

The 4R Robot Manipulator's Kinematic Analysis using Roboanalyzer and CProg

Sridhar Gowda, Bhaskar.B.Katti, Manjunath.S.H

Asst. Prof, Asst. Prof, Asst. Prof

sridharmpheroor@gmail.com, bbkatti@pdit.ac.in, manju.hubballi1988@gmail.com

Department of Mechanical, Proudhadivaraya Institute of Technology, Abheraj Baldota Rd,
Indiranagar, Hosapete, Karnataka-583225

ABSTRACT

A manipulator's kinematic analysis involves the connections between the linkages' locations, velocities, and accelerations. Analysing the robot's kinematics is a time-consuming task. Direct kinematics, often known as forward kinematics, and inverse kinematics are the two main categories of kinematics. Given the length and angle of each link and joint, forward kinematics allows us to determine the location of every point in the robot's work volume. Given the link length and the point's location in the work volume, inverse kinematics asks us to determine the joint's angle. Using Roboanalyzer and CProg, we analysed the 4R robot's kinematics for this project. Then, we created a program for the IGUS robot to choose and position operations. A comparison is performed between the theoretical calculations and the roboanalyzer findings for a single spot.

1. INTRODUCTION

Robot is a machine that collects the information about the environment using some sensors and makes a decision automatically. People prefer it to use different field, such as industry, some dangerous jobs including radioactive effects. In this point, robots are regarded as a server. They can be managed easily and provides many advantages. Robot kinematics is the study of the motion (kinematics) of robots. In a kinematic analysis the position, velocity and acceleration of all the links are calculated without considering the forces that cause this motion. The relationship between motion, and the associated forces and torques is studied in robot dynamics.

In the kinematic analysis of manipulator position, there are two separate problems to solve: direct kinematics, and inverse kinematics. Direct kinematics involves solving the forward transformation equation to find the location of the hand in terms of the angles and displacements between the links. Inverse kinematics involves solving the inverse transformation equation to find the relationships between the links of the

manipulator from the location of the hand in space. In the next chapters, inverse and forward kinematic will be represented in detail. A robot arm is known manipulator. It is composed of a set of nonseparated in space by the arm links. The joints are where the motion in the arm occurs. In basic, a robot arm consists of the parts: base, joints, links, and a gripper. The base is the basic part over the arm, It may be fix or active. The joint is flexible and joins two separated links. The link is fix and supports the gripper. The last part is a gripper. The gripper is used to hold and move the objects. Figure-1 shows these parts. In the report, the manipulator types are defined in details.

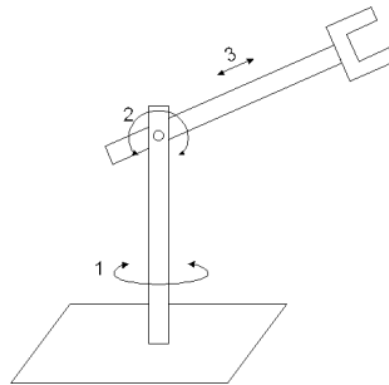


Figure 1.1 Basic robot arm.

Base, 2. joint, 3. link and the last part, gripper. Homogeneous transformation is used to solve kinematic problems. This transformation specifies the location (position and orientation) of the hand in space with respect to the base of the robot, but it does not tell us which configuration of the arm is required to achieve this location. It is often possible to achieve the same hand position with many arm configurations. In the next chapters, this transformation is explained in details with simple examples.

1.1 FEASIBILITY OF THE PROJECT

During the development of the project, I have been researched the feasibility in the different field, especially software and hardware. The feasibility study is below in details.

1.1.1 SOFTWARE FEASIBILITY:

In the software feasibility, I tried to choose the best program that solves my needs. I preferred to use JAVA programming language. Because it is known that it run over any operating system with java virtual machine. I created the user interface by using the java-swings. On the other side, I write the hardware codes by using PIC C program language. MICRO C can be used, too. PICFLASH provides us to load the

program onto development kit. You can use the PROTEUS to design your chip devices.

1.2 HARDWARE FEASIBILITY:

On the hardware side, we should have a development kit with serial communication port to send data and USB port to program the chip. In addition to this, we can use servos to rotate robot arm. The servo rotates different angles. Sometimes it is between -90 and 90 degrees. Some of them can rotate about 90 to 180 degrees. The last one, we can use a sheet of plastic to cut the links. You can see it following picture.

1.3 TECHNICAL FEASIBILITY:

Minimum requirements for the project are given in the table 2.1

Processor	600 MHz processor Recommended: 1 gigahertz (GHz) processor
RAM	512 MB Recommended: 1.5 GB
Available Hard Disk Space	1 GB of available space required on system drive
Video	800 X 600, 256 colors. Recommended: 1024 X 768, High Color 16-bit

Table 1.1 Minimum requirements for the project.

1.4 ECONOMICAL FEASIBILITY:

Economic cost of the project can be separated in two groups. First of them is software cost. Another one is hardware cost. Table 2.2 shows software cost and hardware cost.

1.5 LEGAL FEASIBILITY:

Software needs is usually free. If we have licenses for software, there is no problem. Moreover the part of other articles and researches are referenced at the end of the project.

1.6 BASIC MANIPULATOR GEOMETRIES

In this section, I look at some basic arm geometries. As I said before, a robot arm or manipulator is composed of a set of joints, links, grippers and base part.

The joints are where the motion in the arms occurs, while the links are of fixed construction. Thus the links maintain a fixed relationship between the joints. The joints may be actuated by motors or hydraulic actuators. There are two sorts of robot joints, involving two sorts of motion. A revolute joint is one that allows rotary motion about an axis of rotation. An example is the human elbow. A prismatic joint is one that allows extensions or telescopic motion. An example is a telescoping automobile antenna. There are some types of manipulator kinematic below.

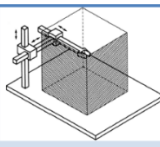
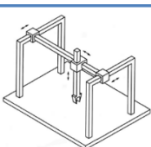
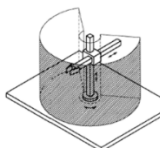
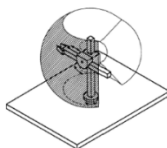
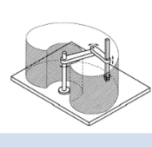
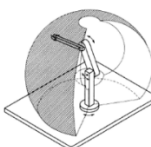
Name	Figure	Name	Figure
Cartesian		Gantry	
Cylindrical		Sphre	
Scara		Anthropomorhic	

Table 1.3. Manipulator kinematic

1.7 OPEN CHAIN MANIPULATOR KINEMATICS:

In this types of the arm, mechanics of a manipulator can be represented as a kinematic chain of rigid bodies (links) connected by revolute or prismatic joints. One end of the chain is constrained to a base, while an end effector is mounted to the other end of the chain. In the open chain robot arm, The resulting motion is obtained by composition of the elementary motions of each link with respect to the previous one. The joints must be controlled individually. Closed Chain Manipulator Kinematics Closed Chain Manipulator is much more difficult than open chain manipulator. Even analysis has to take into account statics, constraints from other links, etc. Parallel robot is a closed chain. For this type of robots, the best example is the Stewart platform. Figure-3.2 shows Stewart platform.

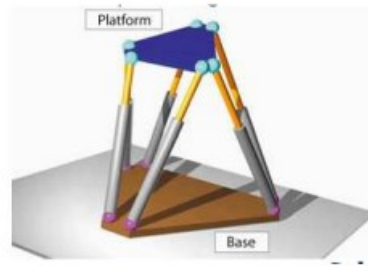


Figure-1.6 Stewart platform

1.8 IGUS ROBOT

IGUS Robot Control enables simple and intuitive robot programming and control and an easy entry into automation. Due to the modular design, different robot kinematics can be controlled, for example, delta robots, linear robots and multi-axis joint robots. The software can be used to simulate the individual movements of the robot on the 3D interface - the robot does not need to be connected for this step.

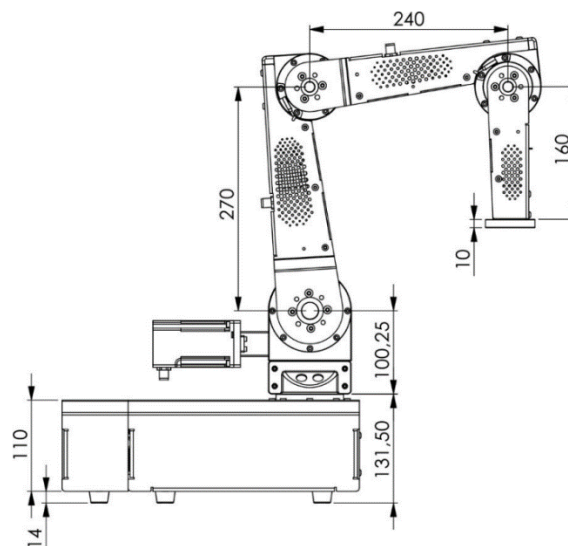


Figure-7.1 IGUS ROBOT

Simple connection between robot and controller

Digital Twin enables direct execution and checking of the motion sequences Camera interface configuration directly via control system and software possible Programming of drylin® delta robots and drylin® linear robots, as well as robolink® DP articulated arm robots Free software enables risk-free testing

SYSTEM REQUIREMENTS:

PC with Windows 10 operating system

Free USB 2.0 port, Ethernet port,

500 storage space

2.KINEMATIC ANALYSIS OF 4 R ROBO MANIPULATOR

2.1 HOMOGENEOUS TRANSFORMATIONS

Homogeneous transformation is used to calculate the new coordinate values for a robot part. Transformation matrix must be in square form. Figure-4.1 shows the transformation matrix.

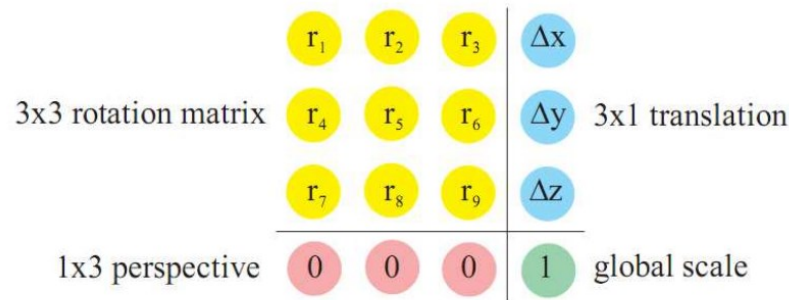


Figure-2.1 Homogeneous Transformation matrix.

3x3 rotation matrix may change with respect to rotation value. 3x1 translation matrix shows the changing value between the coordinate systems. Global scale value is fix and 1. Also 1x3 perspective matrixes is fix.

2.3 FORWARD KINEMATICS

2.3.1 DENAVIT &HARTENBERG (D-H) NOTATION:

The definition of a manipulator with four joint-link parameters for each link and a systematic procedure for assigning right-handed orthonormal coordinate frames, one to each link in an open kinematic chain was proposed by Denavit and Hartenberg (1955) and is known as Denavit-Hartenberg (DH) notation. This notation is presented in this section and followed throughout the text. A frame $\{i\}$ is rigidly attached to distal end of link i and it moves with link i . An n -DOF manipulator will have $(n + 1)$ frames with the frame $\{0\}$ or base frame acting as the reference inertial frame and frame $\{n\}$ being the "tool frame".

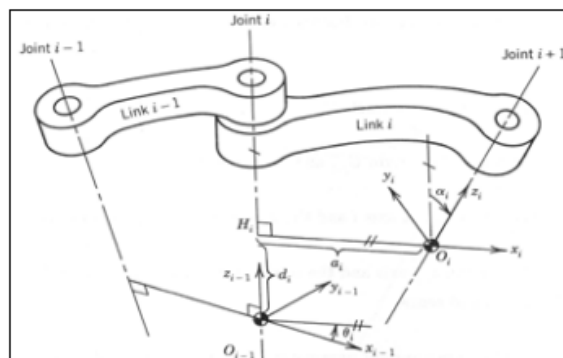


FIGURE 2.2 DH NOTATIONS

Figure 2.2 shows a pair of adjacent links, link $(i-1)$ and link i , their associated joints, joints $(i-1)$, i and $(i+1)$, and axes $(i-2)$, $(i-1)$, and i , respectively. Line AB, in the figure, is the common normal to $(i-2)$ - and $(i-1)$ -axes and line CD is the common normal to $(i-1)$ - and i -axes. A frame $\{i\}$ is assigned to link i as follows: (i) The Z-axis is aligned with axis i , its direction being arbitrary. The choice of direction defines the positive sense of joint variable Θ_i . (ii) The x_i-axis is perpendicular to axis Z_{i-1} and Z_i and points away from axis Z_{i-1} 'that is, X_i axis is directed along the common normal CD. (iii) The origin of the i th coordinate frame, frame $\{i\}$ is located at the intersection of axis of joint $(i+1)$, that is, axis i , and the common normal between axes $(i-1)$ and i (common normal is CD), as shown in the figure. (iv) Finally, y_i-axis completes the right-hand orthonormal coordinate frame $\{i\}$. Note that the frame $\{i\}$ for link i is at the distal end of link i and moves with the link. With respect to frame $\{i-1\}$ and frame $\{i\}$, the four DH-parameters – two link parameters (a_i , α_i) and two joint parameters (d_i , Θ_i) - are defined as: (a) Link Length (a_i) - distance measured along X_{i-1} -axis from the point of intersection of X_i -axis with Z_{i-1} -axis (point C) to the origin of frame $\{i\}$, that is distance CD. (b) Link twist (α_i) - angle between Z_{i-1} and Z_i -axes measured about X_i -axis in the righthand sense. (c) Joint distance (d_i) - distance measured along Z_{i-1} -axis from the origin of frame $\{i-1\}$ (point B) to the intersection of x_i -axis with Z_{i-1} -axis (point C), that is distance BC. (d) Joint angle (Θ_i) - angle between X_{i-1} and X_i axes measured about the Z_{i-1} -axis in the right-hand sense. The convention outlined above does not result in a unique attachment 'of frames to links because alternative choices are available. For example, joint axis i has two choices of direction to point z_i-axis, one pointing upward and other pointing downward. To minimize such options and get a consistent set of frames an algorithm is presented below to assign frames to all links of a manipulator

2.4 ARTICULATED ARM KINEMATIC MODEL

ARTICULATED ROBOTIC ARM An articulated robot is a robot which is fitted with rotary joints. Rotary joints allow a full range of motion, as they rotate through multiple planes, and they increase the capabilities of the robot considerably. An articulated robot can have one or more rotary joints, and other types of joints may be used as well, depending on the design of the robot and its intended function. With rotary joints, a robot can engage in very precise movements. Articulated robots commonly show up on manufacturing lines, where they utilize their flexibility to bend in a variety of directions. Multiple arms can be used for greater control or to conduct multiple tasks at once, for example, and rotary joints allow robots to do things like turning back and forth between different work areas. These robots can also be seen at work in labs and in numerous other settings. Researchers developing robots often work with articulated robots when they want to engage in activities like teaching robots to walk and developing robotic arms. The joints in the robot can be programmed to interact with each

other in addition to activating independently, allowing the robot to have an even higher degree of control. Many next generation robots are articulated because this allows for a high level of functionality.

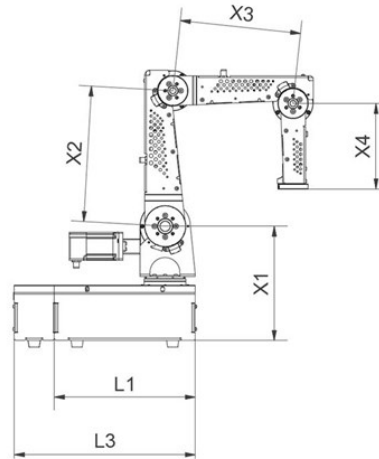


FIGURE 2.3

Above figure shows the schematic diagram of 4R robot manipulator now let us calculate forward kinematics of robotic arm by using DH notations.

SL.NO	Joint angle(Θ) (deg)	Joint offset d_i (mm)	Link length a_i (mm)	Twisting Angle α_i (deg)
1.	Θ_1	d_1	a_1	0
2.	Θ_2	0	a_2	π
3.	Θ_3	0	0	0
4.	Θ_4	0	0	0

Table 2.2 DH PARAMETERS TABLE

Sl.no	Joint type	Joint angle(Θ_i) (deg)	Link offset d_i (mm)	Link length a_i (mm)	Twist angle α_i (deg)	Initial value (Jv) deg or m	Final value (jv)deg or m
1.	Revolute	Θ_1	0.105	0	90	0	0
2.	Revolute	Θ_2	0	0.265	0	30	90
3.	Revolute	Θ_3	0	0.235	0	-60	-90
4.	Revolute	Θ_4	0	0.095	0	-60	90

Table 2.2 DH Parameters Table with Link Values

By Using DH Notation Matrixes We Can Solve Forward Kinematics of 4R Robot Manipulator.

2. ROBOT KINEMATICS USING ROBOANALYZER SOFTWARE

RoboAnalyzer is a 3D Model Based Robotics Learning Software. It has been developed to help the faculty to teach and students to learn the concepts of Robotics. It also acts as a supporting material for the contents on various robotics topics in text book entitled “Introduction to Robotics”, S. K. Saha, 2008 [1].

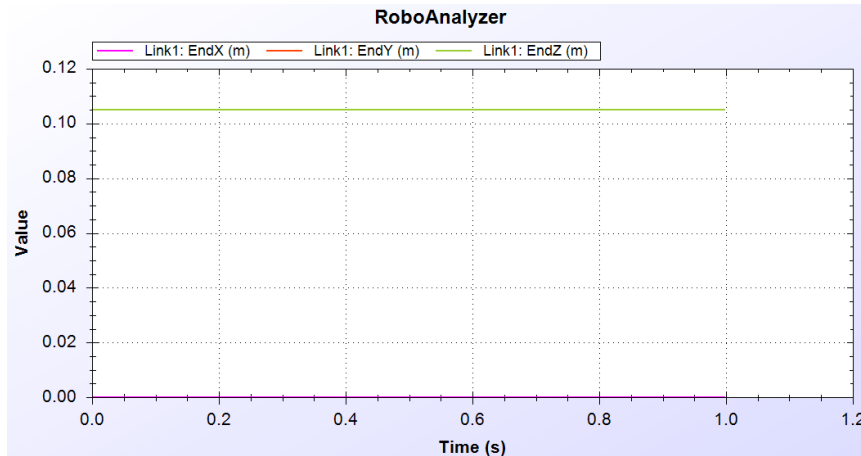
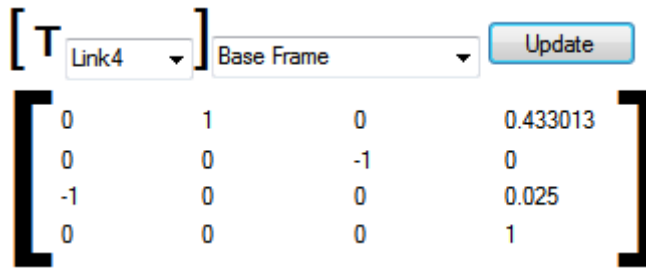
FORWARD KINEMATICS

In the forward or direct kinematics, the joint positions, i.e. the angles of the revolute joints and the displacements of the prismatic joints, are prescribed. The task is to find the end-effector’s configuration/transformation consisting of its position and orientation. More details can be found in Chapter 6 of [1]. After selecting a robot and redefining DH parameters as explained in Section 2.1, forward kinematics (FKin) is performed which updates the 3D model.

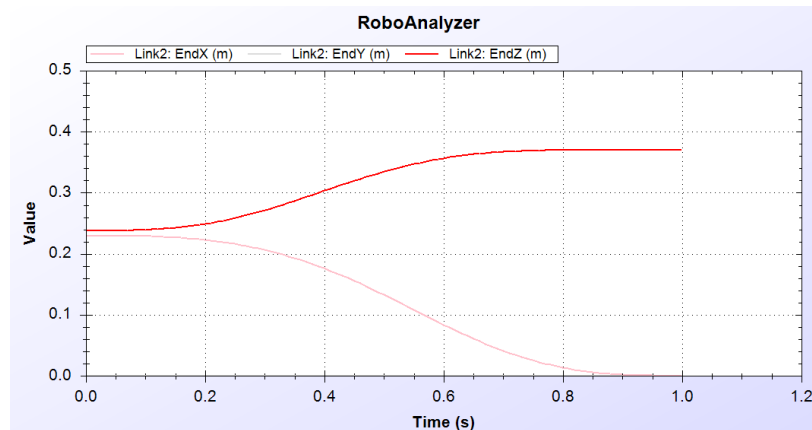
To perform animation of the robot motion between two sets of initial and final values of joint variables, the following are the steps as shown in Figures 10 and 11. The trajectory of joint values, joint velocities and joint accelerations follow Cycloidal trajectory mentioned in Chapter 8 of [1]. The trajectory can be changed as explained in Section 8.

1. Set the initial and final values of joint variables
2. Set Time Duration and Number of Steps
3. Click on FKin button
4. Click on Play button to see the animation
5. The end-effector trace can be viewed

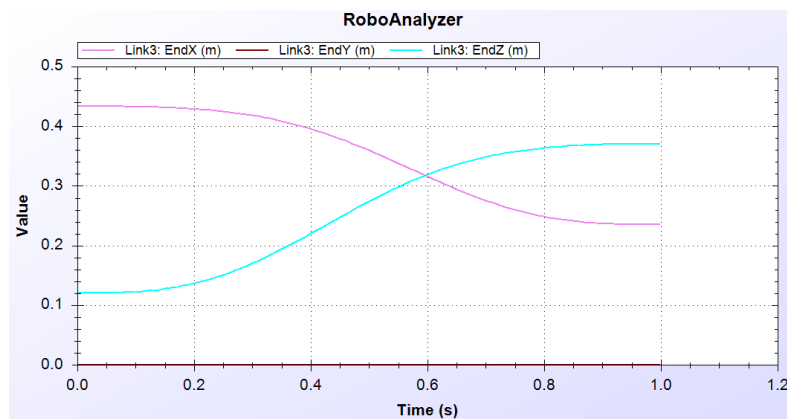
Now let us make use the roboanalyzer for making 4R manipulator IGUS ROBOT and by taking the link lengths of robot and perform forward kinematics for pick and place operation the results of kinematics are shown below which are matching with theoretical solutions.



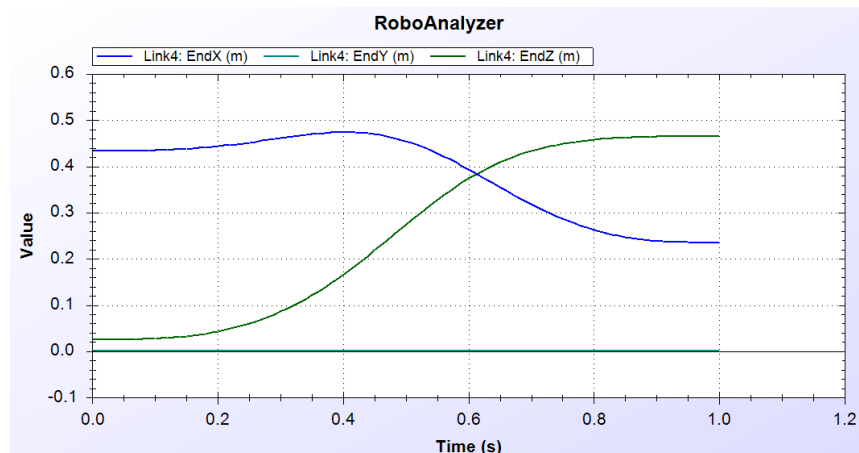
Graph 1 tracing final position of link 1



Graph 2 tracing final position of link 2



Graph 3 tracing final position of link 3



Graph 4 tracing final position of link 4

4. CPROG interface for IGUS ROBOTS

CPRog is a control and programming environment for robots. The 3D user interface allows a quick start into programming. Due to the modular design different kinematics and motor drivers can be controlled.

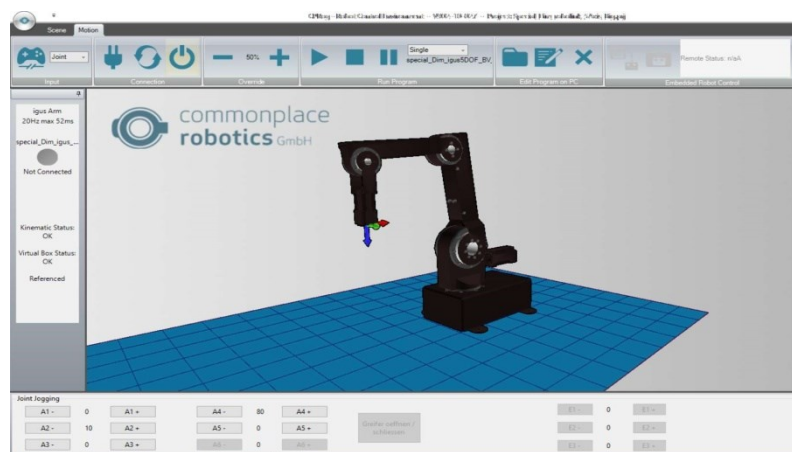


Figure-8.1 UI of CPROG Software

Digital Inputs and Outputs

The simplest connection, e.g. to a PLC, is possible via digital inputs and outputs.

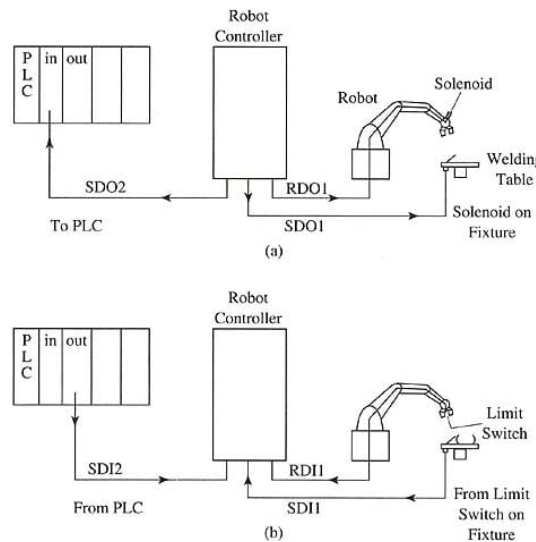
Each robotlink

controller is supplied with a DIO module. One module provides 7 inputs and 7

outputs. A total

of 3 modules can be used.

The outputs are switched by IC relays, capable of up to 500 mA. This value must not be exceeded during the switching process (e.g. by charging currents of capacitors).



Start Robot Programs

The robot program must be loaded and started.

1. Load the program:
Press the folder icon in the "Edit Program" area of the "Motion" tab and select a program, e.g. igus5DOF_TestMotion.Xml
2. Adjust the override:
Before starting a new program, set the override to e.g. 20%. Be particularly attentive during the first complete program run.
3. Start the program:
Press the playback icon in the Run Program area of the Motion tab.
4. Stop or pause the program:
After pressing the pause symbol, the robot continues with the program by pressing the playback symbol again.
After pressing the stop symbol, the program starts with the first command when the playback is pressed again.

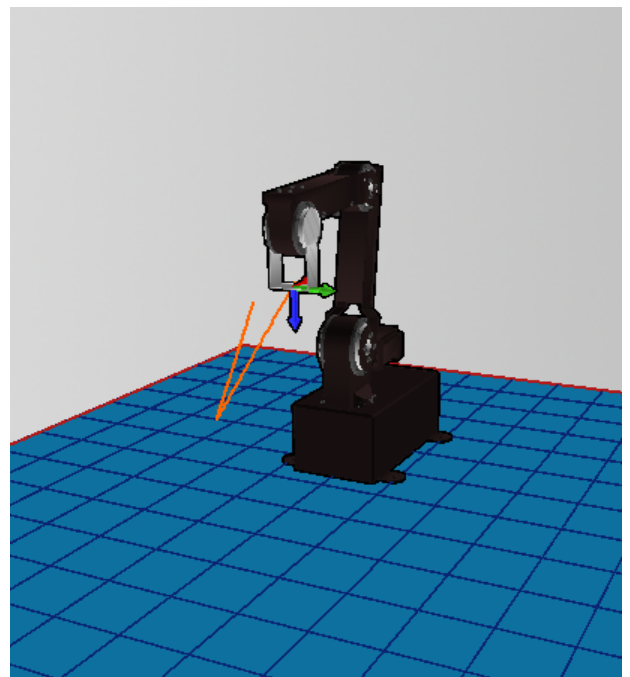
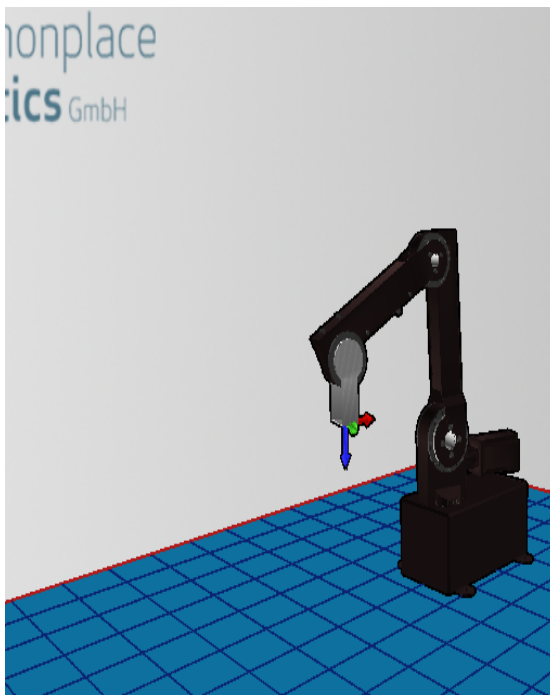
Move the Robot with Joypad and Buttons

The robot can be moved (or "jogged") manually while no program is running. The main controls are the Joypad Connect Panel, the Motion Type Selection List, and the Override. By pressing the joystick button, CPRog connects to a joypad. If the connection was successful, a green OK sign is displayed on the joystick button. The

device must be of the "Joystick" or "Gamepad" type. You can find further information on establishing a connection in the protocol window.

The "Joint" mode allows the individual robot axes to be moved from A1 to A6, if available. In Cart Base mode, the robot moves in straight lines along the X, Y and Z axes of the base coordinate system. In Cart Tool mode, the robot moves according to the current tool coordinate system.

C prog program for making pick and place operation



Program

```
<?xml version="1.0" encoding="utf-8"?>
<!-- values in mm and degree -->
<Program>
  <Header RobotName="igus Arm" RobotType="igus_4DOF_SV" GripperType=""
Software="CPRog V902-10-033"/>
  <Joint Nr="0" a1="0" a2="10" a3="0" a4="80" a5="0" a6="0" e1="0" e2="0" e3="0"
velPercent="50" acc="40" smooth="20" AbortCondition="false" Descr="" />
  <Store Nr="1" Type="position" Name="p1" Value="current" Descr="" />
  <MathOperator Nr="2" Function="Set" FirstOperand="p1.x" SecondOperand="100"
Descr="" />
  <MathOperator Nr="3" Function="Set" FirstOperand="p1.y" SecondOperand="-200"
Descr="" />
</Program>
```

```

<MathOperator Nr="4" Function="Set" FirstOperand="p1.z" SecondOperand="85" Descr=""
/>
<Store Nr="5" Type="position" Name="p2" Value="current" Descr="" />
<MathOperator Nr="6" Function="Set" FirstOperand="p2.x" SecondOperand="200"
Descr="" />
<MathOperator Nr="7" Function="Set" FirstOperand="p2.y" SecondOperand="20" Descr=""
/>
<MathOperator Nr="8" Function="Set" FirstOperand="p2.z" SecondOperand="370"
Descr="" />
<Store Nr="9" Type="position" Name="p3" Value="current" Descr="" />
<MathOperator Nr="10" Function="Set" FirstOperand="p3.x" SecondOperand="200"
Descr="" />
<MathOperator Nr="11" Function="Set" FirstOperand="p3.y" SecondOperand="-60"
Descr="" />
<MathOperator Nr="12" Function="Set" FirstOperand="p3.z" SecondOperand="85"
Descr="" />
<Store Nr="13" Type="position" Name="p4" Value="current" Descr="" />
<MathOperator Nr="14" Function="Set" FirstOperand="p4.x" SecondOperand="230"
Descr="" />
<MathOperator Nr="15" Function="Set" FirstOperand="p4.y" SecondOperand="-60"
Descr="" />
<MathOperator Nr="16" Function="Set" FirstOperand="p4.z" SecondOperand="85"
Descr="" />
<Store Nr="17" Type="position" Name="p5" Value="current" Descr="" />
<MathOperator Nr="18" Function="Set" FirstOperand="p5.x" SecondOperand="260"
Descr="" />
<MathOperator Nr="19" Function="Set" FirstOperand="p5.y" SecondOperand="-60"
Descr="" />
<MathOperator Nr="20" Function="Set" FirstOperand="p5.z" SecondOperand="85"
Descr="" />
<Store Nr="21" Type="position" Name="p6" Value="current" Descr="" />
<MathOperator Nr="22" Function="Set" FirstOperand="p6.x" SecondOperand="200"
Descr="" />
<MathOperator Nr="23" Function="Set" FirstOperand="p6.y" SecondOperand="60"
Descr="" />
<MathOperator Nr="24" Function="Set" FirstOperand="p6.z" SecondOperand="85"
Descr="" />
<Store Nr="25" Type="position" Name="p7" Value="current" Descr="" />
<MathOperator Nr="26" Function="Set" FirstOperand="p7.x" SecondOperand="230"
Descr="" />

```

```

<MathOperator Nr="27" Function="Set" FirstOperand="p7.y" SecondOperand="60"
Descr="" />
<MathOperator Nr="28" Function="Set" FirstOperand="p7.z" SecondOperand="85"
Descr="" />
<Store Nr="29" Type="position" Name="p8" Value="current" Descr="" />
<MathOperator Nr="30" Function="Set" FirstOperand="p8.x" SecondOperand="260"
Descr="" />
<MathOperator Nr="31" Function="Set" FirstOperand="p8.y" SecondOperand="60"
Descr="" />
<MathOperator Nr="32" Function="Set" FirstOperand="p8.z" SecondOperand="85"
Descr="" />
<Store Nr="33" Type="position" Name="p9" Value="current" Descr="" />
<MathOperator Nr="34" Function="Set" FirstOperand="p9.x" SecondOperand="200"
Descr="" />
<MathOperator Nr="35" Function="Set" FirstOperand="p9.y" SecondOperand="90"
Descr="" />
<MathOperator Nr="36" Function="Set" FirstOperand="p9.z" SecondOperand="85"
Descr="" />
<Store Nr="37" Type="position" Name="p10" Value="current" Descr="" />
<MathOperator Nr="38" Function="Set" FirstOperand="p10.x" SecondOperand="230"
Descr="" />
<MathOperator Nr="39" Function="Set" FirstOperand="p10.y" SecondOperand="90"
Descr="" />
<MathOperator Nr="40" Function="Set" FirstOperand="p10.z" SecondOperand="85"
Descr="" />
<Store Nr="41" Type="position" Name="p11" Value="current" Descr="" />
<MathOperator Nr="42" Function="Set" FirstOperand="p11.x" SecondOperand="260"
Descr="" />
<MathOperator Nr="43" Function="Set" FirstOperand="p11.y" SecondOperand="90"
Descr="" />
<MathOperator Nr="44" Function="Set" FirstOperand="p11.z" SecondOperand="85"
Descr="" />
<Linear Nr="45" Source="Variable" VariableName="p1" vel="100" acc="40" smooth="20"
AbortCondition="false" Descr="" />
<Wait Nr="46" Type="Time" Seconds="3" Descr="" />
<Output Nr="47" Channel="DOut25" State="True" Descr="" />
<Linear Nr="48" Source="Variable" VariableName="p2" vel="100" acc="40" smooth="20"
AbortCondition="false" Descr="" />
<Linear Nr="49" Source="Variable" VariableName="p3" vel="100" acc="40" smooth="20"
AbortCondition="false" Descr="" />
<Wait Nr="50" Type="Time" Seconds="3" Descr="" />

```

```
<Output Nr="51" Channel="DOut25" State="False" Descr="" />  
<Linear Nr="52" Source="Variable" VariableName="p2" vel="100" acc="40" smooth="20"  
AbortCondition="false" Descr="" />  
<Linear Nr="53" Source="Variable" VariableName="p1" vel="100" acc="40" smooth="20"  
AbortCondition="false" Descr="" />  
<Wait Nr="54" Type="Time" Seconds="3" Descr="" />  
<Output Nr="55" Channel="DOut25" State="True" Descr="" />  
<Linear Nr="56" Source="Variable" VariableName="p2" vel="100" acc="40"  
<Output Nr="99" Channel="DOut25" State="False" Descr="" />  
<Linear Nr="100" Source="Variable" VariableName="p2" vel="100" acc="40" smooth="20"  
AbortCondition="false" Descr="" />
```

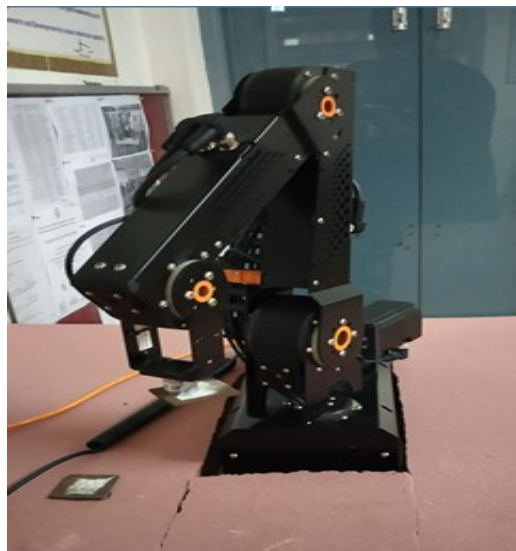


Figure pick and placing operation by IGUS robot

CONCLUSION

With IGUS Robot Control, controlling and programming robots is straightforward and intuitive, and getting started with automation is a breeze. This project's modular architecture allows for the control of a variety of robot kinematics, including linear, delta, and multi-axis joint robots, all of which have four degrees of freedom. manipulator for the IGUS robot, When the forward kinematics of the manipulator are computed using the D-H approach and compared with the findings produced in the roboanalyzer program, no errors are found. A program is created and tested on IGUS robots to execute pick-and-place operations using the cprog software.

Reference

1. Nordkvist N, Sanyal AK (2010) A Lie group variational integrator for rigid body motion in SE (3) with applications to underwater vehicle dynamics. In: The 49th IEEE conference on decision and control (CDC), Atlanta, pp 5414–5419
2. Park FC, Bobrow JE (1994) A recursive algorithm for robot dynamics using Lie groups. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), San Diego, pp 1535–1540
3. Gu YL (1988) Analysis of orientation representations by Lie algebra in robotics. In Proceedings of the IEEE international conference on robotics and automation (ICRA), Philadelphia, pp 874–879
4. Selig JM (2004) Lie groups and Lie algebras in robotics. Computational non-commutative algebra and applications. Springer, Netherlands, pp 101–125
5. Rico JM, Gallardo J, Ravani B (2003) Lie algebra and the mobility of kinematic chains. *J Robot Syst* 20(8):477–499.
6. Coelho P, Nunes U (2003) Lie algebra application to mobile robot control: a tutorial. *Robotica* 21:483–493
7. Sparacino F, Herve JM (1993) Synthesis of parallel manipulators using Lie-groups Y-star and H-robot. Can robots contribute to preventing environmental deterioration? In: Proceedings of the IEEE international workshop on advanced robotics, Tsukuba, pp 75–80
8. Dekret A, Jan B (2001) Applications of line objects in robotics. *Journal of Acta Universitatis Matthiae Belii Math* 9:29–42
9. Singh A, Singla A, Soni S (2014) D–H parameters augmented with dummy frames for serial manipulators containing spatial links. In: The 23rd IEEE international symposium on robot and human interactive communication (RO-MAN 2014), Edinburgh, Scotland, pp 975–980
10. Singh A, Singla A, Soni S (2015) Extension of D–H parameter method to hybrid manipulators used in robot-assisted surgery. *Proc Inst Mech Eng Part H J Eng Med* 229(10):703–712