Avrupa Bilim ve Teknoloji Dergisi
Sayı 31 (Ek Sayı 1), S. 861-869, Aralık 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 31 (Supp. 1), pp. 861-869, December 2021

Research Article

Robot Programming Using Model Based Design Tools

Mustafa Engin[1*], Okan Duymazlar[2], Dilşad Engin[3]

[1*] Ege University, Ege Higher Vocational School, Department of Electronic and Automation, İzmir, Turkey, (ORCID: 0000-0001-7247-4545), mustafa.engin@ege.edu.tr
[2] Ege University, Ege Higher Vocational School, Department of Mechatronics, İzmir, Turkey, (ORCID: 0000-0002-1327-7493), okan.duymazlar@ege.edu.tr
[3] Ege University, Ege Higher Vocational School, Department of Electronic and Automation, İzmir, Turkey, (ORCID: 0000-0003-0159-275X), dilsad.engin@ege.edu.tr

(First received 16 October 2021 and in final form 27 December 2021)

(**DOI:** 10.31590/ejosat.1010444)

ATIF/REFERENCE: Engin, M., Duymazlar, O. & Engin, D. (2021). Robot Programming Using Model Based Design Tools. *European Journal of Science and Technology*, (31), 861-869.

Abstract

The aim of this study is to perform a model-based design for six-revolute (6R) manipulators with six degrees of freedom commonly used in the industrial field and to obtain a simulation and test environment as output. 3D model of the 6R manipulator is designed and drawn using a solid drawing and simulation software. The obtained solid model is transferred to MATLAB™ and integrated to SimMechanics™. With the obtained data output, the forward and inverse kinematic calculations are performed in the computer environment instead of the manipulator controller. The functionality of the simulation and designed visual interface (VI) is tested using the IRB120 6R manipulator. The data transfer between the interface and the manipulator is performed via TCP / IP socket communication.

Keywords: Kinematics, Robotics, Model-based design, Socket communication.

Model Tabanlı Tasarım Araçları ile Robot Programlama

Öz

Bu çalışmada endüstriyel alanda yaygın olarak kullanılan altı serbestlik derecesine sahip 6R manipülatörleri için model tabanlı bir tasarım yapılmış ve çıktı olarak bir simülasyon ve test ortamı elde edilmiştir. 6R manipülatörünün 3 boyutlu modeli, katı model ve simülasyon yazılımı kullanılarak tasarlanmış ve çizilmiştir. Elde edilen katı model MATLAB™'e aktarılmış ve SimMechanics™'e entegre edilmiştir. Elde edilen çıktı ile ileri ve ters kinematik hesaplamalar bilgisayar ortamında gerçekleştirilmiştir. Tasarımı gerçekleştirilen simülasyon ve tasarlanan görsel arayüzün işlevselliği, IRB120 6R manipülatörü kullanılarak doğrulanmıştır. Arayüz ile manipülatör arasındaki veri aktarımı, TCP / IP soket iletişimi ile yapılmıştır.

Anahtar Kelimeler: Kinematik, Robotik, Model tabanlı tasarım, Soket iletişimi.

[*] Corresponding Author: mustafa.engin@ege.edu.tr

1. Introduction

Increasing demand in the digitization of manufacturing, mass customization, and flexible manufacturing systems bring forth the utilization of robots in industry is becoming more widespread. Robots are one of the main components of the industry 4.0, which conceptualize the flexible production systems and the technological transformation with the adoption of computers and automation improved by smart autonomous systems that are equipped with data (Hermann et al., 2016; Mourtzis et al., 2019).

The use of robots in mass production lines, transport, assembly, and quality control areas where repetitive works are carried out, provides a 90% reduction in the unqualified workforce needed in production enterprises, while the faulty production rates allow an 80% reduction in comparison with traditional methods (Alkan, 2018).

When the benefits of integrating robots into existing systems are considered, an adaptation of the robots to the manufacturing and production industry becomes important, regardless of whether they have large or small production quantities. However, it will take time in learning how to control a new robot that will be included in production, packaging, transportation, or quality control operations. Learning cost and time loss will arise for different interfaces and software of each manufacturer's robot. Moreover, the requirements related to the control development of the robot's motion performance result in utilization of simulation and model-based control of industrial robotic applications to enhance the productivity (Brogårdh, 2009).

Although different industrial robot manufacturers produce robots in various forms, the general form of industrial robots is like each other. It will be useful to develop a generalized testing interface that will serve as an input and monitoring interface. This interface also acts like a simulator, which only needs link dimensions of manipulators, which are of the same form.

Researchers work on software interface development and hardware design platforms. These are dedicated to specific industrial robots, only hardware design, or just simulation platforms.

A SolidWorks® Application Programming Interface is utilized where a platform, named as IRoSim, is developed for CAD design and a simulation of serial robot arms with different types of joints (Baizid et al., 2016). Neto et al. present an intuitive robot programming based on CAD drawings to generate offline robot programs equipped with human-robot interface (Neto et al., 2012). A developed toolbox named ARTE (A Robotics Toolbox for Education) is for teaching of serial industrial robotic manipulators, which works under MATLAB and has functions to solve kinematics and dynamics related problems as well as graphics functions to get dynamic visualization of robotic manipulators. This toolbox eases the development of advanced robotic models and simulate them (Gil et al., 2015). 3D CAD design and dynamic simulation of a KUKA robot via Autodesk Inventor, and simulation by MATLAB SimMechanics™ gives very accurate results comparatively (Udai et al., 2011). A CAD-based programming platform and human-robot interface is developed based on open-source libraries where any user can generate a robot path from a CAD model and visualize the simulation (Schou et al., 2013). A Java-based open-source platform, JOpenShowVar, is designed for industrial Kuka robots and robot controllers as a communication interface which allows to read/write the controlled manipulator variables and data structures (Sanfilippo et al., 2015). Rehbein et al. (2019) developed a simplified model-based design for industrial robot programming with offline programming capability (Rehbein, Wrutz, and Biesenbach 2019). This research provides model-based robot programming for KUKA robot manipulators using MATLAB/Simulink with an implemented interface to improve the robot system capabilities by advanced programming languages with the aim to increase the flexibility and ease of robot programming.

These efforts are for less time consuming, and more flexible solutions regarding the traditional online/offline robot programming. To achieve these goals, we need to convey robot programming to a higher level of abstraction rather than a specific robot programming.

In this study, we carried out a design process for a common type six-rotary (6R) axis open chain manipulators produced by different manufacturers, which are only different in link dimensions. The model-based design process includes the mathematical model of the manipulator, the forward and inverse kinematic calculations, as well as the dynamic analysis to determine the torque requirements of the actuators and the implementation of the controller design for use in the simulator. In the design phase, observability is prioritized instead of aesthetic concern. For this purpose, the observed manipulator in the simulation interface is designed as a direct drive in the form of a skeleton. Control of ABB's 6R robot IRB120, which is independent of the designed system, is performed with the test interface. We examined the functionality of this interface and shared the findings with the suggestions.

Section 2 introduces the physical system design, material selection, forward and inverse kinematic analysis and dynamic analysis to determine torque requirements of the manipulator. This section also presents the control interface and programming ABB IRB120 manipulator via Transmission Control Protocol/Internet Protocol (TCP/IP) socket communication with the IRC5 server. We presented our results in Section 3. Section 4 reviews the results and discuss the findings.

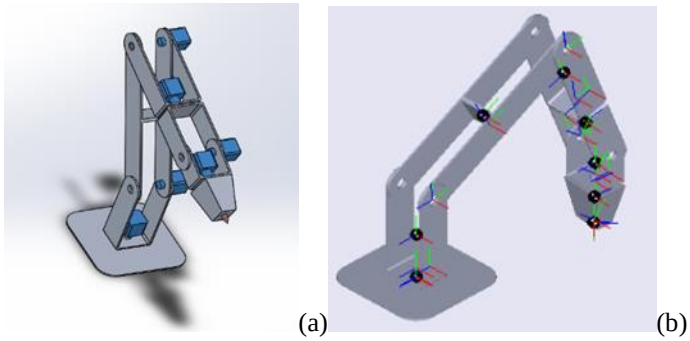
2. Material and Method

The two basic parameters to classify open chain industrial manipulators are the degree of freedom and the joint type. The degree of freedom (DoF) is the most important factor in robotic modeling and increasing the complexity in kinematic analysis. Basically, for open chain manipulators, DoF determined by the total numbers of joints which connect two cascade link/arm (Banka & Lin, 2003). Joint types are classified as spherical, cylindrical, prismatic, helical, planar, and rotating. In open chain industrial robots, commonly rotary joints and prismatic joints are preferred. We preferred a 6R manipulator of 6-DoF for creating a generalized test environment where all joints are rotary.

2.1. Physical System Design

A CAD process is handled to design a 6R serial manipulator using SolidWorks® software as shown in Figure 1(a). The most important aim considered in the design is to decrease the design complexity to observe the movements visually. Then it is

integrated to Simulink using SimMechanics™ add-on library as shown in Figure 1(b).



(a) (b)

Figure 1 6R serial manipulator (a) design using SolidWorks®, (b) Simulink-integrated design using SimMechanics™

The designed 3D manipulator can also be classified as articulated robotic arm in literature, which is human arm structure-based manipulators where all joints are rotary. Articulated robots are the most common type since they manipulate objects in 3D workspace successfully and they are cost effective while having the minimum DOF for 3D manipulation requirements. In industrial use, articulated serial manipulators with six rotary axes, are commonly used for painting, welding, assembly, quality control, etc. and electric motors are used as actuators in drive systems (Craig, 2005). Although the link dimensions change, the basic skeletal structure remains the same. Therefore, we shaped our design for the articulated as simplified to contain all these findings.

The position of a point in space can be expressed in parameters x, y, z in position by [3×1] vector. In the case of rigid bodies, to determine the position in space, there is also a need for the [3×3] orientation matrix. The orientation matrix refers to the orientation of a coordinate system relative to another coordinate system. This orientation is indicated by the parameters Yaw, Pitch and Roll respectively for the x, y and z axes (Fig. 2). In kinematic calculations, an orientation matrix should be created for each joint based on the coordinate of the joint before it. The homogeneous transformation matrix is obtained by combining the position vector and the orientation matrix of each axis (Mikkelsen, 1998). The resulting transformation matrix is used for forward and reverse kinematic calculations.

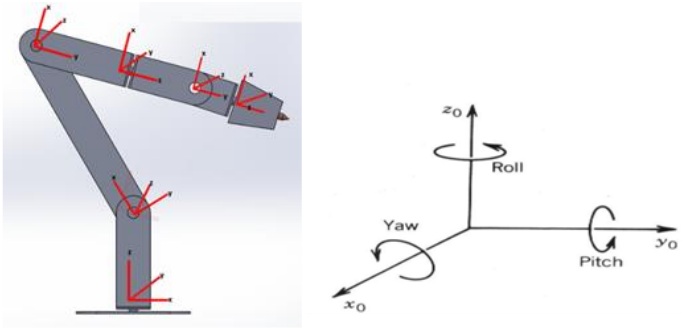


Figure 2 Orientation indicated by the parameters Yaw, Pitch and Roll respectively for the x, y and z axes

2.2 Forward Kinematics

We use the DH method developed by Denavit and Hartenberg (Denavit & Hartenberg, 1955) for kinematic analysis. Forward kinematics is the process of taking the joint angles of the robot as input and obtaining the position and orientation of the end effector of the robot as output. To find the position of the end effector, all the obtained matrices (one for each coordinate frame) are multiplied (Mikkelsen, 1998).

For a 6-DoF articulated arm, six DH parameters are defined as data set with three constants and one variable. Each measurement in the table is calculated by considering the previous joint and coordinate frame.

The DH table for the designed articulated arm, which is used for simulation environment, is shown in Table 1 where a is the distance between x-axis of successive links measured in the z-axis, d is the distance between z-axis of successive links measured in the x-axis, alpha is the angular difference between z-axes of sequential coordinate frames, and theta is the variable that express the rotation of the joint in its own z-axis direction.

Table 1. DH table for the designed articulated arm

Axis (i)	a (mm)	d (mm)	alpha (°)	theta (°)
1	0	425	-90	variable
2	850	0	0	variable
3	425	0	90	variable
4	0	-295	-90	variable
5	0	0	90	variable
6	0	-385	180	variable

Transformation matrix T_i is calculated for each axis using Eq. (1).

$$T_i = \begin{bmatrix} \cos(Q_i) & -\sin(Q_i)*\cos(\alpha_i) & \sin(Q_i)*\sin(\alpha_i) & a_i*\cos(Q_i) \\ \sin(Q_i) & \cos(Q_i)*\cos(\alpha_i) & -\cos(Q_i)*\sin(\alpha_i) & a_i*\sin(Q_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Then each transformation matrices are multiplied consecutively and a [4×4] matrix is obtained, which contains both rotation and the position values as given in Eq. (2).

$$T_0^6 = T_5^6 * T_4^5 * T_3^4 * T_2^3 * T_1^2 * T_0^1$$

$$T_0^6 = \begin{bmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & s & a & P \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

From the resulting matrix, the position of the end effector of the robot is obtained with the P vector and the orientation matrix of size [3×3] formed in the upper left corner.

$n_{x,y,z}$, $s_{x,y,z}$, $a_{x,y,z}$ that forms the orientation matrix correspond to the following statements algebraically as given by Eqs. (3–11).

If the parameters *a* and *d* in the DH table are changed for any robot, i.e., a 6R articulated arm which has the same form with different link dimensions, the same DH table and calculations can also be used for kinematic calculations. For the industrial robot ABB IRB120 having different dimensions from the pre-designed robot for the simulation environment, kinematic calculations and control are performed by updating only the link dimensions.

$$nx = r_{11} = (s\theta4s\theta6 - c\theta4c\theta5c\theta6\,)\,c\theta1((s\theta2s\theta3\; - c\theta2c\theta3\,) - \;s\theta5c\theta6\;(c\theta2s\theta3 + s\theta2c\theta3)) + s\theta1(c\theta4s\theta6 + s\theta4c\theta5c\theta6) \quad (3)$$

$$ny\; = r_{21} = s\theta1((s\theta4s\theta6 - c\theta4c\theta5c\theta6\,)(s\theta2s\theta3 - c\theta2c\theta3\,) - s\theta5c\theta6\;(c\theta2s\theta3 + s\theta2c\theta3)) - \;c\theta1(s\theta4c\theta5c\theta6 + c\theta4s\theta6) \quad (4)$$

$$nz\; = r_{31} = \;(c\theta2\;s\theta3 + s\theta2\;c\theta3\,)(s\theta4s\theta6 - c\theta4c\theta5c\theta6\,) + s\theta5c\theta6\;(s\theta2s\theta3 - c\theta2c\theta3\,) \quad (5)$$

$$sx\; = r_{12} = \;c\theta1((s\theta2s\theta3 - c\theta2c\theta3)(c\theta4c\theta5s\theta6 + s\theta4c\theta6) + s\theta5s\theta6\;(c\theta2s\theta3 + s\theta2c\theta3)) + s\theta1(c\theta4c\theta6 - s\theta4\;c\theta5s\theta6) \quad (6)$$

$$sy\; = r_{22} = s\theta1((s\theta2s\theta3 - c\theta2c\theta3)(c\theta4c\theta5s\theta6 + s\theta4c\theta6) + s\theta5s\theta6(c\theta2s\theta3 + s\theta2c\theta3)) - c\theta1(c\theta4c\theta6 - s\theta4\;c\theta5s\theta6) \quad (7)$$

$$sz\; = r_{23} = \;(c\theta2s\theta3 + s\theta2c\theta3\,)(c\theta4c\theta5s\theta6 + s\theta4c\theta6\,) - s\theta5s\theta6(s\theta2s\theta3 - c\theta2c\theta3\,) \quad (8)$$

$$ax\; = r_{13} = \;c\theta1(c\theta4s\theta5(s\theta2s\theta3 - c\theta2c\theta3) - c\theta5(c\theta2s\theta3 + s\theta2c\theta3\,)) - s\theta1s\theta4s\theta5 \quad (9)$$

$$ay\; = r_{23} = \;s\theta1(c\theta4s\theta5(s\theta2s\theta3 - c\theta2c\theta3\,) - c\theta5(c\theta2\;s\theta3 + s\theta2c\theta3\,)) + c\theta1s\theta4s\theta5 \quad (10)$$

$$az\; = r_{33} = \;c\theta4s\theta5(c\theta2s\theta3 + s\theta2c\theta3\,) + c\theta5(s\theta2s\theta3 - c\theta2c\theta3\,) \quad (11)$$

2.3 Inverse Kinematics

Simply, inverse kinematics is a function that takes the end effector position and orientation matrix as inputs and give the joint angular values as outputs. Commonly, it is preferred to use one of three approaches for inverse kinematic calculation of open chain serial manipulators. These approaches are geometric, algebraic, and iterative (Niku, 2001). In theory, it may be complicated to obtain a closed form solution with using geometric or algebraic approach. Consequently, iterative approach seems preferable because it is superior to the first two approaches in terms of its independency to the physical system parameters. However, iterative approach requires more calculations than the algebraic or geometric one and convergence to the correct solution is not guaranteed (Jones BE, 1990). 6R manipulators commonly have the same physical form. Therefore, if all joints of the manipulator are revolute and last three joint axes intersect at a point, shortly spherical wrist, the obtained closed form turns into a general solution.

In this study, we used kinematic decoupling method to obtain closed-form solution, which is a sub-part of the geometric approach. Kinematic decoupling method handles the robot geometry in two parts as body and wrist.

The body part refers to the first three axes (q1, q2, q3) responsible for positioning and the wrist part (q4, q5, q6) represents the last three axes responsible for orientation as shown in Figure 3.

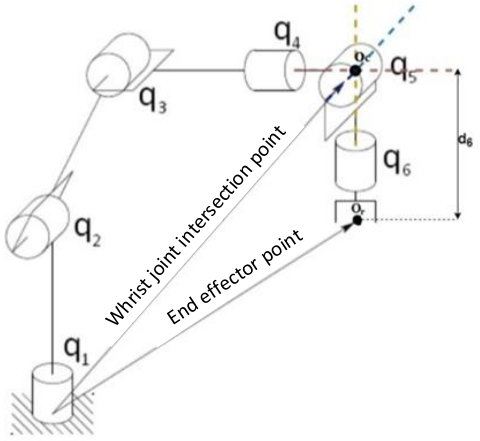


Figure 3 Kinematic Decoupling of 6R Manipulators

Body part calculations of the 6R manipulator are given by Eqns. (12 – 15). For the kinematic calculations *a* and *d* values are taken from the DH table of the manipulator and the A and B variables are for reducing the formulas' complexity. $\mp$ symbol is for obtaining both left- and right-hand solutions.

$$\theta1 = \text{atan2}(p_y, p_x) \quad (12)$$

$$A = ({p_x}^2 + {p_y}^2 + {p_z}^2 - {a_3}^2 - {d_4}^2)\;/\;2{a_3}^2 d_4 \quad (13)$$

$$\theta3 = -atan2(A, \pm\sqrt{(1} - A^2) \quad (14)$$

$$\theta2 = -\text{atan2}(p_z, (\sqrt{{p_x}^2 + {p_y}^2} - atan2(d_4, \sin(\theta3)),\; a_3 + d_4\cos(\theta3)\,) \quad (15)$$

The wrist part calculations are formulated in Eqs. (16–19).

$$\theta5 = \text{atan2}(\pm\sqrt{1 - (r_{13}\sin(\theta1) - r_{23}\cos(\theta1))^2}, r_{13}\sin(\theta1) - r_{23}\cos(\theta1)\,) \quad (16)$$

$$B = r_{13}\cos(\theta1) + r_{23}\sin(\theta1) \quad (17)$$

$$\theta4 = atan2((\mp(B) * sin(\theta2 + \theta3) \mp \big(r_{33}cos(\theta2 + \theta3)\big), (\pm(B)cos(\theta2 + \theta3) \mp r_{23}sin(\theta2 + \theta3))) \quad (18)$$

$$\theta6 = atan2(\pm(r_{12}\sin(\theta1), +r_{22}\cos(\theta1)), \pm(r_{11}\sin(\theta1) - r_{21}\cos(\theta1)) \quad (19)$$

2.4 Dynamic Analysis to Determine Torque Requirements

Torque requirements for each actuator of the designed manipulator, which are subject to kinematic analysis for positioning, were calculated by dynamic analysis. Inverse dynamics utilizes the inertia and the angular acceleration parameters as inputs and calculates the torque values as output.

We determined the material of designed manipulator's skeleton for obtaining the inertial parameters. 1060-H14 Aluminum Alloy with a density of $2.705\;kg/m^3$ was preferred to obtain the inertia matrices used as input.

Six reference coordinates were placed on the rotation axis of each actuator to calculate the inertial matrices. Masses, center of mass positions, and inertia matrices are defined by a [3×3] matrix as in Eq. (20), and the elements of this matrix are reduced to six because of symmetry. The calculated results are presented in Table 2.

$$I_{link} = \begin{bmatrix} Ixx & Ixy & Ixz \\ Iyx & Iyy & Iyz \\ Izx & Izy & Izz \end{bmatrix} \quad (20)$$

Table 2. The elements of the reduced inertial matrix

Link	Mass (kg)	Center of Mass								
		x	*y*	*z*	I_{xx}	I_{yy}	I_{zz}	I_{xy}	I_{xz}	I_{yz}
1	1.19	0.000	0.000	0.094	0.000	0.000	0.021	0.019	0.004	0.000
2	2.235	0.212	0.000	0.050	0.024	0.000	0.011	0.147	0.138	0.000
3	1.192	0.118	0.000	0.062	0.009	0.000	0.008	0.032	0.025	0.000
4	0.855	0.000	0.000	0.092	0.000	0.000	0.010	0.011	0.002	0.000
5	0.561	0.040	0.002	0.051	0.001	0.000	0.002	0.004	0.002	0.000
6	0.454	0.000	0.005	0.060	0.000	0.000	0.002	0.002	0.001	0.000

A worst-case scenario was created for obtaining the second input parameter which was required for inverse dynamics. The manipulator configuration, that causes the maximum momentum and the desired maximum angular velocity for motion were determined.

Resulting inputs were used to obtain torque requirements of the manipulator through the *roboanalyzer* software, which uses recursive method for calculation (Rajeevlochana et al., 2011). The results are shown in Fig. 4. It was obtained as a result of inverse dynamic analysis that the largest torque value was in the second joint motor. Since the layout axis of the base motor is perpendicular to the surface, the first engine's torque requirement was less since the mass effect caused by gravity was negligible.

2.5 Simulation and Control Interface

We designed the simulation and control interface for any 6R manipulator. SolidWorks® is the design platform for the manipulator, and then the obtained model was transferred to Simulink by SimMechanics™ add-on as illustrated in Figure 5.

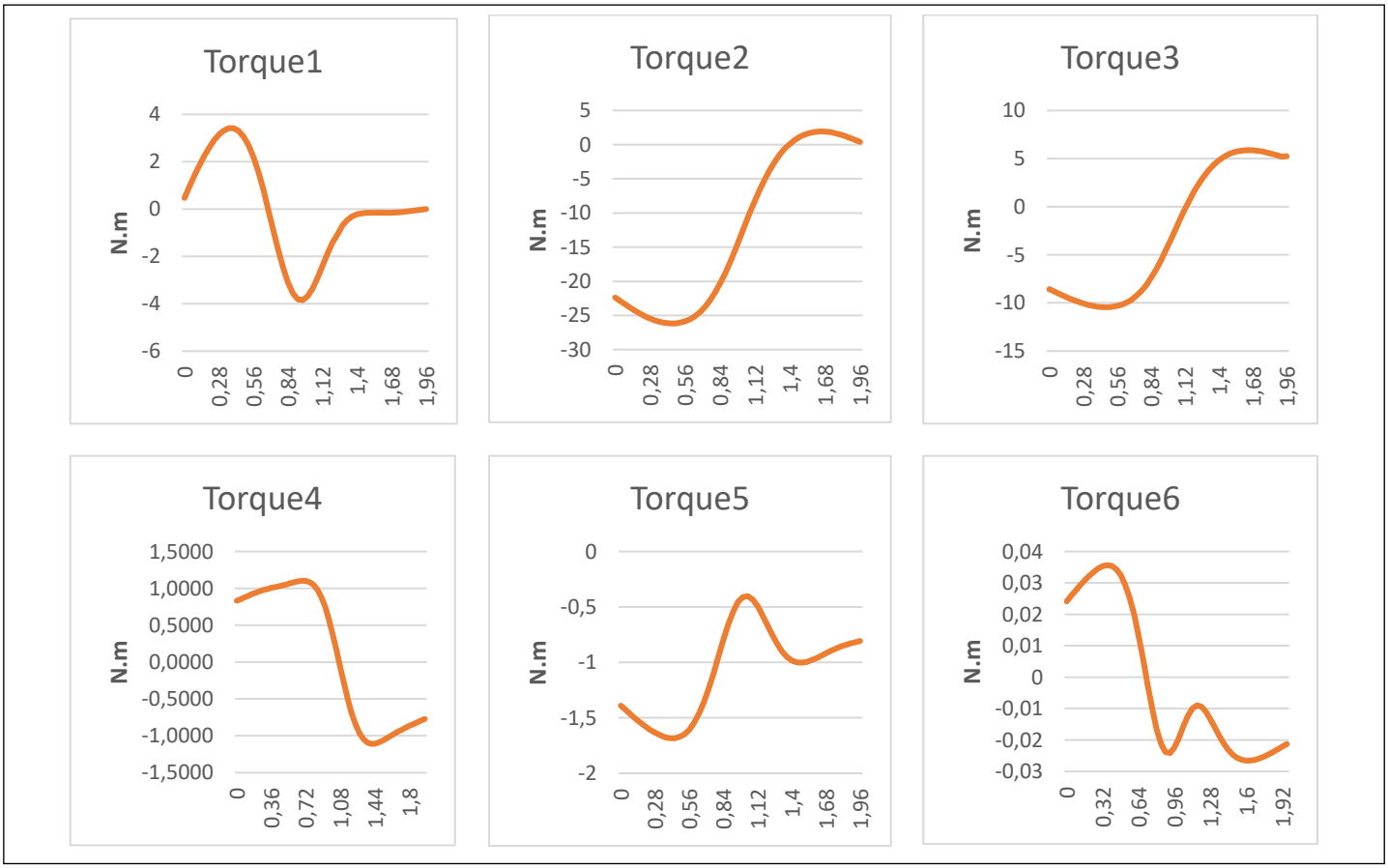


Figure 4 Calculated torque requirements for each actuator

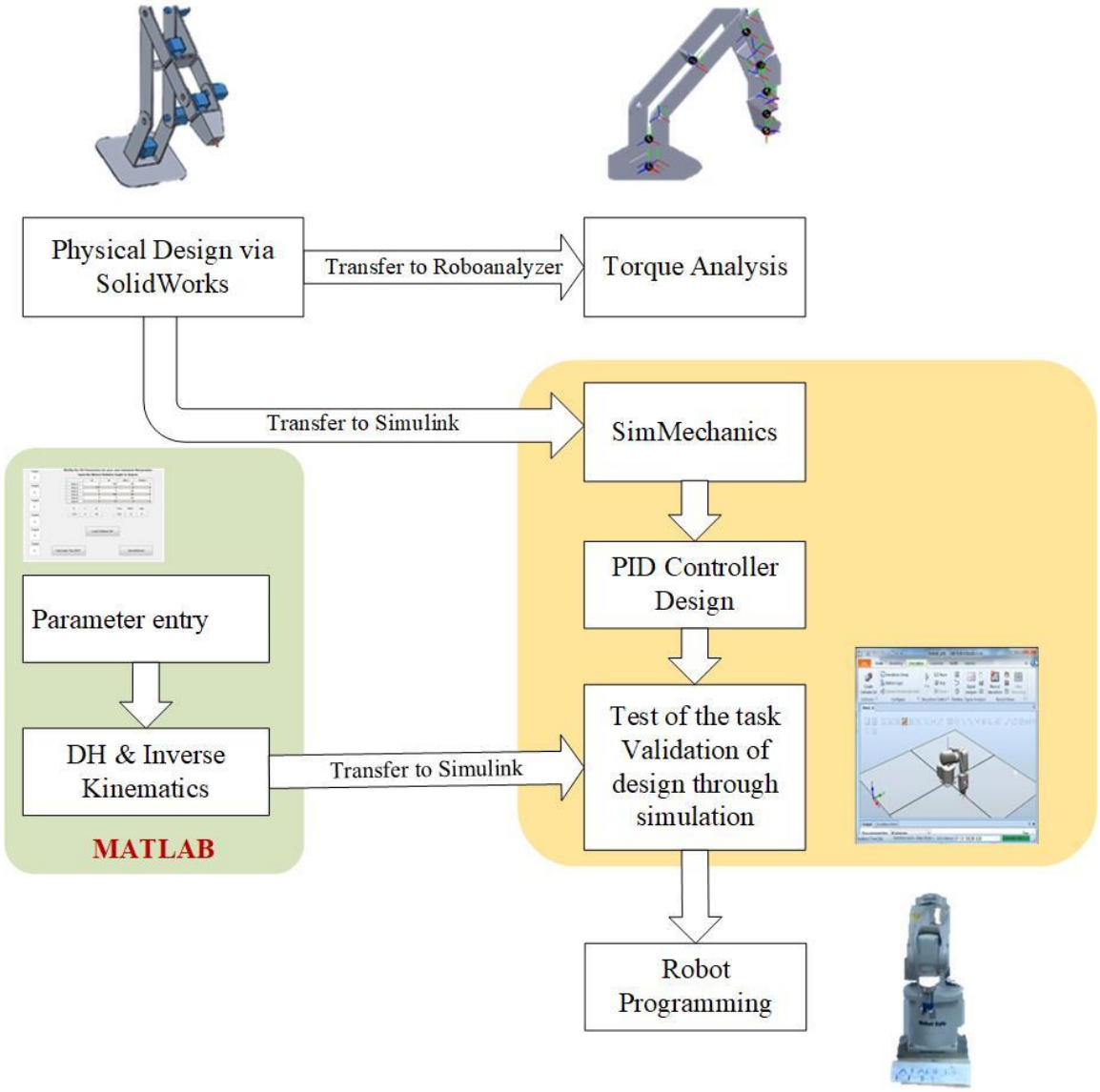


Figure 5 Scheme of the CAD programming and testing

For performing the real-time simulation of motion, we used PID control block, which uses the result of each kinematic calculation as an input. One separate controller was designed for each joint actuator that activates the revolution as shown in Fig. 6.

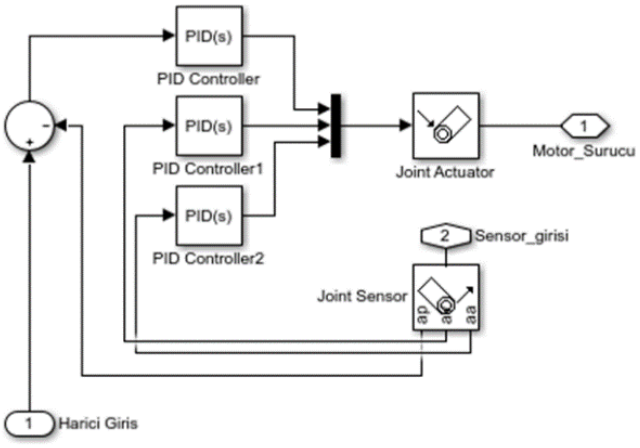


Figure 6 PID controller design for joint actuators

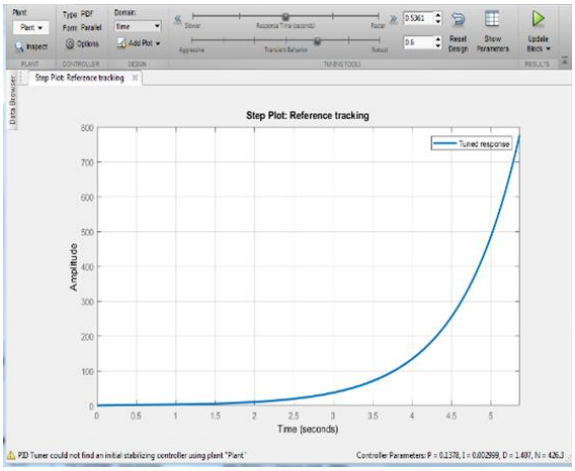
The response time and amplitude parameters of the simulation PID controller were changed to 0.05361s and 0.6 respectively, after auto-tuning for the smooth and stable motion of the 6R manipulator. Untuned response and tuned response for one joint actuator is shown in Figure 7a and Figure 7b, respectively.

Joint parameters entry is through the designed MATLAB Graphical User Interface (GUI) in Fig. 8 to facilitate the movement of the robot manipulator. The GUI was used to supply the target manipulators kinematic model over DH parameters, calculate forward or inverse kinematics and to send joint angular values the robot controller over TCP/IP.

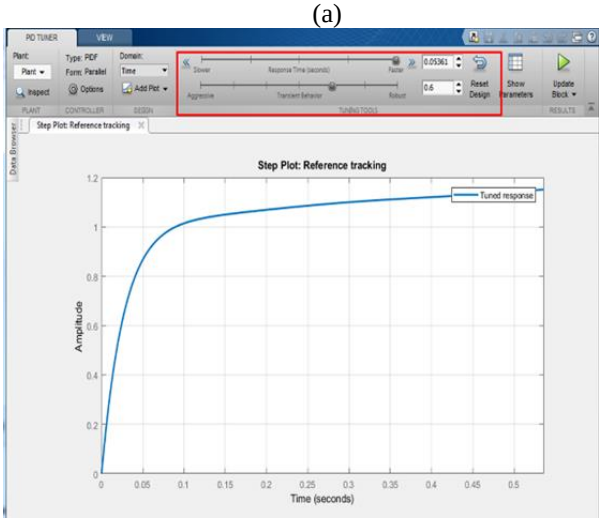
2.3 Communication Method

We tested the simulation environment on ABB IRB120 manipulator via TCP/IP socket communication with the IRC5 server. In the standard internet protocol TCP, IP address is equipped with the socket address, which is the port number, and this type of communication is used to communicate with the IRC5 client server. The ABB IRC5 is a robot controller, which has a unique motion control feature and reliably follows the robot motion path as programmed through the application

interfaces. Electronically linked motors in a robot control master/slave motor configuration actuate the main axes of the robot. Encoders give the exact angular position of each actuator. The IRC5 server is the motion controller (drive) part of the system that enables and positions the actuator via motor drives.



(a)



(b)

Figure 7 (a) Untuned response, (b) tuned response for the joint actuator.

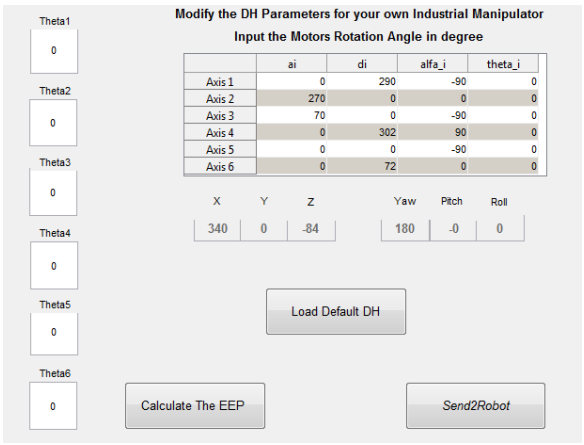


Figure 8 The designed GUI for joint parameters entry and socket communication.

The IRC5 is a multi-robot controller with PC tool support that optimizes the robot performance for short cycle times and precise movements. Communication is real-time, resulting in a continuous communication between PC and IRC5 (Server-Client). Programming interface based on HTML5 communicates with robot from any device, regardless of the operating system. The computer manages the task control and robot simulation whereas IRC5 controller is free of computational complexity and only used for control of the motors. We programmed this server to receive the position and the angle data from the embedded MATLAB program codes and transmit to the motor drives. To send the calculated joint angles to the IRC5 controller for moving manipulator to the desired positions determined by user in the designed GUI in Figure 8, a client software was created with the RAPID programming language as shown in Figure 9 over pseudocode, and motion commands were processed simultaneously with the simulation environment. For any other manipulator belonging to different manufacturers which has the same 6R articulated form, the only requirement is just creating a client-side program with the controller's own robot programming language to receive TCP/IP socket messages by using the pseudocode in Figure 10.

```
Input: IP address and port to receive socket message
(192.168.125.1, 55000)
Start to listen socket and receive the allow to incoming
connections (Socketbind)
While (192.168.125.1, 55000 = no message)
    Keep waiting for the receiving message
End
    Send message to Server (MATLAB) to approve successful
communication
    Read string message sent by Designed GUI and convert
string to float numbers for joint angles
    Send message to Server (MATLAB) to finish successful
reading of socket messages
    Wait 50 ms and close the socket
    Move joints to received joint angles with MoveAbsJ
command of RAPID language
```

Figure 9 IRC5 Client TCP/IP socket Communication pseudo-code

```
MODULE Mainmodule
  PERS tooldata withgripper:=[TRUE, [[0, 0, 103.2], [1, 0, 0,
0]],[0.1, [0, 0, 0.001],[1, 0, 0, 0], 0, 0, 0]];
  CONST jointtarget sifiracilar:=
[[0,0,0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  VAR string receivedFromMatlab;
  VAR socketdev clientABB;
  VAR socketdev serverMATLAB;
  VAR rawbytes data;
  PROC main ()
    SocketCreate clientABB;
    SocketBind clientABB," 192.168.125.1",55000;
    SocketListen clientABB;
    SocketAccept clientABB, serverMATLAB;
    SocketSend serverMATLAB,\Str:="Communication
Established
    SocketReceive serverMATLAB,\Str:= receivedFromMatlab;
    MoveJ receivedFromMatlab,v150,fine, withgripper;
    SocketClose clientABB;
  ENDPROC
ENDMODULE
```

Figure 10 IRC5 Client Program in RAPID Language

3. Results and Discussion

The model-based design process for the generalized 6R articulated manipulator, which is widely used for painting, welding, and assembly operations, was carried out including kinematic and dynamic analysis. Functionality of the graphical interface obtained as a result of the model-based design has been tested for the IRB120 robot arm of ABB company as shown in Figure 11. In the tests carried out, the robot's DH parameters and desired positions provided by the user through the designed GUI were used as inputs, and the robot was moved to randomly selected positions for various tasks. As an indicator of the successful performance of the designed interface, it has been seen that the same graphical outputs were obtained with RobotStudio, ABB's own simulation program, and physical robot arm successfully reached the desired positions and performed various tasks as shown in Figure 11.

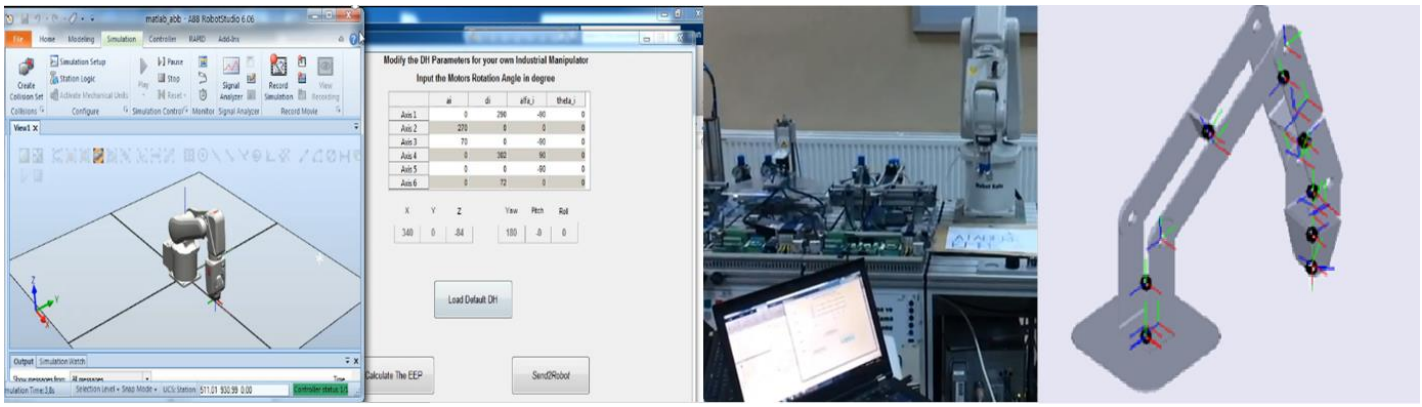


Figure 11 Observations of 6R manipulator over RobotStudio, Designed GUI and Physical Manipulator

4. Conclusions and Recommendations

Model-based design of the 6 DOF articulated robot arm, which is widely used in industrial applications like painting, welding and assembly operations, was carried out including material selection, kinematic analysis and dynamic analyzes for calculating torque requirements. Also, a simplified plate-based 3D model was created for the simulation and clear observation of the any robotic manipulator which has the same kinematic structure. The GUI obtained from the model-based design was tested over ABB IRB120 industrial manipulator. Calculated joint angular values for the desired position and tasks were sent to IRC5 robot controller over TCP/IP socket messages and handled various tasks. It has been shown on the physical robot that the developed interface and real-time tests worked successfully. The functionality of the developed control interface that can be applied to the robots in the same form, has been tested and verified with the applications performed with the IRB120 robot.

This study also showed that, in distance learning and remote task planning, only by writing client-side software for the robot controller in manufacturer's own language to accept TCP/IP socket messages, the control of the kinematic calculation and motion planning accuracy of the users and whether they perform the specified tasks can be observed without the need for any other software.

References

Alkan, M. A. (2018). Karanlık Fabrikalar ile İnsansız Üretim. Endüstri 4.0. https://www.endustri40.com/karanlik-fabrikalar-ile-insansiz-uretim/

Baizid, K., Ćuković, S., Iqbal, J., Yousnadj, A., Chellali, R., Meddahi, A., Devedžić, G., & Ghionea, I. (2016). IRoSim: Industrial Robotics Simulation Design Planning and Optimization platform based on CAD and knowledgeware technologies. Robotics and Computer-Integrated Manufacturing, 42, 121–134. https://doi.org/10.1016/j.rcim.2016.06.003

Banka, N., & Lin, Y. J. (2003). Mechanical design for assembly of a 4-DOF robotic arm utilizing a top-down concept. Robotica, 21(5), 567–573. https://doi.org/10.1017/S0263574702004848

Brogårdh, T. (2009). Robot control overview: An industrial perspective. In Modeling, Identification and Control (Vol. 30, Issue 3, pp. 167–180). https://doi.org/10.4173/mic.2009.3.7

Craig, J. J. (2005). Introduction to Robotics Mechanics and Control (A. Dworkin, Ed.; 3rd ed.). Pearson Prentice Hall.

Denavit, J., & Hartenberg, R. (1955). A kinematic notation for lower-pair mechanisms based on matrices. ASME Journal of Applied Mechanics, 22, 215–221.

Gil, A., Reinoso, O., Marin, J. M., Paya, L., & Ruiz, J. (2015). Development and deployment of a new robotics toolbox for education. Computer Applications in Engineering Education, 23(3), 443–454. https://doi.org/10.1002/cae.21615

Hermann, M., Pentek, T., & Otto, B. (2016). Design principles for industrie 4.0 scenarios. Proceedings of the Annual Hawaii International Conference on System Sciences, 2016-March, 3928–3937. https://doi.org/10.1109/HICSS.2016.488

Jones BE, F. J. (1990). Modelling, Simulation and Identification of an Industrial Manipulator. Dublin City University.

Mikkelsen, J. (1998). A machine vision system controlling a Lynx arm robot along a path. University of Cape Town, South Africa.

Mourtzis, D., Fotia, S., Boli, N., & Vlachou, E. (2019). Modelling and quantification of industry 4.0 manufacturing complexity based on information theory: a robotics case study. International Journal of Production Research, 1–14. https://doi.org/10.1080/00207543.2019.1571686

Neto, P., Mendes, N., Arajo, R., Pires, J. N., & Moreira, A. P. (2012). High-level robot programming based on CAD: Dealing with unpredictable environments. Industrial Robot, 39(3), 294–303. https://doi.org/10.1108/01439911211217125

Niku, S. B. (2001). Introduction to Robotics: Analysis, Systems, Applications (1st ed.). Prentice Hall.

Rajeevlochana, C. G., Jain, A., Shah, S. V, & Saha, S. K. (2011). Recursive Robot Dynamics. In S. Bandopadhyay, S.

Gurunathan, & P. Ramu (Eds.), Introduction to Robotics (pp. 1–9). Narosa Publishing House, New Delhi.

Rajeevlochana, C. G., & Saha, S. K. (2011, February). RoboAnalyzer: 3D model based robotic learning software. In International Conference on Multi Body Dynamics (pp. 3-13).

Sanfilippo, F., Hatledal, L. I., Zhang, H., Fago, M., & Pettersen, K. Y. (2015). Controlling Kuka Industrial Robots: Flexible Communication Interface JOpenShowVar. IEEE Robotics & Automation Magazine, 22(4), 96–109. https://doi.org/10.1109/MRA.2015.2482839

Schou, C., Damgaard, J. S., Bogh, S., & Madsen, O. (2013). Human-robot interface for instructing industrial tasks using kinesthetic teaching. 2013 44th International Symposium on Robotics, ISR 2013, 1–6. https://doi.org/10.1109/ISR.2013.6695599

Udai, A. D., Rajeevlochana, C. G., & Saha, S. K. (2011). Dynamic Simulation of a KUKA KR5 Industrial Robot using MATLAB SimMechanics. 15th National Conference on Machines and Mechanisms, 96, 1–8.