Modeling, control, and simulation of a SCARA PRR-type robot manipulator

M.E. Uk, F.B. Sajjad Ali Shah, M. Soyaslan*, and O. Eldogan

Faculty of Technology, Department of Mechatronics Engineering, Sakarya University, Sakarya, Turkey.

Received 19 June 2018; received in revised form 2 September 2018; accepted 29 October 2018

KEYWORDS
SCARA robot; Real-time control; Modelling; Simulation; Prismatic-Revolute-Revolute (PRR); Servo motor.

Abstract. In this study, a SCARA Prismatic-Revolute-Revolute-type (PRR) robot manipulator is designed and implemented. Firstly, the SCARA robot is designed in accordance with the mechanical calculations. Then, forward and inverse kinematic equations of the robot are derived by using D-H parameters and analytical methods. The software is developed according to the obtained Cartesian velocities from joint velocities and joint velocities from Cartesian velocities. The trajectory planning is designed using the calculated kinematic equations, and the simulation is performed in MATLAB VRML environment. A stepping motor is used for the prismatic joint of the robot, and servo motors are used for revolute joints. While most of the SCARA robot studies focus on the Revolute-Revolute-Prismatic-type (RRP) servo control strategy, this work focuses on PRR-type and both stepper and servo control structures. The objects in the desired points of the workspace are picked and placed to another desired point synchronously with the simulation. Therefore, the performance of the robot is examined experimentally.

* Corresponding author. Tel.: +90 2642956912; Fax: +90 2642956124; E-mail addresses: mustamned.uk@ogr.sakarya.edu.tr (M.E. Uk); faris.ali.shah@outlook.com (F.B. Sajjad Ali Shah); msoyaslan@sakarya.edu.tr (M. Soyaslan); eldogan@sakarya.edu.tr (O. Eldogan)


1. Introduction
Nowadays, the objective of production at high speeds with low costs and low error rates in industrial production lines has gained great importance in terms of competitiveness. For this reason, companies often use different types of robots, such as Cartesian, SCARA, etc., in industrial applications. Cartesian systems are widely used in high-density warehouses and, generally, have both shuttle and aisle robots that generate the Cartesian structure [1,2]. SCARA (Selective Compliance Assembly Robot Arm) manipulators take up less space than Cartesian systems, are easier to install, and can operate without the need for large areas. For this reason, the processes such as packaging, sorting, alignment, planar welding, and assembly in the production lines are usually performed with SCARA-type manipulators. The first SCARA robot was developed in 1978 by Professor Hiroshi Makino at Yamanashi University in Japan [3]. Afterwards, many types of SCARA robots have emerged to be used in the machine, automotive, and robot industries.

In literature studies, kinematic and dynamic modeling, simulation analysis, different control methods, and trajectory planning have been studied both theoretically and experimentally. Different decentralized and centralized (model-based) controllers have been tested with experimental studies of an industrial SCARA robot.

As a result, the performance of decentralized controllers was found to be sufficiently accurate for a large number of industrial applications [4]. Accurate results of experimental studies depend on well-made mathematical modeling. In SCARA robots, which
are generally used in industrial applications, it is very important to make both dynamic and kinematic calculations accurately in order to make the system work properly. While Das and Dulger [5] developed a complete mathematical model with actuator dynamics and motion equations derived by using the Lagrangian mechanics, Abshamasin et al. [6] investigated kinematic modeling and simulation of a SCARA robot by using solid dynamics by means of Matlab/Simulink. Unlike other studies, Urrea and Kern [7] implemented a simulation of a 5-Degree-Of-Freedom (DOF) SCARA manipulator using Matlab/Simulink software. Their study has no physical application, although it bears similarity with the work we have done. This study enjoys some advantages over these types of works, which include only modeling and simulation. Kaleli et al. [8] and Konayem et al. [9] designed a program for simulating and animating the SCARA robot kinematics and dynamics in LabView software. Similar to these works, there are various robot control, simulation, and calculation program studies in the literature [10-20]. While some of them are just based on the analysis and simulation of one type of robot arms, some give results for robots in different types.

SCARA robots with RRP (Revolute-Revolute-Prismatic) or PRR (Prismatic-Revolute-Prismatic) joint configurations are easy to provide linear movement in vertical directions. RRP and PRR types have some advantages and disadvantages. RRP-type SCARA manipulators are very common in light-duty applications that require precision and speed, which is difficult to achieve by human beings [21]. While the prismatic joint motor is only lifting the objects in RRP type, it is lifting the whole robot structure with the objects in PRR type. Therefore, the prismatic joint motor of PRR type has higher torque than that of RRP type. Therefore, the PRR-type SCARA robot configuration is preferred in applications, where lifting heavy weights is a challenge. Since the base is fixed on one point, powerful torque motors for lifting heavy loads linearly can be used easily.

In this study, a PRR-type (Prismatic-Revolute-Revolute) SCARA robot manipulator is designed. In addition, a gripper is placed on the last joint so that the objects can be picked and placed at the desired locations. In the first section, the usage areas of SCARA robots and the studies in the literature are mentioned. In Section 2, the forward kinematics of the robot is obtained by using the Denavit-Hartenberg (D-H) method [22]. Then, the inverse kinematics equations and Jacobian matrix are obtained by using analytical methods. In Section 3, the experimental setup of the robot is explained. In Section 4, the control and simulation software are described. In the conclusion section, the results and discussion are presented.

2. Robot kinematics

2.1. Forward kinematics

Robot forward kinematics deals with the relationship among the positions, velocities, and accelerations of robot joints [23]. A robot consists of links that are attached to each other by prismatic or rotary joints. Coordinate systems are placed to each joint to find the transformation matrices that set the relation between two neighboring joints. The transformation matrix between the two joints is shown like $^{i-1}T_i$. The relation between the base frame and the tool frame is defined by the serial joint transformation matrices. This relation is called forward kinematics and shown in Eq. (1):

$$\mathbf{0}_N T = \prod_{i=2}^{N} T_i T_1 = T_1 T_2 T_3 \ldots T_N .$$

The projected SCARA robot and the axes on the joints are shown in Figure 1. While $d_3$ is the length of the vertical joint, $l_1$ and $l_2$ are the horizontal

![Figure 1. Joint axes and implemented SCARA robot.](image-url)
Table 1. $D-H$ parameters.

<table>
<thead>
<tr>
<th>$i$</th>
<th>$a_i$</th>
<th>$a_{i-1}$</th>
<th>$\theta_i$</th>
<th>$d_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$d_3$</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>$\theta_1$</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>$l_1$</td>
<td>$\theta_2$</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>$l_2$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$a_{i+1}$: Angle between $z_{i+1}$ and $x_i$ around $z_i$;  
$b_{i+1}$: Distance between axes $z_{i+1}$ and $z_i$ throughout $x_{i+1}$;  
$\theta_i$: Angle between $x_{i-1}$ and $x_i$ around $z_i$;  
$d_i$: Distance between axes $x_{i+1}$ and $x_i$ throughout $z_i$.

lengths of the other two joints. The length of $d_3$ is calculated considering the height of the gripper. The transformation matrices are obtained as in Eq. (2) by the D-H method:

$$
T_1 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & d_3 \\
0 & 0 & 0 & 1
\end{bmatrix},
T_2 = \begin{bmatrix}
c_1 & -s_1 & 0 & l_1 \\
s_1 & c_1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
T_3 = \begin{bmatrix}
c_2 & -s_2 & 0 & l_2 \\
s_2 & c_2 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.

The abbreviations $c$ and $s$ represent the terms "$\cosine$" and "$\sinus$". The $D-H$ parameters are given in Table 1.

The forward kinematic matrix is obtained through Eq. (3) by the product of the transformation matrices.

$$
T_3 = \begin{bmatrix}
c_{12} & -s_{12} & 0 & c_1 l_1 + c_{12} l_2 \\
s_{12} & c_{12} & 0 & s_1 l_1 + s_{12} l_2 \\
0 & 0 & 1 & d_3 \\
0 & 0 & 0 & 1
\end{bmatrix}.

2.2. Inverse kinematics

Inverse kinematics is the process of finding the values of joint variables according to the given position and orientation data of the end effector. In other words, for the movement of the end effector to the desired position, we need to find the rotation and linear motion values of joints by means of inverse kinematics. Any found mathematical expression may not be a physical solution. There may also be more than one solution for the end effector to go to the desired position. In other words, the robot manipulator can reach the desired positions with different solutions. In this study, the inverse kinematics solution is obtained by the analytical method. The third column of the forward kinematic matrix is shown in Eq. (4), and it gives the $x$, $y$, and $z$ position coordinates of the robot:

$$
O = P = \begin{bmatrix}
c_{11} l_1 + c_{12} l_2 \\
s_1 l_1 + s_{12} l_2 \\
d_3 \\
1
\end{bmatrix} = \begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
1
\end{bmatrix}.

Eqs. (5)-(7) can be easily understood from Eq. (4):

$$
P_1 = c_1 l_1 + c_{12} l_2, (5)
$$

$$
P_2 = s_1 l_1 + s_{12} l_2, (6)
$$

$$
P_3 = d_3, (7)
$$

2.2.1. Calculation of $\theta_1$ angle

Eqs. (5) and (6) are rearranged as follows:

$$
(c_{12} l_2)^2 = (P_1 - c_1 l_1)^2, (8)
$$

$$
(s_{12} l_2)^2 = (P_2 - s_1 l_1)^2. (9)
$$

Eqs. (8) and (9) are summed up together as follows:

$$
l_2^2 = P_1^2 + P_2^2 + l_1^2 - 2l_1(P_1 c_1 + P_2 s_1), (10)
$$

Eq. (11) is obtained when $b_1$ is used instead of $(P_1 c_1 + P_2 s_1)$:

$$
l_2^2 = P_1^2 + P_2^2 + l_1^2 - 2l_1 b_1, (11)
$$

Figure 2 shows the $\theta_1$ derivation illustration. Due to the existence of $b_1$, $v_1$ also can be derived by analytical methods as in Eq. (13). The variables in Figure 2 are calculated as follows:

$$
b_1 = (P_1 c_1 + P_2 s_1), (12)
$$

$$
v_1 = (-P_1 s_1 + P_2 c_1), (13)
$$

$$
r^2 = v_1^2 + b_1^2, (14)
$$

$$
r^2 = P_1^2 + P_2^2, (15)
$$

$$
v_1 = \pm \sqrt{P_1^2 + P_2^2 - b_1^2}, (16)
$$

Figure 2. Derivation illustration of $\theta_1$. 
\[ \theta_1 = \varphi_1 - \varphi_2, \quad (17) \]
\[ \varphi_1 = A \tan 2(v_1, b_1), \quad (18) \]
\[ \varphi_2 = A \tan 2(P_2, P_1), \quad (19) \]

The inverse kinematics of the first rotational joint is obtained through Eq. (20) when Eqs. (18) and (19) are substituted into Eq. (17). Because of two different values of \( v_1 \) from Eq. (16), there are also two different values of \( \theta_1 \). This shows that there are two solutions.

\[ \theta_1 = A \tan 2(\pm v_1, b_1) - A \tan 2(P_2, P_1). \quad (20) \]

### 2.2.2. Calculation of \( \theta_2 \) angle

**Method 1:**
When Eqs. (5) and (6) are multiplied by \( P_2 \) and \( P_1 \), respectively, Eqs. (21) and (22) are obtained as follows:

\[ P_1 c_{12} b_2 = P_1 c_1 l_1, \quad (21) \]
\[ P_2 s_{12} b_2 = P_2 - s_1 l_1. \quad (22) \]

Eq. (23) is obtained after subtracting Eq. (22) from Eq. (21) and, accordingly, some arrangements are made.

\[ b_2[c_{12} P_2 - s_{12} P_1] = -l_1[P_2 c_1 - P_1 s_1]. \quad (23) \]

When \( v_1 \) is written instead of \( P_2 c_1 - P_1 s_1 \), we get:

\[ c_{12} P_2 - s_{12} P_1 = \frac{-l_1 v_1}{l_2}. \quad (24) \]

**Method 2:**
Eqs. (21) and (22) are summed up together as follows:

\[ b_2[c_{12} P_2 + s_{12} P_1] = P_1 P_2 - l_1[P_2 c_1 + P_1 s_1]. \quad (25) \]

Eq. (26) is obtained when \( b_2 \) is used instead of \( (c_{12} P_2 + s_{12} P_1) \).

\[ b_2 = c_{12} P_2 + s_{12} P_1, \quad (26) \]

\( b_1 \) is rearranged as follows:

\[ b_1 = (P_1 c_1 + P_2 s_1) = \frac{P_1^2 + P_2^2 + r_1^2 - r_2^2}{2 l_1}. \quad (27) \]

After placing \( b_1 \) value in Eq. (25), we get:

\[ c_{12} P_1 + s_{12} P_2 = \frac{P_1^2 + P_2^2 - r_1^2 + r_2^2}{2}. \quad (28) \]

Eq. (29) is obtained with some arrangements in Eq. (28).

\[ b_2 = \frac{P_1^2 + P_2^2 - r_1^2 + r_2^2}{2 l_1}. \quad (29) \]

**Figure 3.** Derivation illustration of \((\theta_1 + \theta_2)\).

The final form of Eq. (28) is as follows:

\[ c_{12} P_1 + s_{12} P_2 = \frac{l_1 b_2}{l_2}. \quad (30) \]

When Eqs. (24) and (30) are solved mathematically together on the same graph, Figure 3 is obtained.

Figure (3) can be solved as follows:

\[ \theta_1 + \theta_2 = A \tan 2(-v_1, b_2) - A \tan 2(P_2, P_1). \quad (31) \]

When \( \theta_1 \) value from Eq. (20) is placed in Eq. (31), Eq. (32) is obtained.

\[ \theta_2 = A \tan 2(-v_1, b_2) - A \tan 2(v_1, b_1). \quad (32) \]

### 2.3. Multiple solution

Two different values of \( v_1 \) observed in Eq. (16) show two-solution ways. These solution ways are discussed in this section. Figure 4 shows the two solutions of object orientation.

- **Left side solution:**

\[ \theta_1^{(1)} = 2 A \tan 2(-v_1, b_1). \quad (33) \]

\[ \theta_2^{(1)} = A \tan 2(P_2, P_1) - \frac{1}{2} \theta_2^{(1)}. \quad (34) \]

**Figure 4.** Two solutions of object orientation.
- Right side solution:

\[ \theta_2^{(2)} = 2A \tan 2(v_1, b_1), \]  
\[ \theta_1^{(2)} = -A \tan 2(P_2, P_1) - \frac{1}{2} \theta_2^{(2)}. \]

2.3.1. Examination of the solution's existence

\[ v_1 = \pm \sqrt{r^2 - b_1^2}, \]  
\[ v_1 = \pm \sqrt{(r - b_1)(r + b_1)}. \]

To ensure solution existence, the condition of \((r - b_1 \geq 0)\) must be provided as in Eq. (38).

Reachable maximum and minimum lengths of arms are shown in Figure 5.

If the analysis is carried out according to Figure 5, then:

\[ r_{\text{max}} = l_1 + l_2, \]  
\[ r_{\text{min}} = l_1 - l_2. \]

The existence of solutions is available under the above conditions.

![Figure 5. Maximum and minimum points achievable by the arms.](image)

The Jacobian matrix in robotics is used for many calculation methods such as smooth trajectory planning and execution, singularity determination, derivation of dynamic equations of motion, and torque calculations. The linear and angular velocities at the SCARA robot can be found in terms of joint velocities. The linear velocity can be defined in terms of the position of the end effector. After conducting the intermediate operations, the Jacobian matrices are obtained through Eqs. (41) and (42):

\[
J_q(\theta) = \begin{bmatrix}
-l_1 \sin \theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) & 0 \\
l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) & 0 \\
0 & 0 & 1
\end{bmatrix},
\]

\[
J_{\omega}(\theta) = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 1 & 0
\end{bmatrix}.
\]

3. Experimental setup

A rigid linear mechanism is preferred for the installation of the SCARA manipulator. This linear mechanism allows the robot arm to move up and down. The reason why this mechanism is preferred includes the ease of control provided by the stepper motor, precise feed steps, and handling load capacity. The effective range of motion of the mechanism on the horizontal axis is 275 mm. The accuracy is 0.05 mm by the applied quality ball screw. The horizontal movement speed in the loaded condition is 100 mm/s, and the maximum horizontal lift load is 10 kg. TB6600 motor driver and Arduino Uno control card are used for stepper motor control, which provides linear motion. In addition, the servo motor used in the gripper is also controlled by the Arduino control card. Figure 6 shows the linear mechanism used in the system and the stepper motor control connection scheme [24] used in the horizontal axis motion.

Two servo motors of Dynamixel AX12A [25] were used for rotary joints of the system. With many feedback functions, these servo motors have programmable

![Figure 6. Linear mechanism and stepper motor connection diagram.](image)
integrated infrastructure, a ready-made network connection system, reducers, ready-made joints, and easy mounting inserts. Motors have a constant torque rate of 1.5 Nm and a speed rate of 59 rpm. The gripping of the motor is accomplished with a small servo motor and the gripper mechanism. The control of Dynamixel servo motors in revolute joints is done with an OpenCM9.04-C control card with an ARM Cortex-M3 32-bit processor and OpenCM485 EXP expansion module. Figure 7 shows the control card, servo motor, and pin connections, and Figure 8 shows the experimental setup.

4. Control and Simulation

Due to the different types of motors used in the SCARA system, the control was done with Arduino and OpenCM9.04 control cards. All kinematic and other calculations used in robot motion were performed with the MATLAB program. The control cards are communicated through MATLAB software and, according to the values entered in the simulation screen, the robot moves at desired speeds. The objects in the working space were taken from their places and moved to the desired coordinates. Figure 9 shows the flow chart of the process until the SCARA manipulator at the home position picks the object from a certain coordinate and places it to the desired coordinate. Figure 10 shows the robot’s main control panel and kinematic calculation interface. Kinematic calculations and automatic and manual control operations are performed via this interface.

The trajectory planning is the planning of the movement of the robot according to the desired trajectory, velocity, acceleration, and time from the present position to the desired position of the end effector. It is desired for the robot to be able to move smoothly and vibrationless without exceeding the limits of the actuator and without crashing any object in the workspace. In the linear trajectory method, even if all robot joints with n degrees of freedom follow a linear trajectory, the end effector does not pass linearly between the two points. By adding parabolic parts to the beginning and end of the trajectory, the continuity of position and velocity is ensured. In addition, a smooth velocity by using a constant acceleration motion at a parabolic trajectory is also ensured. The linear trajectory plans added with parabolic parts for the rotary and linear joints of the robot are shown in Figure 11. The graphs show the displacement, speed, and acceleration values of each joint based on time.

The robot is automatically simulated in the MATLAB VRML [26] environment while simultaneously reaching the desired point through the automatic control panel. Thus, experimental studies can be monitored through the computer software interface. Figure 12 shows the automatic control panel and 3D simulation screen.

5. Conclusion

An academic study was carried out with an implemented experimental setup of the Prismatic-Revolute-Revolute-type (PRR) SCARA manipulator. A robot arm was produced that picked the product from any coordinate on the workspace and placed it to the desired coordinate. Numerous pages of codes were written in the MATLAB environment to perform the required calculations and control operations of the system. Forward and inverse kinematic calculations were solved. In addition, the results of the robot’s trajectory plans were obtained for all joints. The MATLAB program communicated with the control cards OpenCM9.04 and Arduino, and the robot was
Figure 9. Algorithm of the control system.
Figure 10. (a) Main control panel. (b) Kinematic calculations interface.

moved synchronously by the simulation software. Stepper motor in the prismatic joint and servo motors in other joints were used. Although different types of motors made it difficult to control the system, very successful results were achieved. In future studies, the motor powers can be increased to produce a commercial and industrial robot arm. Such a robot like this can be easily used in mass production lines in the industry, where picking and placing operations are done.
Figure 11. Trajectory planning results.

Figure 12. (a) 3D simulation screen. (b) Automatic control panel.
Acknowledgements
The author M. Soyaslan offers acknowledgement to The Scientific and Technological Research Council of Turkey (TUBITAK) for its 2211-C program support.

Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-H</td>
<td>Denavit-Hartenberg</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree Of Freedom</td>
</tr>
<tr>
<td>PRR</td>
<td>Prismatic-Revolute-Revolute</td>
</tr>
<tr>
<td>RRP</td>
<td>Revolute-Revolute-Prismatic</td>
</tr>
<tr>
<td>SCARA</td>
<td>Selective Compliance Assembly Robot</td>
</tr>
<tr>
<td>VRML</td>
<td>Virtual Reality Modelling Language</td>
</tr>
<tr>
<td>Nm</td>
<td>Newton metre</td>
</tr>
<tr>
<td>rpm</td>
<td>Revolutions per minute</td>
</tr>
<tr>
<td>mm</td>
<td>Millimetre</td>
</tr>
<tr>
<td>mm/s</td>
<td>Millimetre/second</td>
</tr>
<tr>
<td>N</td>
<td>Number of degree of freedom</td>
</tr>
<tr>
<td>1-1T</td>
<td>Transformation matrix between two joints</td>
</tr>
<tr>
<td>c</td>
<td>Cosine</td>
</tr>
<tr>
<td>s</td>
<td>Sine</td>
</tr>
<tr>
<td>d3</td>
<td>Length of the vertical joint</td>
</tr>
<tr>
<td>l1, l2</td>
<td>Horizontal lengths of joints</td>
</tr>
<tr>
<td>i</td>
<td>Joint number</td>
</tr>
<tr>
<td>αi-1</td>
<td>Angle between z_i-1 and z_i around x_i-1</td>
</tr>
<tr>
<td>a_i-1</td>
<td>Distance between axes z_i-1 and z_i throughout x_i-1</td>
</tr>
<tr>
<td>θi</td>
<td>Angle between x_i-1 and x_i around z_i</td>
</tr>
<tr>
<td>di</td>
<td>Distance between axes x_i-1 and x_i throughout z_i</td>
</tr>
<tr>
<td>P1, P2, P3</td>
<td>Position coordinates of the robot</td>
</tr>
<tr>
<td>v1, b1, b2</td>
<td>Auxiliary values for coordinates solutions</td>
</tr>
<tr>
<td>r_{max}</td>
<td>l_1 + l_2</td>
</tr>
<tr>
<td>r_{min}</td>
<td>l_1 - l_2</td>
</tr>
<tr>
<td>A tan 2</td>
<td>Arc tangent of the specified x and y coordinates</td>
</tr>
<tr>
<td>J_v(θ)</td>
<td>Jacobian matrix for linear velocity</td>
</tr>
<tr>
<td>J_ω(θ)</td>
<td>Jacobian matrix for angular velocity</td>
</tr>
<tr>
<td>φ1, φ2</td>
<td>Auxiliary values for analytical solutions</td>
</tr>
</tbody>
</table>

References


**Biographies**

**Muhammed Enes Uk** graduated as salutatorian in Mechatronics Engineering from Sakarya University, Turkey in 2017. He is currently working in GUI design and model-in-the-loop testing of vehicle air conditioning systems at Santor A.Ş. (Cooperation of Sherpa Engineering and Figes A.Ş.). His research interests include robotics, control, and GUI design.

**Faris Bin Sajjad Ali Shah** was born in Pakistan, 1994. He received a Turkish Government Scholarship for BSc degree in Mechatronics Engineering from Sakarya University, Turkey and graduated in 2017. He is currently a Robotician and Automation Engineer in TARA Robotics Automation Company. He is an expert on ABB Industrial Robots including Palletizition, Material Handling, and Arc Welding Applications. His research interests include robotics, programming, designing, modeling, and simulation.

**Mucalııt Soyaslan** received BSc and MSc degrees in Mechatronics Engineering from Kocaeli University and Gaziosmanpasa University, Turkey in 2010 and 2012, respectively. He is currently a Research Assistant at the Mechatronics Engineering Department of Sakarya University, Turkey and is working towards PhD degree. His research interests include electrical machine design, robotics, and AS/RS systems.

**Osman Eldogan** received BSc, MSc, and PhD degrees in Machine Engineering from Istanbul Technical University, Selcuk University and Marmara University, Turkey in 1984, 1988, and 1994, respectively. He is currently a Professor and the Head of the Mechatronics Engineering Department of Sakarya University, Turkey. His research interests include machine dynamics, mechanism technique, and vehiæze technology.