Scientia Iranica B (2020) 27(1), 330–340

SCIENTIA
IRANICA

Sharif University of Technology
Scientia Iranica
Transactions B: Mechanical Engineering
http://scientiairanica.sharif.edu

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.

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

*. *Corresponding author. Tel.: +90 2642956912;*
Fax: +90 2642956424
E-mail addresses: muhammed.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)

doi: 10.24200/sci.2018.51214.2065

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, Alshamasin 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 Korayem et al. [9] designed a program for simulating and animating the 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-Revolute) 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 kinematic 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 ${}^{i-1}_{\;\;i}T$. 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):

$$ {}^0_NT = {}^0_1T\,{}^1_2T\,{}^2_3T\,{}^3_4T \cdots {}^{N-1}_{\;\;N}T. \tag{1} $$

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

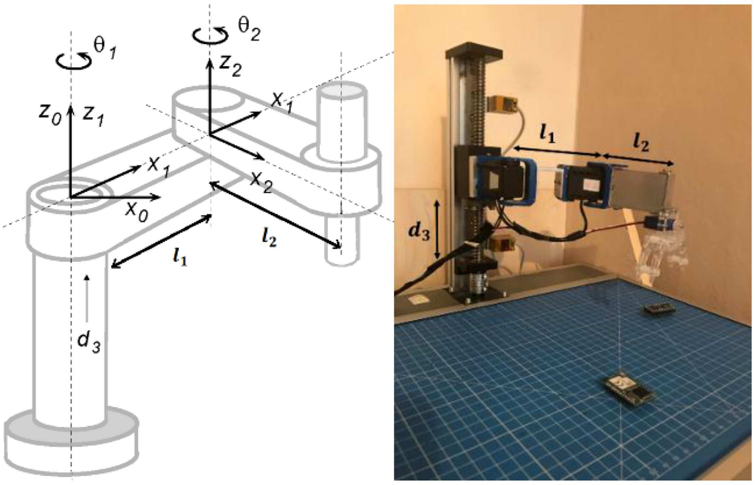


Figure 1. Joint axes and implemented SCARA robot.

Table 1. $D-H$ parameters.

i	α_{i-1} [a]	a_{i-1} [b]	θ_i [c]	d_i [d]
1	0	0	0	d_3
2	0	0	θ_1	0
3	0	l_1	θ_2	0
4	0	l_2	0	0

[a] α_{i-1}: Angle between z_{i-1} and z_i around x_{i-1};

[b] a_{i-1} : Distance between axes z_{i-1} and z_i throughout x_{i-1};

[c] θ_i : Angle between x_{i-1} and x_i around z_i;

[d] 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:

$$ {}^0_1T=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&d_3\\0&0&0&1\end{bmatrix}, \quad {}^1_2T=\begin{bmatrix}c_1&-s_1&0&l_1\\s_1&c_1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}, $$

$$ {}^2_3T=\begin{bmatrix}c_2&-s_2&0&l_2\\s_2&c_2&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}. \tag{2} $$

The abbreviations c and s represent the terms "*cosine*" and "*sine*". 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.

$$ {}^0_3T=\begin{bmatrix}c_{12}&-s_{12}&0&c_1l_1+c_{12}l_2\\s_{12}&c_{12}&0&s_1l_1+s_{12}l_2\\0&0&1&d_3\\0&0&0&1\end{bmatrix}. \tag{3} $$

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.

$$ O=P=\begin{bmatrix}c_1l_1+c_{12}l_2\\s_1l_1+s_{12}l_2\\d_3\\1\end{bmatrix}=\begin{bmatrix}P_1\\P_2\\P_3\\1\end{bmatrix}. \tag{4} $$

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

$$ P_1=c_1l_1+c_{12}l_2, \tag{5} $$

$$ P_2=s_1l_1+s_{12}l_2, \tag{6} $$

$$ P_3=d_3. \tag{7} $$

2.2.1. *Calculation of θ_1 angle*

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

$$ (c_{12}l_2)^2=(P_1-c_1l_1)^2, \tag{8} $$

$$ (s_{12}l_2)^2=(P_2-s_1l_1)^2. \tag{9} $$

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

$$ l_2^2=P_1^2-P_2^2+l_1^2-2l_1(P_1c_1+P_2s_1). \tag{10} $$

Eq. (11) is obtained when b_1 is used instead of $(P_1c_1+P_2s_1)$:

$$ l_2^2=P_1^2-P_2^2+l_1^2-2l_1b_1. \tag{11} $$

Figure 2 shows the θ_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:

$$ b_1=(P_1c_1+P_2s_1), \tag{12} $$

$$ v_1=(-P_1s_1+P_2c_1), \tag{13} $$

$$ r^2=v_1^2+b_1^2, \tag{14} $$

$$ r^2=P_1^2+P_2^2, \tag{15} $$

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

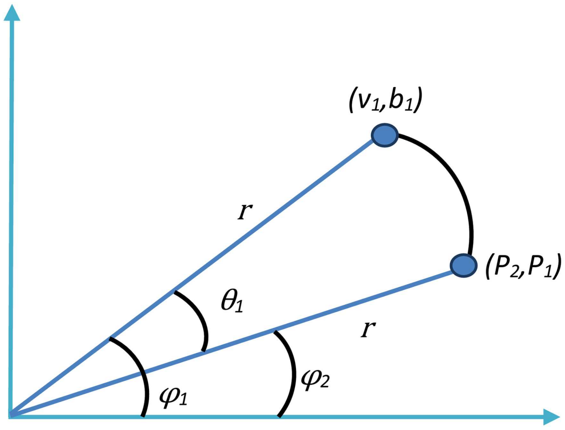


Figure 2. Derivation illustration of θ_1.

$$\theta_1 = \varphi_1 - \varphi_2, \tag{17}$$

$$\varphi_1 = A\tan 2(v_1, b_1), \tag{18}$$

$$\varphi_2 = A\tan 2(P_2, P_1). \tag{19}$$

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 θ_1. This shows that there are two solutions.

$$\theta_1 = A\tan 2(\pm v_1, b_1) - A\tan 2(P_2, P_1). \tag{20}$$

2.2.2. Calculation of θ_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:

$$P_1[c_{12}l_2 = P_1 - c_1 l_1], \tag{21}$$

$$P_2[s_{12}l_2 = P_2 - s_1 l_1]. \tag{22}$$

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

$$l_2[c_{12}P_2 - s_{12}P_1] = -l_1[P_2 c_1 - P_1 s_1]. \tag{23}$$

When v_1 is written instead of $P_2c_1 - P_1s_1$, we get:

$$c_{12}P_2 - s_{12}P_1 = \frac{-l_1 v_1}{l_2}. \tag{24}$$

Method 2:

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

$$l_2[c_{12}P_2 + s_{12}P_1] = P_1P_2 - l_1[P_2c_1 + P_1s_1]. \tag{25}$$

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

$$b_2 = c_{12}P_2 + s_{12}P_1. \tag{26}$$

b_1 is rearranged as follows:

$$b_1 = (P_1c_1 + P_2s_1) = \frac{P_1^2 + P_2^2 + l_1^2 - l_2^2}{2l_1}. \tag{27}$$

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

$$c_{12}P_1 + s_{12}P_2 = \frac{P_1^2 + P_2^2 - l_1^2 + l_2^2}{2}. \tag{28}$$

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

$$b_2 = \frac{P_1^2 + P_2^2 - l_1^2 + l_2^2}{2l_1}. \tag{29}$$

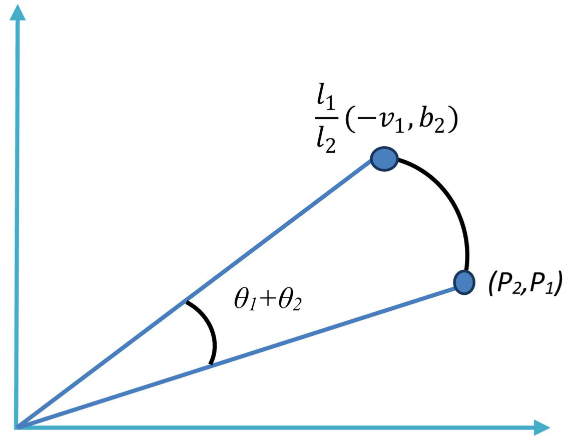


Figure 3. Derivation illustration of $(\theta_1 + \theta_2)$.

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

$$c_{12}P_1 + s_{12}P_2 = \frac{l_1 b_2}{l_2}. \tag{30}$$

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

Figure (3) can be solved as follows:

$$\theta_1 + \theta_2 = A\tan 2(-v_1, b_2) - A\tan 2(P_2, P_1). \tag{31}$$

When θ_1 value from Eq. (20) is placed in Eq. (31), Eq. (32) is obtained.

$$\theta_2 = A\tan 2(-v_1, b_2) - A\tan 2(v_1, b_1). \tag{32}$$

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:

$$\theta_2^{(1)} = 2A\tan 2(-v_1, b_1), \tag{33}$$

$$\theta_1^{(1)} = A\tan 2(P_2, P_1) - \frac{1}{2}\theta_2^{(1)}. \tag{34}$$

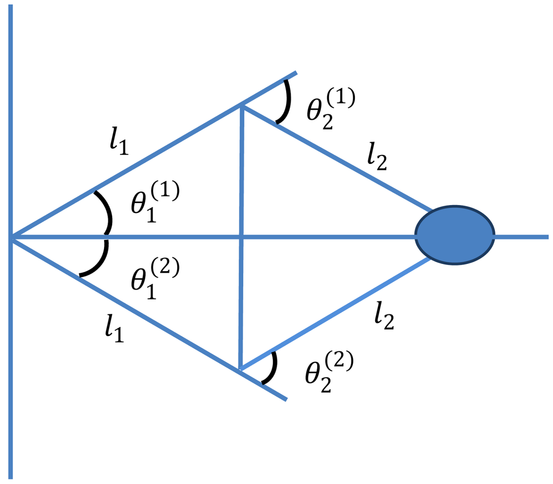


Figure 4. Two solutions of object orientation.

- Right side solution:

$$\theta_2^{(2)} = 2A\tan 2(v_1, b_1), \tag{35}$$

$$\theta_1^{(2)} = -A\tan 2(P_2, P_1) - \frac{1}{2}\theta_2^{(2)}. \tag{36}$$

2.3.1. Examination of the solution's existence

v_1 value from Eq. (16) can be written as follows:

$$v_1 = \pm\sqrt{r^2 - b_1^2}, \tag{37}$$

$$v_1 = \pm\sqrt{(r-b_1)(r+b_1)}. \tag{38}$$

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:

$$r_{\max} = l_1 + l_2, \tag{39}$$

$$r_{\min} = l_1 - l_2. \tag{40}$$

The existence of solutions is available under the above conditions.

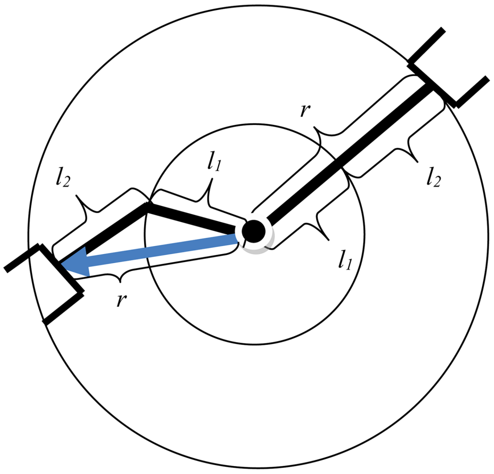


Figure 5. Maximum and minimum points achievable by the arms.

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):

$$J_v(\theta)=\begin{bmatrix} -l_1\sin\theta_1 - l_2\sin(\theta_1+\theta_2) & -l_2\sin(\theta_1+\theta_2) & 0 \\ l_1\cos\theta_1 + l_2\cos(\theta_1+\theta_2) & l_2\cos(\theta_1+\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{41}$$

$$J_w(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}. \tag{42}$$

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. TB 6600 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 feed back functions, these servo motors have programmable

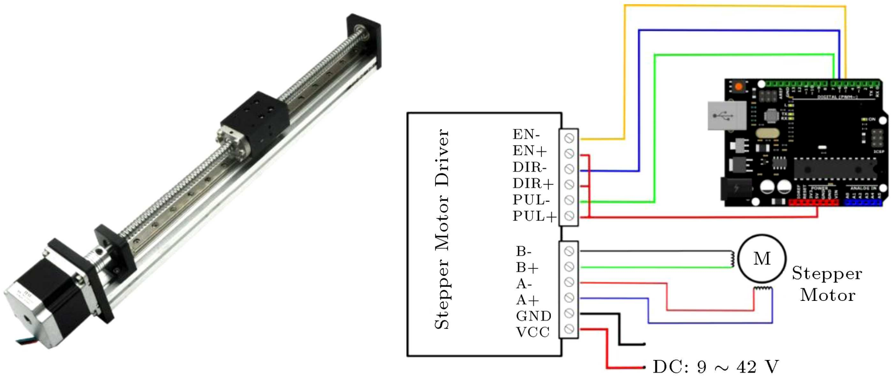


Figure 6. Linear mechanism and stepper motor connection diagram.

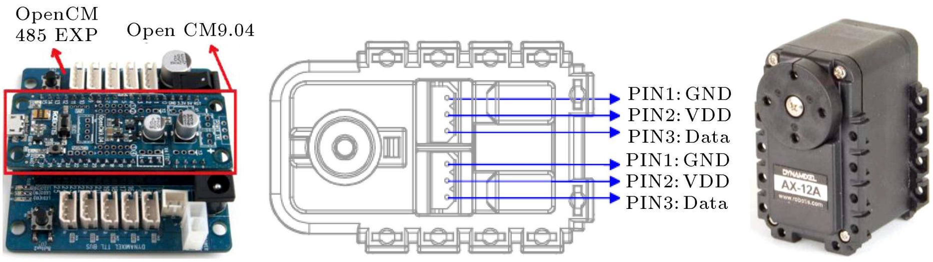


Figure 7. Open CM9.04 and 485 exp. control cards, servo motor, and pin connections.

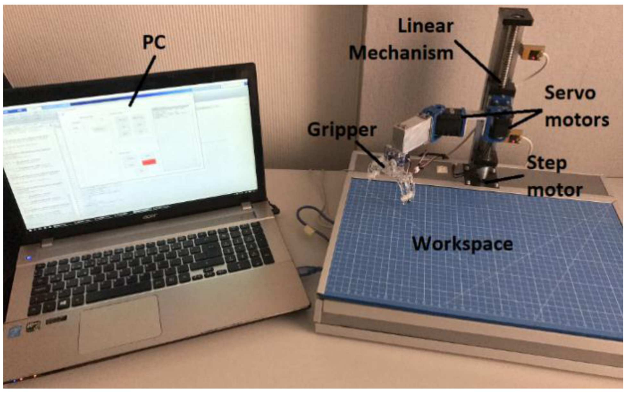


Figure 8. Experimental setup.

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

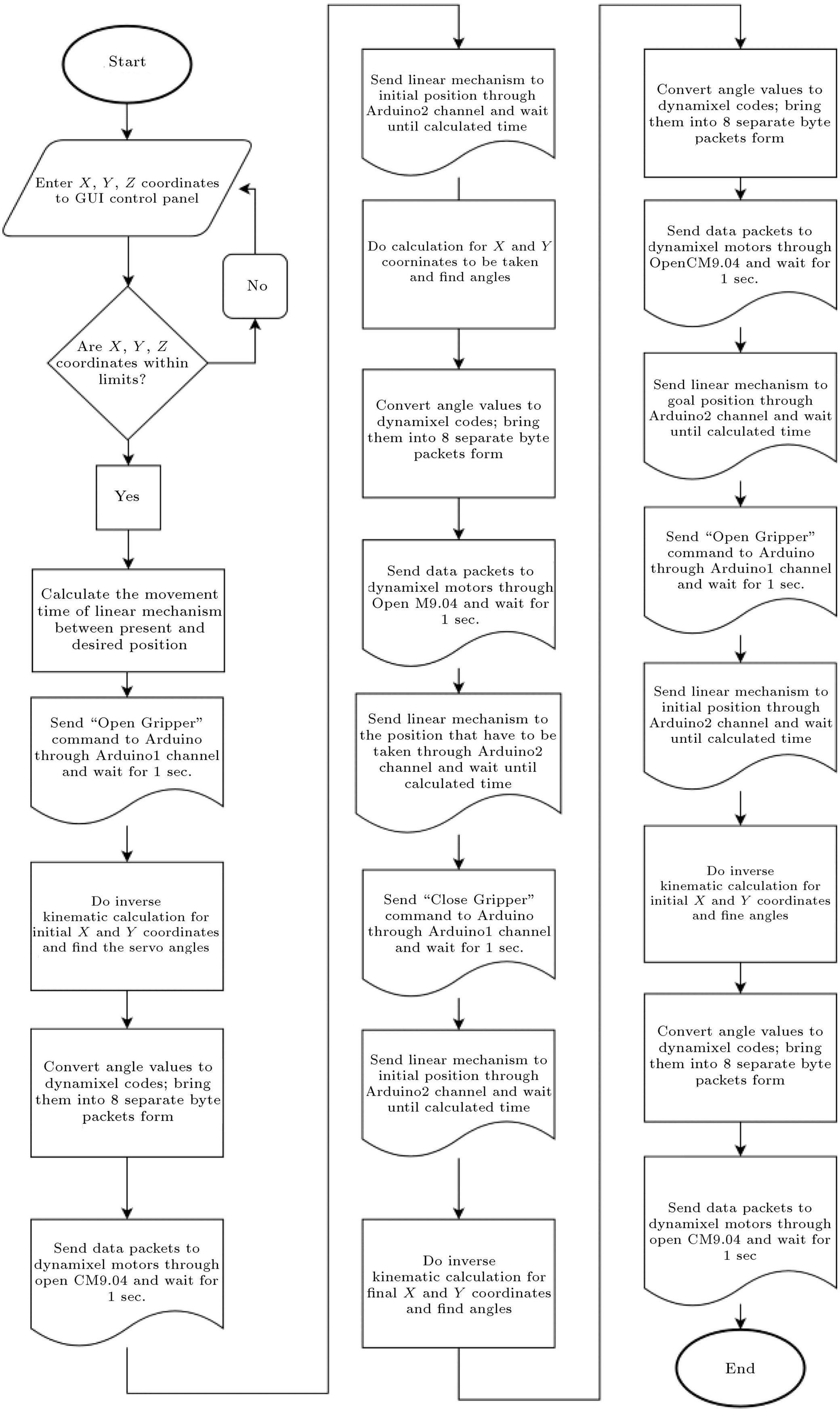
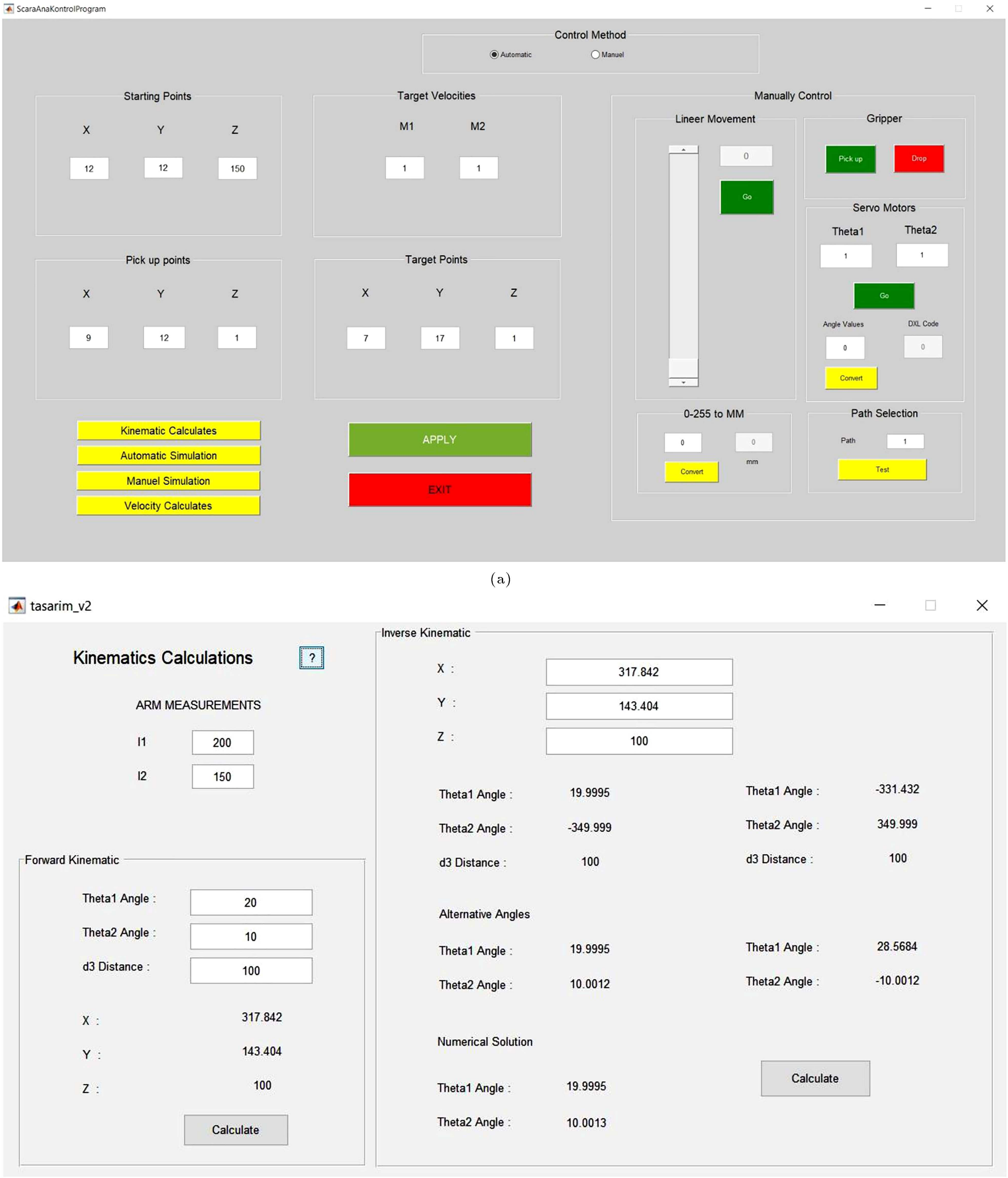

Start
Enter X, Y, Z coordinates to GUI control panel
Are X, Y, Z coordinates within limits?
No
Yes
Calculate the movement time of linear mechanism between present and desired position
Send "Open Gripper" command to Arduino through Arduino1 channel and wait for 1 sec.
Do inverse kinematic calculation for initial X and Y coordinates and find the servo angles
Convert angle values to dynamixel codes; bring them into 8 separate byte packets form
Send data packets to dynamixel motors through open CM9.04 and wait for 1 sec.
Send linear mechanism to initial position through Arduino2 channel and wait until calculated time
Do calculation for X and Y coorninates to be taken and find angles
Convert angle values to dynamixel codes; bring them into 8 separate byte packets form
Send data packets to dynamixel motors through Open M9.04 and wait for 1 sec.
Send linear mechanism to the position that have to be taken through Arduino2 channel and wait until calculated time
Send "Close Gripper" command to Arduino through Arduino1 channel and wait for 1 sec.
Send linear mechanism to initial position through Arduino2 channel and wait until calculated time
Do inverse kinematic calculation for final X and Y coordinates and find angles
Convert angle values to dynamixel codes; bring them into 8 separate byte packets form
Send data packets to dynamixel motors through OpenCM9.04 and wait for 1 sec.
Send linear mechanism to goal position through Arduino2 channel and wait until calculated time
Send "Open Gripper" command to Arduino through Arduino1 channel and wait for 1 sec.
Send linear mechanism to initial position through Arduino2 channel and wait until calculated time
Do inverse kinematic calculation for initial X and Y coordinates and fine angles
Convert angle values to dynamixel codes; bring them into 8 separate byte packets form
Send data packets to dynamixel motors through open CM9.04 and wait for 1 sec
End

Figure 9. Algorithm of the control system .



ScaraAnaKontrolProgram

Control Method: ● Automatic ○ Manuel

Starting Points: X 12, Y 12, Z 150

Target Velocities: M1 1, M2 1

Pick up points: X 9, Y 12, Z 1

Target Points: X 7, Y 17, Z 1

Manually Control

Lineer Movement: 0

Servo Motors: Theta1 1, Theta2 1

Angle Values 0, DXL Code 0

0-255 to MM: 0, 0 mm

Path Selection: Path 1

(a)

tasarim_v2

Kinematics Calculations

ARM MEASUREMENTS

l1 200

l2 150

Forward Kinematic

Theta1 Angle : 20

Theta2 Angle : 10

d3 Distance : 100

X : 317.842

Y : 143.404

Z : 100

Inverse Kinematic

X : 317.842

Y : 143.404

Z : 100

Theta1 Angle :	19.9995	Theta1 Angle :	-331.432
Theta2 Angle :	-349.999	Theta2 Angle :	349.999
d3 Distance :	100	d3 Distance :	100

Alternative Angles

Theta1 Angle :	19.9995	Theta1 Angle :	28.5684
Theta2 Angle :	10.0012	Theta2 Angle :	-10.0012

Numerical Solution

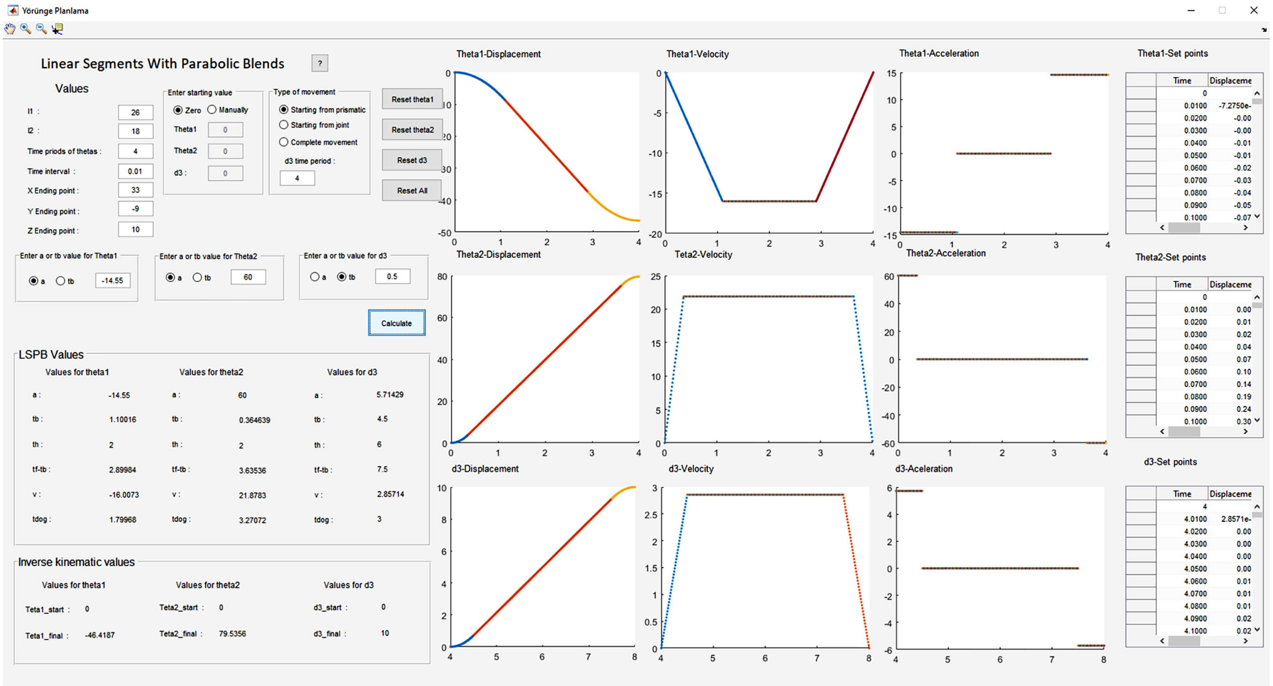
Theta1 Angle : 19.9995

Theta2 Angle : 10.0013

(b)

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.



Linear Segments With Parabolic Blends

Values

l1 : 26
l2 : 18
Time priods of thetas : 4
Time interval : 0.01
X Ending point : 33
Y Ending point : -9
Z Ending point : 10

Enter starting value
● Zero ○ Manually
Theta1 0
Theta2 0
d3 : 0

Type of movement
● Starting from prismatic
○ Starting from joint
○ Complete movement
d3 time period : 4

Enter a or tb value for Theta1
● a ○ tb -14.55

Enter a or tb value for Theta2
● a ○ tb 60

Enter a or tb value for d3
○ a ● tb 0.5

LSPB Values

Values for theta1		Values for theta2		Values for d3	
a :	-14.55	a :	60	a :	5.71429
tb :	1.10016	tb :	0.364639	tb :	4.5
th :	2	th :	2	th :	6
tf-tb :	2.89984	tf-tb :	3.63536	tf-tb :	7.5
v :	-16.0073	v :	21.8783	v :	2.85714
tdog :	1.79968	tdog :	3.27072	tdog :	3

Inverse kinematic values

Values for theta1		Values for theta2		Values for d3	
Teta1_start :	0	Teta2_start :	0	d3_start :	0
Teta1_final :	-46.4187	Teta2_final :	79.5356	d3_final :	10

Theta1-Set points

Time	Displaceme
0	
0.0100	-7.2750e-
0.0200	-0.00
0.0300	-0.00
0.0400	-0.01
0.0500	-0.01
0.0600	-0.02
0.0700	-0.03
0.0800	-0.04
0.0900	-0.05
0.1000	-0.07

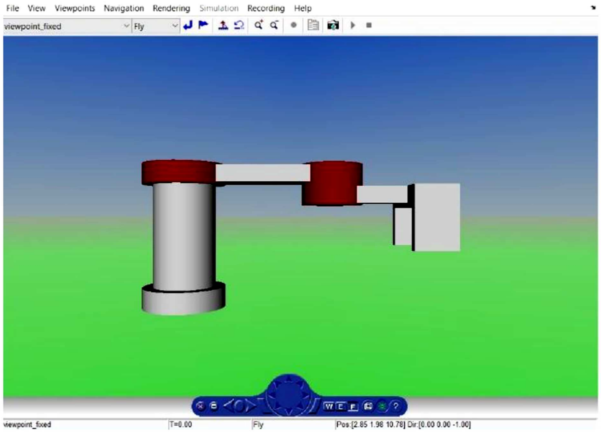
Theta2-Set points

Time	Displaceme
0	
0.0100	0.00
0.0200	0.01
0.0300	0.02
0.0400	0.04
0.0500	0.07
0.0600	0.10
0.0700	0.14
0.0800	0.19
0.0900	0.24
0.1000	0.30

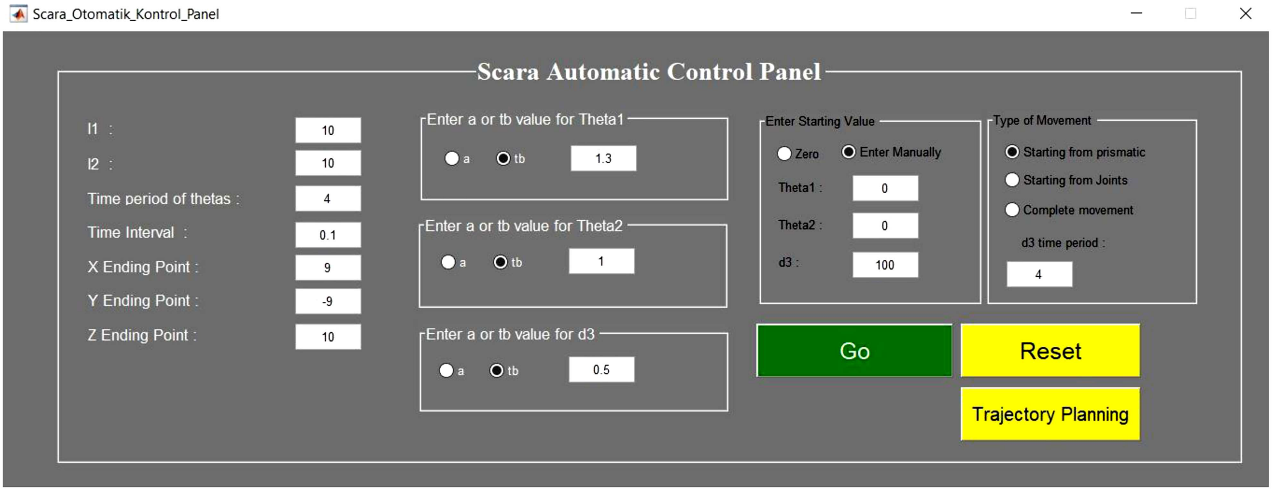
d3-Set points

Time	Displaceme
4	
4.0100	2.8571e-
4.0200	0.00
4.0300	0.00
4.0400	0.00
4.0500	0.00
4.0600	0.01
4.0700	0.01
4.0800	0.01
4.0900	0.02
4.1000	0.02

Figure 11. Trajectory planning results.



(a)



Scara Automatic Control Panel

l1 : 10
l2 : 10
Time period of thetas : 4
Time Interval : 0.1
X Ending Point : 9
Y Ending Point : -9
Z Ending Point : 10

Enter a or tb value for Theta1
○ a ● tb 1.3

Enter a or tb value for Theta2
○ a ● tb 1

Enter a or tb value for d3
○ a ● tb 0.5

Enter Starting Value
○ Zero ● Enter Manually
Theta1 : 0
Theta2 : 0
d3 : 100

Type of Movement
● Starting from prismatic
○ Starting from Joints
○ Complete movement
d3 time period : 4

(b)

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

D-H	Denavit-Hartenberg
DOF	Degree Of Freedom
PRR	Prismatic-Revolute-Revolute
RRP	Revolute-Revolute-Prismatic
SCARA	Selective Compliance Assembly Robot Arm
VRML	Virtual Reality Modelling Language
Nm	Newton metre
rpm	Revolutions per minute
mm	Millimetre
mm/s	Millimetre/second
N	Number of degrees of freedom
${}^{i-1}_{\ \ i}T$	Transformation matrix between two joints
c	Cosine
s	Sine
d_3	Length of the vertical joint
l_1, l_2	Horizontal lengths of joints
i	Joint number
α_{i-1}	Angle between z_{i-1} and z_i around x_{i-1}
a_{i-1}	Distance between axes z_{i-1} and z_i throughout x_{i-1}
θ_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
P_1, P_2, P_3	Position coordinates of the robot
v_1, b_1, b_2	Auxiliary values for coordinates solutions
$r_{\max}$	$l_1 + l_2$
$r_{\min}$	$l_1 - l_2$
$A\tan 2$	Arc tangent of the specified x and y coordinates
$J_v(\theta)$	Jacobian matrix for linear velocity
$J_w(\theta)$	Jacobian matrix for angular velocity
φ_1, φ_2	Auxiliary values for analytical solutions

References

1. Soyaslan, M., Fenercioglu, A., and Kozkurt, C. "A new truck based order picking model for automated storage and retrieval system (AS/RS)", *Journal of Engineering Research*, **5**(4), pp. 169–194 (2017).
2. Soyaslan, M., Kozkurt, C., and Fenercioglu, A. "Automated Storage and Retrieval Systems (ASRS): Research on warehouse configuration and performance studies", *Academic Platform Journal of Engineering and Science - APJES*, **3**(3), pp. 8–26 (2015).
3. Robot Hall of Fame, "Inductees-SCARA", Carnegie Mellon University (2006). http://www. robothalloffame.org/inductees/06inductees /scara.html
4. Visioli, A. and Legnani, G. "On the trajectory tracking control of industrial SCARA robot manipulators", *IEEE Transactions on Industrial Electronics*, **49**(1), pp. 224–232 (2002).
5. Das, M.T. and Dulger, L.C. "Mathematical modelling, simulation and experimental verification of a SCARA robot", *Simulation Modelling Practice and Theory*, **13**(3), pp. 257–271 (2005).
6. Alshamasin, M.S., Ionescu, F., and Al-Kasasbeh, R.T. "Kinematic modelling and simulation of a scara robot by using solid dynamics and verification by Matlab/Simulink", *European Journal of Scientific Research*, **37**(3), pp. 388–405 (2009).
7. Urrea, C. and Kern, J. "Modelling, simulation and control of a redundant SCARA-type manipulator robot", *International Journal of Advanced Robotic Systems*, **9**(2), p. 58 (2012).
8. Kaleli, A., Dumlu, A., Çorapsız, M.F., and Erenturk, K. "Detailed analysis of SCARA-type serial manipulator on a moving base with LabVIEW", *International Journal of Advanced Robotic Systems*, **10**(4), p. 189 (2013).
9. Korayem, M.H., Yousefzadeh, M., and Manteghi, S. "Tracking control and vibration reduction of flexible cable-suspended parallel robots using a robust input shaper", *Scientia Iranica B*, **25**(1), pp. 230–252 (2018).
10. Kozkurt, C. and Soyaslan, M. "Software development for kinematic analysis of scara robot arm with Euler wrist", *6th International Advanced Technologies Symposium (IATS'11)*, Elazıg, Turkey, pp. 27–32 (2011).
11. Kucuk, S. and Bingul, Z. "An off-line robot simulation toolbox", *Computer Applications in Engineering Education*, **18**(1), pp. 41–52 (2009).
12. Adar, N.G. and Kozan, R. "Comparison between real time PID and 2-DOF PID controller for 6-DOF robot arm", *Acta Phys. Pol. A*, **130**(1), pp. 269–271 (2016).
13. Adar, N.G., Tiryaki, A.E., and Kozan, R. "Real time visual servoing of a 6-DOF robotic arm using Fuzzy-PID controller", *Acta Phys. Pol. A*, **128**(2B), pp. 348–351 (2015).
14. Saygın, A. and Rashid, A.M. "Position control of a turret using LabVIEW", *Acta Phys. Pol. A*, **132**(3-II), pp. 970–973 (2017).
15. Karayel, D. and Yegin, V. "Design and prototype manufacturing of a torque measurement system", *Acta Phys. Pol. A*, **130**(1), pp. 272–275 (2016).
16. Fenercioglu, A., Soyaslan, M., and Kozkurt, C. "Automatic storage and retrieval system (AS/RS) based

on Cartesian robot for liquid food industry", *12th International Workshop on Research and Education in Mechatronics*, Kocaeli, Turkey, pp. 283–287 (2011).

17. Korayem, M.H., Maddah, S.M., Taherifar, M., et al. "Design and programming a 3D simulator and controlling graphical user interface of ICaSbot, a cable suspended robot", *Scientia Iranica B*, **21**(3), pp. 663–681 (2014).

18. Sayyaadi, H. and Eftekharian, A.A. "Modeling and intelligent control of a robotic gas metal arc welding system", *Scientia Iranica*, **15**(1), pp. 75–93 (2008).

19. Gulzar, M.M., Murtaza, A.F., Ling, Q., et al. "Kinematic modeling and simulation of an economical scara manipulator by Pro-E and verification using MATLAB/Simulink", *IEEE International Conference on Open Source Systems & Technologies (ICOSST)*, pp. 102–107 (2015).

20. Ibrahim, B.S.K.K. and Zargoun, A.M.A. "Modelling and control of SCARA manipulator", *Procedia Computer Science*, **42**, pp. 106–113 (2014).

21. Urrea, C., Cortés, J., and Pascal, J. "Design, construction and control of a SCARA manipulator with 6 degrees of freedom", *Journal of Applied Research and Technology*, **14**(6), pp. 396–404 (2016).

22. Denavit, J. and Hartenberg, R.S. "A kinematic notation for lower-pair mechanisms based on matrices", *ASME J. Appl. Mechan.*, **77**(2), pp. 215–221 (1955).

23. Bingul, Z. and Kucuk, S. "İleri kinematik, ters kinematik", In *Robot Teknigi I*, pp. 104–200, Birsen Yayınevi, Turkey (2005).

24. TB6600 Stepper Motor Driver (2017). https://www.dfrobot.com/product-1547.html

25. Dynamixel-All in one actuator, Robotis Inc (2014). http://www. robotis.us/dynamixel/

26. Virtual Reality Modeling Language (VRML) - MATLAB & Simulink (2017). https://www.mathworks.com/help/sl3d/vrml.html

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 Roboticist and Automation Engineer in TARA Robotics Automation Company. He is an expert on ABB Industrial Robots including Palletizing, Material Handling, and Arc Welding Applications. His research interests include robotics, programming, designing, modeling, and simulation.

Mucahit 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 vehiazzcle technology.