Annals of Emerging Technologies in Computing (AETiC)
Vol. 8, No. 3, 2024

Research Article

Adaptive Meta-heuristic Framework for Real-time Dynamic Obstacle Avoidance in Redundant Robot Manipulators

Sheik Masthan Shahul Abdul Rahim[1,*], Kanagaraj Ganesan[2] and Mohammed Shafi Kundiladi[3]

[1]Thiagarajar College of Engineering, India
sarsmech@tce.edu; masthansarsheik@gmail.com
[2]Thiagarajar College of Engineering, India
gkmech@tce.edu
[3]Intel Corporation, United States
mohammed.shafi.k@intel.com
*Correspondence: sarsmech@tce.edu

Received: 13th March 2024; Accepted: 23rd June 2024; Published: 1st July 2024

Abstract: Robotic manipulator faces a challenge in navigating dynamic environments while ensuring collision-free trajectories, especially for redundant manipulators. Inverse kinematics involves finding joint angles to reach a specific point in 3D space. The shift from classical analytical and numerical methods to optimization heuristic algorithms is driven by the increasing complexity of robotic systems and the demand for more versatile and adaptive solutions. Meta-heuristic algorithms offer a transformative approach by framing the inverse kinematics problem as an optimization challenge, providing a flexible and robust means to navigate complex solution spaces. Metaheuristic algorithms, known for their ability to explore high-dimensional search spaces and avoid local optima, offer robust solutions for these challenges. They enhance computational efficiency, enabling real-time decision-making and obstacle avoidance, making them ideal for complex robotic applications. These characteristics of the metaheuristic algorithms can used in developing an integrated framework that offers complete solution to robot manipulators. This research article presents a generalized framework leveraging meta-heuristic algorithms to address dynamic obstacle avoidance in redundant manipulators. The framework uses meta-heuristic algorithms as the inverse kinematics solver, 3D trajectory planner, and obstacle avoidance mechanism, encompassing both static and dynamic obstacles. The proposed framework is generalized and gives the user to select the type of robot manipulator, with any number of links with any custom trajectory within the workspace of the robot manipulator. Also, any metaheuristic algorithm can be used in the proposed framework. The proposed framework is implemented in MATLAB's app designer for simulation with six different meta-heuristic algorithms. The effectiveness of the framework was evaluated in terms of its capability to generate 3d path, its ability to follow generated trajectory, while seamlessly adapting to dynamically changing environments. Through simulation, the framework showcased robust performance in navigating workspaces with moving obstacles, ensuring collision-free motion for redundant manipulators.

Keywords: ***Adaptive Trajectory Tracking; Collision-free Trajectory Follower; Dynamic Obstacle Avoidance; Hybrid Meta-heuristic Algorithm; Inverse Kinematics Solver; Real-time Obstacle Avoidance; Redundant Robot Manipulator***

1. Introduction

In today's industrial automation landscape, robotic systems play an increasingly important role. Industrial robots have evolved into indispensable assets, improving production processes, increasing productivity, and ensuring precision in a wide range of applications. As we enter this technological era, the

Sheik Masthan Shahul Abdul Rahim, Kanagaraj Ganesan and Mohammed Shafi Kundiladi, "Adaptive Meta-heuristic Framework for Real-time Dynamic Obstacle Avoidance in Redundant Robot Manipulators", *Annals of Emerging Technologies in Computing (AETiC)*, Print ISSN: 2516-0281, Online ISSN: 2516-029X, pp. 1-16, Vol. 8, No. 3, 1st July 2024, Published by International Association for Educators and Researchers (IAER), DOI: 10.33166/AETiC.2024.03.001, Available: http://aetic.theiaer.org/archive/v8/v8n3/p1.html.

use of robotic manipulators has increased across a wide range of industries, from automotive assembly lines to pharmaceutical manufacturing, demonstrating the revolutionary impact of automation on modern businesses. In the pursuit of efficiency and versatility, redundant robot manipulators have emerged as a significant achievement in robotic design. Redundancy in these manipulators adds degrees of freedom, allowing for greater adaptation to complicated tasks and dexterity in adverse conditions. This characteristic allows robots to navigate complex workspaces and carry out jobs with previously unthinkable agility. However, the deployment of redundant manipulators creates significant issues, especially when faced with dynamic barriers in their operational environment. The dynamic nature of modern industrial environments necessitates powerful algorithms capable of dynamically adjusting robot trajectories to avoid collisions, assuring both safety and continued operation. This requirement is even more obvious in situations where real-time modifications are critical, such as collaborative workspaces where people and robots cohabit. Inverse kinematics, trajectory planning, and obstacle avoidance are the key processes that enable any industrial robot to operate in complex environments, avoid obstacles, and complete tasks with precision and efficiency. A comprehensive review of various control strategies for robotic manipulators and potential solutions for obstacles, with an analytical study explaining the various control strategies, link types, models, applications, and degrees of freedom is done in [1].

Inverse kinematics is the process of determining the joint configurations required to achieve a desired end-effector position and orientation. This mathematical computation is essential for translating task specifications into actionable commands for the robot's joints. The classical analytical methods [2-5] that were proposed for solving inverse kinematics are not versatile, and the robot mechanisms that can be created are limited. For robots with complex kinematic systems, closed-form solutions could not exist or be practical. Furthermore, in some configurations, the analytically derived solutions may not be as reliable due to their sensitivity to singularities. Several numerical methods [6-7] have been put forth to solve inverse kinematics problems when an analytical solution cannot be obtained. However, the numerical methods may have convergence issues, and the choice of initial guesses can affect the solution. Iterative methods may require more computational resources, especially for robots with a large number of degrees of freedom.

In contrast, heuristic and metaheuristic algorithms offer a transformative approach by framing the inverse kinematics problem as an optimization challenge. These algorithms, inspired by natural processes or mathematical optimization techniques, provide a flexible and robust means to navigate the complex solution spaces associated with redundant manipulators. They can incorporate various constraints, including joint limits and task-specific criteria, allowing for a more adaptable and customized approach to inverse kinematics problem-solving. The advantages of these algorithms are their ability to handle redundancy effectively, optimize performance criteria, and offer flexibility in the face of dynamic changes or uncertainties in the environment. Furthermore, these algorithms are well-suited for real-time applications, ensuring adaptability in scenarios where the robot must continuously adjust its configuration.

Metaheuristic algorithms, inspired by natural phenomena have proven to be effective in solving complex optimization problems [8-10]. Researchers have employed various heuristic and metaheuristic algorithms for solving the inverse kinematics of the robot manipulators. To name a few, Genetic Algorithm (GA) [11], modified GA [12], Particle Swarm Optimization (PSO) and its variants [13-14], variant of Artificial Bee Colony (ABC) [15], Firefly [16], beta salp swarm algorithm [17], Bat algorithm and its variants [18], Cuckoo and Imperialist Competitive algorithm [19], etc. Various other optimization algorithms that are used for robot manipulator's inverse kinematics problem, trajectory tracking is available in [20-23].

Metaheuristic methods, inspired by natural or mathematical principles, offer a pragmatic means to navigate the intricate solution spaces associated with inverse kinematics. Furthermore, when extending these approaches to trajectory planning with obstacle avoidance, the adaptability of these algorithms becomes paramount. Metaheuristic-based methods can efficiently explore alternative trajectories, optimizing joint configurations to navigate around obstacles and ensuring collision-free paths. These qualities motivated us in developing a metaheuristic-based framework that can act as inverse kinematics solver, trajectory planner, obstacle avoidance including dynamic obstacles for redundant robot manipulator.

The rest of this paper is organized as follows: Section 2 presents the general flow a typical metaheuristic algorithm. Section 3 explains the proposed framework for redundant robot manipulator and the roles of metaheuristic algorithm in the proposed framework. Section 4 elaborates the implementation of the

proposed framework and discusses about the results obtained through simulation. Section 5 concludes the paper and presents the future work.

2. Metaheuristic Algorithm

Metaheuristic algorithms may differ in their details, but generally they follow a set of common steps. The flow of a typical metaheuristic algorithm is shown in Figure 1.

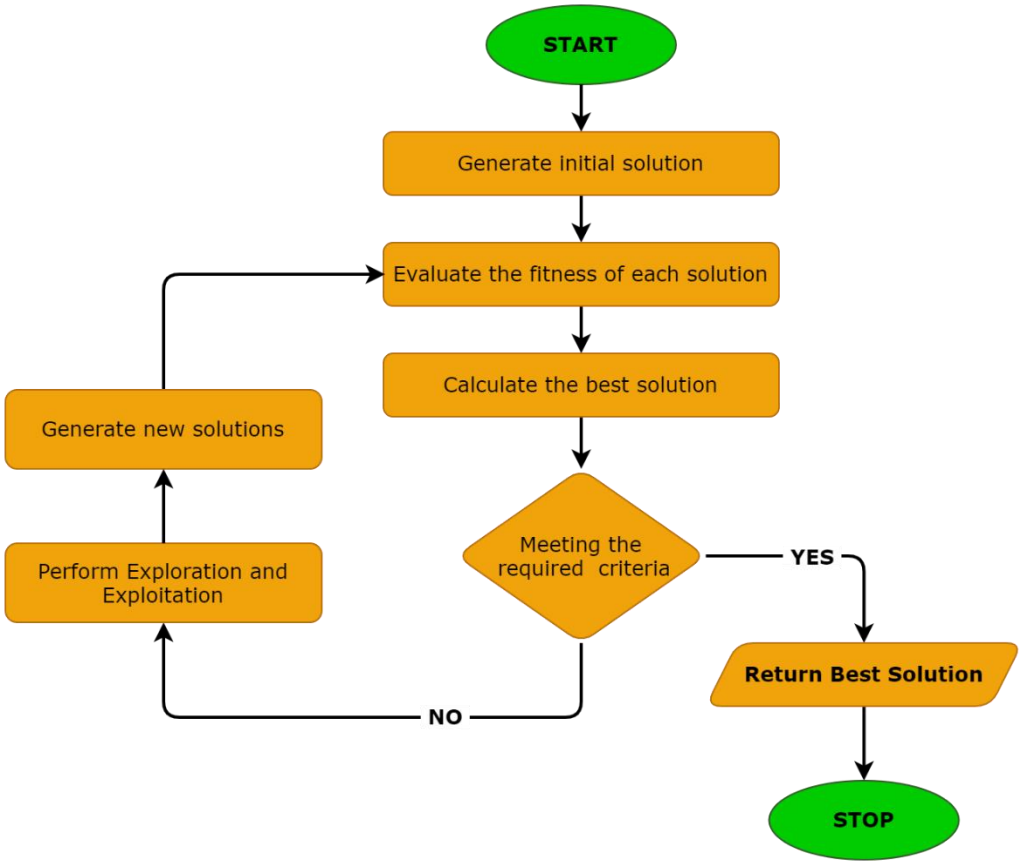


Figure 1. Flow of a typical metaheuristic algorithm

The generalized overview of the typical steps followed in many heuristic algorithms are as follows.

Step 1. Population initialization
Generate an initial set of solutions (population) based on the problem requirements and assign initial values to parameters and variables.

Step 2. Fitness evaluation
Assess the quality of each solution in the population using the objective function or fitness measure associated with the optimization problem. Constrains if any are added while evaluating the fitness of each solution.

Step 3. Global best solution update
Based on the fitness, the top feasible solution is considered as the best solution.

Step 4. Stopping Condition
Define stopping conditions to determine when the algorithm should stop. Common criteria include a maximum number of iterations, reaching a satisfactory solution quality, or a specific level of convergence.

Step 5. Exploration and Exploitation
If the desired solution is not obtained, then create new candidate solutions through operations such as mutation, crossover, random walk or other specific operators relevant to the heuristic algorithm being used.

Step 6. Algorithm termination
Step 1 is a one-time process, whereas steps 2 to 5 are repeated until the termination condition for the algorithm is met.

The effectiveness of any metaheuristic algorithm often depends on careful tuning of parameters and the problem-specific characteristics. Hence, the algorithm parameters must be changed dynamically during the optimization process based on the performance or characteristics of the current solutions. Also, the algorithm has to handle the constraints appropriately and ensure that the generated solutions meet the specified constraints.

3. Proposed Framework - Flow

This section explains the proposed framework which incorporates inverse kinematics solver, 3D trajectory planner and follower, obstacle avoidance for static as well as dynamic obstacles for a redundant robot manipulator. The flow of this proposed framework is shown in Figure 2.

The initial input to the framework encompasses crucial information about the robot manipulator, specifically the Denavit-Hartenberg (DH) parameters. Subsequently, environmental parameters pertaining to the robot's trajectory are considered, including the starting and ending points, obstacle presence and their respective positions. Utilizing these inputs, both the robot and its workspace undergo modelling. The ultimate input integrates the metaheuristic algorithm, functioning as an inverse kinematic solver, 3D path planner, and trajectory follower equipped with static and dynamic obstacle avoidance capabilities

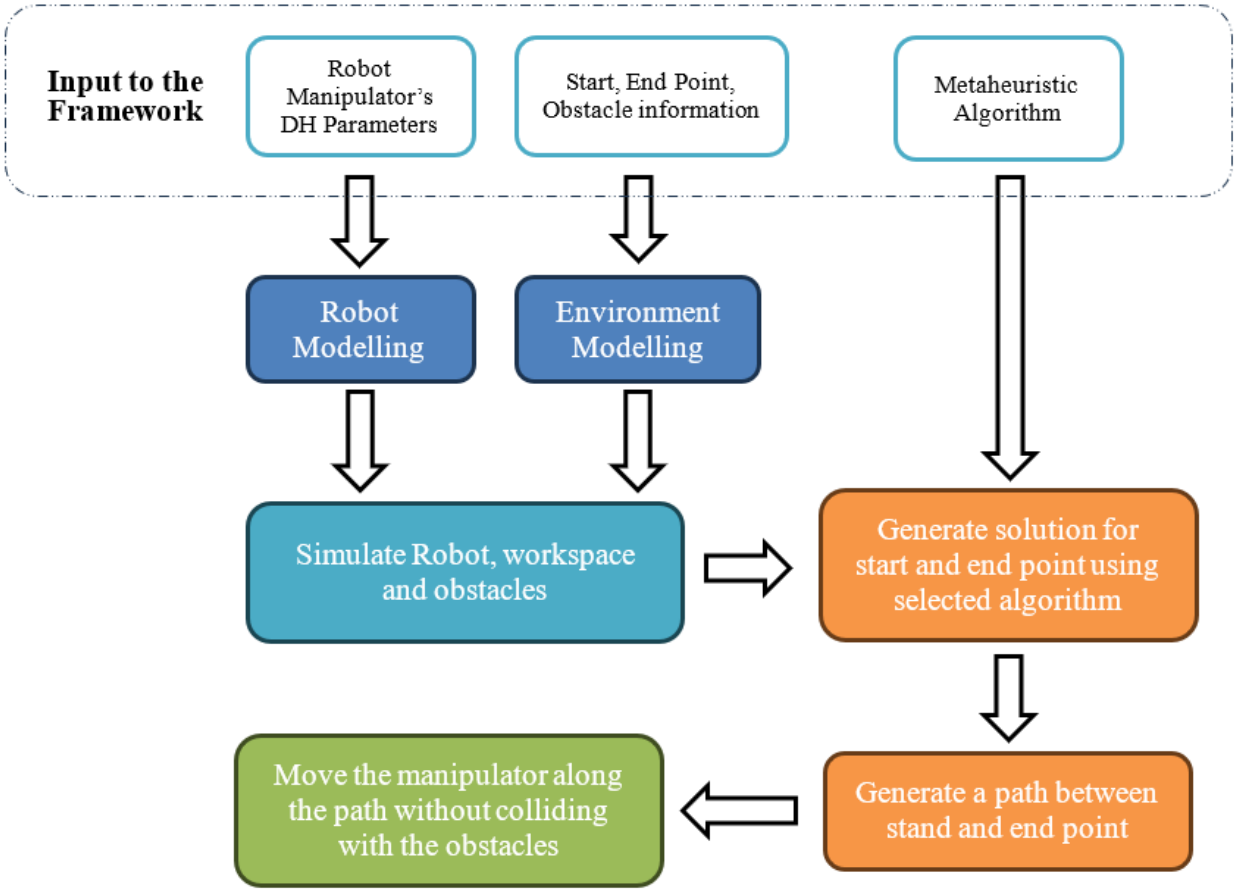


Figure 2. Metaheuristic based framework for redundant robot manipulator

3.1. Robot and Environmental Parameters

The major parameters mentioned in the proposed framework are robot's DH parameters and the environmental parameters.

DH parameters or the Denavit-Hartenberg are a set of standardized parameters used to describe the kinematic structure of a robot manipulator. These parameters facilitate the derivation of transformation matrices between consecutive links in a robotic arm, allowing for the determination of the end-effector's pose based on joint angles. This process is called forward kinematics. The DH parameters consist of four

values for each joint in the robot manipulator. They are link length (a), link twist angle (α), link offset (d) and the joint angle (θ) as shown in Figure 3.

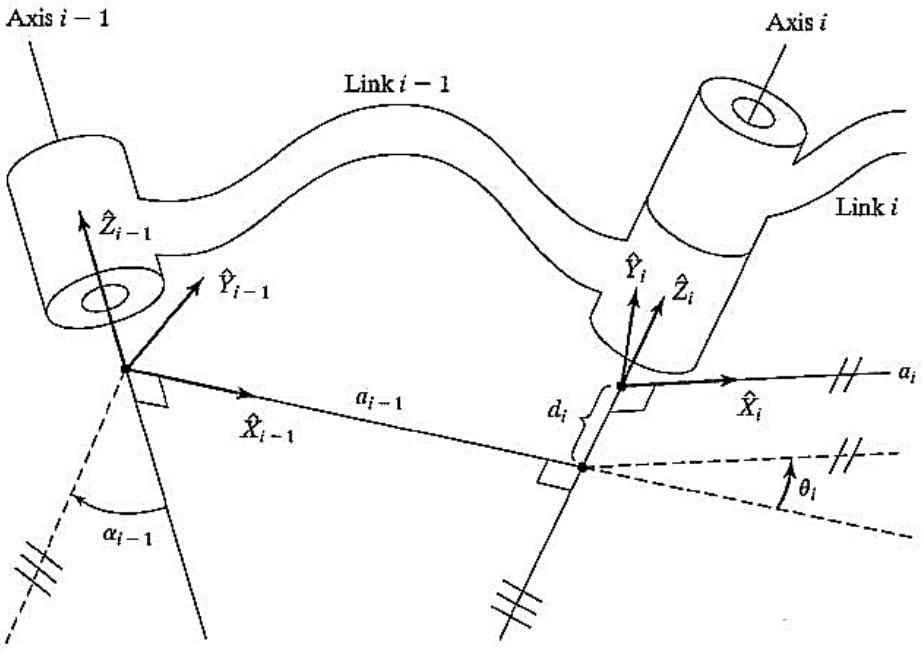


Figure 3. DH parameters - a revolute joint [24]

The link length (a_i) is the distance between the Z-axes of adjacent joints, measured along the common normal. It represents the distance from the i^{th} joint to the $(i+1)^{th}$ joint along the common perpendicular between the Z-axes. The link twist angle (α_i) is the angle between the Z-axes of adjacent joints, measured about the common normal. It represents the twist or rotation between the i^{th} joint and the $(i+1)^{th}$ joint about their common perpendicular. Link offset (d_i) is the offset along the Z-axis from the i^{th} joint to the $(i+1)^{th}$ joint. It represents the distance along the Z-axis from the i^{th} joint to the common perpendicular. Joint angle (θ_i) is the angle of rotation about the common normal, measured from the X-axis of the $(i-1)^{th}$ joint to the X-axis of the i^{th} joint. These parameters are defined with respect to the reference frames assigned to each joint. Typically, the Z-axis is aligned with the joint axis, the X-axis is chosen to form a right-handed coordinate system, and the Y-axis is determined to complete the frame.

The transformation matrix between consecutive links can be derived from these parameters, and by multiplying these matrices, the overall transformation from the base to the end-effector can be obtained which is given by the following equation.

$$ {}^{base}_{ee}T = {}^{base}_{1}T * {}^{1}_{2}T * {}^{2}_{3}T * \dots * {}^{i-1}_{i}T * \dots * {}^{n-1}_{ee}T = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} & p_x \\ r_{yx} & r_{yy} & r_{yz} & p_y \\ r_{zx} & r_{zy} & r_{zz} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1} $$

Here, n represents the number of links and ${}^{i-1}_{i}T$ represents the transformation matrix from link $(i-1)$ to link i which is calculated using equation (2).

$$ {}^{i-1}_{i}T = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

The elements of the transformation matrix can be calculated using the DH parameters of the robot manipulator. Consider a 2-link planar robot with the following DH parameters. The link lengths $a_1, a_2 = 1$; the link twist angles $\alpha_1, \alpha_2 = 0$; link offsets $d_1, d_2 = 0$; joint angle $\theta_1, \theta_2 = 30^o, 45^o$ respectively. By substituting these values, the transformation matrix ${}^{base}_{1}T$ is given by:

$$ {}^{base}_{1}T = \begin{bmatrix} cos\theta_1 & -sin\theta_1 cos\alpha_1 & sin\theta_1 sin\alpha_1 & a_1 cos\theta_1 \\ sin\theta_1 & cos\theta_1 cos\alpha_1 & -cos\theta_1 sin\alpha_1 & a_1 sin\theta_1 \\ 0 & sin\alpha_1 & cos\alpha_1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$= \begin{bmatrix} \cos 30^o & -\sin 30^o \cos 0^o & \sin 30^o \sin 0^o & \cos 30^o \\ \sin 30^o & \cos 30^o \cos 0^o & -\cos 30^o \sin 0^o & \sin 30^o \\ 0 & \sin 30^o & \cos 0^o & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.8660 & -0.5 & 0 & 0.8660 \\ 0.5 & 0.8660 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Similarly, the transformation matrix from link 1 to 2 is given by:

$${}^{1}_{2}T = \begin{bmatrix} \cos 45^o & -\sin 45^o \cos 0^o & \sin 45^o \sin 0^o & \cos 45^o \\ \sin 45^o & \cos 45^o \cos 0^o & -\cos 45^o \sin 0^o & \sin 45^o \\ 0 & \sin 45^o & \cos 0^o & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.7071 & -0.7071 & 0 & 0.7071 \\ 0.7071 & 0.7071 & 0 & 0.7071 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The overall transformation matrix is obtained by multiplying the above two matrices which is given by

$${}^{base}_{2}T = {}^{base}_{1}T * {}^{1}_{2}T = \begin{bmatrix} 0.8660 & -0.5 & 0 & 0.8660 \\ 0.5 & 0.8660 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0.7071 & -0.7071 & 0 & 0.7071 \\ 0.7071 & 0.7071 & 0 & 0.7071 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.2588 & -0.9659 & 0 & 1.125 \\ 0.9659 & 0.2588 & 0 & 1.4659 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For the 2-link planar robot and the DH parameters considered in the example, the end effector position is (1.125,1.4659).

Inverse kinematics is the reverse process of forward kinematics, i.e., it is the process of determining the robot manipulator's joint angles that will lead to the desired end-effector position. Since the complexity of inverse kinematics spikes with increase in robot manipulator's degree of freedom, forward kinematics integrated with metaheuristic algorithms are used as inverse kinematics solver.

The environmental parameters include the start point and end point of the robot manipulator, number of obstacles, their size and position with respect to the robot manipulator. Using the DH parameters and the environmental parameters, the robot and its environment are modelled.

3.2. Metaheuristic Algorithm as Inverse Kinematics Solver

For any robot manipulator, with nL links, inverse kinematics is the process of finding the joint angles $\langle \theta_1, \ldots \theta_{nL} \rangle$ to reach a target point $T(x, y, z)$ within the workspace of the robot manipulator. Due to the complexity of inverse kinematics, these link angles can be found using metaheuristic algorithm and forward kinematics. The block diagram for using forward kinematics with metaheuristic algorithm as inverse kinematics solver is shown in **Figure 4**.

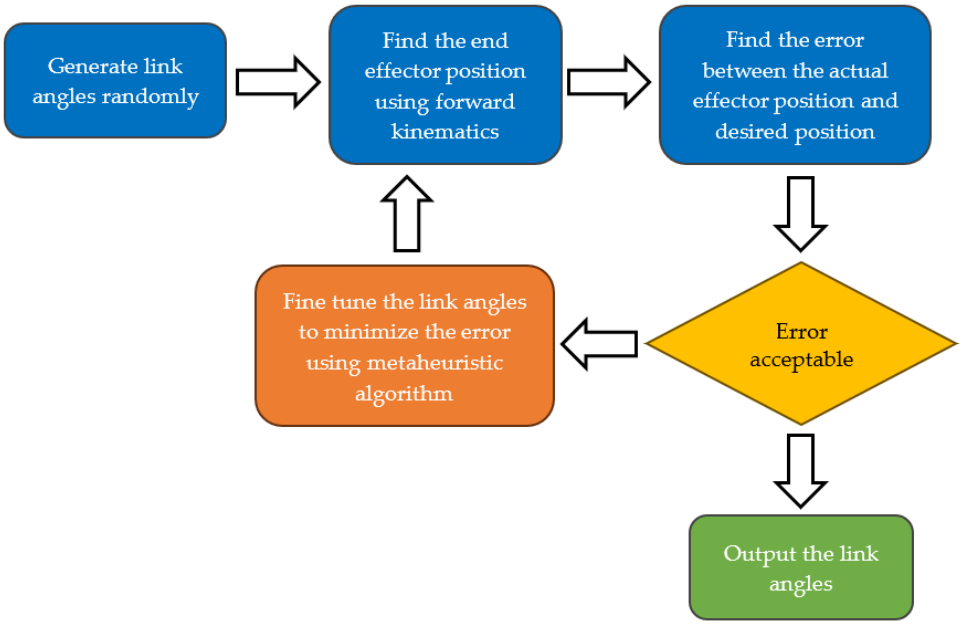


Figure 4. Forward Kinematics + Metaheuristic algorithm as inverse kinematics solver

The first step is to convert the problem of interest, i.e., the inverse kinematics into a maximization or minimization problem. For any random joint angles $\langle\theta_1, \ldots \theta_{nL}\rangle$, $A(x, y, z)$ represent the actual point reached by the robot manipulator which is calculated using the forward kinematics. The 3D distance between the points $T(x, y, z)$ and $A(x, y, z)$ represents the error which needs to be minimized. This minimization problem can be taken as the objective for the metaheuristic algorithm.

The constrains involved in this problem are the limits of the joint angles $\langle\theta_1, \ldots \theta_{nL}\rangle$ and the obstacles. There will be lower and upper limit to the each of the joint angles in the robot manipulator which is provided by the manufacturer. The next constrain is that the robot manipulator must reach the target point $T(x, y, z)$ without colliding with any of the obstacles present in its workspace.

Thus, the inverse kinematics problem is converted into a minimization problem with two constrains which is stated as follows.

Objective: Minimize the 3D distance error between the target point $T(x, y, z)$ and the actual point $A(x, y, z)$ reached by the robot manipulator.

Constrain 1: The generated joint angles $\langle\theta_1, \ldots \theta_{nL}\rangle$ must be within the limits provided by the manufacturer.

Constrain 2: The robot manipulator must not collide with the obstacles present in the workspace at any point of time.

The pseudo code to use any metaheuristic algorithm as inverse kinematic solver is shown in Algorithm 1.

Algorithm 1. Metaheuristic algorithm as inverse kinematics solver

```
Initialize algorithm related parameters
Generate initial solution:
A population with nP solutions, i.e., joint angle vectors with
dimension nL is generated randomly
```

$$\begin{matrix}1\\2\\\vdots\\nP\end{matrix}\begin{bmatrix}\langle\theta_1, \ldots \theta_{nL}\rangle\\\langle\theta_1, \ldots \theta_{nL}\rangle\\\vdots\\\langle\theta_1, \ldots \theta_{nL}\rangle\end{bmatrix}$$

```
Apply forward kinematics and generate A(x,y,z) for each of the nP
solution
Fitness evaluation:
Evaluate each solution by finding the error between the target point
T(x,y,z) and the achieved point A(x,y,z) by satisfying the constrains
Assign the solution with minimum error as the best solution
While (stopping criteria is not met)
{
    Perform exploration and exploitation
    Generate new solutions
    Apply forward kinematics for the new solutions (as in step 3)
    Fitness evaluation (as in step 4)
    Assign the solution with minimum error as the best solution
    Update algorithm related parameters (if required)
}
```

3.3. Trajectory Planner

The 3D path is generated using trapezoidal velocity profile. The path is generated based on the start / end point, and the intermediate points. By using a trapezoidal velocity profile, trajectories can be generated that ensure smooth and efficient motion while adhering to the acceleration and deceleration limits of the robot. This approach minimizes jerky movements and reduces the risk of overshooting or oscillations, leading to more precise and controlled motion. Trajectories with trapezoidal velocity profiles are commonly employed in various robotic applications, including industrial automation, pick-and-place tasks, and robotic manipulation, where smooth and accurate motion is crucial for task performance and safety [25]. Once the path is generated, the joint angles required to reach each and every point in the path is obtained using the inverse kinematics solver as explained in section 3.2.

3.4. Dynamic Obstacle Avoidance

The joint angles required for the robot manipulator to reach each and every point in the path are generated using metaheuristic algorithm. Hence, the robot manipulator continues to follow the generated trajectory unless otherwise faced with an obstacle.

3.4.1. Obstacle Colliding with Robot Manipulator

Any moving obstacle within the workspace of the robot manipulator could colliding with one of more links of the robot manipulator. During such instance, a new solution, i.e., new joint angles for the robot manipulator have to be generated for that particular target point, as explained in the section 3.2. The path of the robot manipulator remains unchanged. If the solution for the subsequent points in the trajectory undergoes collision, then the solution for those points is also generated using the metaheuristic algorithm. This is illustrated in Figure 5.

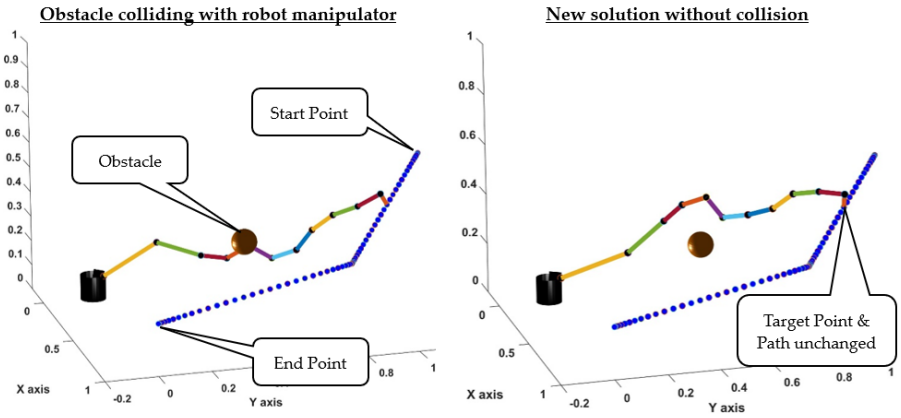


Figure 5. Obstacle colliding with robot manipulator

3.4.2. Obstacle in the path of the Robot Manipulator

The second scenario is where the obstacle could appear in the path of the robot manipulator. In this case the path has to replanned in such a way that it avoids the obstacle and reaches the destination. This new optimized path is also generated using the metaheuristic algorithm. Furthermore, for this new path, new solutions have to be generated for the robot manipulator. This scenario is illustrated in Figure 6.

To ensure a smooth and optimized trajectory, a special logic is implemented which is as follows. The path of the robot manipulator consists of a start / end point, one or more intermediate stop points. Using these few points, the entire trajectory is constructed. Consider a case with one start point, one end point and two intermediate stop points. To ensure smooth movement of the robot manipulator, using these four points, a path with 100 points is generated. Of these 100 points, if the obstacle has collided with nC points in the path, in addition to the collided nC points, points before the collision point and after the collision point are taken. That is, a total of $3 * nC$ points are taken from the path and replaced by new points which are generated using metaheuristic algorithm. This scenario is illustrated in Figure 7. The steps for generating an optimized new path using metaheuristic algorithm is shown in Algorithm 2.

Algorithm 2. Metaheuristic algorithm as 3D path planner

```
1.  Input: 3*nObsP points from current path
2.  Initialize algorithm related parameters
3.  newPath[3*nObsP] = 0
4.  for i = 1 to 3*nObsP
5.  {
6.      Generate initial solution:
        A population with nP 3D points are generated randomly
7.      Calculate the distance between the generated points and the i^th input point
8.      Fitness evaluation:
        The point which is nearest to the i^th input point is taken as the best
        solution with the constrain that it is not colliding with the obstacle
9.      While (stopping criteria is not met)
10.     {
11.         Perform exploration and exploitation
12.         Generate new solutions
13.         Calculate the distance between the i^th input point and the new
            solutions (as in step 7)
14.         Fitness evaluation (as in step 8)
15.         Assign the solution with minimum distance as the best solution
16.         Update algorithm related parameters (if required)
17.     }
18.     newPath[i] = best solution
19. }
20. Output: newPath with 3*nObsP points
```

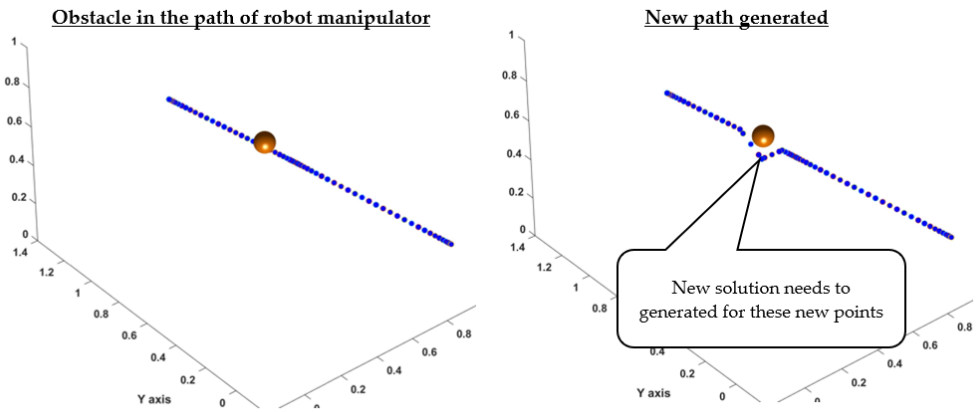


Figure 6. Obstacle in the path of robot manipulator

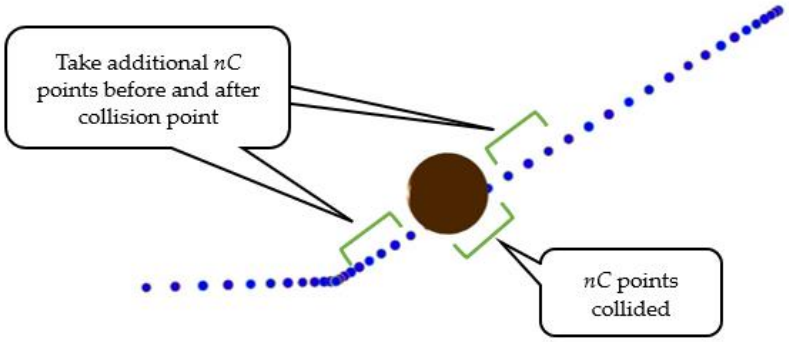


Figure 7. 3D path planner after colliding with the obstacle

The key features of this proposed framework are as follows:

- Metaheuristic algorithm can handle high-dimensional search spaces, making them suitable for redundant robot manipulators
- Many metaheuristic algorithms are computationally efficient and suitable for real-time implementation as they can generate obstacle-avoidance trajectories quickly, making them suitable for applications where rapid decision-making is essential, such as human-robot collaboration or agile manufacturing environments
- The proposed framework offers a complete solution for redundant robot manipulator's 3D path planning and path following with static and dynamic obstacle avoidance
- The proposed framework is generalised and works for any type of robot manipulator, with any number of links for any trajectory within the workspace of the robot manipulator
- Furthermore, any metaheuristic algorithm that is appropriate for a specific scenario or application can be used with the proposed framework

4. Implementation and Results

The proposed framework is implemented using MATLAB R2023a. A Graphical User Interface (GUI), as shown in Figure 8, is developed in MATLAB using App Designer[1]. The GUI is grouped into seven parts which are listed in Table 1**Error! Reference source not found.**.

The "Select Input Parameters" button is for getting input parameters of the robot manipulator and modelling it. Clicking the button will open a separate dialog box as shown in Figure 9.

The number of links for the robot manipulator can be selected. The default values of link offset (d), link length (a), link twist angle (α), lower and upper limits of the joint angle (θ) will be populated in the table. The table is editable and user can enter their own parameters of their choice. Clicking the "Save

[1] https://in.mathworks.com/products/matlab/app-designer.html

Parameters" button completes the robot modelling. The presence of obstacles in the environment can be selected using the "No. of Obstacles" drop down menu. The obstacle's type could be static or dynamic which can also be chosen using the radio buttons. And finally, the metaheuristic algorithm, the brain of the framework, is selected. This algorithm acts as inverse kinematics solver, 3D path planner, trajectory follower with dynamic obstacle avoidance. Particle Swarm Optimization (PSO), Bat Algorithm (BAT), Gravitational Seach Algorithm (GSA), Whale Optimization Algorithm (WOA), Aquilla Optimizer (AO), enhanced exploration and exploitation GSA (e^3GSA) are the few algorithms that are implemented in the proposed framework. After selecting the required algorithm, clicking "OK" button closes the dialog box and returns to the main window. The selected parameters are displayed in the status box at the top of the window.

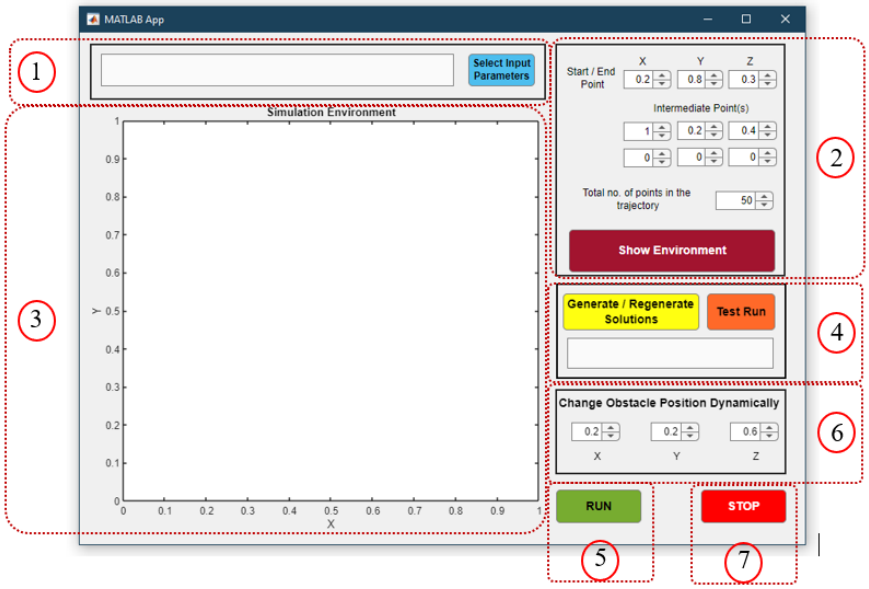


Figure 8. GUI – Proposed Framework

Table 1. Parts in the Proposed Framework GUI

Part	Usage
1	Input part for getting Robot's DH parameters, metaheuristic algorithm selection, obstacle's presence
2	3D Path Planner: with inputs such as start / end point, intermediate points, total number of points in the trajectory
3	Simulation Environment Window that displays the simulation of robot manipulator
4	Inverse kinematics solver
6	Input part for getting the obstacle's position (dynamic obstacle avoidance part)
5, 7	Start and stop of simulation

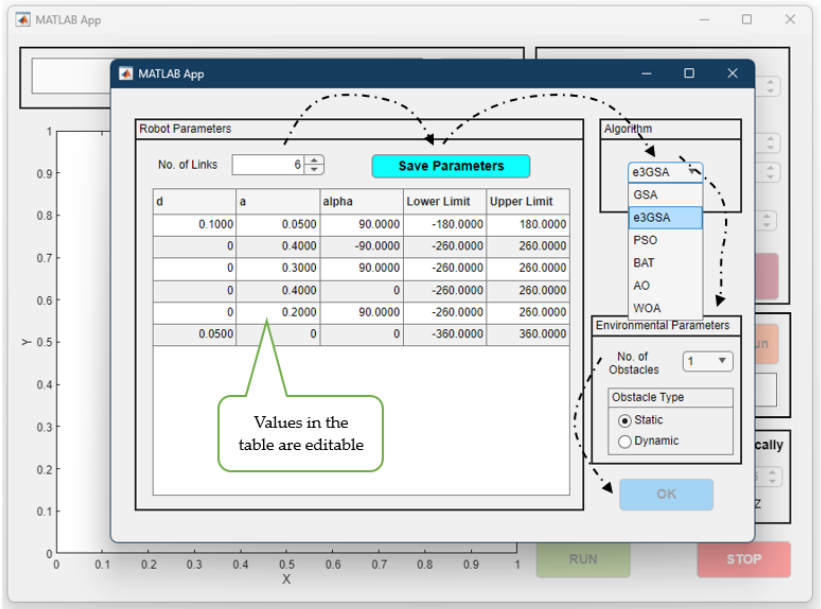


Figure 9. GUI – Parameter selection window

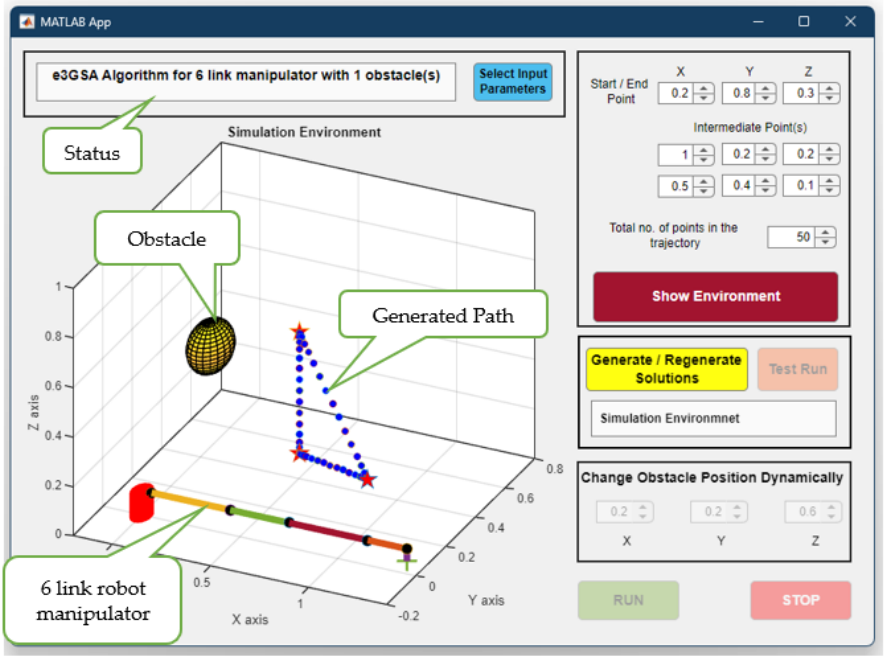


Figure 10. GUI – Generated path with robot manipulator and obstacle

The path that needs to followed by the robot manipulator can be generated using the data from the second part of the GUI. The start / end point, and two intermediate points and the total number of points in the trajectory are given by the user. Using these, the path is generated by trapezoidal velocity profile. The total number of points generated is based on the user's input in the field "Total no. of points in the trajectory". By clicking the "Show Environment" button, the generated path, along with the robot manipulator and the obstacle can be viewed in the simulation environment section of the GUI as shown in Figure 10.

The next part is to find solutions for the robot manipulator to follow the generated path. This can be achieved by using the selected metaheuristic algorithm as inverse kinetics solver. The steps for achieving this is mentioned under section 3.2, Algorithm 1. Clicking "Generate / Regenerate Solutions" button will create solution. i.e., robot manipulator's joint angles $\langle \theta_1, \ldots \theta_{nL} \rangle$ for each and every point in path. The solution of the robot manipulator for the generated path can be visualized by clicking the "Test Run" button.

For simulation, a 6-link robot manipulator is modeled with the parameters mentioned in Table 2. The environmental parameters used for the simulation are listed in Table 3. The metaheuristic algorithm used is e^{3}GSA [25].

Table 2. Robot Manipulator's Parameters

Parameter	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6
Link offset (d *in mm*)	0.1	0.0	0.0	0.0	0.0	0.05
Link length (a *in mm*)	0.05	0.4	0.3	0.4	0.2	0.0
Link twist angle (α *in degree*)	90	-90	90	0	90	0
Lower limit of the joint angle (θ *in degree*)	-180	-260	-260	-260	-260	-360
Upper limit of the joint angle (θ *in degree*)	+180	+260	+260	+260	+260	+360

Table 3. Environmental Parameters

Parameter	Value
Start / End Point	⟨0.2, 0.8, 0.3⟩
Intermediate Point 1	⟨1.0, 0.2, 0.2⟩
Intermediate Point 2	⟨0.5, 0.4, 0.1⟩
No. of points in the trajectory	50
Obstacle Presence	Yes, Dynamic

The solution generated for the robot manipulator using e^{3}GSA algorithm at selected points of the trajectory are shown in Figure 11. The solution can be regenerated by clicking "Generate / Regenerate Solutions" button any number of times. Every time the algorithm generates different solution for the robot manipulator to follow the generated path.

The final part of the GUI is the dynamic obstacle avoidance. With the solution generated for the entire path, "Run" button will simulate the robot manipulator to follow the generated path continuously. During

this time, the obstacle's position can be varied dynamically using the X, Y, Z spinners which are marked 6 in Figure 8. The initial solutions are generated with the obstacle in the position $\langle 0.2, 0.2, 0.6 \rangle$. After clicking the "Run" button, the robot manipulator starts following the path based on the solution, i.e., link angles generated by the selected e^{3}GSA algorithm. Now, the position of the obstacle is changed as the robot manipulator moves along the path. The proposed framework test whether the current solution of the robot manipulator collides with the new position of the obstacle throughout its movement along the path. If the collision is predicted based on the new position of the obstacle, then the solution of the robot manipulator is regenerated using the selected metaheuristic algorithm such that it avoids the obstacle along its path. Figure 12 shows the robot manipulator at different points in the trajectory for two different scenarios, viz., obstacle position at $\langle 0.2, 0.2, 0.6 \rangle$ and second with obstacle moved to $\langle 0.1, 0.15, 0.4 \rangle$. The figure clearly shows that the robot manipulator's solutions are regenerated to avoid collision with the obstacle when it is moved. The number in the right top of the figure represent the i^{th} point in the trajectory.

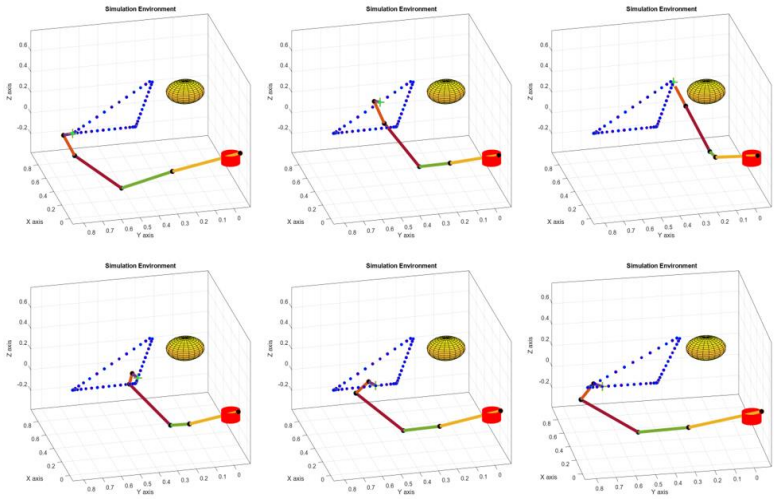
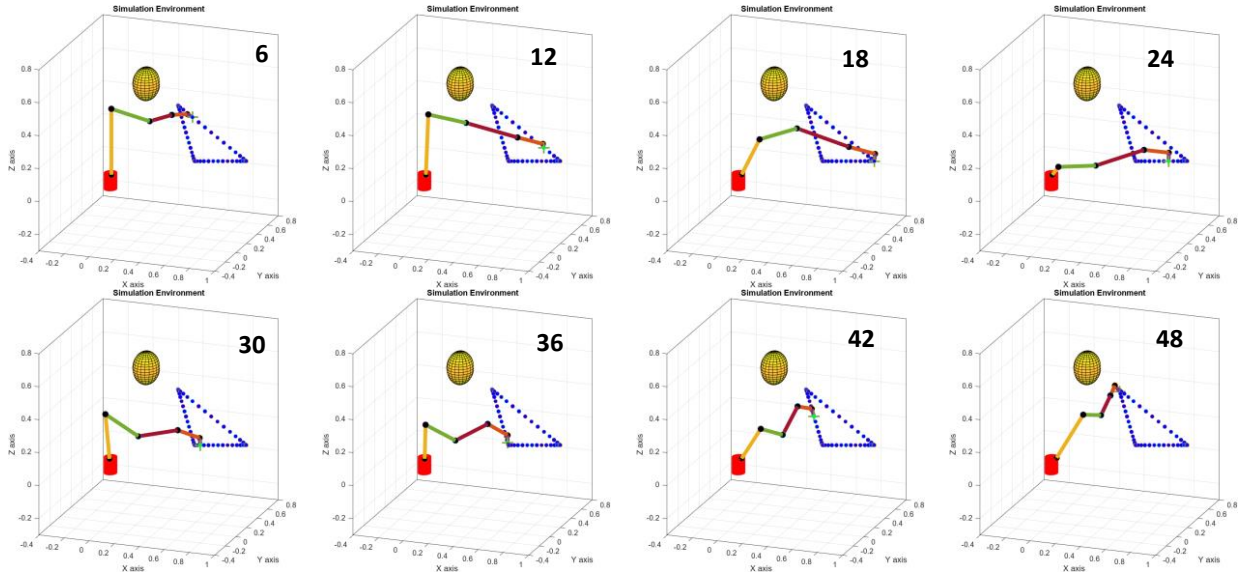
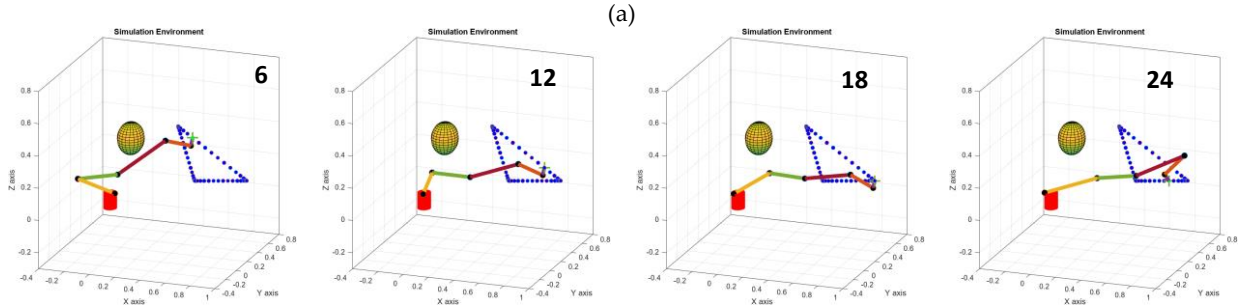


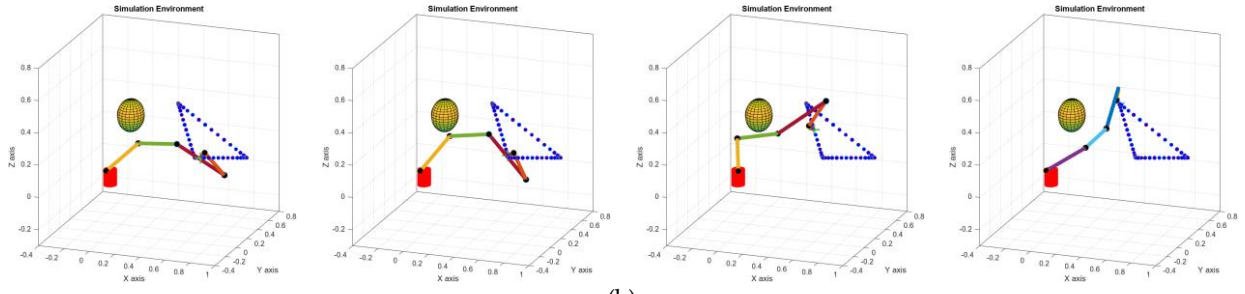
Figure 11. Robot Manipulator following the generated path with static obstacle



(a)

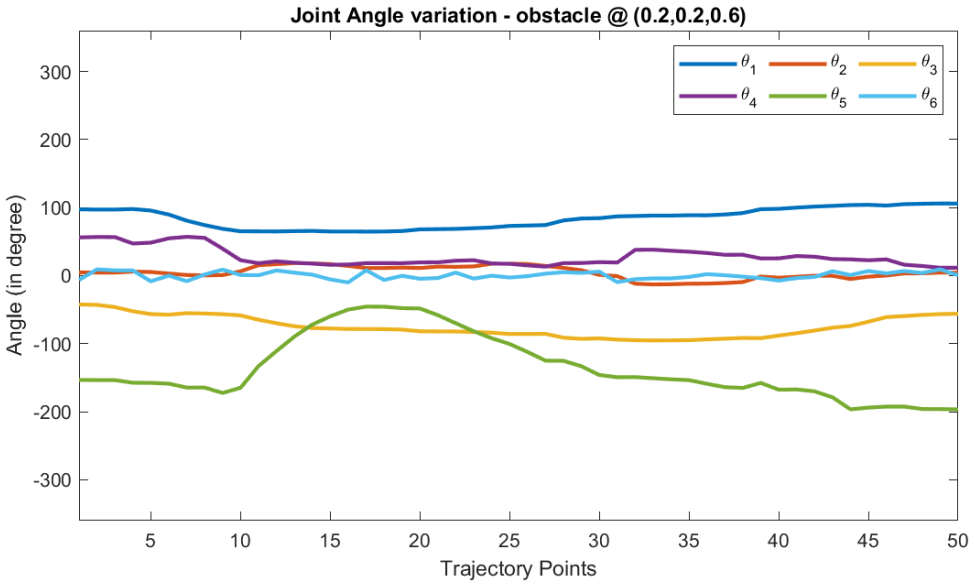


30 36 42 48

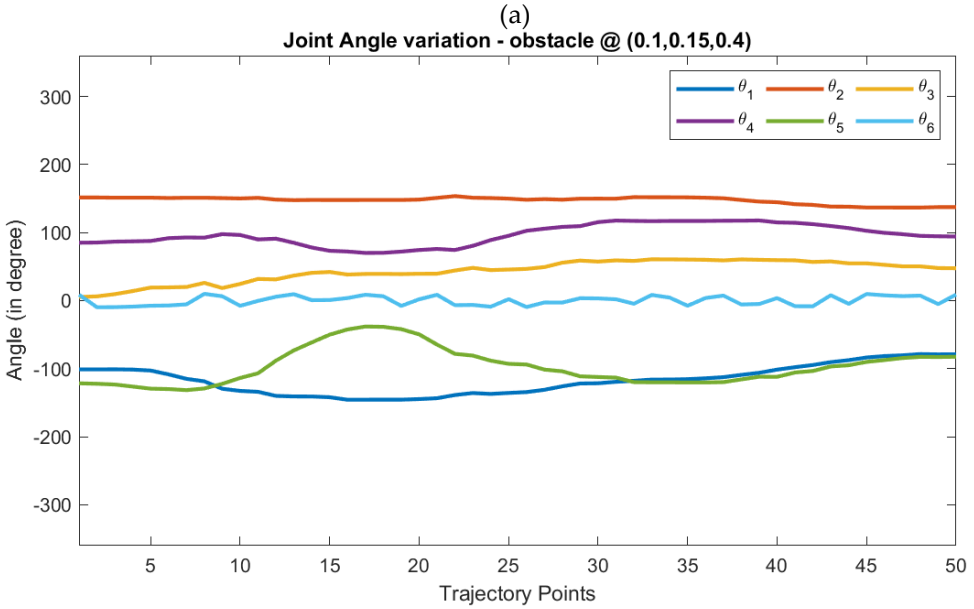


(b)

Figure 12. Robot Manipulator following the generated path with dynamic obstacle (a) obstacle position $\langle \mathbf{0.2}, \mathbf{0.2}, \mathbf{0.6} \rangle$ (b) obstacle position $\langle \mathbf{0.1}, \mathbf{0.15}, \mathbf{0.4} \rangle$



(a)



(b)

Figure 13. Robot Manipulator's joint angles with dynamic obstacle (a) obstacle position $\langle \mathbf{0.2}, \mathbf{0.2}, \mathbf{0.6} \rangle$ (b) obstacle position $\langle \mathbf{0.1}, \mathbf{0.15}, \mathbf{0.4} \rangle$

To study the characteristics of the solution generated, the joint angles of the robot manipulator are recorded throughout the trajectory. The variation of the joint angles when the obstacle is in the position ⟨0.2,0.2,0.6⟩ and when the obstacle is moved to ⟨0.1,0.15,0.4⟩ are plotted in Figure 13. The joint angle variation plot shown in Figure 13 shows that there is no abrupt variation of the joint angles even when the obstacle is moved. This shows that the proposed framework generates optimized solution avoiding dynamic obstacle exerting with less effort on the robot manipulator.

5. Conclusion

This research article proposes a generalized framework for redundant robot manipulators, leveraging metaheuristic algorithms as versatile tools for inverse kinematics solving, 3D trajectory planning, and dynamic obstacle avoidance. This innovative framework is designed to be adaptable for any type of robot with any number of links, ensuring broad applicability across various robotic systems. It empowers users with the flexibility to generate customized 3D paths by adjusting start points, end points, and intermediate stop points, thereby catering to specific task requirements.

One of the significant advantages of this framework is its ability to handle dynamic and static obstacles. Users can alter the positions of these obstacles within the environment, enabling the robot to adapt to changing conditions and ensuring safe navigation. This adaptability is crucial for real-world applications where the environment may be unpredictable or subject to change. The core strength of the proposed framework lies in its integration with metaheuristic algorithms. These algorithms, known for their robustness in handling high-dimensional search spaces and avoiding local optima, can be easily implemented and tailored to specific scenarios or applications. This flexibility allows for the selection of the most suitable metaheuristic algorithm based on the specific needs of the task, enhancing the overall efficiency and effectiveness of the robotic manipulator.

The comprehensive nature of this framework addresses the complete spectrum of challenges faced by redundant robot manipulators. From solving complex inverse kinematics problems to planning precise 3D trajectories and navigating through dynamic obstacles, this framework offers a holistic solution. Its versatility and adaptability make it a valuable tool for a wide range of industrial and research applications.

To validate the framework's efficacy, extensive simulations were conducted in MATLAB. These simulations demonstrated the framework's capability to generate optimal joint configurations and trajectories, dynamically adapting to changing environments. The results underscore the potential of this generalized framework to significantly improve the performance and reliability of redundant robot manipulators in diverse and dynamic settings. Thus, this research paves the way for more advanced and adaptable robotic systems, capable of performing complex tasks with high precision and efficiency.

6. Future Work

Moving forward, several avenues for future research emerge from this work. One key area is the performance and scalability of different metaheuristic algorithms. By examining how these algorithms perform under various conditions and tasks, researchers can better understand their suitability for specific robotic applications and environments. This could involve comparative studies to identify which algorithms are most efficient and effective in different scenarios, ultimately leading to more optimized solutions for robotic manipulation.

Another important direction is the development of hardware implementations and experimental validations in real-world environments. While simulations in MATLAB have demonstrated the framework's potential, real-world testing is crucial for assessing its practical viability. By deploying the framework in industrial settings, researchers can observe its performance in actual operations, identify any limitations or areas for improvement, and refine the system accordingly. This step is essential for bridging the gap between theoretical research and practical application, ensuring that the framework can be reliably used in diverse industrial contexts.

Furthermore, this research opens up new possibilities for advancing dynamic obstacle avoidance in redundant manipulators. Exploring innovative approaches to obstacle detection and avoidance, integrating advanced sensors and machine learning techniques, could significantly enhance the framework's capabilities. By continuing to innovate and expand on the current research, future work can contribute to the development of more sophisticated and adaptable robotic systems.

Overall, this work lays the groundwork for future innovations in robotic manipulation, offering a robust foundation for further exploration and development. By pursuing these research directions, the field can achieve greater advancements in dynamic obstacle avoidance and overall robotic efficiency and effectiveness.

References

[1] Sherif I. Abdelmaksoud, Mohammed H. Al-Mola, Ghulam E. Mustafa Abro and Vijanth S. Asirvadam, "In-Depth Review of Advanced Control Strategies and Cutting-Edge Trends in Robot Manipulators: Analyzing the Latest Developments and Techniques", in *IEEE Access*, Electronic ISSN: 2169-3536, pp. 47672-47701, vol. 12, 2024, Published by IEEE, DOI: 10.1109/ACCESS.2024.3383782, Available: https://ieeexplore.ieee.org/document/10486927.

[2] Yaowen Zhang, Yechao Liu, Baoshi Cao, Yang Liu, Boyu Ma *et al*, "Joint Limit Optimal Inverse Kinematics of the 7-DoF Manipulator with Link Offset based on Semi-analytical Solution", in *Proceedings of the 2021 IEEE International Conference on Robotics and Biomimetics, ROBIO*, 27-31 December 2021, Sanya, China, E-ISBN: 978-1-6654-0535-5, Print ISBN: 978-1-6654-0536-2, DOI: 10.1109/ROBIO54168.2021.9739314, Published by IEEE, Available: https://ieeexplore.ieee.org/document/9739314.

[3] Mingde Gong, Xiangdong Li and Lei Zhang, "Analytical Inverse Kinematics and Self-Motion Application for 7-DOF Redundant Manipulator", *IEEE Access*, E-ISSN: 2169-3536, vol. 7, 2019, Published by IEEE, DOI: 10.1109/ACCESS.2019.2895741, Available: https://ieeexplore.ieee.org/document/8630999.

[4] Masayuki Shimizu, Hiromu Kakuya, Woo-Keun Yoon, Kosei Kitagaki and Kazuhiro Kosuge, "Analytical Inverse Kinematics for 7 DOF Redundant Manipulators with Joint Limits", *Journal of the Robotics Society of Japan*, Online ISSN: 1884-7145, Print ISSN: 0289-1824, vol. 25, no. 4, 2007, Published by The Robotics Society of Japan, DOI: 10.7210/jrsj.25.606, Available: https://www.jstage.jst.go.jp/article/jrsj1983/25/4/25_4_606/_article/-char/en.

[5] T. Asfour and R. Dillmann, "Human-like Motion of a Humanoid Robot Arm Based on a Closed-Form Solution of the Inverse Kinematics Problem", in *IEEE International Conference on Intelligent Robots and Systems*, 27-31 October 2003, Las Vegas, NV, USA, Print ISBN: 0-7803-7860-1, DOI: 10.1109/iros.2003.1248841, Published by IEEE, Available: https://ieeexplore.ieee.org/document/1248841.

[6] Jonghoon Park, Youngjin Choi, Wan Kyun Chung and Youngil Youm, "Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators", in *Proceedings of the IEEE International Conference on Robotics and Automation*, 21-26 May 2001, Seoul, South Korea, Print ISBN: 0-7803-6576-3, Print ISSN: 1050-4729, DOI: 10.1109/robot.2001.933249, Published by IEEE, Available: https://ieeexplore.ieee.org/document/933249.

[7] Tomomichi Sugihara, "Robust solution of prioritized inverse kinematics based on Hestenes-Powell multiplier method", in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 14-18 September 2014, Chicago, IL, USA, Print ISSN: 2153-0858, Electronic ISSN: 2153-0866, DOI: 10.1109/IROS.2014.6942607, Published by IEEE, Available: https://ieeexplore.ieee.org/document/6942607.

[8] Kaushik Kumar, Divya Zindani and J. Paulo Davim, *Optimizing Engineering Problems through Heuristic Techniques*, 1st ed. Boca Raton, US: CRC Press, 2019, eBook ISBN: 9781351049580, DOI: 10.1201/9781351049580.

[9] Sachin Desale, Akhtar Rasool, Sunil Andhale and Priti Rane, "Heuristic and Meta-Heuristic Algorithms and Their Relevance to the Real World: A Survey", *International Journal of Computer Engineering in Research Trends*, Online ISSN: 2349-7084, vol. 351, no. 5, 2015, Published by IJCERT Publication House, Available: https://www.ijcert.org/issue_des.php?id=317.

[10] Omid Bozorg-Haddad, Mohammad Solgi and Hugo A. Loáiciga, *Meta-Heuristic and Evolutionary Algorithms for Engineering Optimization*, 1st ed. New Jersey, US: John Wiley & Sons, 2017, Print ISBN: 9781119386995, Online ISBN: 9781119387053, DOI: 10.1002/9781119387053, Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9781119387053.

[11] Shaher Momani, Zaer S. Abo-Hammour and Othman M. K. Alsmadi, "Solution of inverse kinematics problem using genetic algorithms", *Applied Mathematics and Information Sciences*, Online ISSN: 23250399, vol. 10, no. 1, 2016, Published by Natural Sciences Publishing, DOI: 10.18576/amis/100122, Available: https://www.naturalspublishing.com/files/published/07761y0s0f4pv8.pdf.

[12] Andreas C. Nearchou, "Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm", *Mechanism and Machine Theory*, Print ISSN: 0094-114X, Online ISSN: 1873-3999, vol. 33, no. 3, 1998, Published by Elsevier, DOI: 10.1016/S0094-114X (97)00034-7, Available: https://www.sciencedirect.com/science/article/abs/pii/S0094114X97000347.

[13] Goh Shyh Chyan and S. G. Ponnambalam, "Obstacle avoidance control of redundant robots using variants of particle swarm optimization", *Robotic and Computer-Integrated Manufacturing*, Print ISSN: 0736-5845, Online ISSN: 1879-2537, vol. 28, no. 2, 2012, Published by Elsevier, DOI: 10.1016/j.rcim.2011.08.001, Available: https://www.sciencedirect.com/science/article/pii/S0736584511000974.

[14] Ze Fan Cai, Dao Ping Huang and Yi Qi Liu, "Inverse Kinematics of Multi-joint Robot Based on Particle Swarm Optimization Algorithm", *Journal of Automation and Control Engineering*, Print ISSN: 2301-3702, 2016, Published by Science and Engineering Research Support Society, DOI: 10.18178/joace.4.4.305-308, Available: https://www.joace.org/uploadfile/2015/1023/20151023022156323.pdf.

[15] Ahmed El-Sherbiny, Mostafa A. Elhosseini and Amira Y. Haikal, "A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm", *Applied Soft Computing Journal*, Print ISSN: 1568-4946, Online ISSN: 1872-

9681, vol. 73, 2018, Published by Elsevier, DOI: 10.1016/j.asoc.2018.08.028, Available: https://www.sciencedirect.com/science/article/pii/S1568494618304885?via%3Dihub.

[16] Serkan Dereli and Raşit Köker, "A meta-heuristic proposal for inverse kinematics solution of 7-DOF serial robotic manipulator: quantum behaved particle swarm algorithm", *Artificial Intelligence Review*, Electronic ISSN: 1573-7462, Print ISSN: 0269-2821, vol. 53, no. 2, 2020, Published by Springer, DOI: 10.1007/s10462-019-09683-x, Available: https://link.springer.com/article/10.1007/s10462-019-09683-x.

[17] Nizar Rokbani, Seyedali Mirjalili, Mohammed Slim and Adel M. Alimi, "A beta salp swarm algorithm meta-heuristic for inverse kinematics and optimization", *Applied Intelligence*, Electronic ISSN: 1573-7497, Print ISSN: 0924-669X, vol. 52, no. 9, 2022, Published by Springer, DOI: 10.1007/s10489-021-02831-3, Available: https://link.springer.com/article/10.1007/s10489-021-02831-3.

[18] Ganesan Kanagaraj, Shahul Abdul Rahim Sheik Masthan and Vincent F. Yu, "Inverse kinematic solution of obstacle avoidance redundant Robot manipulator by Batalgorithms", *International Journal of Robotics and Automation*, Online ISSN: 1925-7090, Hardcopy ISSN: 0826-8185, vol. 36, no. 1, 2021, Published by ACTA Press, DOI: 10.2316/J.2021.206-0425, Available: https://www.actapress.com/PaperInfo.aspx?paperId=50680.

[19] Mostafa Bayati, "Using cuckoo optimization algorithm and imperialist competitive algorithm to solve inverse kinematics problem for numerical control of robotic manipulators", in *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, ISSN: 0959-6518, Online ISSN: 2041-3041, vol. 229, no. 5, 2015, DOI: 10.1177/0959651814568364, Available: https://journals.sagepub.com/doi/10.1177/0959651814568364.

[20] Mustafa Ayyıldız and Kerim Çetinkaya, "Comparison of four different heuristic optimization algorithms for the inverse kinematics solution of a real 4-DOF serial robot manipulator", *Neural Computing and Applications*, Electronic ISSN: 1433-3058, Print ISSN: 0941-0643, vol. 27, no. 4, 2016, Published by Springer, DOI: 10.1007/s00521-015-1898-8, Available: https://link.springer.com/article/10.1007/s00521-015-1898-8.

[21] Ahmed El-Sherbiny, Mostafa A. Elhosseini and Amira Y. Haikal, "A comparative study of soft computing methods to solve inverse kinematics problem", *Ain Shams Engineering Journal*, Online ISSN: 2090-4495, vol. 9, no. 4, 2018, DOI: 10.1016/j.asej.2017.08.001, Available: https://www.sciencedirect.com/science/article/pii/S2090447917300965.

[22] Ganesan Kanagaraj, SAR Sheik Masthan and Vincent F. Yu, "Meta-Heuristics Based Inverse Kinematics of Robot Manipulator's Path Tracking Capability Under Joint Limits", *Mendel*, Online ISSN: 2571-3701, vol. 28, no. 1, 2022, Published by Brno University of Technology, DOI: 10.13164/mendel.2022.1.041, Available: https://mendel-journal.org/index.php/mendel/article/view/173.

[23] Mahmoud. A. A. Mousa, Abdelrahman T. Elgohr and Hatem A. Khater, "Trajectory Optimization for a 6 DOF Robotic Arm Based on Reachability Time", *Annals of Emerging Technologies in Computing (AETiC)*, Print ISSN: 2516-0281, Online ISSN: 2516-029X, vol. 8, no. 1, pp. 22–35, 2024, DOI: 10.33166/AETiC.2024.01.003, Available: https://aetic.theiaer.org/archive/v8/v8n1/p3.html.

[24] John J. Craig, *Introduction to Robotics: Mechanics and Control*, 3rd ed. London, UK: Pearson/Prentice Hall, 2005, ISBN: 0-13-123629-6.

[25] Hyun Joong Yoon, Seong Youb Chung, Han Sol Kang and Myun Joong Hwang, "Trapezoidal motion profile to suppress residual vibration of flexible object moved by robot", *Electronics*, Electronic ISSN: 2079-9292, vol. 8, no. 1, 2019, Published by MDPI, DOI: 10.3390/electronics8010030, Available: https://www.mdpi.com/2079-9292/8/1/30.

[26] Sheik Masthan S A R, G Kanagaraj and Vincent F Yu, "Gravitation search-based hybrid algorithm for solving inverse kinematics of an n-link redundant manipulator", *Journal of Computational Design and Engineering*, Online ISSN: 2288-5048, vol. 10, no. 5, pp. 2019-2035, 2023, Published by Oxford University Press, DOI: 10.1093/jcde/qwad087, Available: https://academic.oup.com/jcde/article/10/5/2019/7285804.