Engineering Applications of Artificial Intelligence 122 (2023) 106099

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Trajectory planning for a 6-axis robotic arm with particle swarm optimization algorithm

Özge Ekrem, Bekir Aksoy *

Department of Mechatronics Engineering, Isparta University of Applied Sciences, Isparta, Turkey

ARTICLE INFO

Keywords:
Robotic arm
Robot optimization
PSO algorithm
Matlab
Trajectory planning
Cartesian space
Joint space

ABSTRACT

Robotic arms, which are favored for usage in both large- and small-scale industrial regions, run into issues with numerous limits between the starting and ending locations in the working space when attempting to complete a particular task. With the solution to one of these problems, trajectory planning, the robotic arm manipulator can move from the starting point to the target point vibration-free, without hitting obstacles, and by choosing the shortest way. In this study, a robotic arm with 6 degrees of freedom, which is in the Mechatronics Engineering laboratory of Isparta University of Applied Sciences and whose prototype was realized by Acrome company, was used. In the study, the trajectory planning of the robotic arm was carried out using the MATLAB program and particle swarm optimization (PSO). Trajectory planning is developed using the PSO algorithm to determine the position of the robot at each point as it moves from its starting point to its target. Thus, time optimization was achieved by choosing the shortest path between the two points. Trajectory planning in joint space is aimed to ensure that the position, speed, and acceleration between the starting and ending points are continuous by using the fifth-order polynomial. The instant values of the joint variables used to determine the points followed by the manipulator were obtained by forward kinematics through the MATLAB program. Using forward kinematics, the position information of the manipulator was obtained by providing a transition from joint space to Cartesian space.

1. Introduction

Today, with the rapid development of industrial automation, robotic systems are gaining a prominent place in production systems and our daily lives (Ağralı and Çavaş, 2020). In the age of the intelligent manufacturing industry, the controlled operation of robotic manipulators in different tasks and stages such as production efficiency, error-free assembly accuracy, and vibration-free and barrier-free transportation of materials ensures time optimization and increases the working potential (Huang et al., 2019). Robotic manipulators consist of a group of rigid arms connected by rotating or prismatic joints that can be programmed and self-controlled in three or more axes, which can be used to move any object from one point to another (Tonbul and Sarıtaş, 2003; Sadiq and Raheem, 2017).

The widespread use of robotic manipulators in many areas today creates the problem of trajectory control and necessitates the necessary trajectory planning. Transporting the manipulator from the starting point to the desired endpoint in a smooth and controlled manner without vibration, and without hitting any obstacles constitutes the purpose of trajectory planning (Uzuner et al., 2017). Trajectory planning is especially important in the workspace to have more than one path option between the start and end points and to determine the path with the shortest distance among them (Beşkirli and Tefek, 2019).

Trajectory planning in robotic manipulators is carried out with two approaches: joint space and Cartesian space (Ma et al., 2021). When trajectory planning is carried out with Cartesian space, linear movement is provided between the starting and target positions of the manipulator. On the defined linear path, the positions and speeds of the robot manipulator at the intermediate points defined at close distances are determined (Ergür, 2015). In trajectory planning in joint space, the movement of the end manipulator from the starting point to the target point occurs at a certain time interval, and three or higher degree polynomials are used. By using polynomial functions, the position, speed, and acceleration parameters of the joint are ensured to be continuous, and a smooth joint trajectory can be obtained (Uzuner et al., 2017).

Determination of the intermediate points between the starting and ending point; The speed of the robot is determined by the parameters of acceleration and distance, which poses a difficulty depending on the degree of freedom of the robot (Tonbul and Sarıtaş, 2003). In recent years, with the studies carried out in the field of robotics, trajectory

* Corresponding author.
E-mail address: bekiraksoy@isparta.edu.tr (B. Aksoy).

https://doi.org/10.1016/j.engappai.2023.106099
Received 8 November 2022; Received in revised form 11 February 2023; Accepted 3 March 2023
Available online 15 March 2023
0952-1976/

planning problems; Heuristic optimization methods such as genetic algorithm, ant colony algorithm, artificial bee colony, and particle swarm optimization can produce solutions (Savsani et al., 2014). In the study, the PSO algorithm is preferred because it allows for up-to-date multi-information exchange, reduces the computational load, and mapping, avoids obstacles, going to the desired destination in the fastest, shortest, and most convenient position (Gürgüze and Türkoğlu, 2019; Pires et al., 2006), Compared to algorithms such as genetic algorithm (GA) and simulated annealing (SA), being more understandable and easier to implement (Kim and Lee, 2015), having fewer parameters (Wang et al., 2016).

Within the scope of this study, the technical information of the 6-degrees-of-freedom robot arm, which is in the Mechatronics Engineering laboratory of Isparta University of Applied Sciences and prototyped by Acrome, was used. For the robot manipulator to reach the target point from the starting point in a controlled manner, trajectory planning was carried out in the joint space. Continuity with fifth-order polynomial interpolation and trajectory control is provided using the PSO algorithm. The robotic manipulator is simulated in MATLAB programming language. Various obstacles such as circular and rectangular were placed in the simulation and the trajectories formed by the manipulator in the presence of these obstacles were examined depending on the number of iterations and the RNG parameter. When the graphs obtained were examined, it was observed that the PSO algorithm gave effective results in the trajectories created between the starting and target points of the 6-axis robotic arm.

2. Related studies

When the literature is examined, the PSO algorithm is among the most preferred heuristic algorithms in various application areas such as artificial intelligence and robotics (Zhang et al., 2015). In the study, the studies in the literature for the PSO algorithm used to solve the trajectory planning problem were examined. In line with the studies examined, it was observed that the PSO algorithm gave effective results in trajectory planning for robotic systems, and in this study, trajectory planning was carried out by using the PSO algorithm for the 6-axis robotic arm. Doctor performed a collective robotic search process by providing single and multiple target searches with the PSO algorithm. As a result of the study, it has been proven that PSO is very reliable in target search applications (Doctor et al., 2004). Min et al., proposed a method based on a quadratic motion model and a multi-object optimization algorithm. They came up with a model based on the PSO algorithm for robots to avoid obstacles in the dynamic environment. In the simulation experiment, it was observed that the method based on the PSO algorithm gave better results than the artificial potential field algorithm (APF) and the genetic algorithm (Min et al., 2005). In their study, Chen and Li used the PSO algorithm to solve the path planning problem of the mobile robot in an environment where static obstacles exist and obtained an effective result (Chen and Li, 2006). Nasrollahy and Javadi, proposed a PSO-based path planning method for the mobile robot that can reach the moving target position in the shortest way over time and avoids local optimums in an environment where both static and dynamic objects are located. The accuracy and demonstration of the proposed method were simulated on the MATLAB application and they found that it performed better compared to similar studies (Nasrollahy and Javadi, 2009). Masehian and Sedighizadeh, presented a robotic motion planning algorithm that handles two target points from the shortest route without vibration using the PSO algorithm. In the study, the PSO algorithm was used as a global planner and the Probabilistic Roadmap Method (PRM) was used as a local planner. As a result of the study, it was obtained that the combination of PSO+PRM was 50% faster than the classical PRM method (Masehian and Sedighizadeh, 2010). In their study, Zhang et al., proposed a multi-purpose road planning algorithm based on the PSO, which is used for situations such as fire, mines, and enemies on the battlefield, for rescue tasks of robots. As a result of the study, it was concluded that the algorithm performed well in uncertain dangerous situations (Zhang et al., 2013). In their study, Rath, and Deepak, used the mobile robot's PSO algorithm to propose a new fitness function that considers parameters such as obstacle-robot, robot-target and distance between robot-robot in an unknown dynamic environment where both static and dynamic obstacles exist (Rath and Deepak, 2015). Wang et al., proposed a PSO algorithm in which the suitability value is based on the length of the path to perform the optimal path planning of the robot in an unknown environment. In the proposed method, they found that the robot effectively escaped all obstacles and achieved the optimum path (Wang et al., 2017). Xin et al., have ensured that the vibration level is reduced to a minimum for the space manipulator modeled using the PSO algorithm. The proposed method was processed through the simulation of a 3-axis planar manipulator and was observed to give effective results (Xin et al., 2017). In their study, Han et al., used the PSO algorithm to perform the trajectory planning of the mobile robot. In experiments on simulation, they concluded that the PSO algorithm does not always give the most appropriate way and that the study can be improved (Han et al., 2019). In their study, Ulusoy and Güneş designed a mobile robot and controlled the control of the designed mobile robot in two ways: classic PID controller and PSO algorithm + PID controller. When they compared the results observed, they found that the PSO+PID controller performed better than the classic PID (Ulusoy and Güneş, 2019). Beşkirli and Tefek simulated the orbital planning process using the PSO algorithm. In the study, circle-shaped obstacles were used and the mathematical formula of the distance between a point and a line was used to find the distance between the starting and target points. As a result of the study, the shortest calculations of the robot path for three different situations were obtained and it was shown that the PSO algorithm was applicable for robot path planning (Beşkirli and Tefek, 2019). Lopez-Franco et al. in their study, to solve the inverse kinematics problem in the path tracing of robot manipulators: Artificial Bee Colony (ABC), Bat Algorithm (BA), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), Cuckoo Search (CS), Differential Evolution (DE), Differential Search (DS), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) have used different artificial intelligence techniques and observed that the CMA-ES algorithm gives the best results in solving the problem (Lopez-Franco et al., 2018). In most cases (especially when dealing with large-scale problems), meta-heuristic methods are considered one of the most effective alternatives for searching and determining a near-optimal solution. Many of them, such as Genetic Algorithms, Cuckoo Search, Ant Colony Optimization, Simulated Annealing, Particle Swarm Optimization or Harmony Search, are inspired by natural phenomena and have been developed by mimicking the intelligence properties of biological and physical agents (Manjarres et al., 2013). Ulises et al. The MemPBPF algorithm proposed by MemPBPF contains dynamic membranes with a pseudo-bacterial genetic algorithm to develop the necessary parameters in the artificial potential field method. This hybridization between membrane computation, pseudo-bacterial genetic algorithm and artificial potential field method provides a better performing path planning algorithm for autonomous mobile robots. The computer simulation results show the effectiveness of the proposed MemPBPF algorithm in terms of path length considering collision avoidance and smoothness (Ulises et al., 2019a). Ulises et al. in their study, proposed an evolutionary artificial potential field (memEAPF) approach that combines membrane computation with a genetic algorithm (a membrane-inspired evolutionary algorithm with a single-level membrane structure) and artificial potential to solve the mobile robot path planning problem. The proposed approach gave better results in terms of path length when compared to artificial potential area-based path planning methods (Ulises et al., 2019b). Montiel-Ross et al. in their study, present the navigation software Ant Colony Test Center, which is designed to teach the different stages involved in mobile robotics. It has been determined that the study has been effective in solving discrete optimization problems recently (Montiel-Ross et al., 2013). Garcia et al. in their study, observed

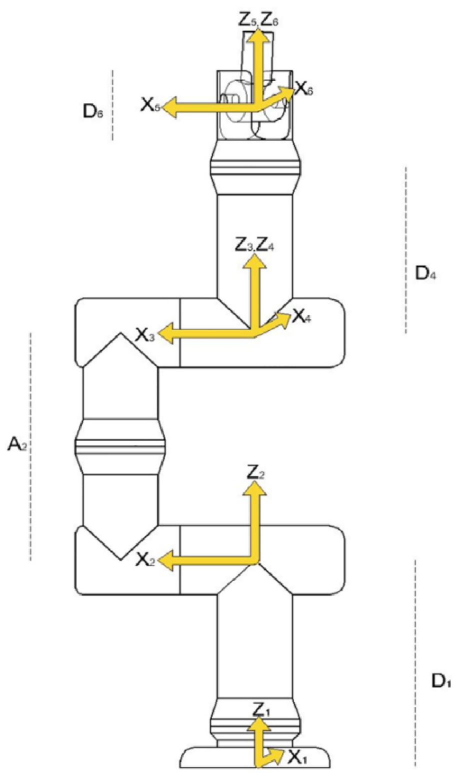

Fig. 1. 6 DOF robotic arm with joint coordinate spaces.

that the simple ant colony optimization metaheuristic-based (SACO-MH) method developed by mobile robots for trajectory planning gave effective results for path planning (Garcia et al., 2009). Amador-Angulo et al. In their study, they developed a Chicken Search Optimization (CSO) algorithm to be used in the field of fuzzy logic controllers and obtained effective results (Amador-Angulo et al., 2021). Bernal et al. carried out a study on the application of generalized type-2 fuzzy systems in the dynamic tuning of the parameters of a new metaheuristic method, the fuzzy sea hunter algorithm (FMPA). They concluded that the technique used improves performance (Bernal et al., 2020).

3. Material and method

3.1. Robot kinematics

A robotic manipulator consists of a group of rigid arms connected to each other by rotating or prismatic joints, which can be used to move any object from one point to another, can be programmed in three or more axes, and are self-controlled (Tonbul and Sarıtaş, 2003; Sadiq and Raheem, 2017). Each independently controllable joint represents one degree of freedom. Kinematics describes the relationship between joint positions and end-effector position and orientation. The kinematic chain ends are delimited between the base and the end effector, and the manipulator motion is achieved by combining the fundamental motion of each link with respect to the previous link (Beşkirli and Tefek, 2019). In order to establish the relationship between the joints, a coordinate system is placed on each joint. The position and relationship between the coordinate systems of the 6-degree-of-freedom robotic arm shown in Fig. 1 can be conveniently described with the homogeneous transformation matrix. DH parameters are observed in Table 1 (Demir, 2020; Zheng et al., 2019).

In the Denavit Hartenberg method, robot kinematics is created using four main variables (Demir, 2020; https://acrome.net/)

i. The link length between the two axes(link length) = a_{i-1},

Table 1
Denavit Hartenberg table of 6 DOF robotic arm.

Link number	Twist angle (α_{i-1})	Link length (a_{i-1})	Joint offset (d_i)	Joint angle (Θ_{i-1})
1	90	0	0, 191 m	Θ1 = 0
2	0	0, 278 m	0	Θ2 = 90
3	−90	0	0	Θ3 = −90
4	90	0	0, 202 m	Θ4 = 0
5	−90	0	0	Θ5 = 0
6	0	0	0, 105 m	Θ6 = 0

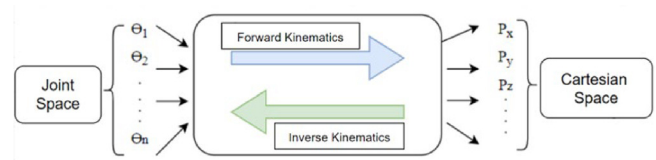

Fig. 2. Relationship between forward and inverse kinematics.

ii. Axis angle between two neighboring axes (link twist) = α_{i-1},
iii. Misalignment between overlapping axes (joint offset) = d_i
iv. The joint angle between two adjacent limbs (joint angle) = Θ_{i-1}

3.1.1. Robot forward kinematics

In robotic arms, each joint can rotate on a certain axis and the state of each joint is measured as an angle. The forward kinematics of the robot is based on each joint variable and coordinate system of the robot to determine the position of the end manipulator. This situation is associated with the cartesian space definition of the robotic arm. Each joint of the robotic arm is rotated at a certain angle, allowing the end manipulator to reach different target points in the working space (Bayrak, 2007).

3.1.2. Robot inverse kinematics

The transition from Cartesian space, which applies the position and orientation data of the end effector with forward kinematics, to the joint space, which contains only one angle of each joint, is called the inverse kinematics problem. This relationship between forward and inverse kinematics is shown in Fig. 2. Inverse kinematics problems are difficult to solve compared to advanced kinematics problems, and techniques involving trial and error are used, although they do not always produce exact solutions (Beyhan, 2021; Yılmaz, 2010; Özkarakoç, 2009).

4. Trajectory planning

During the movement of the end manipulators of the robotic arms from the starting point to the target point; Trajectory planning is the calculation of which waypoints it will pass, at what speed and acceleration it will move, and its orientation at those points. The aim of trajectory planning is to transport the tip manipulator from the starting point to the target point in a smooth and controlled manner without vibration, hitting any obstacle. Trajectory planning is considered as one of the important problems in robotic application studies (Uzuner et al., 2017). The robot does not go directly from its initial position to its final position. The reason for this is the difference in the axis of movement of each joint, the difference in their speed, or some obstacles in the environment. For this reason, it is necessary to create a trajectory planning passing through certain intermediate points for the robot arm to reach the desired target. In identifying waypoints, the speed of the robot is determined depending on the acceleration and distance parameters, and it poses a challenge depending on the robot's degree of freedom (Tonbul and Sarıtaş, 2003).

Trajectory planning is defined using two different ways, Cartesian space, and joint space.

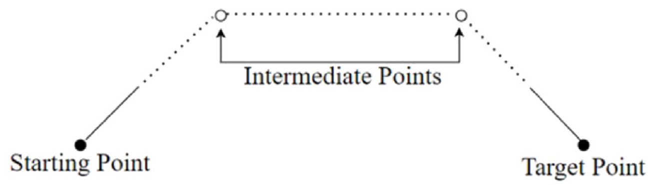

Fig. 3. Trajectory formed in Cartesian space.

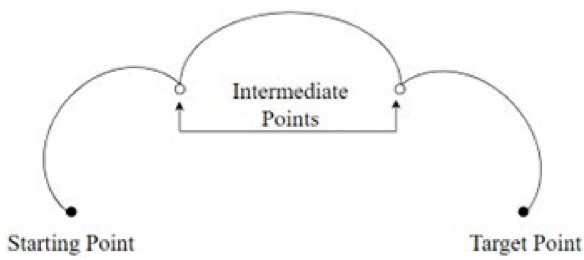

Fig. 4. Trajectory formed in joint space.

4.1. Trajectory planning in Cartesian space

Path planning in Cartesian space is defined in terms of functions of x, y and z coordinates. The Cartesian space is only concerned with the position of the manipulator. While planning the trajectory in Cartesian space, linear motion is provided between the starting and target positions of the manipulator. On the defined linear path, the position and velocity of the robot manipulator at the intermediate points defined at close distances are determined. Each axis of rotation must not exceed its own speed and acceleration limits. Thus, the joints are directed in accordance with this velocity profile. By using position, velocity and acceleration information in Cartesian space, joint information can be obtained with inverse kinematics (Dewi et al., 2020). The trajectory created by passing through intermediate points in the cartesian space is shown in Fig. 3.

4.2. Trajectory planning in joint space

The instant values of the joint variables are obtained by using advanced kinematic equations to find the points followed by the end manipulator reaching the target point from the starting point in a certain time interval. Thus, the position and position of the starting and target points are determined (https://acrome.net/). With orbital planning in joint space, the robot tip manipulator will complete its motion by creating the optimum trajectory between two points and will instantly generate position, velocity, and acceleration values. The operator will not have any interference with the created trajectory and will not have any effect on the speed of the end manipulator (Chen et al., 2015). The trajectory created by passing through intermediate points in the joint space is shown in Fig. 4.

If the manipulator movement will take place between a start and target point, the velocity and acceleration are taken as zero at the start and target points. If transition points are defined between the start and target points, the velocity value is selected and the initial, intermediate, and final acceleration values can be taken as zero to ensure the continuity of the acceleration (Tonbul and Sarıtaş, 2003).

5. Particle swarm optimization

Particle Swarm Optimization (PSO), developed by Kenedy and Elberhart in (1995), inspired by the behavior of bird and fish swarms, is a population-based stochastic optimization technique (Xia et al., 2023). The PSO algorithm is used to find solutions to multivariate and parameterized nonlinear optimization problems (Uzuner et al., 2017). The fact that flocks of birds, fish, etc., display random behaviors in situations such as searching for a source of nutrition and ensuring their safety, by communicating with each other, enables them to reach their goal in the shortest way and affects their speed (Savsani et al., 2014). The algorithm starts with a population of random solutions and searches for the optimal solution by updating the generations. Possible solutions, called particles in PSO, circulate in the problem space by following the optimum particle at that moment. The most important difference between PSO from classical optimization techniques is that it does not need derivative information. This social interaction within the herd is modeled with PSO (Öztürk, 2010; Çavuşlu et al., 2010).

In the PSO algorithm, each individual solution particle in the swarm is named as a particle, and each particle adjusts its position towards the best position in the swarm, taking advantage of its previous experience However, PSO is based on the logic of approximating the position of each of the particles in the swarm by updating it relative to the particle with the best position in the swarm (Gürgüze and Türkoğlu, 2019). For example, for the fitness function calculation of a problem consisting of D (dimension) parameters, the particle-matrix consisting of n particles is given in Eq. (1) (Dorigo et al., 2008).

$$x = \begin{bmatrix} x_{11} & x_{12} & \dots & \dots & x_{1D} \\ x_{21} & x_{22} & \dots & \dots & X_{2D} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & \dots & x_{nD} \end{bmatrix}_{nxD} \tag{1}$$

In the above matrix, particle i, $x_i = [x_{i1}, x_{i2}, \mathrm{I}x_{iD}]$,

The position of the i, particle giving the previous best fitness value, $P_{best} = [p_{i1}, p_{i2}, \dots, p_{iD}]$,

gbest is unique for all particles in each iteration, and $G_{bset} = [p_1, p_2, \dots, p_D]$,

The velocity of the i. particle (the amount of change in its position in each dimension) is expressed as $= v_i = \left[v_{i1}, v_{i2}, \dots, v_{iD}\right]$ (Erdoğmuş and Yalcin, 2015)

PSO flowchart is given in Fig. 5.

The algorithm consists of the following steps (Tamer and Karakuzu, 2006);

i. The starting swarm is created with randomly generated starting positions and velocities.
ii. The fitness values of all particles in the swarm are calculated.
iii. For each particle, the local best (pbest) from the current generation is found and the number of the best in the swarm is equal to the number of particles.
iv. The global best (gbest) is selected among the local bests in the current generation.
v. Positions and velocities are being renewed.

Step 2, 3, 4, 5 is repeated until the stopping criterion is met.

In the preference of PSO algorithm in the study; It allows for up-to-date multi-information exchange, being successful in reducing the computational load, mapping, avoiding obstacles, reaching the desired target in the fastest, shortest and most suitable position, algorithms such as genetic algorithm (Genetic Algorithm — GA) and simulated annealing (SA). Factors such as being more understandable and easier to implement, success in search operations, speed in terms of control performance and less parameters have been effective (Gürgüze and Türkoğlu, 2019; Pires et al., 2006; Kim and Lee, 2015; Wong et al., 2008; Şerbet, 2018).

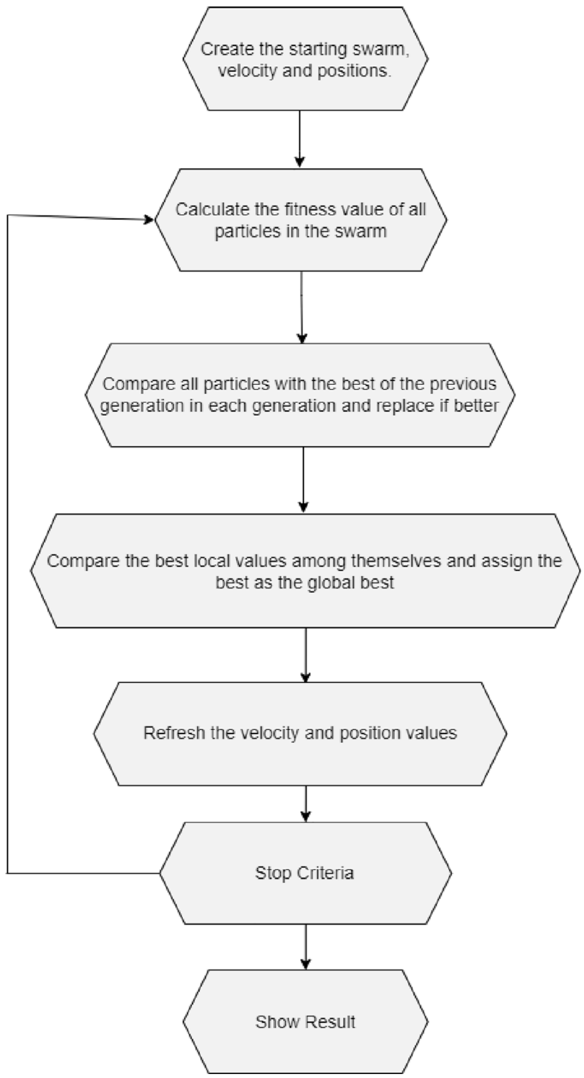

Fig. 5. PSO algorithm flow chart.

6. Method

In the study, the Particle swarm optimization (PSO) algorithm is the main method used to perform trajectory planning for a serial robot arm with 6 degrees of freedom. PSO algorithm was used to reach the target point from the starting point in the optimum time and to make each joint's position, velocity, and acceleration parameters more sensitive. Trajectory planning in the joint space was performed using a fifth-order polynomial so that the end actuator of the serial robot moves from the starting point to the target point without vibration and all the joints of the robot arm are finished simultaneously. Using forward kinematics and joint angles, the initial and final positions of the end manipulator without and in the presence of obstacles were investigated in Cartesian space. Method flowchart is given in Fig. 6.

In the flow diagram created with the PSO algorithm, the end manipulator, which moves from the starting point, checks whether there is any obstacle while moving towards the target. In the presence of an obstacle, with the PSO algorithm; In the search for the optimum trajectory of the particles in the motion space, the pbest and gbest values are determined and it is ensured that the target is moved over the optimum trajectory in the best particle tracking.

Acrome robot, designed and prototyped by Acrome company located in Isparta University of Applied Sciences, Faculty of Technology, Mechatronics Engineering Laboratory, is a type of robot with 6 degrees of freedom. In the study, a simulation model of the robot was created, as shown in Fig. 7, using the Robotic Toolbox library on the MATLAB program.

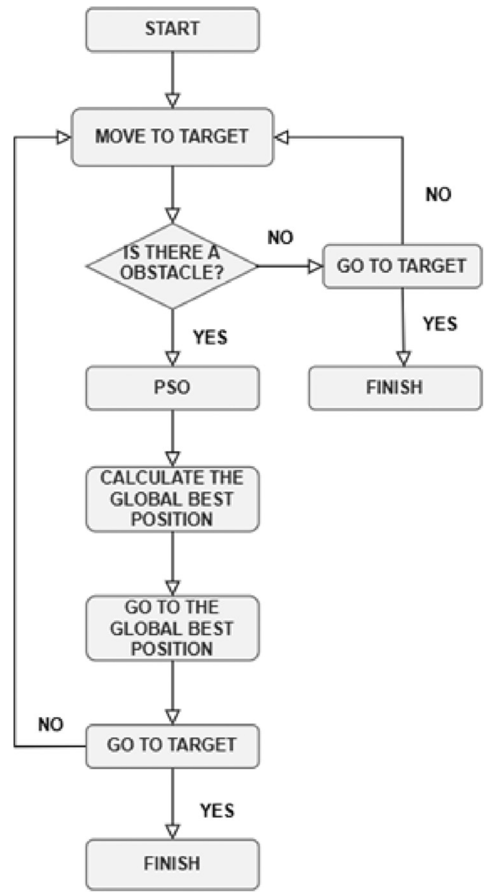

Fig. 6. Flow chart of the algorithm.

With the use of library functions, the starting and ending angles for each joint of the robot were determined. Forward kinematics enabled us to obtain the position and rotation information of the robot's end functionalist for the start and end points in Cartesian space by using the joint angles.

The robot's trajectory planning includes two methods as point-to-point (PTP) and continuous path-CP: While only the start and end points are specified with the PTP method, the intermediate points between two points are also specified with the CP method. In this study, the PTP method was used and a five-polynomial algorithm in MATLAB was used to perform joint space interpolation every 2 s. The created trajectory is shown in Fig. 8.

With the fifth-degree polynomial determined by the max and min constraints created depending on the joint angles; Smooth and continuous trajectory estimation of the end manipulator from the starting point to the endpoint is performed. The polynomial describes the motion trajectory of the end manipulator between two points. The angle-time graph in Fig. 9 below shows the nonlinear motion obtained from the fifth-degree polynomial equation.

The path traveled by the robot manipulator is estimated by determining the linear equation for each joint angle. The goal is to get the linear equation that gives the shortest distance and avoids possible obstacles. The PSO algorithm searches for the best linear or non-linear equation for the trajectory. The equations obtained can vary from a linear equation to a multidimensional polynomial. This result: means that the joint angles can move from the starting point to the target point using a simple linear equation or nonlinear polynomial equation.

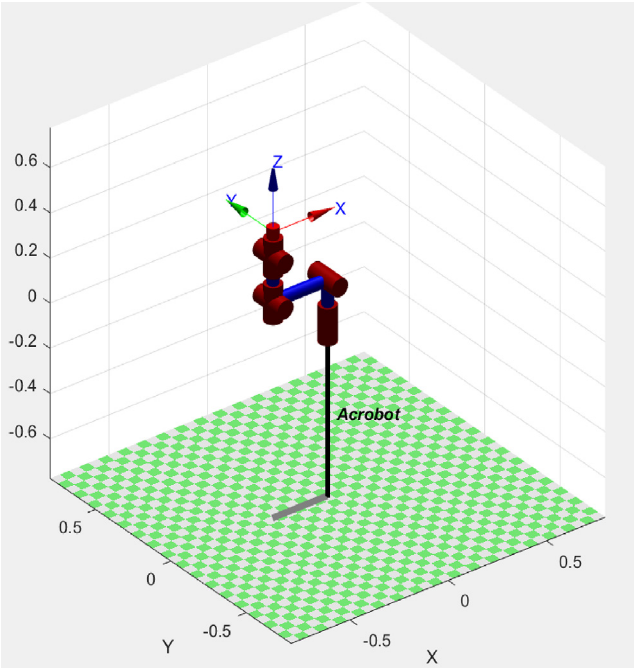

Fig. 7. Acrobot 6 DOF robotic.

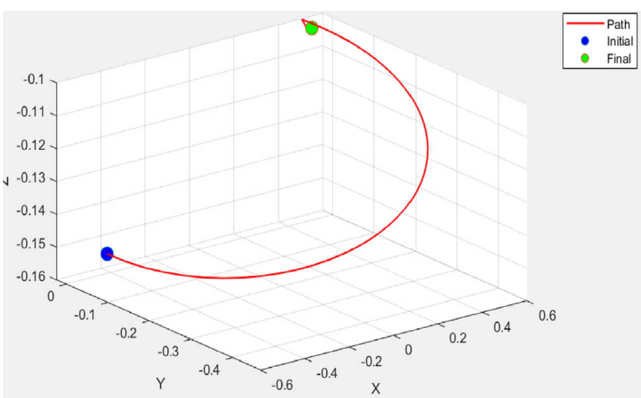

Fig. 8. Trajectory created by the end effector.

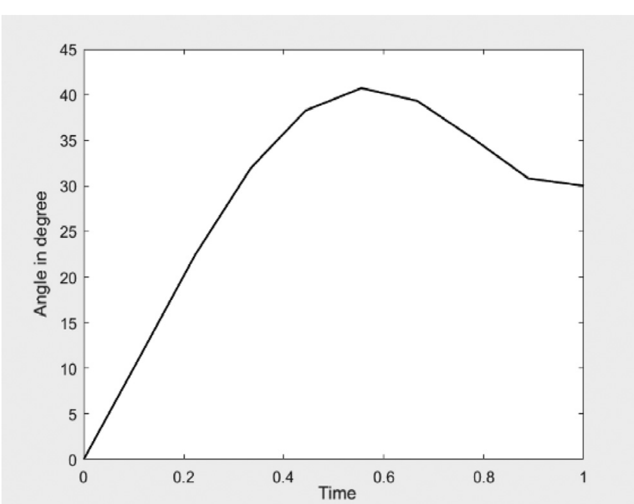

Fig. 9. Angle-time graph obtained with a fifth-degree polynomial.

Table 2
An example of a cross-section from the iteration table.

Iteration	f-count	Best f(x)	Mean f(x)	Stall iterations
31	3200	292.1	312.2	1
32	3300	292.1	313.3	0
33	3400	292.1	318.3	1
34	3500	292.1	320.6	2
35	3600	292	317.7	0
36	3700	292	316	0
37	3800	292.5	334.9	0
38	3900	291	335.4	0
39	4000	290.9	333	0

The purpose of using a fifth-degree polynomial is It is to minimize the vibration that may occur during the movement of the manipulator and to ensure optimum continuity.

All joints belong to the robot; The objective function is used to minimize the orbital time and the total distance depending on the parameters of restrictions such as speed, acceleration, and vibration. The objective function created by calculating the distance between all points drawn by the tip manipulator to the target point is shown in Eqs. (2).

$$d = \sum_{i=1}^{n} (Pxi, yi, zi - Qx, y, z) \tag{2}$$

Here,

Qx, y, z target coordinates,

Pxi, yi, zi represent the position coordinates created by the robot while moving to the target point.

The PSO algorithm is here.

• evaluates the objective function at each particle location and determines the best (lowest) function value and the best location.

• Selects new velocities based on the current velocity, the individual best positions of the particles, and the best positions of their neighbors.

• It then iteratively updates the particle positions (the new position is the old position plus the velocity, the velocity modified to keep the particles within bounds), velocities, and neighbors.

• Iterations continue until the algorithm reaches a stopping criterion.

Following its steps. An example iteration table is given in Table 2.

7. Findings and discussion

The proposed PSO algorithm method has been applied for the robot manipulator to move from the starting position to the target position on an optimal trajectory without hitting obstacles. The simulation process was performed with the MATLAB program to the proposed method and the trajectories created in the presence of obstacles, and the following graphics and results were obtained.

The path drawn for the trajectory planning created in the presence of a single obstacle is given in Fig. 10, the velocity graph of the joint angles in Fig. 11, and the motion graph of the joint angles depending on the time in Fig. 12.

A single spherical obstacle with a radius of 0.3 and a center of [−0.2,0.35,−0.1] is placed on the simulation. Maximum iteration number: 100 and RNG number: 10. When Figure 3.13 is examined, it is seen that the optimum trajectory is created from the starting point to the ending point without hitting the obstacle.

The path drawn for the trajectory planning including two circular obstacles is given in Fig. 13, the velocity graph of the joint angles in Fig. 14, and the motion graph of the joint angles depending on the time in Fig. 15.

Two spherical obstacles with 0.3 radius, centers [−0.2,0.35,−0.1] and [0.1, 0, −0.1] are placed on the simulation. Maximum iteration number: 100 and RNG number: 20. In Figure 3.16, it is seen that the optimum trajectory is created for the robotic manipulator from

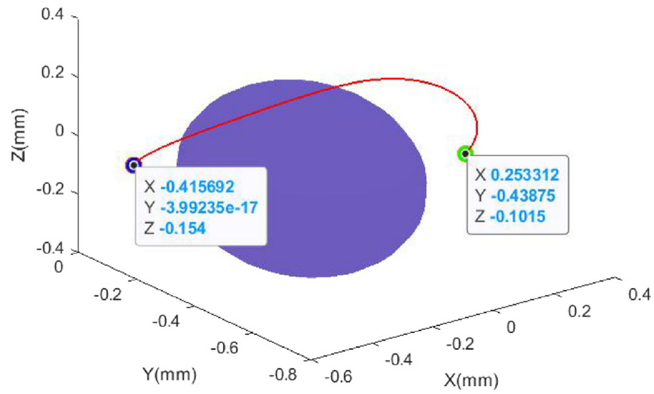

Fig. 10. A path is drawn by the robot's end effector.

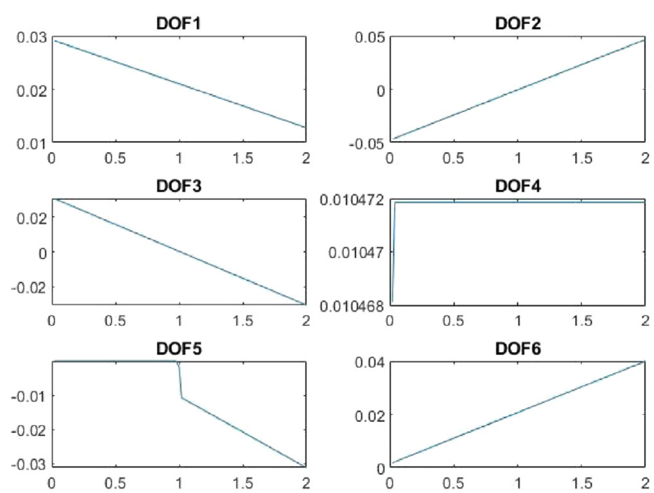

Fig. 11. Velocity graph of joint angles (rad/s).

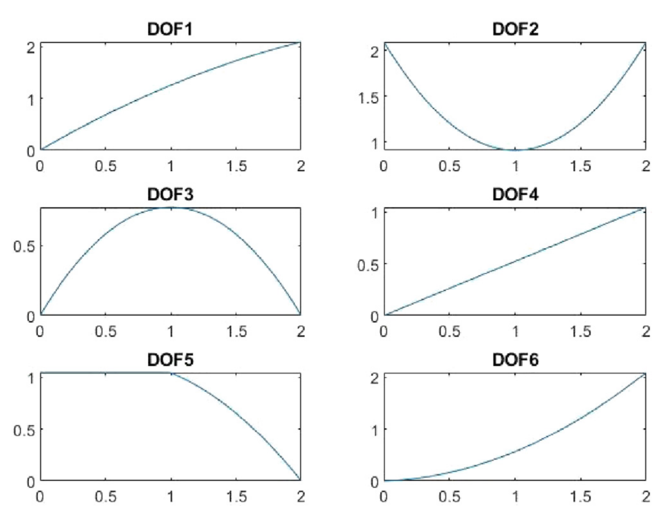

Fig. 12. Time-dependent graph of joint angles (rad/s).

the starting point to the ending point without touching the obstacles. However, when the velocity graph depending on the joint angles given in Figure 3.17 was examined, it was seen that the continuity could not be ensured.

The path drawn for the trajectory planning including two circular obstacles and a quadrilateral obstacle is given in Fig. 16, the velocity graph of the joint angles in Fig. 17, and the motion graph of the joint angles depending on time is given in Fig. 18.

Two spherical obstacles with a radius of 0.3 and one rectangular obstacle are placed on the simulation. Max. the number of iterations was determined as 100 and the number of RNG: 20. When the results obtained are examined, it is seen that the end manipulator moves from the starting point to the end point in the optimum trajectory without touching the obstacles.

The path drawn for the trajectory planning including two circular obstacles is given in Fig. 19, the velocity graph of the joint angles in Fig. 20, and the motion graph of the joint angles depending on time is given in Fig. 21.

Two spherical obstacles with a radius of 0.3 are placed on the simulation. Maximum number of iterations: 100 and number of RNG: 100. When the results obtained are examined, it is seen that the end manipulator moves from the starting point to the end point in the optimum trajectory without touching the obstacles.

The path drawn for the trajectory planning including two circular obstacles is given in Fig. 22, the velocity graph of the joint angles in Fig. 23, and the motion graph of the joint angles depending on time is given in Fig. 24.

Two spherical obstacles with a radius of 0.3 are placed on the simulation. Maximum number of iterations: 60 and number of rng: 20. When the results obtained are examined, it is seen that the end manipulator moves from the starting point to the end point in the optimum trajectory without touching the obstacles.

When the results obtained with the Matlab program are examined; It has been observed that the robot, which is optimized with fifth-order high-order interpolation and particle swarm optimization, creates successful trajectories despite some limitations encountered in disabled environments.

The results of the trajectory planning of the PSO algorithm in the optimum time of 40 s without hitting the obstacles are given in Table 3.

The academic literature related to the study is examined and the studies carried out with metaheuristic algorithms are given in Table 4.

Finally, the time optimization process of the PSO algorithm was performed in the study. For the time saving of the PSO algorithm, academic studies are examined, and the results are given in Table 5.

When Table 5 is examined, the results obtained from the PSO algorithms differ according to the obstacle and operating status compared to the hybrid algorithms. For this reason, it has been determined that the PSO algorithm is slower than the hybrid algorithms, albeit partially.

8. Conclusion

The main purpose of orbital planning is to minimize the travel time, the vibration, and the energy consumed during the journey, depending on the location, speed, and acceleration restrictions, without damaging the system. For this reason, heuristic methods can produce solutions to trajectory planning problems in studies in the field of robotics in recent years. This scope of work; Particle swarm optimization was used in the trajectory planning for the 6-axis robotic arm to reach the target in a controlled manner. In this study, the PSO algorithm is proposed to optimize the trajectory planning of the 6-degree-of-freedom Acrobot robot in the Isparta University of Applied Sciences laboratory with 5th-degree polynomial interpolation in the joint space.

The results obtained from the conducted study are given in detailed articles.

• Using the technical information of the 6-degree-of-freedom robot arm, a simulation model was created on the MATLAB Robotics Toolbox program, which can be simulated with the PSO algorithm.

• In order to ensure that the end manipulator of the 6-axis robotic arm can move in the optimum trajectory between certain starting and target points, trajectory planning in joint space has been carried out by

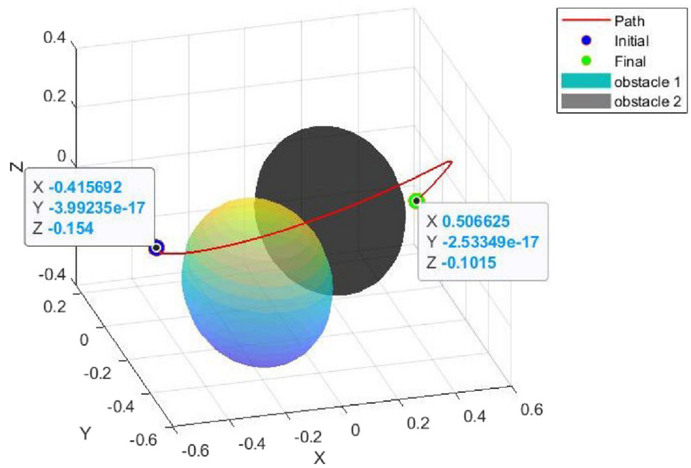

Fig. 13. A path is drawn by the robot's end effector.

Table 3
Timetable for static barriers in PSO algorithm.

Algorithm	Obstacle shape	Obstacle situation	Target situation	Time (s)
PSO	Single circular obstacle	Static	Static	19
	Two circulars obstacle	Static	Static	23
	Two circulars + one quadrilateral obstacle	Static	Static	40

Table 4
Comparison table of algorithms.

Author	Working area	Algorithms used	Time (s)	Result	Year
Panov and Koceski	Mobile Robot (Dynamic environment with 50% obstacles)	PSO GA ACO SA QHS	~2200 ~1900 ~10 ~3200 ~10	The developed QHS algorithm gave the best results in terms of time.	2015
Arora and Gigras	Mobile Robot and Robotics	PSO GA ACO SA TS(Tabu Search)	220 200 250 101 140	SA algorithm gave the best result in CPU time calculation.	2013
Altan	Unmanned aerial vehicle	PID PSO+PID HHO+PID	0.12×10^{-1} 0.17×10^{-3} 0.09×10^{-5}	It has been seen that the PID-based HHO algorithm gives more effective results.	2020
Abdor-Sierra et al.	6-DOF UR5 robot	SCAPSO DE/best/1/L DE/rand/1/bin PSO QPSO	0.141 0.121 0.293 0.381 0.270	The given algorithms have the fastest execution time with a low iteration count of 50 optimal solutions.	2022
Hao et al.	Robot	PSO GA PPSO	2.991 2.631 1.547	PPSO algorithm gave the fastest solution.	2007
Dewang et al.	Mobil robot	PSO APSO	3.466 3.360	The APSO algorithm gave success in a shorter time than the traditional PSO algorithm.	2018
In the study	6-DOF robotic arm	PSO	40	It gave a slow result compared to the improved PSO algorithms and hybrid algorithms.	2022

Table 5
Temporal comparison of metaheuristic algorithms in robotic systems.

Algorithm	Author	Obstacle shape	Obstacle status	Target status	Simulation	Real-time application	Success status	Year
PSO	Zhang et al.	Optional obstacle shapes	Static	Static	Available	–	Success	2013
	Hao et al.	Optional obstacle shapes	Static and dynamic	Static	Available	Available	More successful than the GA algorithm (PPSO hybrid model is more successful)	2007
	Wang et al.	Circular obstacles	Dynamic	Static	Available	–	Successful in taking the shortest path and converging very fast while maintaining the V-formation of the PSO	2009
	Dewang et al.	Square and circular obstacles	Static	Static	Available	–	The APSO algorithm is more successful than the PSO algorithm in terms of energy consumption and time.	2018
	Gong et al.	Various polygons	Static	Static	Available	–	Success	2011
	Rath and Deepak	Various polygons	Static and	Dynamic	Available	–	It has given effective results.	2015
	Ahmadzadeh and Ghanavati	Optional obstacle shapes	Static and dynamic	Static and dynamic	Available	–	Movement between desired points is provided by PSO.	2012
	Ekmen	–	–	Dynamic	Available	–	PSO, CLA, and CCA algorithms were used, and the PSO algorithm gave more successful results in 4 and 5 Drone studies.	2020
	Gigras and Gupta	Rectangular	Dynamic	Static	Available	–	The KKA algorithm gave the closest optimal solution in a shorter time than the PSO algorithm.	2014
	Mousavi et al.	–	–	Dynamic	Available	–	GA was 69.4%, PSO was 74% and the GAPSO algorithm was 79.8%.	2017
	Kim and Lee	Circular	Static	Static	Available	Available	According to STOMP and DE algorithms, the PSO algorithm gave the most effective results.	2015

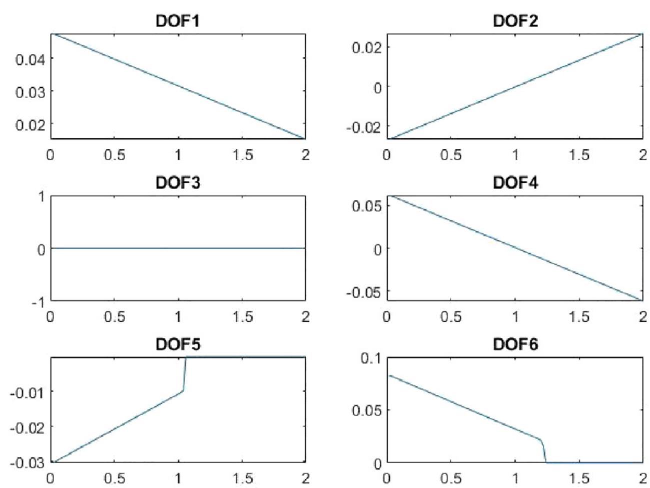

Fig. 14. Velocity graph of joint angles (rad/s).

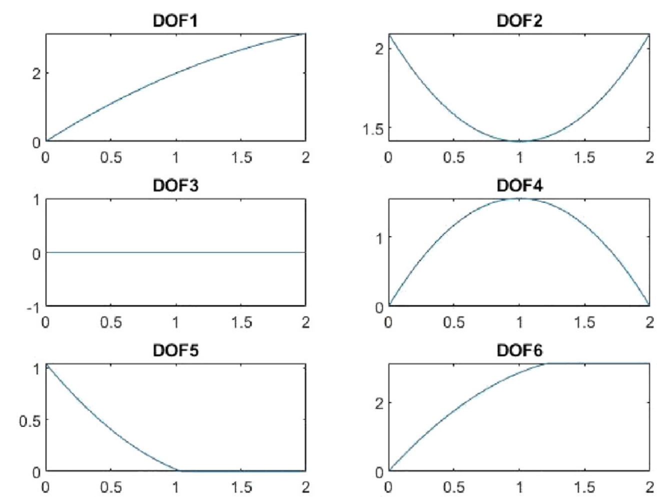

Fig. 15. Time-dependent graph of joint angles (rad/s).

using PSO algorithm, which is one of the metaheuristic methods, and fifth-order polynomial interpolation.

• The trajectory followed by the end manipulator of the robot was plotted in the Cartesian coordinate system, and the behavior of the end manipulator between the determined start and end points in the working environment containing circular and quadrangular obstacles was examined.

• With the obtained figures and graphics, it was concluded that the end manipulator reached the end point from the optimum path avoiding obstacles under constraints.

• The purpose of using a fifth-degree polynomial in the study is to minimize the vibration that may occur during the movement of the manipulator and to provide optimum continuity.

As a result, the study has contributed to the academic literature by making trajectory planning in the joint space using a fifth-degree polynomial with the PSO algorithm, taking into account the specific technical information of the 6-axis robotic arm.

In future studies, it is planned to carry out new studies according to the suggestions and restrictions given below.

• The proposed method in this thesis study can be developed and compared with different optimization methods.

• By strengthening the weaknesses of the algorithm and changing the operating parameters, a hybrid model can be created with two or more optimization methods. Thus, by using hybrid algorithms, the trajectory planning problem can be optimized and the global best solution can be reached.

• It can be used in fields such as industrial, medical, education and health in order to serve various image processing and artificial intelligence technologies by integrating the work performed with camera systems on real robots.

• Different results can be obtained by making the obstacles in the working environment and the starting and ending points dynamic.

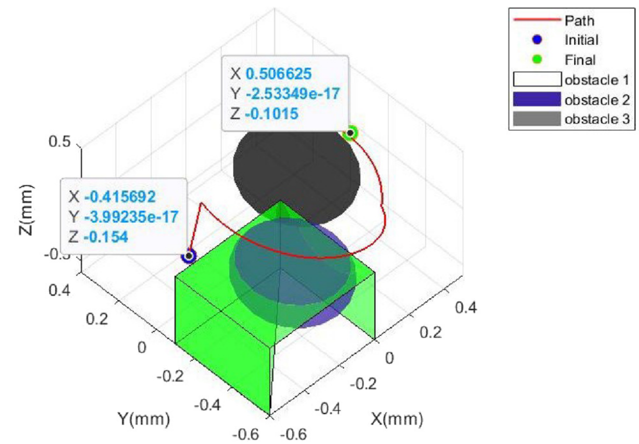

Fig. 16. A path is drawn by the robot's end effector.

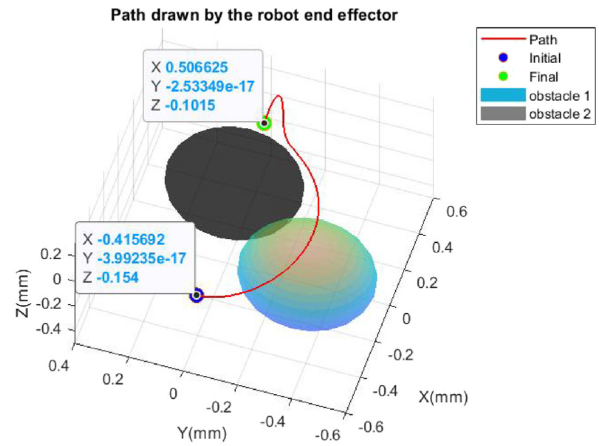

Fig. 19. A path is drawn by the robot's end effector.

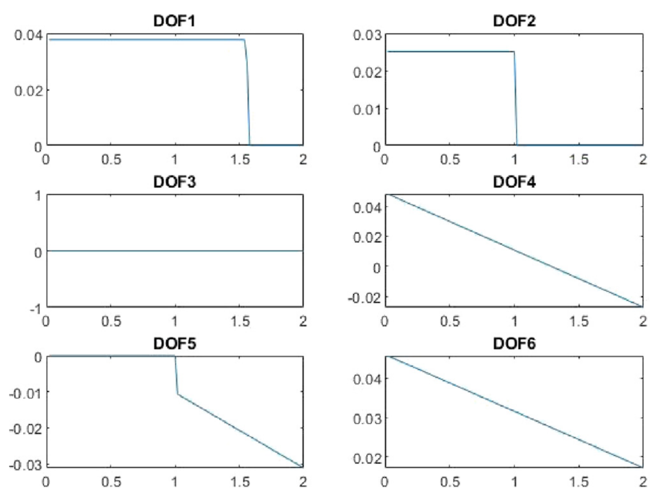

Fig. 17. Velocity graph of joint angles (rad/s).

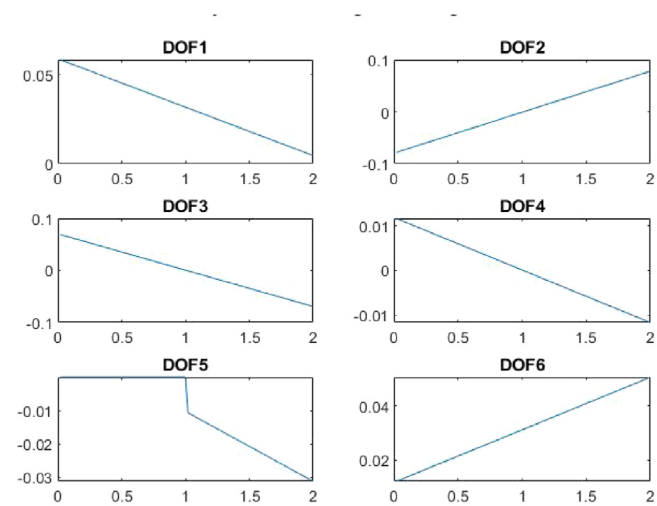

Fig. 20. Velocity graph of joint angles (rad/s).

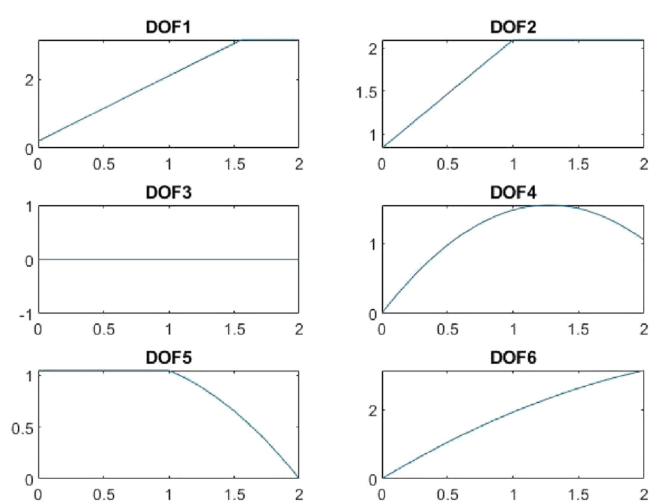

Fig. 18. Time-dependent graph of joint angles (rad/s).

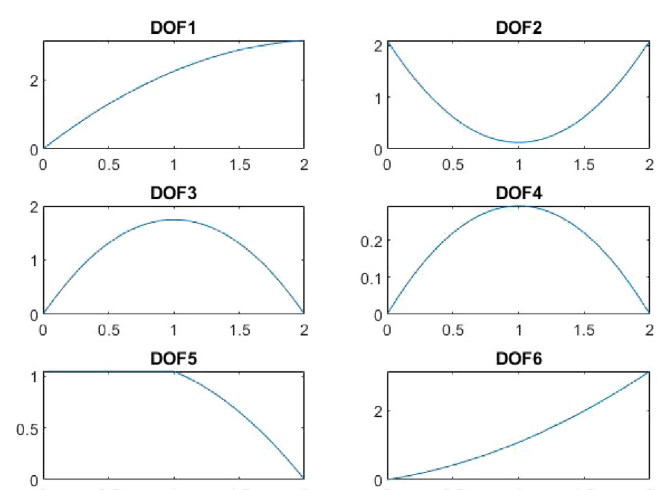

Fig. 21. Time-dependent graph of joint angles (rad/s).

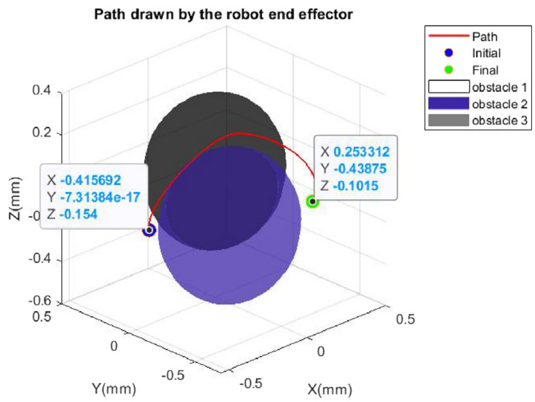

Fig. 22. A path is drawn by the robot's end effector.

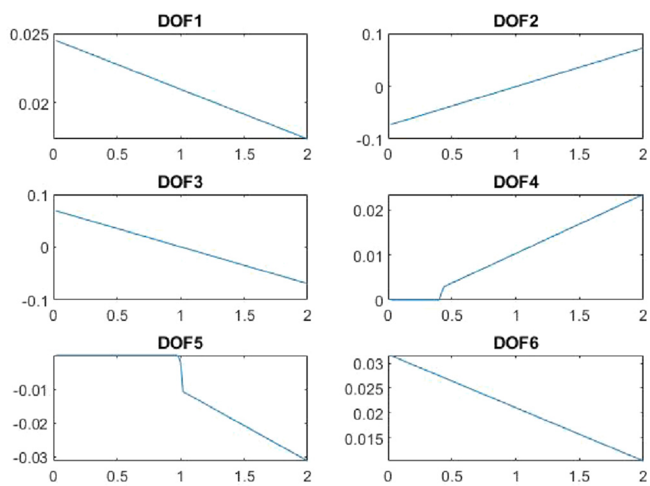

Fig. 23. Velocity graph of joint angles (rad/s).

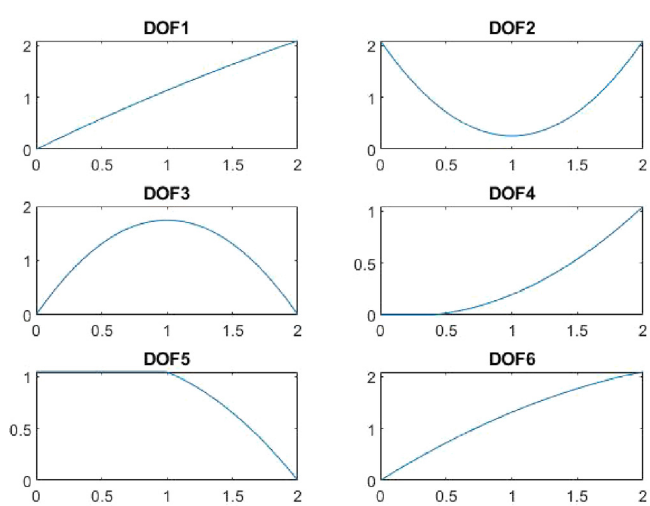

Fig. 24. Time-dependent graph of joint angles (rad/s).

CRediT authorship contribution statement

Özge Ekrem: Carried out a literature review and simulation application as a Master's thesis. **Bekir Aksoy:** As a master's thesis supervisor, research, Software, editing, Reviewing and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

We would like to thank Isparta University of Applied Sciences, Graduate Education Institute, Department of Mechatronics Engineering for permission to use the robotic arm and licensed MATLAB program in the study.

Data availability

No data was used for the research described in the article.

References

Abdor-Sierra, J.A., Merchán-Cruz, E.A., Rodríguez-Cañizo, R.G., 2022. A comparative analysis of metaheuristic algorithms for solving the inverse kinematics of robot manipulators. Results Eng. 16 (100597), 1–11.

Ahmadzadeh, S., Ghanavati, M., 2012. Navigation of mobile robot using the PSO particle swarm optimization. J. Acad. Appl. Stud. (JAAS) 2 (1), 32–38.

Altan, A., 2020. Performance of metaheuristic optimization algorithms based on swarm intelligence in attitude and altitude control of unmanned aerial vehicle for path following. In: 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies. ISMSIT, IEEE, pp. 1–6.

Amador-Angulo, L., Castillo, O., Peraza, C., Ochoa, P., 2021. An efficient chicken search optimization algorithm for the optimal design of fuzzy controllers. Axioms 10 (1), 1–26.

Arora, T., Gigras, Y., 2013. A survey of comparison between various metaheuristic techniques for path planning problem. Int. J. Comput. Eng. Sci. 3 (2), 62–66.

Ağralı, E., Çavaş, M., 2020. V-REP robotik simülatör ile robot kol simülasyonu. Firat Univ. J. Eng. 32 (2), 435–444.

Bayrak, A., 2007. Beş Eksenli Bir Robot Kolunun Simülasyonu ve Kontrolü. Gazi Üniversitesi. Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara.

Bernal, E., Castillo, O., Soria, J., Valdez, F., 2020. Generalized type-2 fuzzy logic in galactic swarm optimization: Design of an optimal ball and beam fuzzy controller. J. Intell. Fuzzy Systems 39 (3), 3545–3559.

Beşkirli, M., Tefek, M.F., 2019. Parçacık sürü optimizasyon algoritması kullanılarak optimum robot yolu planlama. Avrupa Bilim Teknol. Derg. 201–213, Special Issue.

Beyhan, A., 2021. Üç Serbestlik Dereceli Robotik Kolun Kartezyen Uzayda Empedans Kontrolü (Master's thesis). BTÜ, Fen Bilimleri Enstitüsü.

Çavuşlu, M.A., Karakuzu, C., Şahin, S., 2010. Parçacık sürü optimizasyonu algoritması ile yapay sinir ağı eğitiminin FPGA üzerinde donanımsal gerçeklenmesi. Politek. Derg. 13 (2), 83–92.

Chen, X., Li, Y., 2006. Smooth path planning of a mobile robot using stochastic particle swarm optimization. In: 2006 International Conference on Mechatronics and Automation. IEEE, pp. 1722–1727.

Chen, D., Zhang, B., Wang, M., 2015. Cartesian space trajectory planning on 7-DOF manipulator. In: 2015 IEEE International Conference on Robotics and Biomimetics. ROBIO, IEEE, pp. 940–945.

Demir, H. Path, 2020. Planning for 3 Degree of Freedom (RRR) Robot and its Application. Aksaray University/ Fen Bilimleri Enstitüsü/ Elektrik-Elektronik ve Bilgisayar Mühendisliği Ana Bilim Dalı.

Dewang, H.S., Mohanty, P.K., Kundu, S., 2018. A robust path planning for mobile robot using smart particle swarm optimization. Procedia Comput. Sci. 133, 290–297.

Dewi, T., Nurmaini, S., Risma, P., Oktarina, Y., Roriz, M., 2020. Inverse kinematic analysis of 4 DOF pick and place arm robot manipulator using fuzzy logic controller. Int. J. Electr. Comput. Eng. 10 (2), 1376–1386.

Doctor, S., Venayagamoorthy, G.K., Gudise, V.G., 2004. Optimal PSO for collective robotic search applications. In: Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753), Vol. 2. IEEE, pp. 1390–1395.

Dorigo, M., de Oca, M.A.M., Engelbrecht, A., 2008. Particle swarm optimization. Scholarpedia 3 (11), 1486.

Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory. In: Mhs95 Sixth International Symposium on Micro. Machine & Human Science.

Ekmen, M.İ., 2020. Drone Sürüsü Ile Hedef Takip Optimizasyonu. Yüksek Lisans Tezi, Konya Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü.

Erdoğmuş, P., Yalcin, E., 2015. Parçacik sürü optimizasyonu ile kisitsiz optimizasyon test problemlerinin çözümü. İleri Teknol. Bilim. Derg. 4 (1), 14–22.

Ergür, S., 2015. Endüstriyel Robot Kollarıyla 3B Modelleme Amaçlı Benzetim Ortamı Tasarımı (Master's thesis). ESOGÜ, Fen Bilimleri Enstitüsü.

Garcia, M.P., Montiel, O., Castillo, O., Sepulveda, R., Melin, P., 2009. Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. Appl. Soft Comput. 9 (3), 1102–1110.

Gigras, Y., Gupta, K., 2014. Metaheuristic algorithm for robotic path planning. Int. J. Comput. Appl. 85 (3), 26–29.

Gong, D., Zhang, J., Zhang, Y., 2011. Multi-objective particle swarm optimization for robot path planning in environment with danger sources. J. Comput. 6, 1554–1561.
Gürgüze, G., Türkoğlu, İ., 2019. Robot sistemlerinde kullanılan algoritmalar. Türk Doğa Fen Derg. 8 (1), 17–31.
Han, Y., Zhang, L., Tan, H., Xue, X., 2019. Mobile robot path planning based on improved particle swarm optimization. In: 2019 Chinese Control Conference. CCC, IEEE, pp. 4354–4358.
Hao, Y., Zu, W., Zhao, Y., 2007. Real-time obstacle avoidance method based on polar coordination particle swarm optimization in dynamic environment. In: 2007 2nd IEEE Conference on Industrial Electronics and Applications. IEEE, pp. 1612–1617.
https://acrome.net/ (accessed June 20, 2022).
Huang, S., Wang, M., Zou, Y., 2019. V-REP-based virtual platform on multi-pattern neural control of robotic manipulator. In: 2019 Chinese Control and Decision Conference. CCDC, IEEE, pp. 1570–1575.
Kim, J.J., Lee, J.J., 2015. Trajectory optimization with particle swarm optimization for manipulator motion planning. IEEE Trans. Ind. Inform. 11 (3), 620–631.
Lopez-Franco, C., Hernandez-Barragan, J., Alanis, A.Y., Arana-Daniel, N., 2018. A soft computing approach for inverse kinematics of robot manipulators. Eng. Appl. Artif. Intell. 74, 104–120.
Ma, C., Zhang, Y., Cheng, S., 2021. Time optimal trajectory planning based on redundant manipulator. In: 2021 6th International Conference on Control, Robotics and Cybernetics. CRC, IEEE, pp. 125–129.
Manjarres, D., Landa-Torres, I., Gil-Lopez, S., Del Ser, J., Bilbao, M.N., Salcedo-Sanz, S., Geem, Z.W., 2013. A survey on applications of the harmony search algorithm. Eng. Appl. Artif. Intell. 26 (8), 1818–1831.
Masehian, E., Sedighizadeh, D., 2010. A multi-objective PSO-based algorithm for robot path planning. In: 2010 IEEE International Conference on Industrial Technology. IEEE, pp. 465–470.
Min, H.Q., Zhu, J.H., Zheng, X.J., 2005. Obstacle avoidance with multi-objective optimization by PSO in dynamic environment. In: 2005 International Conference on Machine Learning and Cybernetics, Vol. 5. IEEE, pp. 2950–2956.
Montiel-Ross, O., Sepúlveda, R., Castillo, O., Melin, P., 2013. Ant colony test center for planning autonomous mobile robot navigation. Comput. Appl. Eng. Educ. 21 (2), 214–229.
Mousavi, M., Yap, H.J., Musa, S.N., Tahriri, F., Dawal, S.Z., 2017. Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization. pp. 1–25.
Nasrollahy, A.Z., Javadi, H.H.S., 2009. Using particle swarm optimization for robot path planning in dynamic environments with moving obstacles and target. In: 2009 Third UKSim European Symposium on Computer Modeling and Simulation. IEEE, pp. 60–65.
Özkarakoç, M., 2009. Scara Robot Manipulatör Ile Seçme ve Yerleştirme Uygulaması. Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü Yüksek Lisans Tezi, İzmir.
Öztürk, M., 2010. Robotik Yapının Kinematik Modellenmesi. Lisans Tezi, İstanbul Teknik Üniversitesi, Uçak ve Uzay Bilimleri Fakültesi, İstanbul.
Panov, S., Koceski, S., 2015. Metaheuristic global path planning algorithm for mobile robots. Int. J. Reason.-Based Intell. Syst. 7 (1–2), 35–41.
Pires, E.J., Oliveira, P.B., Tenreiro Machado, J.A., ve Cunha, J.B., 2006. Manipülatör yörünge planlamasında parçacık sürüsü optimizasyonuna karşı genetik algoritma. In: Otomatik Kontrolü CONTROLO 2006-7th Portekizce Konferansı. pp. 1–7.
Rath, M.K., Deepak, B.B.V.L., 2015. PSO based system architecture for path planning of mobile robot in dynamic environment. In: 2015 Global Conference on Communication Technologies,. GCCT, IEEE, pp. 797–801.
Sadiq, A.T., Raheem, F.A., 2017. Robot arm path planning using modified particle swarm optimization based on D* algorithm. Al-Khwarizmi Eng. J. 13 (3), 27–37.
Savsani, P., Jhala, R.L., Savsani, V.J., 2014. Comparative study of different metaheuristics for the trajectory planning of a robotic arm. IEEE Syst. J. 10 (2), 697–708.
Şerbet, F., 2018. Parçacık Sürü Optimizasyon Algoritmasıyla Labvıew Ortamında Sonlu İmpals Cevaplı Sayısal Filtre İçin Gerçek Zamanlı Pencere Fonksiyonu Tasarımı. Yüksek Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü.
Tamer, S., Karakuzu, 2006. Parçacık sürüsü optimizasyon algoritmasi ve benzetim örnekleri.
Tonbul, T., Sarıtaş, M., 2003. Beş eksenli bir edubot robot kolunda ters kinematik hesaplamalar ve yörünge planlamasi. Gazi Üniv. Mühendis. Mimarlık Fak. Derg. 18 (1), 145–167.
Ulises, O.R.U., Montiel, O., Sepúlveda, R., 2019b. Mobile robot path planning using membrane evolutionary artificial potential field. Appl. Soft Comput. 77, 236–251.
Ulises, O.R., Picos, K., Montiel, O., 2019a. Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots. IEEE Access 7, 156787–156803. http://dx.doi.org/10.1109/ACCESS.2019.2949835.
Ulusoy, A., Güneş, M., 2019. Mobil robot kolunun PSO ile stabilizasyonu. Kahramanmaraş Sütçüimam Üniv. Mühendis. Bilim. Derg. 22 (4), 288–297.
Uzuner, S., Akkus, N., Toz, M., 2017. Trajectory planning of a 5-DOF serial robot manipulator in joint-space. J. Polytech. 20 (1), 151–157.
Wang, Y., Cai, F., Wang, Y., 2017. Application of particle swarm optimization in path planning of mobile robot. In: AIP Conference Proceedings, Vol. 1864. AIP Publishing LLC, pp. 1–7, (1).
Wang, Y., Chen, P., Jin, Y., 2009. Trajectory planning for an unmanned ground vehicle group using augmented particle swarm optimization in a dynamic environment. In: 2009 IEEE International Conference on Systems, Man and Cybernetics. IEEE, pp. 4341–4346.
Wang, X., Shi, Y., Ding, D., Gu, X., 2016. Çift global optimum genetik algoritma - partikül sürüsü optimizasyonuna dayalı kaynak robotu yol planlamasi. Mühendis. Optim. 48 (2), 299–316.
Wong, C.C., Wang, H.H., Li, S.A., 2008. PSO-based motion fuzzy controller design for mobile robots. Int. J. Fuzzy Syst. 10, 24–32.
Xia, X., Fu, X., Zhong, S., Bai, Z., Wang, Y., 2023. Gravity particle swarm optimization algorithm for solving shop visit balancing problem for repairable equipment. Eng. Appl. Artif. Intell. 117, 1–20.
Xin, P., Rong, J., Yang, Y., Xiang, D., Xiang, Y., 2017. Trajectory planning with residual vibration suppression for space manipulator based on particle swarm optimization algorithm. Adv. Mech. Eng. 9 (4), 1–16.
Yılmaz, D., 2010. Bir Robot Kolunun Bilgisayar Destekli Kinematik Analizi (Master's thesis). Sakarya Fen Bilimleri Enstitüsü.
Zhang, Y., Gong, D.W., Zhang, J.H., 2013. Robot path planning in uncertain environment using multi-objective particle swarm optimization. Neurocomputing 103, 172–185.
Zhang, Y., Wang, S., Ji, G., 2015. A comprehensive survey on particle swarm optimization algorithm and its applications. Math. Probl. Eng. 2015, 1–39.
Zheng, X., Zheng, Y., Shuai, Y., Yang, J., Yang, S., Tian, Y., 2019. Kinematics analysis and trajectory planning of 6-DOF robot. In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference. ITNEC, IEEE, pp. 1749–1754.