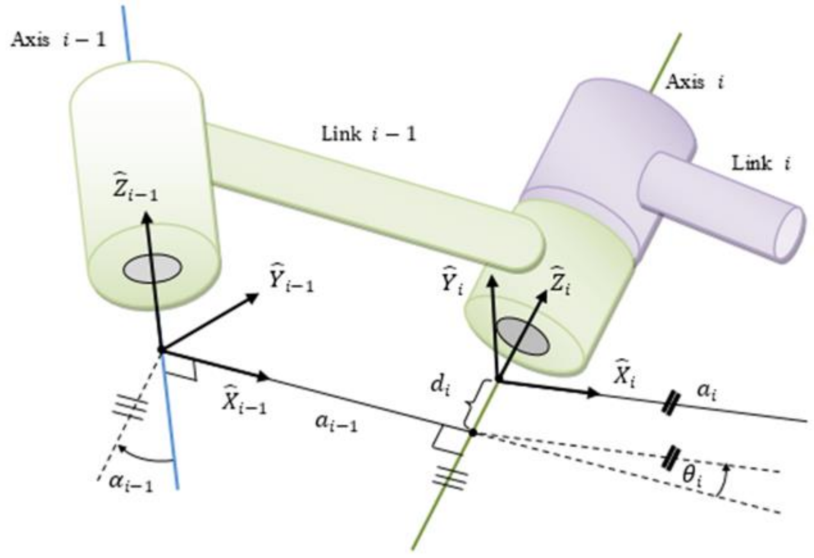


Figure 3.1 Coordinate frame assignment (Adapted from (Rahman 2012)*)*

- locating the X_i axis (at link frame origin O_i) as pointing along the common normal line between the axes Z_{i-1} and Z_i. If the joint axes intersect, it is required to establish X_i in a direction *normal* to the plane containing both axes (Z_{i-1} and Z_i);

- establishing the Y_i axis through the origin O_i to complete a right-hand coordinate system.

3.2.2 Definition of D-H Parameters

A link of a robot can be described by four parameters (two parameters for describing the link itself and other two for describing the link's relation to a neighboring link) if we assign the coordinate frames as described above (Denavit 1955). These parameters are known as Denavit-Hartenberg (DH) parameters. The definitions of the DH parameters are given below (Hartenberg and Denavit 1964):

- Link Length (a_{i-1}): the length measured along X_{i-1}, from axis Z_{i-1} to axis Z_i;
- Link Twist (α_{i-1}): the angle measured about X_{i-1}, from axis Z_{i-1} to axis Z_i;
- Link Offset (d_i): the distance measured along the axis Z_i; from X_{i-1} to axis X_i, and
- Joint Angle (θ_i): the angle measured about Z_i, from X_{i-1} to axis X_i

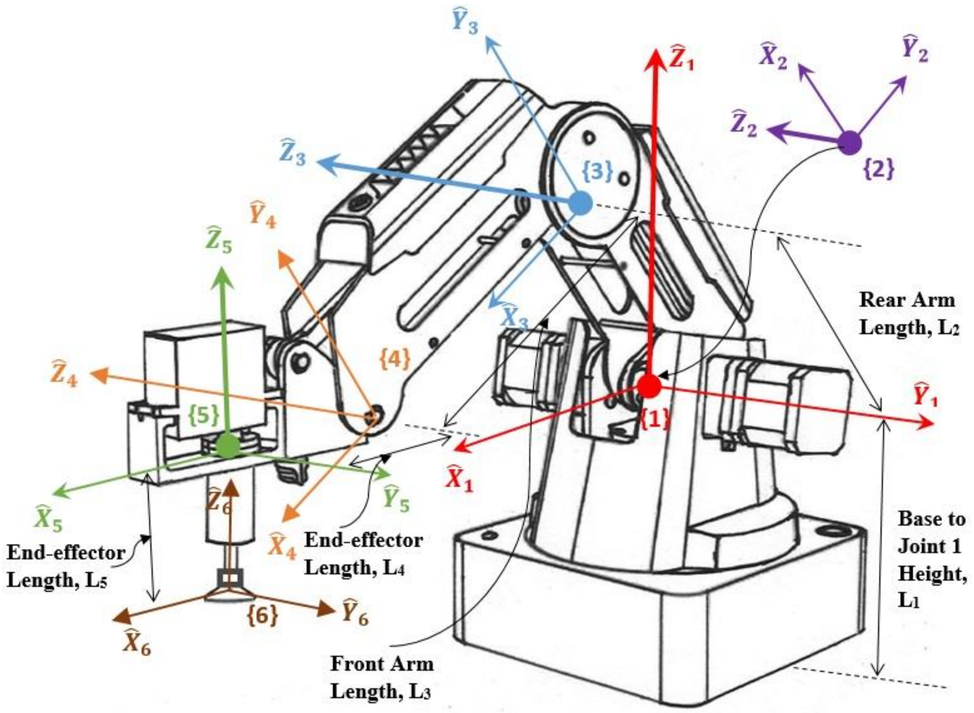


Figure 3.2 Frame Assignment to Dobot Magician

To obtain the DH parameters, we assume that the coordinate frames (i.e., the link-frames which map between the successive axes of rotation) coincide with the joint axes of rotation and have the same order, i.e., frame {1} coincides with joint 1, frame {2} with joint 2, and so on.

As shown in figure 3.2, the joint axes of rotation of the Dobot Magician corresponding to joint 1, joint 2, joint 3, and joint 4 are indicated by darker arrows with solid circle origin where the rotations are around Z axes (i.e., Z_1, Z_2, Z_3, and Z_5). In this model, frame {1}, {2}, {3}, and {5} correspond to joint 1, joint 2, joint 3, and joint 4 respectively. Joint 1 and joint 4 provide horizontal rotation, and joint 2 and joint 3 represent vertical rotations.

The origin of the frame {2} coincides with frame {1} which is located at a distance L_1 from the base. The distance between frame {2} to frame {3} is L_2 (rear arm length), and the distance between frame {3} to frame {4} is L_3 (front arm length). Note that the end-effector attachment in Dobot magician remains in the horizontal position even after vertical rotation of the joint 2 and the joint 3.

The modified DH parameters corresponding to the placement of the link frames (in figure 3.2) are summarized in Table 3.1. These DH parameters are used to get the homogeneous transformation matrix, which represents the positions and orientations of the reference frame concerning the fixed reference frame. It is assumed that the fixed reference frame {0} is located at distance L_1 apart from the first reference frame {1}.

Table 3.1 Modified DH Parameter

Frame (i)	α_{i-1}	a_{i-1}	d_i	θ_i
1	0°	0	L_1	θ_1
2	90°	0	0	θ_2
3	0°	L_2	0	θ_3
4	0°	L_3	0	$\theta_4 = -\theta_2 - \theta_3$
5	-90°	L_4	0	θ_5
6	0°	0	$-L_5$	0°

In table 3.1, α_{i-1} is the link twist, a_{i-1} corresponds to link length, d_i stands for link offset, and θ_i is the joint angle of the Dobot Magician.

3.2.3 Homogeneous Transformation Matrices

We know that the general form of a link transformation that relates frame $\{i\}$ relative to the frame $\{i-1\}$ (Craig 2014) is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} {}^{i-1}_{i}R^{3\times3} & {}^{i-1}_{i}P^{3\times1} \\ 0^{1\times3} & 1 \end{bmatrix} \tag{3.1} $$

where, ${}^{i-1}_{i}R$ is the rotation matrix that describes frame $\{i\}$ relative to frame $\{i-1\}$ and can be expressed as:

$$ {}^{i-1}_{i}R = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} \end{bmatrix} \tag{3.2} $$

and, ${}^{i-1}_{i}P$ is the vector that locates the origin of frame $\{i\}$ relative to frame $\{i-1\}$ and can be expressed as:

$$ {}^{i-1}_{i}P = [a_{i-1} \quad -sin\,\alpha_{i-1}\,d_i \quad cos\,\alpha_{i-1}\,d_i]^T \tag{3.3} $$

Using equations (3.1) to (3.3) the individual homogeneous transfer matrix that relates two successive frame (figure 3.2), we get

$$ {}^0_1T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^1_2T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^2_3T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^3_4T = \begin{bmatrix} cos\,\theta_4 & -sin\,\theta_4 & 0 & L_3 \\ sin\,\theta_4 & cos\,\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4} $$

$$ {}^4_5T = \begin{bmatrix} cos\,\theta_5 & -sin\,\theta_5 & 0 & L_4 \\ 0 & 0 & 1 & 0 \\ -sin\,\theta_5 & -cos\,\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^5_6T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -L_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The homogenous transformation matrix that relates frame {6} to frame {0} can be obtained by multiplying individual transformation matrices.

$$ {}^0_6T = [{}^0_1T.\,{}^1_2T.\,{}^2_3T.\,{}^3_4T.\,{}^4_5T.\,{}^5_6T] \tag{3.5} $$

The single transformation matrix thus found from equation (3.5) represents the positions and orientations of the reference frame attached to the suction cup with respect to the fixed reference frame {0}. After solving (matrix multiplication) equation (3.5) using equation (3.4), we get

$$ {}^0_6T = \begin{bmatrix} \cos(\theta_1+\theta_5) & -\sin(\theta_1+\theta_5) & 0 & cos\,\theta_1\,(L_4 + L_3\cos(\theta_2+\theta_3) + L_2\cos\theta_2) \\ \sin(\theta_1+\theta_5) & \cos(\theta_1+\theta_5) & 0 & sin\,\theta_1\,(L_4 + L_3\cos(\theta_2+\theta_3) + L_2\cos\theta_2) \\ 0 & 0 & 1 & L_1 - L_5 + L_3\sin(\theta_2+\theta_3) + L_2\sin\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6} $$

Where, ${}^0_6R$ is the rotation matrix that describes frame {6} relative to frame {0}:

$$ {}_{6}^{0}R = \begin{bmatrix} cos(\theta_1+\theta_5) & -sin(\theta_1+\theta_5) & 0 \\ sin(\theta_1+\theta_5) & cos(\theta_1+\theta_5) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (3.7) $$

and, ${}_{6}^{0}P$ is the vector that locates the origin of frame {6} relative to frame {0}:

$$ {}_{6}^{0}P = \begin{bmatrix} cos\,\theta_1\,(L_4 + L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \\ sin\,\theta_1\,(L_4 + L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \\ L_1 - L_5 + L_3\,sin(\theta_2+\theta_3) + L_2\,sin\,\theta_2 \end{bmatrix} \qquad (3.8) $$

The position of suction cup tip (frame {6}) is always constant with respect to the end-effector attachment joint to the front arm tip (frame {4}). However, if there is any rotation in joint 4 (frame {5}), only orientation of frame {6} changes with respect to frame {4}. We can depict this in the form of the transformation matrix of frame {6} with respect to frame {4} in equation (3.9),

$$ {}_{6}^{4}T = \begin{bmatrix} cos\,\theta_5 & -sin\,\theta_5 & 0 & L_4 \\ 0 & 0 & 1 & -L_5 \\ -sin\,\theta_5 & -cos\,\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.9) $$

We also get the transformation matrix of wrist frame {4} with respect to the base frame {0},

$$ {}_{4}^{0}T = \begin{bmatrix} \cos\theta_1 & 0 & sin\,\theta_1 & cos\,\theta_1\,(L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \\ \sin\theta_1 & 0 & -\cos\theta_1 & sin\,\theta_1\,(L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \\ 0 & 1 & 0 & L_1 + L_3\,sin(\theta_2+\theta_3) + L_2\,sin\,\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.10) $$

Where the last column is the position vector of wrist frame {4} with respect to the base frame {0} which we can represent as -

$$
{}_4^0P = \begin{bmatrix} \cos\theta_1\,(L_3\cos(\theta_2+\theta_3) + L_2\cos\theta_2) \\ \sin\theta_1\,(L_3\cos(\theta_2+\theta_3) + L_2\cos\theta_2) \\ L_1 + L_3\sin(\theta_2+\theta_3) + L_2\sin\theta_2 \end{bmatrix} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.11}
$$

3.2.4 Inverse Kinematics

The inverse kinematics solution for a robotic manipulator is computationally complex compared to direct kinematics. It is often difficult to find a closed-form solution due to the nonlinear nature of the equations to solve. Furthermore, an inverse kinematics problem for a redundant robotic manipulator is much more complex since it gives infinite solutions.

If we have the desired position and orientation of the object relative to the base, we can imply inverse kinematics to compute the joint angles required to reach to the desired position. Solving the problem is divided into two parts. First of all, it is required to do frame transformation to get the wrist frame (frame {4}). Then, the inverse kinematics are required to solve for the joint angles.

If the desired position of the top face of an object (frame {6}) is given as (x, y, z) with respect to the base frame from figure 6, we get the desired wrist frame position using the frame transformations:

$$
\begin{gathered} P_x = x - L_4 \\ P_y = y \\ P_z = z + L_5 \end{gathered} \tag{3.12}
$$

Now, we will consider two different approaches, algebraic and geometric, to acquire the desired joint angle position.

3.2.4.1 Algebraic Solution

After recalling equation (3.11), we get a set of equation,

$$P_x = cos\,\theta_1\,(L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \qquad (3.13)$$

$$P_y = sin\,\theta_1\,(L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2) \qquad (3.14)$$

$$P_z = L_1 + L_3\,sin(\theta_2+\theta_3) + L_2\,sin\,\theta_2 \qquad (3.15)$$

Solving equation (3.13) and (3.14), we get the desired angular position (θ_1) of joint 1,

$$\theta_1 = \tan^{-1}\left(\frac{P_y}{P_x}\right) \qquad (3.16)$$

Reshaping equation (3.13) and (3.15), we get

$$a = \frac{P_x}{cos\,\theta_1} = L_3 cos(\theta_2+\theta_3) + L_2\,cos\,\theta_2 \qquad (3.17)$$

$$b = P_z - L_1 = L_3\,sin(\theta_2+\theta_3) + L_2\,sin\,\theta_2 \qquad (3.18)$$

From equation (3.17) and (3.18), we get

$$a^2 + b^2 = {L_2}^2 + {L_3}^2 + 2L_2L_3\cos\theta_3 \qquad (3.19)$$

$$\cos\theta_3 = \frac{a^2 + b^2 - {L_2}^2 - {L_3}^2}{2L_2L_3} \qquad (3.20)$$

There will be a solution of equation (3.20) when the right side of the equation gives a value between -1 to 1. The object is out of the robot's workspace if the value is outside of this constraint. Assuming the object is in the workspace, we get

$$\sin\theta_3 = \pm\sqrt{1-\cos^2\theta_3} \tag{3.21}$$

Now using equation (3.20) and (3.21), we calculate the desired angular position (θ_3) of joint 3,

$$\theta_3 = \tan^{-1}\left(\frac{\sin\theta_3}{\cos\theta_3}\right) \tag{3.22}$$

For solving θ_2, we rearrange equation (3.17) and (3.18) we get

$$a = (L_3\cos\theta_3 + L_2)\, cos\,\theta_2 - (L_3\sin\theta_3)\sin\theta_2 \tag{3.23}$$

$$b = (L_3\cos\theta_3 + L_2)\sin\theta_2 - (L_3\sin\theta_3)\, cos\,\theta_2 \tag{3.24}$$

Rewriting equation (3.23) and (3.24) we get

$$a = m\, cos\,\theta_2 - n\sin\theta_2 \tag{3.25}$$

$$b = m\sin\theta_2 - n\, cos\,\theta_2 \tag{3.26}$$

where

$$m = L_3\cos\theta_3 + L_2 \tag{3.27}$$

$$n = L_3\sin\theta_3 \tag{3.28}$$

To solve problem in this form, performing changes in variable helpful. If

$$p = +\sqrt{m^2 + n^2} \tag{3.29}$$

and $$\delta = \tan^{-1}\left(\frac{n}{m}\right) \tag{3.30}$$

then $$\begin{aligned} m &= p\cos\delta \\ n &= p\sin\delta \end{aligned} \tag{3.31}$$

Equation (3.23) and (3.24) now can be written as

$$a = p\, cos\,\delta\, cos\,\theta_2 - p\, sin\,\delta \sin\theta_2 \tag{3.32}$$

$$b = p\, cos\,\delta \sin\theta_2 - p\, sin\,\delta\, cos\,\theta_2 \tag{3.33}$$

thus

$$a = p\cos(\delta + \theta_2) \tag{3.34}$$

$$b = p\sin(\delta + \theta_2) \tag{3.35}$$

From equation (3.34) and (3.35), we get

$$\delta + \theta_2 = \tan^{-1}\left(\frac{b}{a}\right) \tag{3.36}$$

and so $$\theta_2 = \tan^{-1}\left(\frac{b}{a}\right) - \tan^{-1}\left(\frac{n}{m}\right) \tag{3.37}$$

3.2.4.2 Geometric Solution

We will decompose the spatial geometry of the arm into two plane geometry problems. First, we will solve for vertical rotations on joint 2 and joint 3. Then, we will solve horizontal rotation on joint 1.

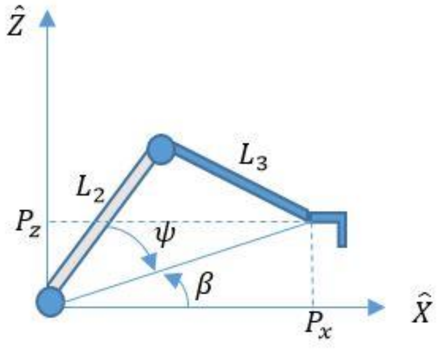


Figure 3.3 Plane geometry of vertical rotations

Applying 'law of cosines' to solve for θ_3:

$$P_x{}^2 + P_y{}^2 + P_z{}^2 = L_2{}^2 + L_3{}^2 - 2L_2L_3\cos(180° - \theta_3) \qquad (3.38)$$

Now, $\cos(180° - \theta_3) = -\cos\theta_3$, so we have

$$\cos\theta_3 = \frac{P_x{}^2 + P_y{}^2 + P_z{}^2 - L_2{}^2 - L_3{}^2}{2L_2L_3} \qquad (3.39)$$

Above equation (3.39) is valid when $P_x{}^2 + P_y{}^2 + P_z{}^2 \le L_2 + L_3$

Solution for θ_2:

From figure 3.3, we get

$$\beta = \tan^{-1}\left(\frac{P_z}{\sqrt{P_x{}^2 + P_y{}^2}_x}\right) \qquad (3.40)$$

Applying 'law of cosine' again

$$L_3{}^2 = L_2{}^2 + P_x{}^2 + P_y{}^2 + P_z{}^2 - 2L_2\sqrt{P_x{}^2 + P_y{}^2 + P_z{}^2}\cos\psi \quad (3.41)$$

Hence,
$$\cos\psi = \frac{L_2{}^2 + P_x{}^2 + P_y{}^2 + P_z{}^2 - L_3{}^2}{2L_2\sqrt{P_x{}^2 + P_y{}^2 + P_z{}^2}} \quad (3.42)$$

and
$$\sin\psi = \pm\sqrt{1 - \cos^2\psi} \quad (3.43)$$

From equation (3.42) and (3.43) we get,

$$\psi = \tan^{-1}\left(\frac{\sin\psi}{\cos\psi}\right) \quad (3.44)$$

From figure (3.3) we get,

$$\theta_2 = \beta + \psi \quad (3.45)$$

$$\theta_2 = tan^{-1}\left(\frac{P_z}{\sqrt{P_x{}^2 + P_y{}^2}}\right) + tan^{-1}\left(\frac{sin\,\psi}{cos\,\psi}\right)$$

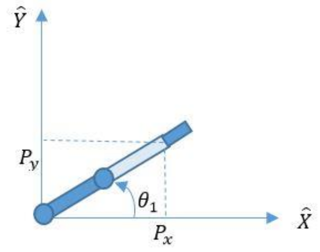


Figure 3.4 Plane geometry of horizontal rotation

Now from figure 3.4, we get the horizontal rotation (θ_1) for joint 1,

$$\theta_1 = \tan^{-1}\left(\frac{P_y}{P_x}\right) \quad (3.46)$$

CHAPTER 4

CONTROL

4.1 Background

This chapter focuses on a linear control (Proportional-Integral-Derivative Control), and a nonlinear model-based control (Computed Torque Control) approaches to simulate the dynamic model of the Dobot to follow a reference trajectory.

4.2 Proportional-Integral-Derivative (PID) Control

The general layout of the PID control approach is depicted in figure 4.1. The PID control technique is the most widely used control technique for industrial applications (Craig 2014). It is simple in design, efficient in computation, and it does not require a dynamic model of the system. As shown in the schematic in figure 4.1, with a PID control, the joint torque commands to drive the Dobot can be expressed by the following equation:

$$\tau = K_P(\theta_d - \theta) + K_V(\dot{\theta}_d - \dot{\theta}) + K_I \int (\theta_d - \theta)\, dt \tag{4.1}$$

where $\theta_d, \theta \in \mathbb{R}^4$are the vectors of desired and measured joint angles respectively, $\dot{\theta}_d,\ \dot{\theta} \in \mathbb{R}^4$are the vectors of desired and measured joint velocities respectively, K_P, K_V, K_I are the diagonal positive definite gain matrices, and $\tau \in \mathbb{R}^4$is the generalized torque vector. Let the error vector E and its derivative be: Therefore, this equation (4.1) can be re-formulated as an error equation:

$$E = \theta_d - \theta;\ \dot{E} = \dot{\theta_d} - \dot{\theta} \tag{4.2}$$

Therefore, this equation (4.1) can be re-formulated as an error equation:

$$\tau = K_P E + K_V \dot{E} + K_I \int E\, dt \tag{4.3}$$

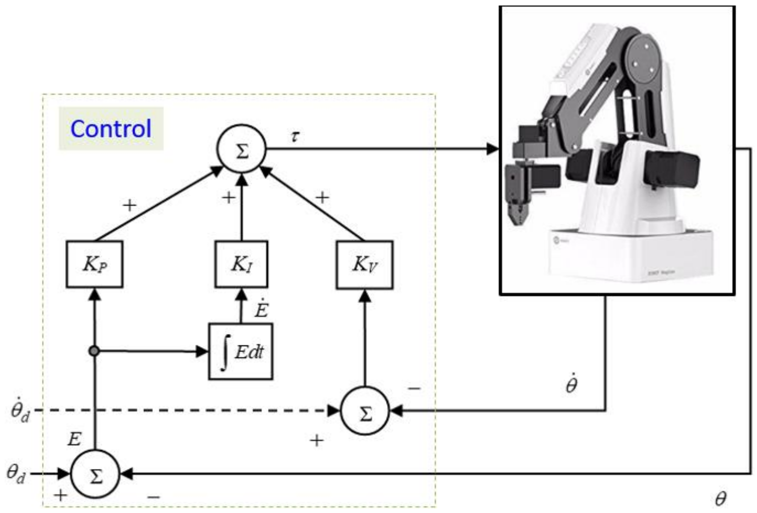


Figure 4.1 Schematic diagram of PID control

The relation (4.3) is decoupled, therefore individual torque command for each joint would be as follows.

$$\tau_i = K_{P_i} e_i + K_{V_i} \dot{e_\iota} + K_{I_i} \int e_i\, dt \tag{4.4}$$

where $e_i = \theta_{d_i} - \theta_i \cdots (i = 1, 2, \cdots 4)$; $\theta_i,\ \theta_{d_i}$ are the measured and desired trajectory for joint *i* respectively, and

$\dot{E} = [\dot{e_1} \quad \dot{e_2} \quad \cdots \quad \dot{e_4}]^T$,

$\theta_d = [\theta_{d_1} \quad \theta_{d_2} \quad \cdots \quad \theta_{d_4}]^T,\ \theta = [\theta_1 \quad \theta_2 \quad \cdots \quad \theta_4]^T$;

$K_P = diag[K_{P_1} \quad K_{P_2} \quad \cdots \quad K_{P_4}]^T$, $K_V = diag[K_{V_1} \quad K_{V_2} \quad \cdots \quad K_{V_4}]^T$, and

$K_I = diag[K_{I_1} \quad K_{I_2} \quad \cdots \quad K_{I_4}]^T$.

4.3 Computed Torque Control (CTC)

Computed torque control includes the dynamic model of a system, therefore with a CTC better tracking performance can be realized as long as the dynamic model is accurate. The dynamic behavior of the Dobot can be expressed by the well-known rigid body dynamic equation as:

$$M(\theta)\ddot{\theta} + V\left(\theta, \dot{\theta}\right) + G(\theta)\tau \tag{4.5}$$

Where

$\theta \in \mathbb{R}^4$ is the joint variables vector,

τ is the generalized torque vector,

$M(\theta) \in \mathbb{R}^{4\times4}$ is the inertia matrix,

$V\left(\theta, \dot{\theta}\right) \in \mathbb{R}^4$ is the Coriolis/centrifugal vector, and

$G(\theta) \in \mathbb{R}^4$ is the gravity vector.

$$\ddot{\theta} = -M^{-1}(\theta)\left[V\left(\theta, \dot{\theta}\right) + G(\theta)\right] + M^{-1}(\theta)\tau \tag{4.6}$$

$M^{-1}(\theta)$ always exists since $M(\theta)$ is symmetrical and positive definite.

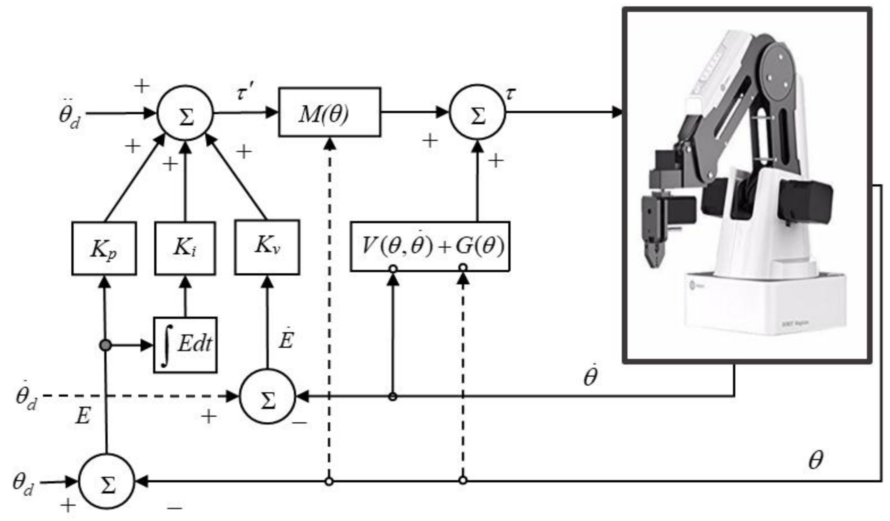


Figure 4.2 Schematic Diagram of Computed Torque Control

The layout of the modified computed torque control technique is depicted in figure 4.2. Unlike the conventional computed torque control approach, here we have added an integral term to have a better tracking performance and to compensate the trajectory tracking error that usually occurs due to imperfect dynamic modeling, parameter estimation, and also for external disturbances. The control torque in figure 4.2 can be written as:

$$\tau = M(\theta)\left[\ddot{\theta}_d + K_v(\dot{\theta}_d - \dot{\theta}) + K_p(\theta_d - \theta) + K_i \int (\theta_d - \theta)\, dt\right] + V(\theta, \dot{\theta}) + G(\theta) \qquad (4.7)$$

From relations (4.5) and (4.7), we may write:

$$\ddot{\theta} = \ddot{\theta}_d + K_v(\dot{\theta}_d - \dot{\theta}) + K_p(\theta_d - \theta) + K_i \int (\theta_d - \theta)\, dt \qquad (4.8)$$

where $\theta_d, \dot{\theta}_d,$ and $\ddot{\theta}_d$ are the desired position, velocity and acceleration, respectively, and K_p, K_v, and K_i diagonal positive definite matrices. Let the error vector E and its derivative be:

$$E = \theta_d - \theta;\ \dot{E} = \dot{\theta}_d - \dot{\theta},\ \ddot{E} = \ddot{\theta}_d - \ddot{\theta} \tag{4.9}$$

Therefore, equation (4.8) can be rewritten in the following form:

$$\ddot{E} + K_v\dot{E} + K_pE + K_i\int E\,dt = 0 \tag{4.10}$$

Where the control gains K_p, K_v, and K_i are positive definite matrices. Therefore, the proper choice of these matrices ensures the stability of the system.

4.4 Cartesian Trajectory Tracking with Joint Based Control

The general layout of the control architecture for 'Cartesian trajectory tracking with joint-based control' is given in figure 4.3 where $\theta_d,\ \dot{\theta}_d, \ddot{\theta}_d$ represent desired joint position, velocity and acceleration respectively and those for Cartesian coordinates are represented by $x_d,\ \dot{x}_d, \ddot{x}_d$ respectively.

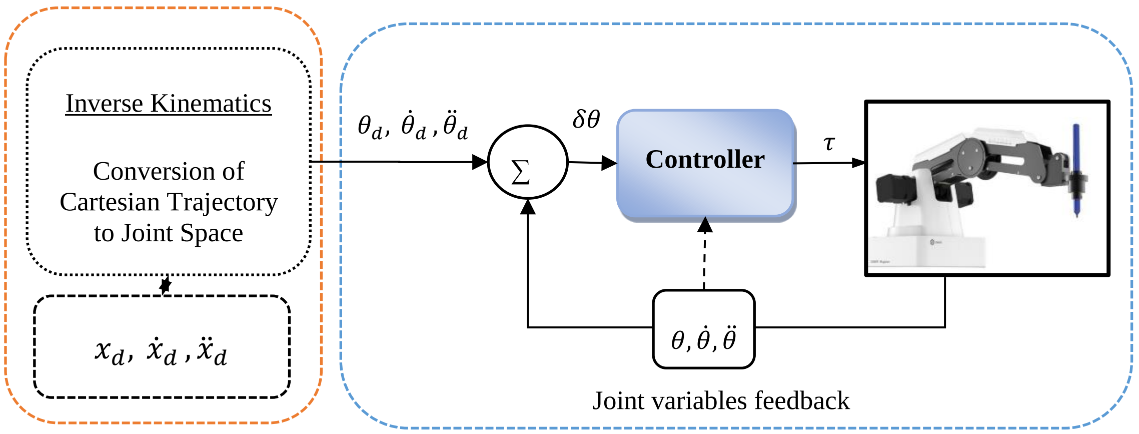

Inverse Kinematics
Conversion of Cartesian Trajectory to Joint Space
$x_d,\ \dot{x}_d, \ddot{x}_d$
$\theta_d,\ \dot{\theta}_d, \ddot{\theta}_d$
Σ
$\delta\theta$
Controller
τ
$\theta, \dot{\theta}, \ddot{\theta}$
Joint variables feedback

Figure 4.3 Cartesian trajectory tracking with joint based control

The left dotted box (figure 4.3) indicates the Cartesian to joint space trajectory conversion process (inverse kinematics). Given an end-effector position and orientation (x_d), desired Cartesian velocities ($\dot{x}_d$) and accelerations ($\ddot{x}_d$) can be found using cubic polynomial method (Craig 2014). The inverse-kinematic solution of Dobot is presented in Chapter 3. It can be seen from above schematic (the right dotted square box) that once the desired joint variables are found the control scheme followed the principle of joint-based control approach (Craig 2014). As seen from figure 4.3 the inputs to the controllers are joint errors ($\delta\theta$) and output is the torque command (τ) to the Dobot.

CHAPTER 5

SOFTWARE DEVELOPMENT

5.1 Background

Dobot Magician comes with its own software, DobotStudio, to perform the built-in operations like Write & Draw, 3D print, Laser Engraving, and controlling using leap motion and mouse. DobotStudio is not sufficient to complete all the tasks of four aims of this research interactively. The manufacture of Dobot Magician gives users SDKs in different programming languages for secondary development of the Dobot Magician. In this research, Dobotdll.dll and C# SDK are used to get access to the sensors data and executing point to point movement of the end-effector of the Dobot and the sliding rail. The C# SDK of Kinect V2 is also used in this project to access the speech feature of the Kinect sensor.

5.2 Tools for Dobot C# App Development

Dobot DLL, Dobot API, Dobot communication protocol, C# programming language, and Kinect SDK are required for the secondary development of the Dobot

5.2.1 Dobot DLL

The DobotDll.dll is a supported DLL file which is required to get access to the MCU of the motherboard of the Dobot Magician. The full form of DLL is Dynamic-Link Library, which contains a set of data and code to execute certain operations in Windows operating systems (Hart 2005). The precompiled source code of the DLL file is studied intuitively to understand the working principle of the Dobot Magician (Shenzhen Yuejiang Technology Co. Ltd. 2019).

5.2.2 Dobot API

An application programming interface (API), in computer programming, is a set of communication protocols, subroutine definitions, and tools for developing software (Monperrus et al. 2012). The Dobot controller supports two types of commands: immediate command and queue command (Shenzhen Yuejiang Technology Co. Ltd. 2018a). In the Dobot C# software development, both commands are being implemented. Immediate command is executed by the Dobot controller right after receiving the command whether other commands are processing or not. On the other hand, the queued commands are executed by the Dobot controller maintaining the sequence of receiving the command.

5.2.3 Dobot Communication Protocol

Dobot communication protocol is used in this research to build communication with the Dobot. The communication can be established by USB serial port, TTL level serial port, Wi-Fi (UDP), where the physical layer receives eight bite length raw data to determine the starting and end of the data, and to verify the accuracy of the data. The communication protocol is the combination of the packet header, packet load, and checksum to ensure correct communication (Shenzhen Yuejiang Technology Co. Ltd. 2018b).

5.2.4 C# Programming Language

C# is a powerful programming language for application development on the .NET framework (Sharp 2015). It is a modern programming language which some features like the ability to do object-oriented programming, rich library, and user-friendly coding environment making it preferable to the app developers. In this research, Visual Studio 2019 community edition is used to develop the graphical user interface (GUI) to control Dobot.

5.2.5 Kinect SDK

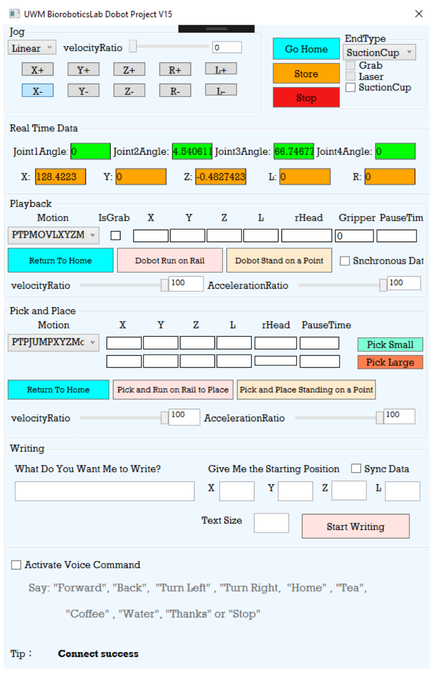
Kinect SDK seems like the eyes and ears of an application based on the Kinect sensor. It provides tools and APIs to develop Kinect based application. The SDK has features like tracking people and their skeletal movement, determining the distance between objects, capturing the location of an audio source, and voice recognition. In this research project, the voice-activated application is developed using the voice recognition technology of Microsoft.

5.3 Application Development

A desktop application is build using C# programming language to perform all the aims of this research.

5.3.1 Desktop Application

The desktop application is developed using Window Presentation Foundation (WPF) which facilitates creating a user interface (UI) easily. Figure 5.1 shows the user interface to control the Dobot robot. It is created based on the SDK provided for the secondary development of Dobot. In figure 5.1, the interface contains bunch buttons, text boxes, check boxes, sliding bar, combo boxes, text blocks, and group boxes for giving the Dobot different set of operating instructions.



Jog: Linear velocityRatio 0

EndType: SuctionCup

Joint1Angle: 0 Joint2Angle: 4.540611 Joint3Angle: 66.74677 Joint4Angle: 0

X: 128.4223 Y: 0 Z: -0.4827423 L: 0 R: 0

Playback: Motion PTPMOVLXYZM IsGrab X Y Z L rHead Gripper 0 PauseTim

velocityRatio 100 AccelerationRatio 100

Pick and Place: Motion PTPJUMPXYZMc X Y Z L rHead PauseTime

velocityRatio 100 AccelerationRatio 100

What Do You Want Me to Write? Give Me the Starting Position Sync Data X Y Z L Text Size

Activate Voice Command

Say: "Forward", "Back", "Turn Left" , "Turn Right, "Home" , "Tea", "Coffee" , "Water", "Thanks" or "Stop"

Tip : **Connect success**

Figure 5.1 Windows application for controlling Dobot developed with C# programming language

5.3.2 Pseudo Code

Pseudo codes for writing tasks, pick and place tasks, and voice command task are briefly described below.

5.3.2.1 Pseudo Code for Writing Tasks

```
//This program will write pseudo code for writing text using Dobot
if (start writing button is pressed)
then
Read 'What Do You Want Me to Write'
Set to towrite

Read X, Y, Z, L
Set to x,y,z,l

foreach(character in towrite)
{
Switch(character)
{
case 'A'
        write A
        break
case 'B'
        write B
        break
case 'C'
        write C
        break
case 'D'
        write D
        break
case 'E'
        write E
        break
case 'F'
        write F
        break
case 'G'
        write G
        break
case 'H'
        write H
        break
case 'I'
        write I
case 'J'
        write J
        break
case 'K'
        write K
break
case 'L'
        write L
        break
case 'M'
        write M
break
case 'N'
        write N
        break
case 'O'
        write O
        break
case 'P'
        write P
        break
case 'Q'
        write Q
        break
case 'R'
        write R
case 'S'
        write S
        break
case 'T'
        write T
        break
case 'U'
        write U
        break
case 'V'
        write V
        break
case 'W'
        write W
        break
case 'X'
        write X
        break
case 'Y'
        write Z
        break
case 'Z'
        write Z
        break
case '1'
        write 2
case '2'
        write 2
        break
case '3'
        write 3
        break
case '4'
        write 4
        break
case '5'
        write 5
        break
case '6'
        write 6
        break
case '7'
        write 7
        break
case '8'
        write 8
        break
case '9'
        write 9
        break
case '0'
        write 0
```

break	break	break	break

```
case 'NULL'
        move the sliding rail without writing
        break
}
move the sliding rail to write next character
}
// end of Pseudo Code
```

In the writing program, the formulas for writing ‘A’, ‘B’, ‘C’, ‘1’, ‘2’, and ‘3’, for instances, used in the above ‘switch’ statements are given in figure 5.2 below.

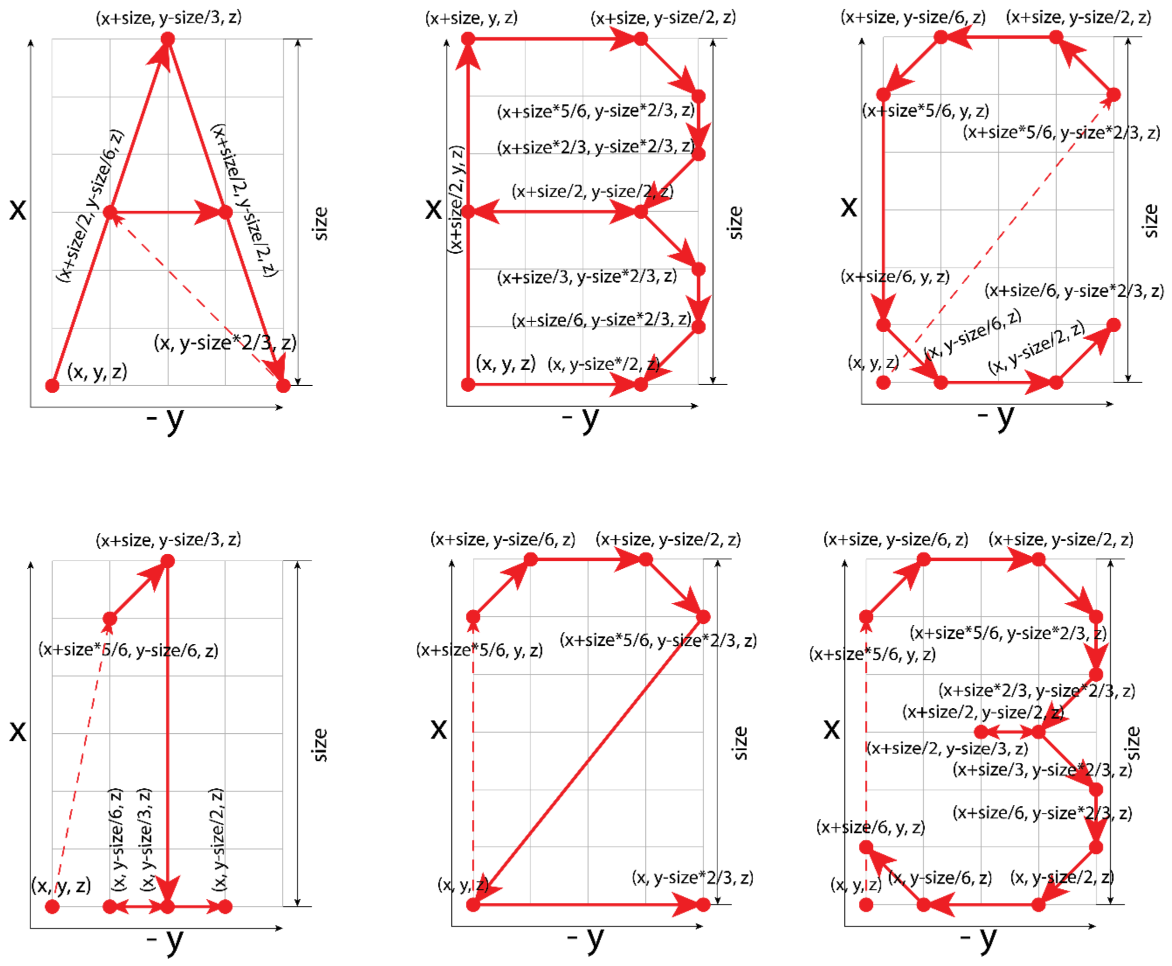


Figure 5.2 Cartesian coordinate grid generation for writing ‘A’, ‘B’, ‘C’, ‘1’, ‘2’, and ‘3’

5.3.2.2 Pseudo Code for Pick & Place Task

```
//Pseudo code for Pick and Place Task
Read Motion mode (PTPJUMPXYZMode)

Read X1, Y1, Z1, L1, rHead1, PauseTime1(Picking Initial Conditions)
Read X2, Y2, Z2, L2, rHead2, PauseTime2(Placing Initial Conditions)

if('Pick and Run on Rail to Place' is pressed)
        execute ptpwithL to run Dobot on rail
else if ('Pick and Place Standing on a Point' is pressed)
        execute ptp to run Dobot without moving on rail
else if ('Pick Small' is pressed)
        execute ptp to pick 9 colored cubes from horizontal position
        execute ptp to place 9 colored cubes to vertical position
else if ('Pick Large' is pressed)
        execute ptp to pick 9 colored rectangular boxes from horizontal position
        execute ptp to place 9 colored rectangular boxes to vertical position
else if ('Return To Home' is pressed)
        execute setHomecmd operation to go to homing position
//End of Pseudo code for Pick and Place Task
```

5.3.2.3 Pseudo Code for Voice Command

```
//Pseudo code for Voice Command
if('Activate Voice Command' is checked && SpeechRecognized)
Switch(Semantics value)
{
case 'FORWARD'
        execute ptpwithL to run Dobot on rail To go to user
        break
case 'BACKWARD'
        execute ptpwithL to run Dobot on rail To go to home position
        break
case 'LEFT'
        execute ptp to turn joint 1 -90 degree
        break
case 'RIGHT'
        execute ptp to turn joint 1 90 degree
        break
case 'TEA'
        execute ptpwithL to pick up tea
        execute ptpwithL to bring the tea to the user
        break
case 'COFFEE'
```

```
        execute ptpwithL to pick up coffee
        execute ptpwithL to bring the coffee to the user
        break
case 'WATER'
        execute ptpwithL to pick up water
        execute ptpwithL to bring the water to the user
        break
case 'STOP'
        stop executing
        break
case 'HOME'
        execute setHomecmd to go homing position
        break

//End of Pseudo code for Voice Command Task
```

CHAPTER 6

EXPERIMENTS DESIGN

6.1 Experimental Setup

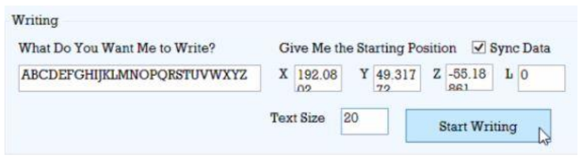
In this research, the Dobot Magician was mounted on a linear sliding rail which has a one-meter linear working distance to give the Dobot ample working space. There are two stages of experiments. In the 1st stage, to accomplish the Aim 1, a drawing pen was mounted on the end-effector using the pen holder. In the 2nd stage, to achieve the Aim 2, Aim 3, and Aim 4, a suction cup kit was mounted on the end-effector. The Kinect sensor, for giving voice-command, was placed three feet left from the Dobot's initial position on the sliding rail.

6.2 Writing Tasks

In the software interface, a group box named 'Writing' is designed to give instructions for writing alphabets, numbers, words, and sentences where users can give desired writing output, text size, and initializing coordinate for start writing.

6.2.1 Writing for Children

Task 1: Writing all the alphabet from A to Z. In figure 6.1, in the left text box the desired writing


Writing

What Do You Want Me to Write? ABCDEFGHIJKLMNOPQRSTUVWXYZ

Give Me the Starting Position ☑ Sync Data

X 192.0802 Y 49.31772 Z -55.18861 L 0

Text Size 20

Figure 6.1 User Interface for writing-writing alphabets

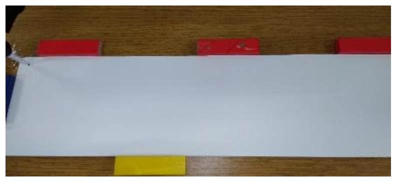
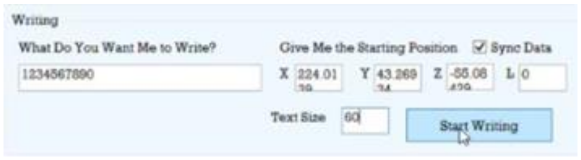


Figure 6.2 Initializing the starting coordinate to write the alphabets

output “ABCDEFGHIJKLMNOPQRSTUVWXYZ” is given. A checkbox ‘Sync Data’ is checked to get the real-time end-effector position in ‘X’, ‘Y’, and ‘Z’ text boxes and the sliding rail position in the ‘L’ text box. In the ‘Text Size’ text box, the text height is given in mm to write the text in desired size. In figure 6.2, the pen tip is placed on the paper from where the writing task will start to begin after clicking on ‘Start Writing’ button.


Writing

What Do You Want Me to Write? 1234567890

Give Me the Starting Position [✓ Sync Data]

X 224.01 39 Y 43.269 34 Z -55.06 439 L 0

Text Size 60

Figure 6.3 User Interface for writing-writing numbers

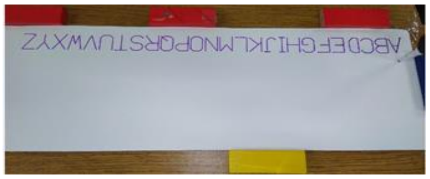
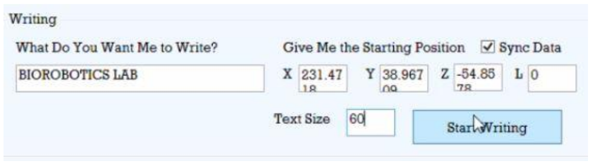


Figure 6.4 Initializing the starting coordinate to write the numbers

Task 2: Writing all the numbers from 1 to 0. In figure 6.3, in the left text box the desired writing output “1234567890” is given. A checkbox ‘Sync Data’ is checked to get the real-time end-effector position in ‘X’, ‘Y’, and ‘Z’ text boxes and the sliding rail position in the ‘L’ text box. In the ‘Text Size’ text box, the text height is given in mm. In figure 6.4, the pen tip is placed on the paper from where the writing task will start to begin after clicking on ‘Start Writing’ button.

6.2.2 Signboard Writing


Writing

What Do You Want Me to Write? BIOROBOTICS LAB

Give Me the Starting Position [✓ Sync Data]

X 231.47 18 Y 38.967 09 Z -54.85 78 L 0

Text Size 60

Figure 6. 5 User Interface for writing-Two separate words at the same time

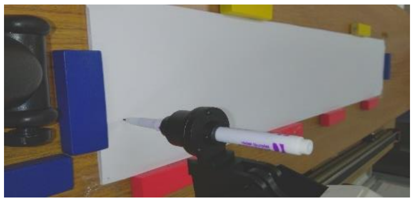
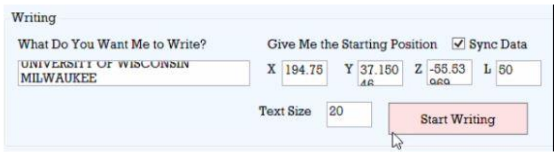


Figure 6.6 Initializing the starting coordinate to write the words

Task 1: Writing words or titles. In figure 6.5, in the left text box the desired writing output “BIOROBOTICS LAB” is given. A checkbox ‘Sync Data’ is checked to get the real-time end-effector position in ‘X’, ‘Y’, and ‘Z’ text boxes and the sliding rail position in the ‘L’ text box. In the ‘Text Size’ text box, the text height is given in mm. In figure 6.6, the pen tip is placed on the paper from where the writing task will start to begin after clicking on ‘Start Writing’ button.

Task 2: Writing a sentence. In figure 6.7, in the left text box the desired writing output “University of Wisconsin Milwaukee” is given. A checkbox ‘Sync Data’ is checked to get the real-time end-effector position in ‘X’, ‘Y’, and ‘Z’ text boxes and the sliding rail position in the ‘L’ text box. In the ‘Text Size’ text box, the text height is given in mm. In figure 6.8, the pen tip is placed on the paper from where the writing task will start to begin after clicking on ‘Start Writing’ button.


Writing

What Do You Want Me to Write? UNIVERSITY OF WISCONSIN MILWAUKEE

Give Me the Starting Position ☑ Sync Data

X 194.75 Y 37.150 46 Z -55.53 969 L 50

Text Size 20

Figure 6.7 User Interface for writing-writing several words or a sentence

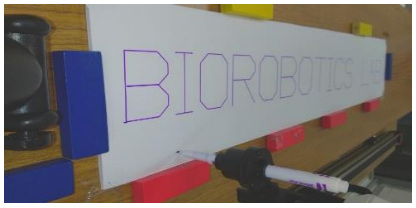
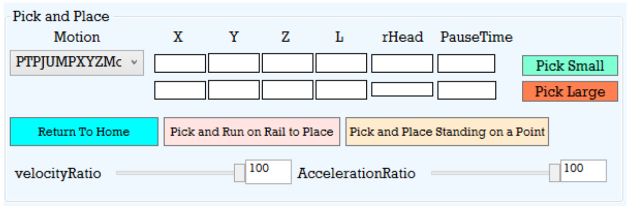


Figure 6.8 Initializing the starting coordinate to write the sentence

6.3 Pick and Place Tasks

In the software interface, a group box named ‘Pick and Place’ is designed to give commands for picking objects from one place and drop it to another place. In figure 6.9, there is a drop-down menu to select the mode pick and place task. If ‘PTPJUMPXYZMode’ is selected, the pick and place will be done in a way that after picking an object the robot end-effector will elevate to a certain height and after reaching to the desired location, it will drop the object lowering


Pick and Place

Motion: PTPJUMPXYZMc

X Y Z L rHead PauseTime

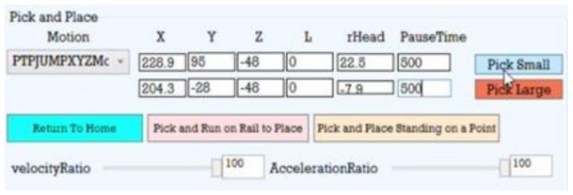
velocityRatio 100 AccelerationRatio 100

Pick Small
Pick Large
Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

Figure 6. 9User Interface for giving pick and place command the end-effector height. There are two rows of text boxes for ‘X’, ‘Y’, ‘Z’, ‘L’, ‘rHead’, and ‘Pause Time’, where the first row is for picking position coordinates and the second row is for dropping position. There are two sliding bars for giving a percentage of velocity ratio and acceleration ratio respectively. If ‘Pick and Place Standing on a Point’ button is clicked, the Dobot will execute the pick and place task without running on the sliding rail. On the other hand, if the ‘Pick and Run on Rail to Place’ button is clicked, the Dobot will run on the sliding rail for a distance given for ‘L’ in the second row. The function of ‘Pick Small’ and ‘Pick Large’ buttons are specially designed for teaching colors and stacking block tasks.

6.3.1 Teaching Colors & Stacking Blocks

Task 1: Pick and place nine small cubes. In this task, it is required to stack the cubes based on color. In figure 6.10, the initializing coordinates for picking and placing are shown in the first



Pick and Place

Motion	X	Y	Z	L	rHead	PauseTime
PTPJUMPXYZMc	228.9	95	-48	0	22.5	500
	204.3	-28	-48	0	-7.9	500

velocityRatio 100
AccelerationRatio 100

Pick Small
Pick Large
Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

Figure 6.10 User Interface for pick and place of 9 colored cubes

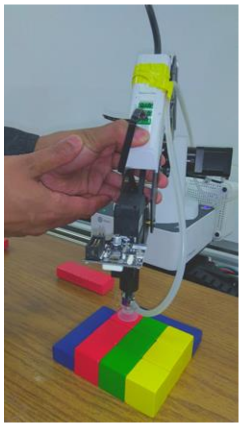


Figure 6.11 Initializing the starting coordinate of picking 1st cubic shape colored box

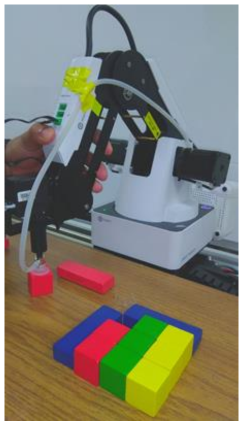


Figure 6.12 Initializing the starting coordinate of placing 1st cubic shape colored box

and the second row of text boxes respectively. Velocity ratio and acceleration ratio both are kept 100 percent. In figure 6.11, a button is pressed to move the forearm freely to place on a block to initialize the starting pick up coordinates. Similarly, in figure 6.12, the forearm is moved to initialize the starting placing coordinates. From figure 6.13 to figure 6.16 are showing different aspects of picking, placing, and stacking cubes.

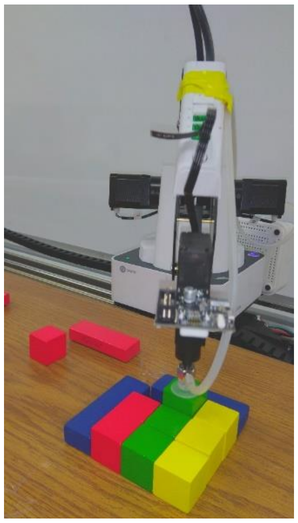


Figure 6.13 After placing the 1st cube (red), the end-effector is picking the green cube from next available position

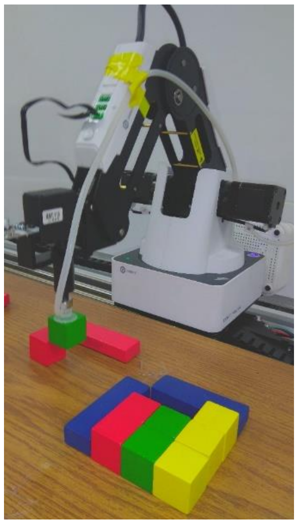


Figure 6.14 After picking the 2nd cube (green), the end-effector is placing it to the next available placing position

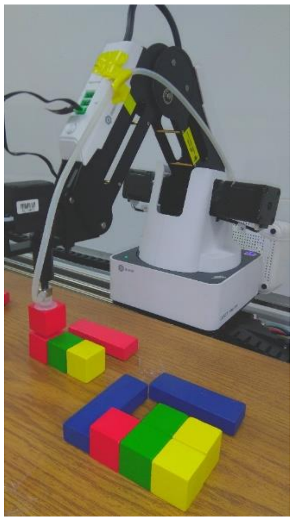


Figure 6. 15 After placing the 1st row of cubes, the end-effector is placing the red cube on another red cube placed at the beginning

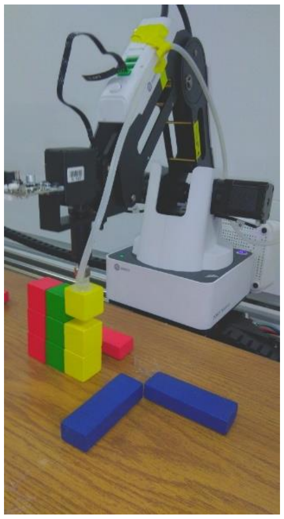
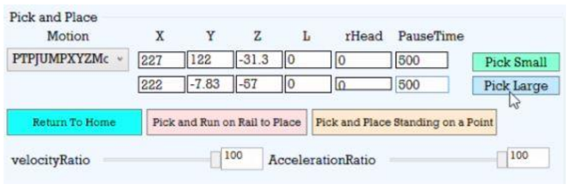


Figure 6.16 The end-effector stacked all red cubes together, all green cubes together and now stacking the last yellow cube

Task 2: Pick and place nine colored rectangular boxes. In this task, it is required to stack the boxes based on color. In figure 6.17, the initializing coordinates for picking and placing are shown in the first and the second row of text boxes respectively. Velocity ratio and acceleration ratio both are kept 100%. In figure 6.18, a button is pressed to move the forearm freely to place on a block to initialize the starting pick up coordinates. Similarly, in figure 6.19, the forearm is moved to initialize the starting placing coordinates. From figure 6.20 to figure 6.25 are shown and briefly explained different aspects of picking, placing, and stacking rectangular boxes.



Pick and Place

Motion	X	Y	Z	L	rHead	PauseTime
PTPJUMPXYZMc	227	122	-31.3	0	0	500
	222	-7.83	-57	0	0	500

velocityRatio 100 AccelerationRatio 100

Pick Small
Pick Large
Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

Figure 6.17 User Interface for pick and place of 9 colored rectangular box

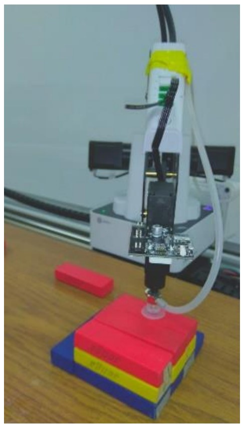


Figure 6.18 Initializing the starting coordinate of picking 1st colored rectangular box

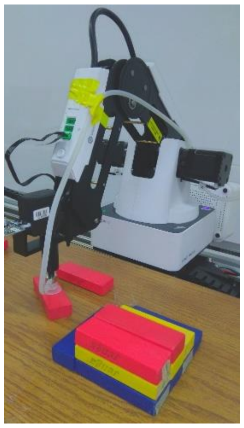


Figure 6.19 Initializing the starting coordinate of placing 1st colored rectangular box

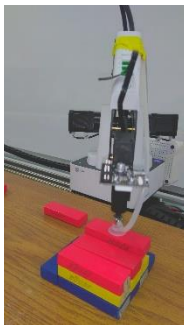


Figure 6.20: The suction cup *end-effector is picking the 1st rectangular box (red) from the starting picking coordinate*

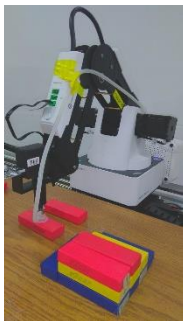


Figure 6.21: The suction cup *end-effector is placing the 1st rectangular box (red) to the starting placing coordinate*

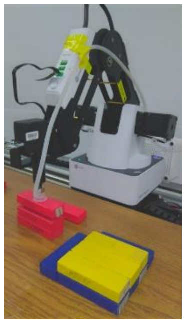


Figure 6.22: The suction cup *end-effector is stacking the last red rectangular box on other red boxes.*

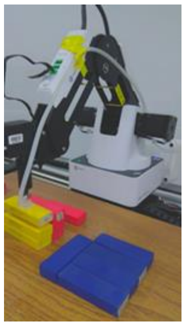


Figure 6.23: The suction cup *end-effector is stacking the last yellow rectangular box on other yellow boxes.*

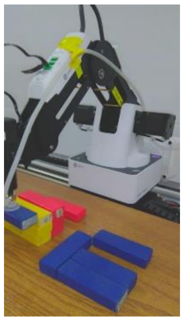


Figure 6.24: The suction cup *end-effector is starting to place blue boxes adjacent to the yellow box columns.*

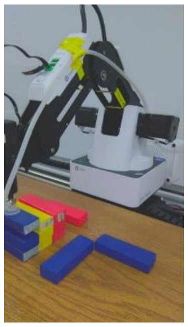
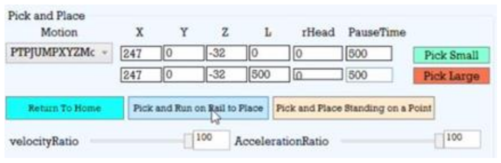


Figure 6.25: The suction cup *end-effector is stacking the last blue rectangular box on other blue boxes.*

6.3.2 Pick and Place Different Weighted Objects

Task 1: Pick and place an object of 184 gm weight. In this task, it is required to pick the object using the suction cup from its top face and drop it to 0.5-meter distance away from its picking position moving on the sliding rail. In figure 6.26, the initializing coordinates for picking and placing are shown in the first and the second row of text boxes respectively. Velocity ratio and acceleration ratio both are kept 100%. From figure 6.27 to figure 6.31 are sequentially shown and briefly explained different aspects of picking and placing the object and running on the sliding rail.


Pick and Place

Motion	X	Y	Z	L	rHead	PauseTime
PTPJUMPXYZMc	247	0	-32	0	0	500
	247	0	-32	500	0	500

velocityRatio 100 AccelerationRatio 100

Pick Small
Pick Large
Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

Figure 6.25 User Interface for pick and place of 184 gm weight

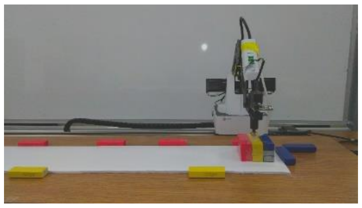


Figure 6.26 Picking up the 184 gm weight object from the given picking coordinate

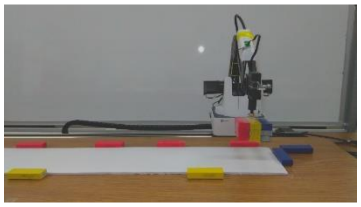


Figure 6.28: After grabbing the 184 gm weight elevated to the level which will be maintained constant while running on rail

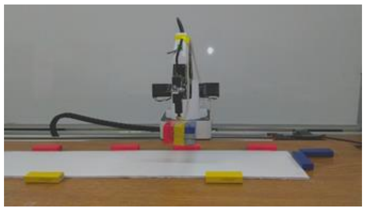


Figure 6.29: After reaching the elevated level, running on the sliding rail keeping the same level to drop it to the placing coordinate

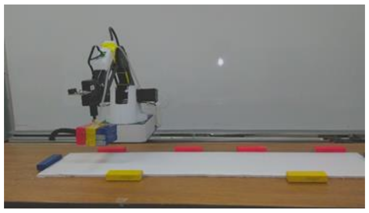


Figure 6.30: Reached to the placing X, and Y coordinate staying in the elevated level

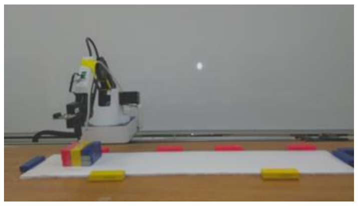
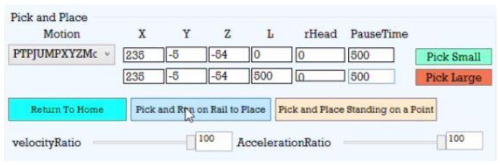


Figure 6.31: Placed at the desired location

Task 2: Pick and place an object of 282 gm weight. In this task, it is required to pick the object using the suction cup from its top face and drop it to 0.5-meter distance away from its picking position moving on the sliding rail. In figure 6.32, the initializing coordinates for picking and placing are shown in the first and the second row of text boxes respectively. Velocity ratio and acceleration ratio both are kept 100 percent. From figure 6.33 to figure 6.37 are sequentially shown and briefly explained different aspects of picking and placing the object and running on the sliding rail.


Pick and Place

Motion	X	Y	Z	L	rHead	PauseTime	
PTPJUMPXYZMc	235	-5	-54	0	0	500	Pick Small
	235	-5	-54	500	0	500	Pick Large

Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

velocityRatio 100 AccelerationRatio 100

Figure 6.32 User Interface for pick and place of 282 gm weight

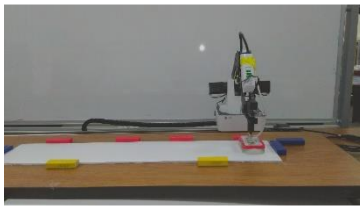


Figure 6.33: *Picking up the 282 gm weight object from the given picking coordinate*

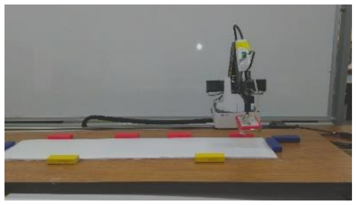


Figure 6.34: *After grabbing the 282 gm weight elevated to the level which will be maintained constant while running on rail*

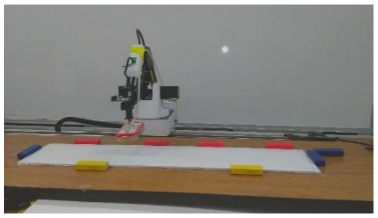


Figure 6.35: *After reaching the elevated level, running on the sliding rail keeping the same level to drop it to the placing coordinate*

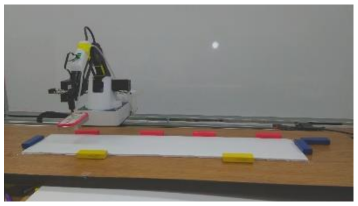


Figure 6.36: *Reached to the placing X, and Y coordinate staying in the elevated level*

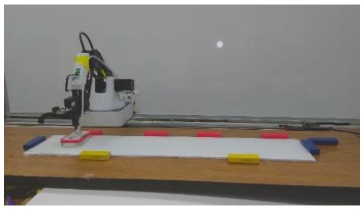
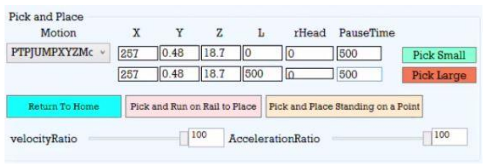


Figure 6.37: *Placed at the desired location*

Task 3: Pick and place an object of 467 gm weight. In this task, it is required to pick the object using the suction cup from its top face and drop it to 0.5-meter distance away from its picking position moving on the sliding rail. In figure 6.38, the initializing coordinates for picking and placing are shown in the first and the second row of text boxes respectively. Velocity ratio and acceleration ratio both are kept 100 percent. From figure 6.39 to 6.43 are sequentially shown and briefly explained different aspects of picking and placing the object and running on the sliding rail.



Pick and Place

Motion	X	Y	Z	L	rHead	PauseTime	
PTPJUMPXYZMc	257	0.48	18.7	0	0	500	Pick Small
	257	0.48	18.7	500	0	500	Pick Large

Return To Home
Pick and Run on Rail to Place
Pick and Place Standing on a Point

velocityRatio 100 AccelerationRatio 100

Figure 6.38: User Interface for pick and place of 467 gm weight

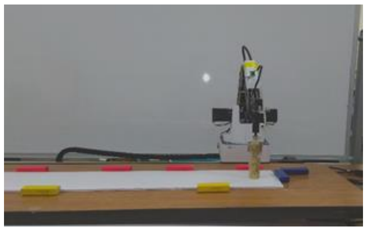


Figure 6.39: *Picking up the 282 gm weight object from the given picking coordinate*

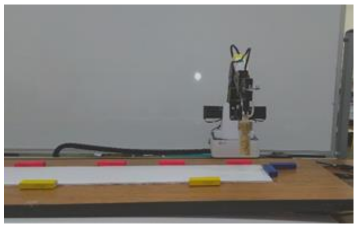


Figure 6.40: *After grabbing the 282 gm weight elevated to the level which will be maintained constant while running on rail*

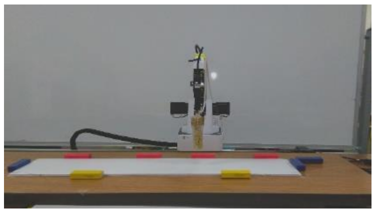


Figure 6.41 After reaching the elevated level, running on the sliding rail keeping the same level to drop it to the placing coordinate

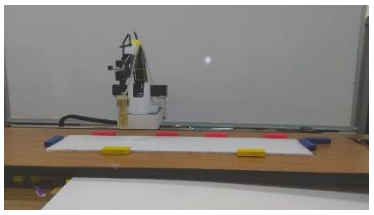


Figure 6.42: *Reached the placing X, and Y coordinate staying in the elevated level*

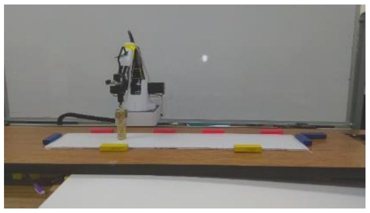
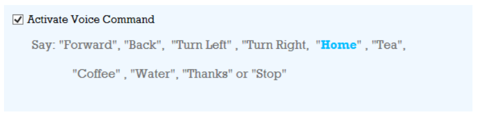


Figure 6.43 Placed at the desired location

6.4 Desired Objects Carrier

6.4.1 Pick and Carry Desired Drinks

This section is the implementation of pick and place through voice command. In this voice command enabled task, a user can give voice command to the Dobot using the Kinect sensor to bring either tea, coffee, or water to the user running on the sliding rail and running back to its original position to perform a execute a new command. In figure 6.44, it is shown that voice command is activated after checking the 'Activate Voice Command'. When the Kinect sensor detects a voice command, the corresponding command turns into blue. Figure 6.45 is showing the relative distance between the Kinect sensor and the Dobot. From figure 6.47 to 6.54 are shown briefly explained different aspects of picking the user's desired object and bring it to the users running on the sliding rail.


☑ Activate Voice Command

Say: "Forward", "Back", "Turn Left", "Turn Right, "Home", "Tea", "Coffee", "Water", "Thanks" or "Stop"

Figure 6.44: User Interface for giving a voice command, where different available commands are shown.

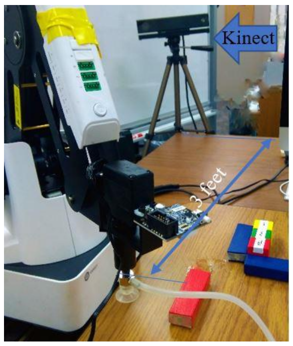


Figure 6.45: The Kinect Sensor is placed 3 feet away from the initial position of the Dobot on the sliding rail

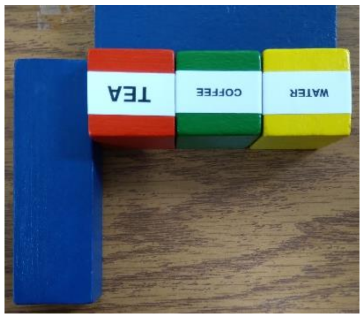


Figure 6.46: Model of Tea, Coffee, and Water which predefined position coordinates are known by the Dobot

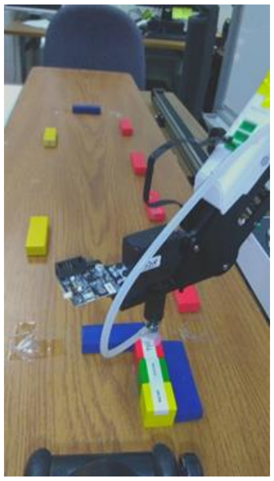


Figure 6.417 When the user gives a voice command ''Tea', the suction cup end-effector reaches to the predefined location of Tea to grab it

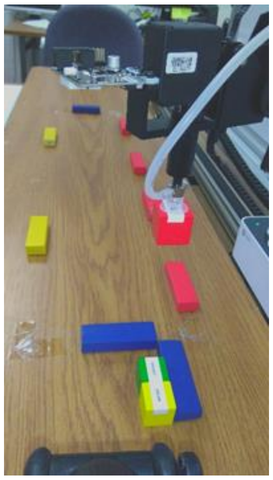


Figure 6.48 After grabbing the tea, it elevated to an upper-level before running on the sliding rail.

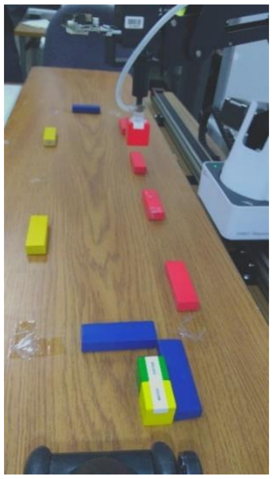


Figure 6.49 The Dobot is running on the rail keeping the same level of height to carry the tea to the user

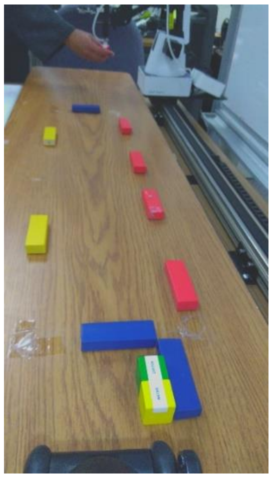


Figure 6.50 The Dobot is running on the rail keeping the same level of height to carry the tea to the user

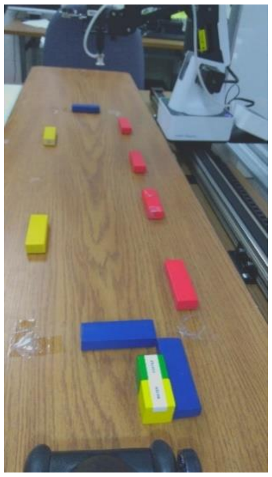


Figure 6.51 The Dobot is running on the rail keeping the same level of height to carry the tea to the user

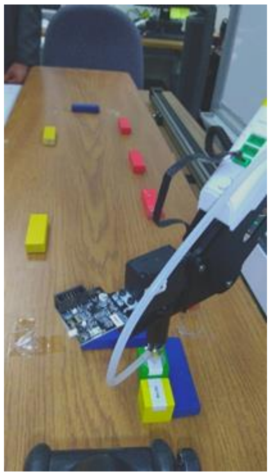


Figure 6.52 When the user gives a voice command 'Coffee', the suction cup end-effector reaches to the predefined location of the coffee to grab it

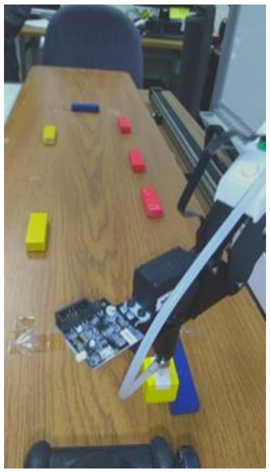


Figure 6.53 When the user gives a voice command 'Water', the suction cup end-effector reaches the predefined location of the water to grab it

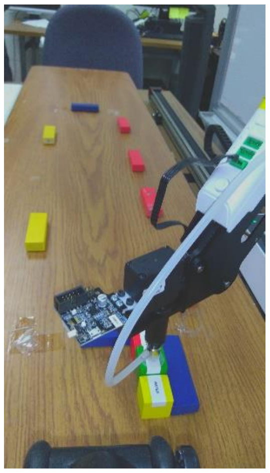


Figure 6.54 When the user gives a voice command 'Coffee', and the coffee is in adjacent to tea and water, while picking it up using the suction cup it does not interrupt the position of tea and water

6.4.2 More with Voice Command

It is always essential to keep an emergency stop for running a robot to stop it immediately. Even though the software is designed with emergency stop using a red ‘Stop’ button, providing a ‘Stop’ voice command has added an extra feather to the secondary development of the Dobot. There are few more commands like ‘Home’, ‘Forward’, ‘Back’, ‘Turn Right’, ‘Turn Left’, and ‘Thanks’ used in this software to make it more voice supported.

CHAPTER 7

RESULTS AND DISCUSSION

7.1 Writing Tasks Results

Figure 7.1, figure 7.4, and figure 7.7 show the three dimensional (3D) view of trajectories for writing A, B, and C respectively, where red lines are representing the actual trajectory the Dobot followed to write the characters and the black dashed lines are showing the desired trajectories to

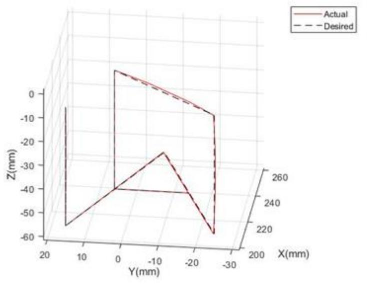


Figure 7.1: 3D view of the desired and actual trajectory for writing 'A'

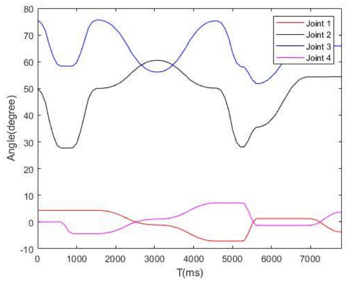


Figure 7.2: 4 Joints angular displacement tracking for writing 'A'

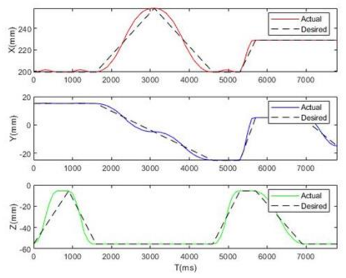


Figure 7.3: Desired and actual cartesian coordinates for writing 'A'

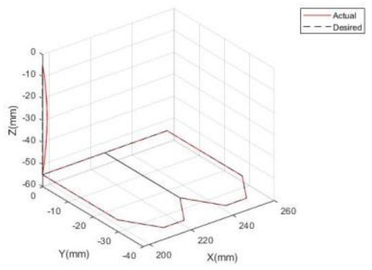


Figure 7.4: 3D view of the desired and actual trajectory for writing 'B'

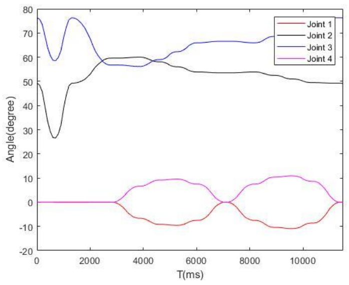


Figure 7.5: 4 Joints angular displacement tracking for writing 'B'

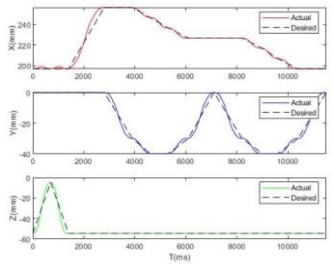
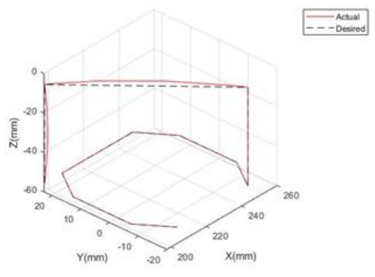
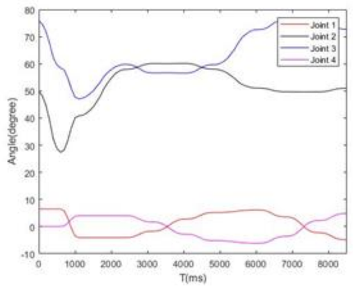


Figure 7.6: Desired and actual cartesian coordinates for writing 'B'

write those characters. Figure 7.3, figure 7.6, and figure 7.9 are showing the cartesian coordinates of the writing pen tip for writing A, B, and C respectively, where red, blue, and green lines are representing the actual trajectory of X, Y, and Z coordinates respectively in writing the characters, and the black dashed lines are showing the desired trajectories to write those characters. It can be seen from those figures that while moving from one point to another point the actual path is





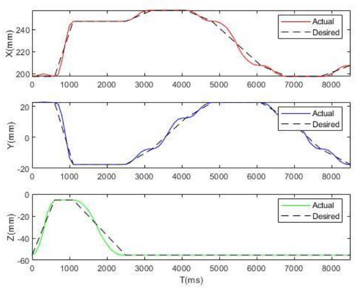


Figure 7.7: 3D view of the desired and actual trajectory for writing 'C'

Figure 7.8: 4 Joints angular displacement tracking for writing 'C'

Figure 7.9: Desired and actual cartesian coordinates for writing 'C'

following a cubic polynomial trajectory; however, the desired path is following a straight line trajectory. Finally, figures 7.2, 7.5, and 7.8 are depicting the changes in angular displacements of four joints (joint 1, 2, 3, and 4) when writing A, B, and C respectively. Comparing the figures 7.1, 7.2, and 7.3 for writing A, figures 7.4, 7.5, and 7.6 for writing B, and figures 7.7, 7.8, and 7.9 for writing C, it can be seen that when there is any elevation along Z axis without changes in X, Y values, the joint 1 angles remain constant as there is no angular motion in joint 1 motor. When there are changes only in X without any changes in Y, and Z, joint 2 increases and joint 3 decreases proportionally and joint 1 remains constant. On the other hand, when Y stays constant without changes in X, and Z, joint 2 and joint 3 remains constant, but joint 1 increases. When there are increases in both X and Y without any changes in Z, joint 2 increases and joint 3 decreases proportionally with joint 1 decreases. Similarly, when there are decreases in both X and Y without any changes in Z, joint 1 and joint 3 increases and joint 2 decreases.

Figure 7.10, figure 7.13, and figure 7.16 show the three dimensional (3D) view of trajectories for writing 1, 2, and 3 respectively, where red lines are representing the actual trajectory the Dobot followed to write the numbers and the black dashed lines are showing the desired trajectories to write those characters. Figure 7.12, figure 7.15, and figure 7.18 are showing the cartesian coordinates of the writing pen tip for writing 1, 2, and 3 respectively, where red, blue, and green lines are representing the actual trajectory of X, Y, and Z coordinates respectively in writing the characters, and the black dashed lines are showing the desired trajectories to write those characters.

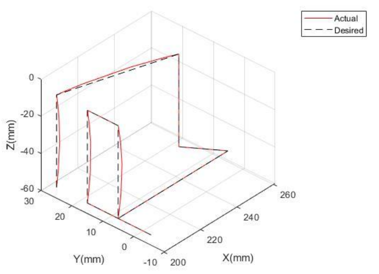


Figure 7.10: 3D view of the desired and actual trajectory for writing '1'

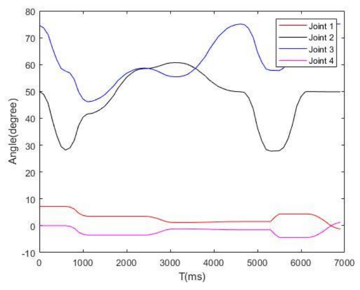


Figure 7.11: Four Joints angular displacement tracking for writing '1'

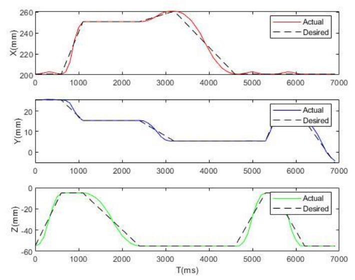


Figure 7.12: Desired and actual cartesian coordinates for writing '1'

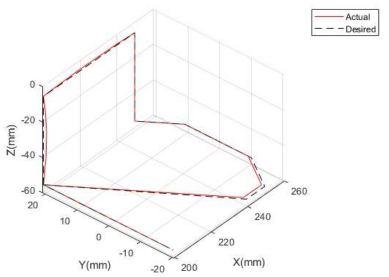


Figure 7.13: 3D view of the desired and actual trajectory for writing '2'

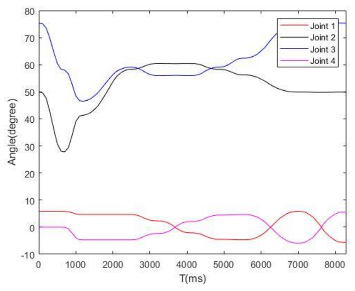


Figure 7.14: Four Joints angular displacement tracking for writing '2'

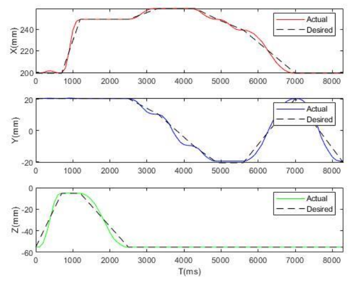


Figure 7.15: Desired and actual cartesian coordinates for writing '2'

It can be seen from those figures that while moving from one point to another point the actual path is following a cubic polynomial trajectory; however, the desired path is following a straight line trajectory. Finally, figure 7.11, figure 7.14, and figure 7.17 are depicting the changes in angular displacements of four joints (joint 1, 2, 3, and 4) when writing 1, 2, and 3 respectively. Comparing the figures 7.10, 7.11, and 7.12 for writing 1, figures 7.13, 7.14, and 7.15 for writing 2, and figures 7.16, 7.17, and 7.18 for writing 3, it can be seen that when there is any elevation along Z axis without changes in X, Y values, the joint 1 angles remain constant as there is no angular motion in joint 1 motor. When there are changes only in X without any changes in Y, and Z, joint 2 increases and joint 3 decreases proportionally and joint 1 remains constant. On the other hand,

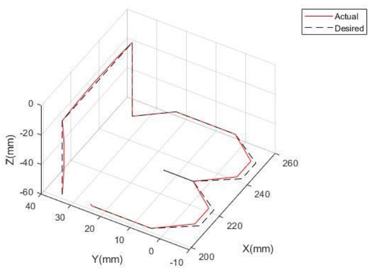


Figure 7.16: 3D view of the desired and actual trajectory for writing '3'

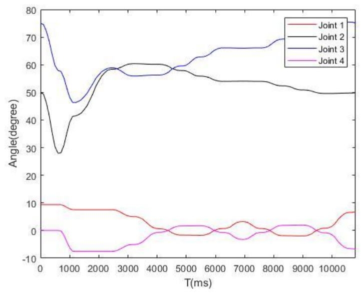


Figure 7.17: 4 Joints angular displacement tracking for writing '3'

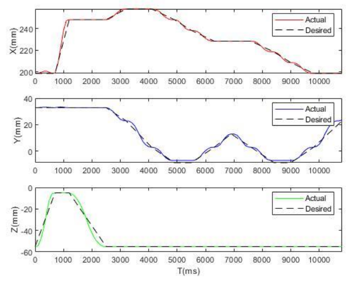


Figure 7.18: Desired and actual cartesian coordinates for writing '3'

when Y stays constant without changes in X, and Z, joint 2 and joint 3 remains constant, but joint 1 increases. When there are increases in both X and Y without any changes in Z, joint 2 increases and joint 3 decreases proportionally with joint 1 decreases. Similarly, when there are decreases in both X and Y without any changes in Z, joint 1 and joint 3 increases and joint 2 decreases.

Figure 7.19 shows the three dimensional (3D) view of trajectories for writing 'BIOROBOTICS LAB', where red lines are representing the elevation of Z axis, and the blue lines are showing the trajectories on a 2D (plane) surface. Figure 7.20 is showing the joint and cartesian coordinates of the writing pen tip for writing 'BIOROBOTICS LAB'. It can be seen from those graphs that while moving from one point to another point the actual path is following a cubic polynomial trajectory;

however, the desired path is following a straight line trajectory. Moreover, when there is any elevation along Z axis without changes in X, Y values, the joint 1 angles remain constant as there is no angular motion in joint 1 motor. When there are changes only in X without any changes in Y, and Z, joint 2 increases and joint 3 decreases proportionally and joint 1 remains constant. On the other hand, when Y stays constant without changes in X, and Z, joint 2 and joint 3 remains constant, but joint 1 increases. When there are increases in both X and Y without any changes in Z, joint 2 increases and joint 3 decreases proportionally with joint 1 decreases. Similarly, when there are decreases in both X and Y without any changes in Z, joint 1 and joint 3 increases and joint 2 decreases.

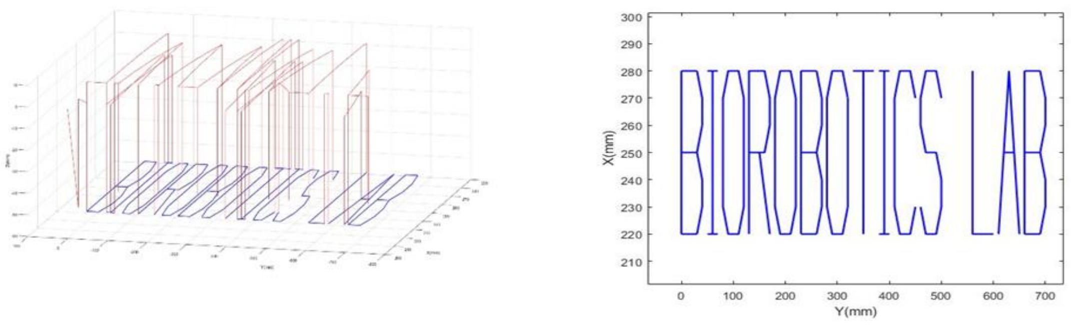
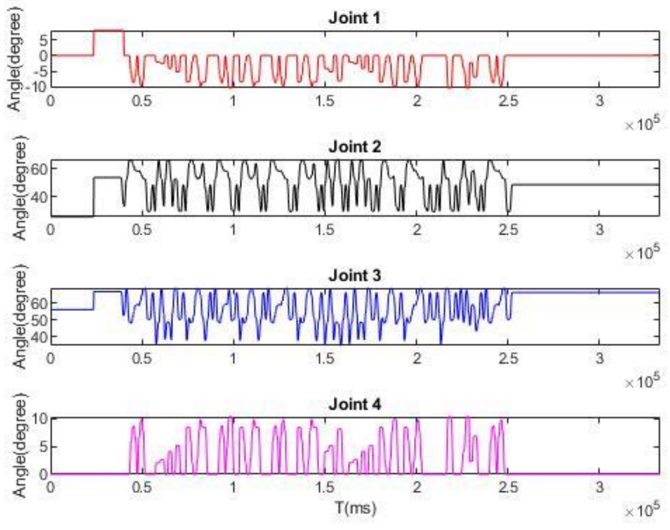


Figure 7.19: 3D view of the desired and actual trajectory for writing 'BIOROBOTICS LAB' and the output on the plane surface



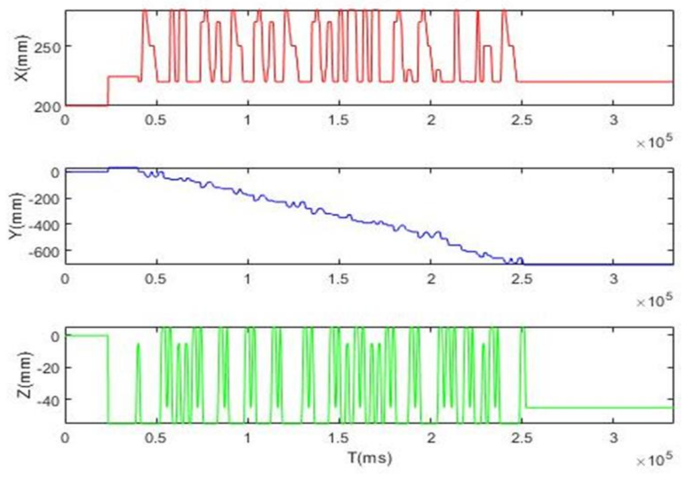


Figure 7.20: Joint coordinates and cartesian coordinates for writing 'BIOROBOTICS LAB'

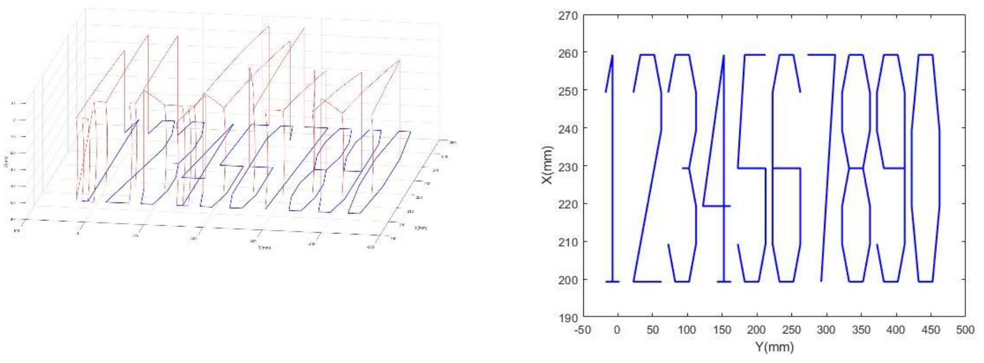
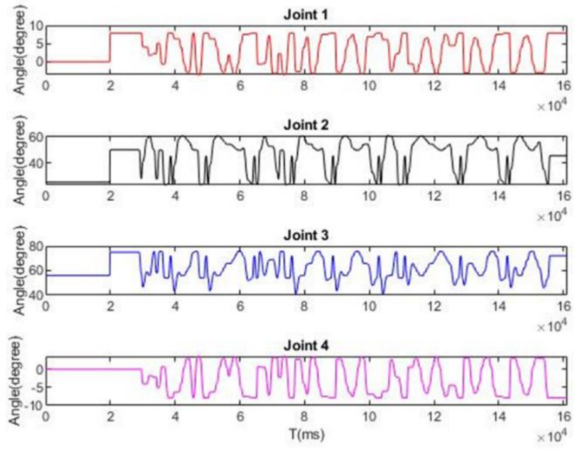


Figure 7.21: 3D view of the desired and actual trajectory for writing '1234567890' and the output on the plane surface



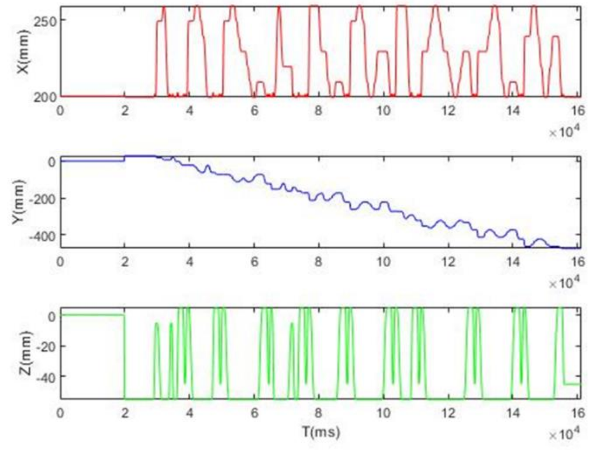


Figure 7.22: Joint coordinates and cartesian coordinates for writing '1234567890'

Figure 7.22 shows the three dimensional (3D) view of trajectories for writing '1234567890', where red lines are representing the elevation of Z axis, and the blue lines are showing the trajectories on a 2D (plane) surface. Figure 7.23 is showing the joint and cartesian coordinates of the writing pen tip for writing '1234567890'. It can be seen from those graphs that while moving from one point to another point the actual path is following a cubic polynomial trajectory; however, the desired path is following a straight line trajectory. Moreover, when there is any elevation along Z axis without changes in X, Y values, the joint 1 angles remain constant as there is no angular motion in joint 1 motor. When there

are changes only in X without any changes in Y, and Z, joint 2 increases and joint 3 decreases proportionally and joint 1 remains constant. On the other hand, when Y stays constant without changes in X, and Z, joint 2 and joint 3 remains constant, but joint 1 increases. When there are increases in both X and Y without any changes in Z, joint 2 increases and joint 3 decreases proportionally with joint 1 decreases. Similarly, when there are decreases in both X and Y without any changes in Z, joint 1 and joint 3 increases and joint 2 decreases.

7.2 Pick and Place Tasks Results

Figure 7.24, figure 7.27, and figure 7.30 show the three-dimensional (3D) view of trajectories for pick and place of three different weights of 184 gm, 282 gm, and 467 gm respectively, where red lines are representing the actual trajectory the Dobot followed and the black dashed lines are

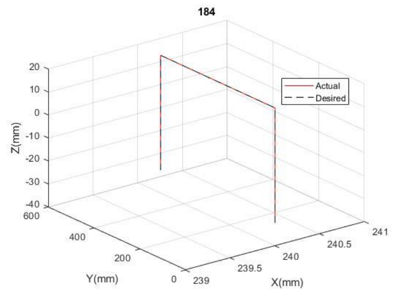


Figure 7.23: 3D view of the desired and actual trajectory for pick and place 184 gm weight

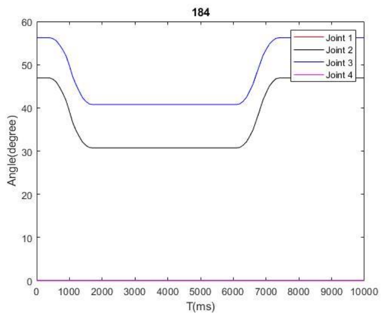


Figure 7.24: 4 Joints angular displacement tracking for pick and place 184 gm weight

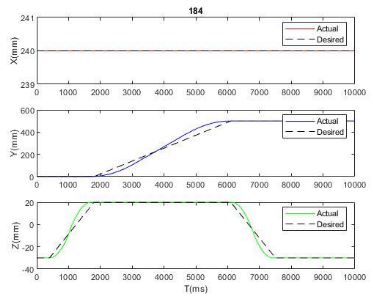


Figure 7.25: Desired and actual cartesian coordinates for pick and place 184 gm weight

showing the desired trajectories to perform pick and place (PP) tasks. Figure 7.26, figure 7.29, and figure 7.32 are showing the cartesian coordinates of the suction cup for performing PP for the weights 184 gm, 282 gm, and 467 gm respectively, where red, blue, and green lines are representing the actual trajectory of X, Y, and Z coordinates, and the black dashed lines are showing the desired trajectories. It can be

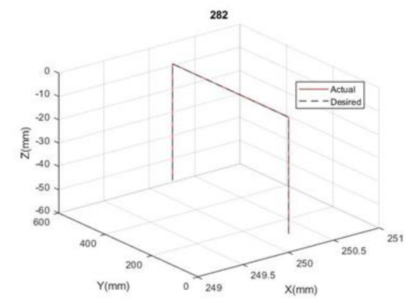


Figure 7.26: 3D view of the desired and actual trajectory for pick and place 282 gm weight

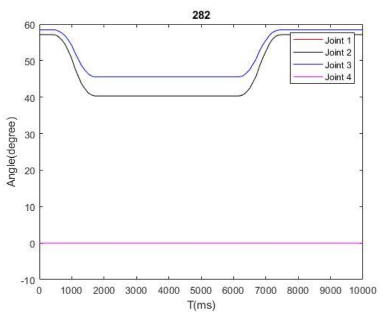


Figure 7.27: Four Joints angular displacement tracking for pick and place 282 gm weight

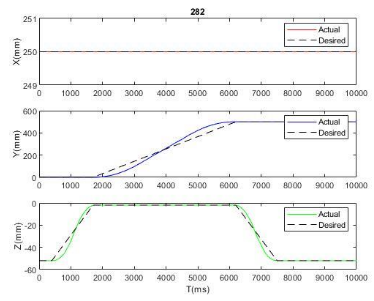


Figure 7.28: Desired and actual cartesian coordinates for pick and place 282 gm weight

seen from those figures that while moving from one point to another point the actual path is following a cubic polynomial trajectory; however, the desired path is following a straight-line trajectory. Finally, figure 7.25, 7.28, and 7.31 are depicting the changes in angular displacements of four joints (joint 1, 2, 3, and 4) when performing PP tasks for the weights 184 gm, 282 gm, and 467 gm respectively. Comparing the figures 7.24, 7.25, and 7.26 for 184 gm weight, figures 7.27,

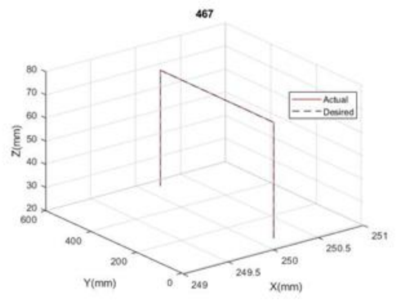


Figure 7.29: 3D view of the desired and actual trajectory for pick and place 467 gm weight

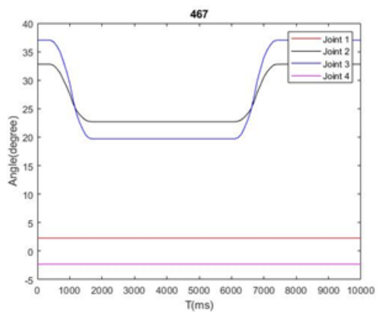


Figure 7.30: 4 Joints angular displacement tracking for pick and place 467 gm weight

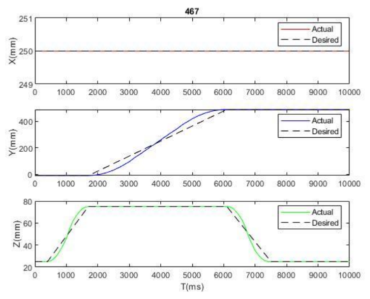


Figure 7.31: Desired and actual cartesian coordinates for pick and place 467 gm weight

7.28, and 7.29 for 282 gm weight, and figures 7.30, 7.31, and 7.32 for 467 gm weight, the actual trajectories follows the desired trajectories everywhere regardless the amount of weight.

7.3 Experiment Video Links

https://www.youtube.com/watch?v=0Ysfv12RWBQ

https://www.youtube.com/watch?v=6HjqYmynqAE

https://www.youtube.com/watch?v=fXjMpQuoUeE

https://www.youtube.com/watch?v=AZ_xn8mx48A

7.4 Discussions

The Dobot Magician itself has 500 gm payload capacity, and in this research several pick and place tasks for different weights under the 500 gm limit are carried out. The results lead us to conclude that the Dobot Magician follows the desired trajectories for all weights under the limit of its payload capacity and it can be utilized for multiple application, for instance, in our research teaching color to the children, stacking and sorting loads in industrial use, and integrating to work with voice commands for carrying objects to the users successfully implemented.

CHAPTER 8

CONCLUSION AND FUTURE WORKS

8.1 Conclusion

A commercially available four degree of freedom robotic manipulator, Dobot Magician, is used in this research to demonstrate the prospect of accommodating more functionality in a single robot. A motion sensing device, Kinect, is synchronized to work along with the Dobot utilizing its voice sensing ability to control the Dobot with voice-activated commands.

On the way to add more features in the Dobot, the forward kinematics of the Dobot is derived using modified Denavit-Hartenberg parameters; the inverse kinematics is solved using both algebraic and geometric approach; the Proportional-Integral-Derivative (PID) control, Computed Torque Control (CTC), and Cartesian Trajectory Tracking with Joint Based Control were studied.

A multifunctionality software interface for the Dobot is developed using C# programming language. Dobot API, Kinect V2 API, and Speech SDK are used for developing the software. Eventually, the Dobot has been turned out to an intelligent robotic manipulator will be more accessible to all ages. To accomplish all the aims of this research, several experiments have been performed using the Dobot. A novel approach of writing is applied while developing the code for writing alphabets, numbers, words, and sentences through which children, especially having a learning disability, will become more interested in learning alphabets, numbers, words, and sentence as well as playing with a robot. The old people, especially who has a disability, can adopt a Dobot for rest of their life as it can be controlled by voice command and can be another hand to carry stuffs to the users.

8.2 Future Works

The Dobot has the potential to be more productive with more features than before. In this research, only the speech recognition technology is blended to make it a voice-controlled robot. Future studies can also be expanded as follows:

- using motion detection technology of the Kinect sensor, the Dobot can be controlled through human body motion;
- using the audio source sensing technology of the Kinect sensor, the Dobot will be able to respond towards the source of audio;
- using the color sensing and coordinate mapping technology of the Kinect sensor, the Dobot will be able to detect objects;
- integrating other low-cost sensors, it can become more intelligent to implement in different applications.

BIBLIOGRAPHY

Beran, Tanya N., Alejandro Ramirez-Serrano, Roman Kuzyk, Meghann Fior, and Sarah Nugent. 2011. “Understanding How Children Understand Robots: Perceived Animism in Childrobot Interaction.” *International Journal of Human Computer Studies* 69 (7–8): 539–50. https://doi.org/10.1016/j.ijhcs.2011.04.003.

Broekens, Joost, Joost Broekens, Marcel Heerink, and Henk Rosendal. 2009. “Assistive Social Robots in Elderly Care: A Review.” *GERONTECHNOLOGY*, 94--103. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.714.6939.

Craig, John J. 2014. *Introduction to Robotics Mechanics and Control*. Pearson Education Limited. https://books.google.com/books/about/Introduction_to_Robotics_Pearson_New_Int.html?id =ZTqpBwAAQBAJ&source=kp_book_description.

Denavit, Jacques. 1955. “A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices.” *Trans. of the ASME. Journal of Applied Mechanics* 22: 215–21. http://ci.nii.ac.jp/naid/10008019314/en/.

Deutsch, Inbal, Hadas Erel, Michal Paz, Guy Hoffman, and Oren Zuckerman. 2019. “Home Robotic Devices for Older Adults: Opportunities and Concerns.” *Computers in Human Behavior*. https://doi.org/10.1016/j.chb.2019.04.002.

Foresi, Gabriele, Alessandro Freddi, Andrea Monteriu, Davide Ortenzi, and Daniele Proietti Pagnotta. 2018. “Improving Mobility and Autonomy of Disabled Users via Cooperation of Assistive Robots.” *2018 IEEE International Conference on Consumer Electronics, ICCE 2018* 2018-Janua: 1–2. https://doi.org/10.1109/ICCE.2018.8326291.

Grossi, Giuliano, Raffaella Lanzarotti, Paolo Napoletano, Nicoletta Noceti, and Francesca Odone. 2019. “Positive Technology for Elderly Well-Being: A Review.” *Pattern Recognition Letters*, no. xxxx: 1–10. https://doi.org/10.1016/j.patrec.2019.03.016.

Hart, Johnson M. 2005. *Windows System Programming*. Addison-Wesley.

Hartenberg, Richard Scheunemann., and Jacques Denavit. 1964. *Kinematic Synthesis of Linkages*. New York: McGraw-Hill.

Highfield, Kate. 2010. “Australian Primary Mathematics Classroom : APMC.” *Australian Primary Mathematics Classroom* 15 (2): 22–27. https://researchers.mq.edu.au/en/publications/robotic-toys-as-a-catalyst-for-mathematical-problem-solving.

Islam, Md Rasedul, Christopher Spiewak, Mohammad Habibur Rahman, and Raouf Fareh. 2017. “A Brief Review on Robotic Exoskeletons for Upper Extremity Rehabilitation to Find the Gap between Research Porotype and Commercial Type.” *Advances in Robotics & Automation* 06 (03): 1–12. https://doi.org/10.4172/2168-9695.1000177.

McComb, Gordon. 2008. “Getting Kids into Robotics.” *Servo Magazine*, October 2008. https://doi.org/10.2008.

Monperrus, Martin, Michael Eichberg, Elif Tekes, and Mira Mezini. 2012. “What Should Developers Be Aware of? An Empirical Study on the Directives of API Documentation.” *Empirical Software Engineering* 17 (6): 703–37. https://doi.org/10.1007/s10664-011-9186-4.

Mousa, Ahmed Amin, Tamer M Ismail, and M Abd El Salam. 2017. “A Robotic Cube to Preschool Children for Acquiring the Mathematical and Colours Concepts.” *International Journal of Educational and Pedagogical Sciences* 11 (7): 1516–19.

Rahman, Mohammad Habibur. 2012. “Development of an Exoskeleton Robot for Upper-Limb Rehabilitation.” *ProQuest Dissertations and Theses*. Ann Arbor: Ecole de Technologie Superieure (Canada). https://ezproxy.lib.uwm.edu/login?url=https://search.proquest.com/docview/1268161964?accountid=15078.

Riener, Robert, Marco Guidali, Urs Keller, Alexander Duschau-Wicke, Verena Klamroth, and Tobias Nef. 2011. “Transferring ARMin to the Clinics and Industry.” *Topics in Spinal Cord Injury Rehabilitation* 17 (1): 54–59. https://doi.org/10.1310/sci1701-54.

Sarkar, Swati, and Amarendra Kumar Das. 2019. “Modular Assistive Devices for Elderly People to Overcome Age-Related Problems BT - Research into Design for a Connected World.” In , edited by Amaresh Chakrabarti, 463–74. Singapore: Springer Singapore.

Sharp, John. 2015. *Microsoft Visual C# Step by Step, 8th Edition*. https://www.microsoftpressstore.com/store/microsoft-visual-c-sharp-step-by-step-9781509301089.

Shenzhen Yuejiang Technology Co. Ltd. 2018a. “Dobot Magician API Description.”

———. 2018b. “Dobot Magician Communication Protocol.”

———. 2019. “Dobot Magician Demo Description.”

Wei, Chun Wang, I. Chun Hung, Ling Lee, and Nian Shing Chen. 2011. “A Joyful Classroom Learning System with Robot Learning Companion for Children to Learn Mathematics Multiplication.” *Turkish Online Journal of Educational Technology* 10 (2): 11–23.