An Analysis Review: Optimal Trajectory for 6-DOF-based Intelligent Controller in Biomedical Application

Kian Raheem qasim [1]
University of information technology and Communication
kian@uoitc.edu.iq

Esam T. Yassen
College of Computer Science and Information Technology
co.esamtah@uoanbar.edu.iq

Yousif I. Al Mashhadany
Department of Electrical Engineering
University of Anbar

yousif.mohammed@uoanbar.edu.iq

ABSTRACT: With technological advancements and the development of robots have begun to be utilized in numerous sectors, including industrial, agricultural, and medical. Optimizing the path planning of robot manipulators is a fundamental aspect of robot research with promising future prospects. The precise robot manipulator tracks can enhance the efficacy of a variety of robot duties, such as workshop operations, crop harvesting, and medical procedures, among others. Trajectory planning for robot manipulators is one of the fundamental robot technologies, and manipulator trajectory accuracy can be enhanced by the design of their controllers. However, the majority of controllers devised up to this point were incapable of effectively resolving the nonlinearity and uncertainty issues of high-degree freedom manipulators in order to overcome these issues and enhance the track performance of high-degree freedom manipulators. Developing practical path-planning algorithms to efficiently complete robot functions in autonomous robotics is critical. In addition, designing a collision-free path in conjunction with the physical limitations of the robot is a very challenging challenge due to the complex environment surrounding the dynamics and kinetics of robots with different degrees of freedom (DoF) and/or multiple arms. The advantages and disadvantages of current robot motion planning methods, incompleteness, scalability, safety, stability, smoothness, accuracy, optimization, and efficiency are examined in this paper.

Index Items: Bio-mechatronics systems, Biomedical Robotics 6-Degree of Freedom (6DoF), Optimal Trajectory Planning

1. INTRODUCTION

For more than three decades, researchers have been studying bio-mechatronics systems and applications to solve practical and theoretical obstacles, particularly those provided by robotic and mechatronic applications in the medical and healthcare sectors. Biomedical Robotics and bio-mechatronic research include a wide range of rapidly developing multidisciplinary fields, such as bio-inspired robots for medical, military, industrial, and rehabilitative applications. In addition, designing and developing bio-inspired systems and devices, including innovative and high-performance, have been investigated in various applications. This special issue highlights recent advances and the most exciting mechatronic and robotic applications in many fields. It combines various topics that cover several definitions, developments, controls, and bio-mechatronic/robot deployment practices. Including wearable robot systems like exoskeletons, social robots, rehabilitation robots, and telerobots. This special issue aims to examine bio-mechatronic systems that are biological from a perspective, as well as the engineering and scientific concepts that underpin their exceptional performance. This work covers high-quality original papers with novel concepts and ideas that use bio-mechatronic and biomedical robotic systems. The recent growth of multidisciplinary research will advance bio-mechatronic and biomedical robotic systems and their applications in rehabilitation, personal assistance, prosthetics, assistive technology, surgery, diagnosis, transportation, and health care in hospitals and laboratories [1].

Many fast-growing multidisciplinary disciplines are applied in biomedical robotics and biomechanics, including biology-inspired robots for medical, military, industrial, and rehabilitative systems. The most recent and exciting discoveries in robotics and mechatronics in many sectors encompass various issues such as definitions, control, development, and deploying mechatronic/bio-robotic systems involving social. Exoskeletons, rehabilitative robots, remote robots, and various systems engineering techniques like control, optimization, and modeling are examples of robotics and wearable robot systems [2]. Like other mechatronic devices, Robotic arms are widely produced and sold worldwide. Hundreds of different types of arms are available from various firms, most of which are designed to aid in industrial manufacturing and assembly lines. These arms are utilized in various applications requiring high accuracy and repeatability by mounting various tools on their standard wrist connections (e.g., welding, manipulation, spraying). Robotic businesses and their cooperating laboratories do most investigations on commercially available arms to enhance the design/performance of mechanical/control systems [3]. Due to their dependability and resilience, these arms are an essential product and a low-cost testbed for research models researching intelligent control systems and innovative applications. As a result, selecting an appropriate robot arm with its controller becomes a critical study choice from a researcher's standpoint. Aside from the arm standard parameters, such as workspace requirements, weight, repeatability, payload, and degrees of freedom (DOF), the after-sales service, availability of the software simulator, ability to combine existing and new software, and adding new sensing devices flexibility, and controller, among others, must be considered. Evaluating these characteristics early on is critical before making a final purchasing choice to ensure future scalability.

The need for 6 degrees of freedom stages in precision engineering is significant. These stages must be exact, with a nanometer-level precision (Willson and Roman, 2008). Precision-compliant stages with 3, 5, and 6 degrees of freedom have been discovered in the literature. In each of the six DoF phases, there were three degrees of freedom for both translational(x, y, z) and rotational(x, y, z). Every stage in the 3 DoF has one rotational (z) and two translational (x, y) degrees of freedom, while in considering the rotation around the z-axis, there is no degree of freedom for one 5 DoF stage (Wang et al., 2005). In the literature, all of the designs were entirely compatible. In another way, no traditional joints were employed to transmit motion. Aside from that, all of the designs were quite symmetrical.

Trajectory planning is moving a robot between two alternative configurations over time to complete a job while staying within the robot's restrictions. For the robot manipulator, a specific configuration involves a set of joint angles, while the collection of every available joint angle represents the space configuration. The physical restrictions of robots are included in the constraints. Geometric restrictions, which may be described in robot joint angles, are among them (i.e., avoiding collision with the environment, and bounds on the joint angles) [4]. They also contain higher-order time derivatives of joint angles and kinematics and dynamics restrictions (i.e., motor current inputs, torques, accelerations, or bounds on the joint velocities).

Additionally, the work should be done effectively and precisely across consecutive configurations while achieving a specific goal, such as jerk, or maximizing smoothness, decreasing energy consumption (or effort of the actuator), and path travelling distance or execution time. Accurate dynamic models of parallel robots are required for dynamics-based control techniques [5]. In actuality, an incompatibility between the developed dynamic model and the actual system is unavoidable. The control's performance will suffer due to this mistake [6, 7]. Furthermore, The parameter values of dynamic parallel robot models are not fixed but continually revised by the robot motions, resulting in expensive computations and increased demands on the controllers' processing capacity. The parallel-robot actuators are assumed to be self-governing of each other in kinematic-based control techniques [8]. The intended path and location of the robot are accomplished by manipulating each target tracing trigger [9]. The design of the kinematic-based control technique is nearly identical to that of the uniaxial motion controller. Therefore, no need to think about a parallel robot-dynamic model. It is simple to put into practice.

On the other hand, the motors are not independent in practice. They are connected via the CPU, and the moveable manipulator applies constantly changing load pressures to each operator. As a result, the anti-disturbance ability of the uniaxial motion controller has an essential effect on the performance of the kinematic-based control technique.

Much of the current research has been on building more sophisticated robotic systems and degrees of freedom to aid locomotor training for more complex motions, including gait [10] and multi-joint arm and hand movements. They also make gadgets portable in everyday activities [10]. Additionally, progress has been made in establishing control techniques influencing how these gadgets interact with individuals. Control approaches for parallel robots are critical, particularly for direct drive actuators, because uncertain system disturbances directly affect them [9, 10]. Based on kinematics and dynamics, control techniques for parallel robots may be categorized [11, 12].

2. BACKGROUND

As mentioned in the linked works below, multiple distinct optimum trajectories dominate the 6-DOF-based intelligent controller in biomedical applications and other studies:

Two alternative path optimizers are derived using the free-time optimization control method. Using a seven-speed DOF robot, the two resultant algorithms that encode two different game styles are the defensive player and the central player [13]. In dynamic systems, the ANFIS technique is an application of Artificial Neural Networks (ANN) that is prepared for any frame of care by adding a training signal system to the diverse learning base, using a mixture of slope ratios and small square error (LSE) technology. They also employed a 7-DOF controller. [14] The neural network approach was used to navigate a humanoid robot. The parameters are left and right obstacle distances, the front obstacle distance, and the heading angle. The neural network's output is the steering angle [15]. Based on the 6-DOF parallel robot kinematics, a control approach regulated six linear motors to trace their respective intended trajectories underneath a developed Fractional Order Active Disturbance Rejection Controller (FOADRC)[16]. The forward kinematic model uses the Denavit Hartenberg (DH) 6 DOF robot arm placement parametric scheme. The applied inverse kinematic model on an actual robot arm, while the forward kinematic model is examined in MATLAB using Robotics Toolbox [17]. The quintic polynomial function is used for each robot joint to fulfil multiple specific points, where mathematical expressions are generated for each common robot variable over time. The optimized adaptive genetic algorithm improves each joint's interval of interval points to achieve the optimum path planning time. [18]. The robot trajectory is updated, and its real-time target with no need for extra offline iterations. Based on control optimization and gradient ratios, it extends to controllers of linear quadratic regulators to circulate locally or globally controllers modulate the robot's core path [19]. An improved cetacean improvement algorithm (IWOA) based on the algorithm of differential evolution (DE) and the algorithm of whale improvement (WOA) has been proposed [20]. Artificial neural networks and evolutionary algorithms are used to control the lower limb rehabilitation system built on a parallel Stewart 6 DOF robot (smart strategy). [21]. Kinematics creates a six-degree-of-freedom (DOF) arm of the robot and evaluates its working space. The suggested approach allows manipulator control to accomplish any position and direction [22]. Finding an optimum trajectory from a beginning and end location in the operating space for a particular route. Using a genetic algorithm to calculate the optimum trajectory for a 3DoF spherical wrist with a concurrent axis [23]. A novel grid-based technique for robot path rolling planning in an unknown and static environment Ant colony optimization (ACO) coupled with the ecological knowledge of the local view of the robot and target information is used to intelligently design the robot's local navigation optimization path [24]. A novel framework for generating a trajectory for robotic table tennis that does not use a fixed striking plane. Two alternative trajectory optimizers are created using a free-time optimum control approach[25]. A metaheuristic optimization technique for the planning mobile robot path problem. A comparison of ACO and particle swarm algorithms is carried out. [26]. A novel method and test its efficacy in optimal grinding robot trajectory planning. An objective function was created based on jerk and time. The differential evolution algorithm (DE) and the whale optimization algorithm (WOA) are used to develop an improved whale optimization algorithm (IWOA) [29]. Analyzing Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Ant Colony Optimization (ACO) metaheuristic techniques to select the multicriteria ad hoc network path pairs. As multicriteria, the power consumption, load variation, and signal-to-noise ratio (SNR) are utilized [30]. Adapt a global attraction phrase that directs ants to the goal site. The Ant colony Algorithm is used in several approaches for robot path planning. [31]. A humanoid navigational controller has been created utilizing an intelligent algorithm of fuzzy logic to avoid obstacles in the surroundings and securely reach the intended destination point [32]. A neural network approach guided humanoid robots through a crowded environment filled with numerous hazards. In a crowded environment, the "ROBONOVA" humanoid can travel to the target position in a safe mode by employing the neural network approach, as has been discovered [33]. Making the case that physical human interaction is instructive: it provides information about how the robot should perform its function. Formalize learning from these interactions as a dynamical system in which the task objective has hidden state parameters, and physical human interactions are observations about these parameters [34]. Explores applications to energy and power systems, intending to offer a complete overview of the key challenges surrounding utilizing global optimization metaheuristics problems [35]. The algorithm's performance is evaluated and compared to one another to solve the selection of features in biomedicine successfully. In order to evaluate the searchability of the algorithm fairly and precisely, a set of benchmark problems' feature selection is constructed and provided for performance testing [36]. An improved ant colony algorithm was used for developing the capabilities of effective path planning for mobile robots' difficult maps. In the improved ant colony algorithm, the features of the Ant system's MAX-MIN and the A algorithm are combined [37]. The MVA method is a novel meta-heuristic algorithm based on Multiverse Theory that can tackle NP-hard optimization problems such as nonlinear and multi-level programming issues and practical optimization challenges for CPS systems. [38] In path planning, the suggested method is compared to other algorithms. It was subjected to a performance review to achieve the best learning algorithm [39]. Path optimization addresses the route planning challenge for autonomous mobile robots [40]. The cuckoo optimization algorithm in a dynamic environment is suggested as a novel technique for tackling the mobile robot route planning issue [41]. An adaptive robot gripper and a six-degree-of-freedom robot arm are created by the IRAGS [42]. It is a four-link biped system with an actuated hip joint and passive knee joints that walk independently [43]. Developing optimum trajectory planning algorithms for autonomous robots is critical for completing robot jobs efficiently. Polynomial trajectory planning for a 6-DOF robotic arm is described in [44]. Trajectory planning and joint space simulation were carried out based on kinematics analysis. Kinematics of a three-degree-of-freedom robotic manipulator [45]. The Artificial Neural Networks (ANNs), Genetic Algorithms (GAs), and algebraic method function are the methodologies [46]. Besides the basic optimization algorithms such as GA, PSO, and tabu search [47], there is a continual development of novel optimization algorithms (between 2000 and 2020). To investigate the kinematics and dynamics of degree-of-freedom multiple situations of surgical micro-robot, as well as the best trajectory planning [48]. The challenge of planning path algorithms and optimize dynamic and static situations is the focus of mobile robotics research [49]. The method of smooth trajectory planning and the time-optimal is developed and subsequently applied to robot manipulators [50]. Introduces collaborative robots, improving work performance, dependability, and cost [51]. Based on the Online Trajectory Generation (OTG) concept, the suggested trajectory controller provides real-time trajectory computation and easy communication with various components, such as the path planner, trajectory generator, collision checker, and controller [52]. suggest smooth pathways from the robot trajectories observation by utilizing Differential Evolution with different initialization forms, selection pressure, exploration, and exploitation to optimize fitting and smoothness requirements [53]. Potential functions are used in an improved technique to plan a wheeled mobile manipulator's collision-free trajectory in blocked settings [54]. A flexible snake-like robot design is given, as well as the constraints model provided for kinematics and obstacle avoidance while moving, motion trajectory planning, and dynamics control [55]. A novel potential function-based technique for wheeled mobile manipulators' collision-free trajectory planning in blocked settings [56]. The development of a novel "whip-lashing" approach on the base of the robot arm's motion is comparable to the action of the whip [57]. It proposes and shows a technique for designing smooth path-constrained time optimum manipulator trajectories. These trajectories are produced by minimizing the jerks needed by the intended motion [58]. an effective approach for determining the best robot base location for performing necessary trajectories of end-effector while redundant copy the wrist partitioned 6 R articulated robots [59]. To increase the speed of industrial robots by generating the shortest-time trajectories that fulfil different restrictions typically used in industrial robot applications [60]. A new planning technique of robot route-rolling in unknown and static environments is suggested on the base of grid modelling. For the robot in an unfamiliar scenario, generating an intelligent local navigation optimization path is accomplished by combining (ACO) with the ecological knowledge of the local view of the robot and the information target [62]. A new planning technique of robot route-rolling in an unknown and static environment is suggested on the base of grid modelling.

3. TRAJECTORY PLANNING ANALYSIS

Robot trajectory planning generally refers to tracking points based on assumptions and a goal posture. Later, and then adjusting the rotational angle of each robot connection to the effector's end along a trajectory led by each point to reach its targeted point ultimately. Cartesian space trajectory planning is more suitable and time-consuming than joint space trajectory planning. As a result, numerous fixed locations at the ends of several robotic arms are often provided [64]. The robot's track points are then calculated using inverse kinematics to transform them from Cartesian space to joint-coordinate space. Following that, the track points are utilized to do interpolation using various polynomial, spline functions, or other formulas and curves for each joint variable time of the generated robot. Furthermore, because of the mechanical properties of the robot, each arm's joint acceleration and speed should be confined to the permissible range. As a result, optimizing the arm's joint acceleration and speed is required to ensure the smooth functioning of the joint arm and decrease wear and influence on extending the robot's working life[64].

The robotic control system should provide reference inputs to ensure the planned motion is carried out to solve the trajectory planning problem. Typically, the path created by the path planner, the robot's kinematic and dynamic restrictions, are sent into the trajectory planning algorithm. The trajectory of the end effector or the joints, in the series of location form, data acceleration, and velocity, is the trajectory planning module output [65].

Because the task to be completed and the difficulties to overcome are stated more naturally in the operating over the joint space, the geometric path is generally established in the robot's operational space. As a result, creating a set of values specifying the position and orientation that the robot's end-effector should acquire at each interval is required while planning the route in the operative environment. When motion follows a path with specific geometric properties established in the functional space, the trajectory is usually scheduled in space operation. In this situation, paths can be characterized precisely (by following the original path) or roughly (by assigning specific route points and utilizing polynomial sequences to link them). In most cases, the path is planned in the joint space of the robot because the manipulator's control action is performed on that joint. Planning in the operational space necessitates a kinematic inversion to convert the end-effector's orientation and position information into a common value. Trajectory planning procedures can be pretty different [66]:

- with a pre-defined path
- point to point

Or:

- in the joint space;
- in the workspace,
- Specifying certain places of interest (starting and ending locations, through a set of points) or the entire geometric path x = x (t). Two factors must be specified to plan the desired trajectory:
- A geometric route
- A low of motion

with constraints on the path's smoothness (continuity) and time derivatives up to a recognized degree.

The geometric path's definition is either the joint space or the workspace. It is usually stated in a parameter form like

$$p = p(s) \qquad workspace$$

$$p = p(\sigma) \qquad workspace$$

The law of motion is s = s(t) (σ = (t)) which is derived once the parameter s (σ) is specified as a time function. As can be seen in Figure 1, the geometric paths are shown.

.

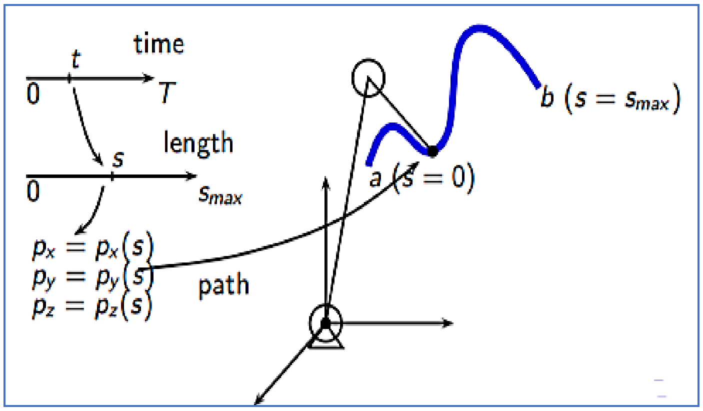


Figure (1) Geometric paths [55]

Parabolic segments, circular, Linear or, more broadly, analysis function tracts are examples of geometric pathways (in the workspace). Geometric pathways are created in the joint space by giving the joint variables initial/final (also, in some circumstances, intermediate) values and the desired law motion. In the motion law, continuous functions up to a specific order of derivations are required (often at least first and second order, i.e., acceleration and velocity). Correct degree n polynomial functions are commonly used:

$$s(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_n t^n$$

These results are in a "smooth" points' interpolation that defines the route geometry.

In most cases, the user has to provide basic information regarding the trajectory, such as the start and stop positions, motion length, maximum velocity, etc.

1. **Workspace trajectories** allow for direct consideration of path constraints (route geometry, obstacles, etc.) in the same space that are more challenging to address (because of the nonlinear kinematics)
2. **Joint space trajectories** are more computationally straightforward and enable concerns like singular configurations, actuation redundancy, and velocity/acceleration limitations.

Specific characteristic points define joint space trajectories, directly allocated by some requirements, defined by describing desired configurations x in the workspace, and then translated into joint space using the inverse kinematic model. The algorithm that evaluates a function q(t) interposes the given points must have important attributes. The paths should be computationally efficient, while the velocity and position patterns should be at least continuous-time functions, and undesirable effects (like non-regular bends) should be reduced or avoided entirely. A single joint is considered in the following discussion. If there are multiple joints, a coordinated motion must be planned, such as considering the same initial and final time instant for each of them or assessing the joint that is most stressed (with the greatest displacement) and then appropriately scaling the motion of the remaining joints

In most cases, the joint space method is used, and the paths are formed by various interpolation functions, like polynomials.

The polynomial trajectories: In the most specific circumstances, a segment of a path is defined as (period) specifying initial conditions and the final ones for a time, acceleration, velocity, position, and so. The difficulty then becomes determining a function.

$$q = q(t) \quad or \quad q = q(\sigma) \qquad \sigma = \sigma(t)$$

as a result of which those prerequisites are met. It is a problem of boundary conditions that can be handled quickly by using polynomial functions like:

$$q(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_n t^n$$

The polynomial's degree n (3, 5...) determines the boundary conditions' number that needs to be confirmed and the required trajectory smoothness.

In addition to the starting and end values, other restrictions might be put on various time derivatives' values (jerk, acceleration, velocity, etc.) in generic instants tj. In other words, creating a polynomial function q(t) whose kth derivative has a defined value q k (tj) at a specific time, tj might be of relevance. These requirements can be represented mathematically as:

$$k!\,a_k + (k+1)!\,a_{k+1} t_j + \cdots + \frac{n!}{(n-k)!} a_n t_j^{n-k} = q^k(t_j)$$

or, in matrix form: $M_{a=b}$

Where:

- M is matrix (n + 1) × (n + 1),
- b is the vector with constraints of n + 1 on the path (the data known),
- a = [a0, a1, . . . , an] T contains the parameters that are unknown in order to be computed

$$a = M^{-1} b$$

- **Third-order polynomial paths:** Given an initial instant ti and a final one tf, a (segment of a) path can be determined by allocating initial conditions and final ones:

- Initial position q_i , velocity $˙q_i$
- Final position q_f and velocity $˙q_f$

Because there are four boundaries, a polynomial of degree 3 (or higher) should be recognized.

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

The four parameters a0, a1, a2, and a3 must be defined to satisfy the boundary conditions. It follows from the boundary conditions that

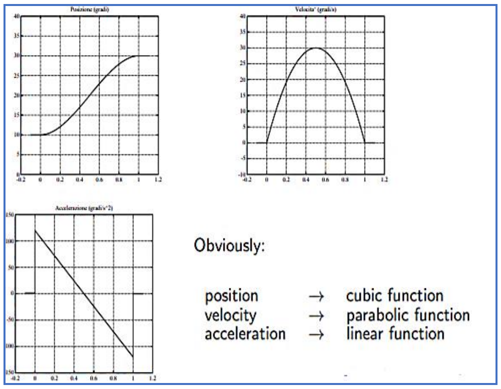
$$q(t_i) = a_0 + a_1 t_i + a_2 t^2{}_i + a_3 t^3{}_i \qquad = q_i$$

$$q(t_i) = \quad a_1 \ + a_2 t^2{}_i + a_3 t^3{}_i \qquad = q_i$$

$$q(t_i) = a_0 + a_1 t_f + a_2 t^2{}_f + a_3 t^3{}_f \qquad = q_f$$

$$q(t_i) = \qquad 2a_2 t_f + 3a_3 t^2{}_f \qquad = q_f$$

Let us assume that ti = 0 for the time being in order to solve these equations. Therefore:
Profiles of acceleration, velocity, and position obtained to use a cubic polynomial and boundary conditions: qi = 10o , qf = 30o , ˙qi = ˙qf= 0 o /s, t_i = 0,t_f= 1s: The polynomial functions are shown in Figure 2.



Obviously:

position → cubic function
velocity → parabolic function
acceleration → linear function

Figure (2) polynomial [56]

The obtained results with both polynomial (1) and the coefficients (3)-(6) are generalizable to the case where ti 6= 0. One receives:

$$q(t) = a_0 + a_1(t - t_i) + a_2(t - t_i)^2 + a_3(t - t_i)^3 \quad t_i \le t \le \ t_f$$

With coefficients

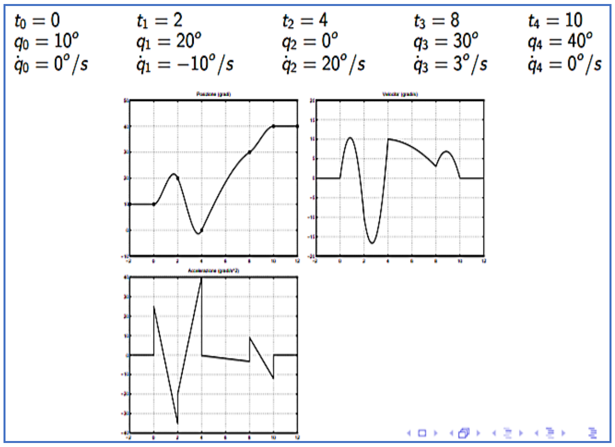
$$a_0 = q_i$$

$$a_1 = q_i$$

$$a_2 = \frac{-3(q_i - q_f) - (2q_i - q_f)(t_f - t_i)}{(t_f - t_i)^2}$$

$$a_3 = \frac{-2(q_i - q_f) - (2q_i - q_f)(t_f - t_i)}{(t_f - t_i)^3}$$

It is simple to plan a trajectory that passes through a series of intermediate points. Profiles of position, velocity, and acceleration. The polynomial of six boundaries is depicted in Figure 3.



$t_0 = 0$	$t_1 = 2$	$t_2 = 4$	$t_3 = 8$	$t_4 = 10$
$q_0 = 10^o$	$q_1 = 20^o$	$q_2 = 0^o$	$q_3 = 30^o$	$q_4 = 40^o$
$\dot{q}_0 = 0^o/s$	$\dot{q}_1 = -10^o/s$	$\dot{q}_2 = 20^o/s$	$\dot{q}_3 = 3^o/s$	$\dot{q}_4 = 0^o/s$

Figure (3) polynomial of six boundaries [57]

- **Fifth-order polynomial trajectories**: profiles of location and velocity are continuous-time functions, as seen. It is not the case with acceleration, which results in discontinuities between segments. Furthermore, giving appropriate initial/final values for this signal in each segment is impossible. These features are not a concern in many applications because the trajectories are "smooth" enough. If, on the other hand, initial and final acceleration values are required (for example, to obtain acceleration profiles), then fifth-order polynomial functions must be considered (at least).

$$q(t) = a_0 + a_1 t + a_2 t^2{}_i + a_3 t^3 + a_4 t^4 + a_5 t^5$$

With conditions of six boundaries:

$$q(t_i) = q_i \quad q(t_f) = q_f$$

$$\dot{q}(t_i) = \dot{q}_i \quad \dot{q}(t_f) = \dot{q}_f$$

$$\ddot{q}(t_i) = \ddot{q}_i \quad \ddot{q}(t_f) = \ddot{q}_f$$

In this situation (if T = tf- ti), the polynomial coefficients are

$$a_0 = q_i$$

$$a_1 = \dot{q}_i$$

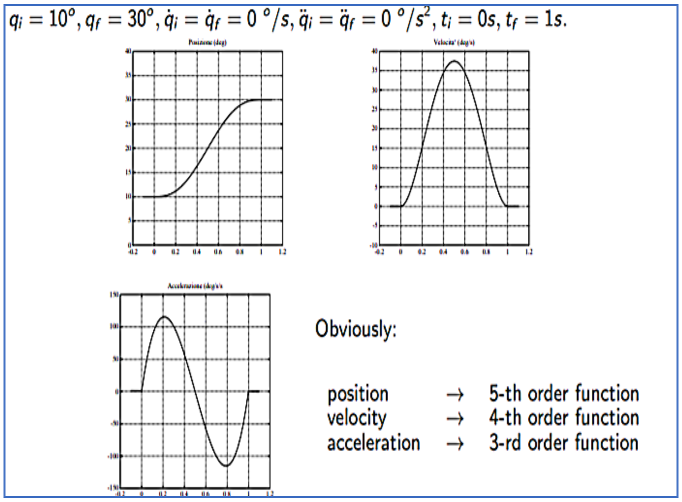
$$a_2 = \frac{1}{2}\ddot{q}_i$$

$$a_3 = \frac{1}{2T^3}\left[20(q_f - q_i) - (8\dot{q}_f - 1\dot{2}q_i)T - (3\ddot{q}_f - \ddot{q}_i)T^2\right]$$

$$a_4 = \frac{1}{2T^4}\left[30(q_f - q_i) - (14\dot{q}_f - 1\dot{6}q_i)T - (3\ddot{q}_f - 2\ddot{q}_i)T^2\right]$$

$$a_5 = \frac{1}{2T^5}\left[12(q_f - q_i) - 6(\dot{q}_f - \dot{q}_i)T - (\ddot{q}_f - \ddot{q}_i)T^2\right]$$

If a set of points is provided, the intermediate velocity values can be computed using the same considerations as for third-order polynomials trajectories. Polynomial trajectories are a type of polynomial trajectory. The following is a fifth-order trajectory with boundary conditions. The polynomial of four boundaries is depicted in Figure 4.


$q_i = 10^o, q_f = 30^o, \dot{q}_i = \dot{q}_f = 0\ ^o/s, \ddot{q}_i = \ddot{q}_f = 0\ ^o/s^2, t_i = 0s, t_f = 1s.$

Obviously:

position → 5-th order function
velocity → 4-th order function
acceleration → 3-rd order function

Figure (4) polynomial of four boundaries [57]

3.1. Classic Approaches

Creating a geometric path without a temporal law is what trajectory planning entails, whereas trajectory planning involves giving the geometric path a time law. Off-line and online are two algorithms of trajectory planning classified depending on the information provided. Offline robots could calculate the target's path before motion occurs when the environment is completely known (i.e., information about obstacles is known from the beginning), resulting in a globally optimum solution. This category covers a wide range of topics. The topics include optimality (locally and globally) and completeness. The solution will be discovered if one exists. Moreover, cost and computational efficiency (permit changings without replanning or recomputing everything). Online robots progressively develop the route to the goal during mobility, resulting in local optimum solutions at their finest. In this case, the mobile robot gathers data from sensors across the environment. This category includes issues such as completeness (is the robot guaranteed to reach the aim if a solution is possible), computational efficiency and costs at each level, and optimality (how far is a solution from the optimal, and is it limited by an upper bound). Trajectory planning can be done in a variety of ways. The most important classical methodologies are sampling-based, potential field methods, combinatorial approaches, and bug-like algorithms. The earliest and most fundamental sensor-based algorithms are bug-like algorithms [66]. The robot is believed to be a perfectly positioned point in space with a restricted workspace. They are equipped with a contact sensor that detects the obstacle boundary when it comes into contact with it. They are simple to execute because they must move towards the objective unless an obstacle is met. In that situation, they navigate around the obstruction until a movement toward the goal is once again permitted. It can be done by calculating the distance between two places. Most offline robots employ combinatorial approaches; space configuration as the basic notion is used by geometric representation planners. Geometric representations of the environment might take the form of graphs or roadmaps that encapsulates the open space topology and are constructed using many well-known approaches like a visibility graph ([69], [70], [71], [72]), the Voronoi diagram ([67], [68]); a tangent graph [73]; the Subgoal Network [23], Silhouette [82], and cell decomposition and grid method ([74], [75], [76], [77], [78], [79], [80], [81]). They portray free spaces (non collisions space) differently. However, arranged on the basis of an interlinked network of trajectory segments, which might be tracked from the beginning to the end. Including the obstacle maps, the fundamental complexity in terms of the computational cost relating to these techniques is the free space representation. Dijkstra's search [83] or A* [84] are examples of traditional methods that track the optimal path after creating the road map. The advantage of these strategies is that they make the broad motion planning problem NP-complete. However, they have the drawbacks of being too sluggish for practical usage, particularly in high-dimensional issues and requiring an explicit description of obstacles, which is difficult to get in most actual problems. Another technique is to characterize the search space as an undirected graph[2] using a uniform grid. These approaches impose high costs on edges that cross barriers, allowing nodes in the free space to be effectively separated from inaccessible nodes. According to the search space discrete representation, the resolution of this method is full, as it is for all systems that explain the mechanism of how the pathways are traversed at low grid resolution via narrow space in between obstacles that might be neglected.

An increase in graph resolution, on the other hand, would result in a significant increase in computational effort. The uniform grid representation has a substantially higher number of nodes than the roadmap-based techniques, which is a drawback. However, this method can be used when impediments are not defined well, like with a mobile robot. The method of potential field creates a high-potential field near the barriers and a low-potential field near the configurable goal ([1], [26]). By allowing the robot configuration to evolve in that potential field, the robot is steered to the aim configuration while averting impediments. In other words, although the obstacles repel the robot, it is drawn to the appropriate configuration. A vector that is pointing to the current position is named the gradient vector. Contrary to the aforementioned techniques, probabilistic completeness might be accepted using the sampling-oriented planner (it is not a condition that the target might be approached within a specific time chunk). Any solution might be accepted not only the optimal one, and at the same time, the explicitly represented geometric solutions might be neglected in terms of the graph or roadmaps free configurations

A roadmap is a graph connecting vertices representing free-space configurations with a path running through the free space. The roadmap can be created in two ways: deterministic or probabilistic. In the Probabilistic Roadmap Planner, samples are chosen randomly in free space rather than following a regular grid (PRM). There are numerous approaches for selecting the pairs of vertices to connect due to the lack of a prior grid structure. This method is effective for a wide range of issues ([27], [28]). It is based on the notion that it is less computationally expensive to see if there is a single robot arrangement in the free space. PRM generates a roadmap in free space by sampling the nodes and edges of the roadmaps with coarse and fine sampling, respectively (i.e., the free trajectory between the configuration of the node). Then, by connecting the target configurations and the initial roadmap, queries about planning could be answered. A uniform random distribution ensures the planner's probabilistic completeness [29]. Other planners that are sample-based and use the nodes sampling method include the planner Rapidly Exploring Random Tree (RRT) [24] and the planner Expansive Spaces Tree (EST) [30], which may be more successful for single-query planning.

3.2 Heuristic Approaches

Metaheuristic and heuristic techniques have been developed to address the shortcomings of traditional approaches. Probabilistic Roadmaps (PR) methods are included. Rapid exploration of Random Trees (RRT) contains the following:
Ant Colony Optimization (ACO) uses ants' foraging behaviour to find the shortest food source path ([32], [33], [34]). Genetic Algorithms (GA) that are based on selection and natural genetics mechanisms ([38], [39], [40]), Simulated Annealing (SA), which is the approach of heuristic random search that mimics the annealing processes for cooling molten metal ([35], [36]) and Neural Network [37]. More techniques include Particle Swarm Optimization (PSO) is inspired by the social behaviour of bird flocking or fish schooling for more straightforward implementation using fewer adjusted parameters, as compared to the GA technique ([41], [42], [43], [44], [45], [46]). Also, Wavelet is based on the theory of wave-like oscillations [48]. Fuzzy logic is many-valued logic wherein the variables' truth values might range from 0 to 1 [49]. Finally, Tabu Search is a mathematical optimization method based on local search [36]. Heuristic algorithms do not promise that a solution can be found, but if they want to, they will do it far more quickly than deterministic methods.

3.3 **Decision-making process technique for path planning**

There are numerous ways to estimate the path of a moving item when accurate knowledge about the robot's environment and geometric description is unavailable. The environment's knowledge is derived from the provided measurements by a set of imprecise sensors, which are noisy in such instances. As a result, trajectory planning is done in the face of uncertainty, which must be predicted. This uncertainty impacts the robot's and its environment's predictability of present states and the states in the future (in continuous or discrete state spaces and continuous time). These states are determined by the memory of activities that were previously performed, sensors, and initial conditions. As a result, under uncertainty, trajectory planning approaches include problems like manipulation, map creation, pursuit-evasion, and localization [2]. To a greater or lesser extent, some strategies can account for the decision-making process and uncertainty. For example, more advanced methods involve sequential decision-making (a sequence of fundamental problems of decision-making), artificial intelligence, and control theory. Probabilistic models, expected-case, or Worst-case, analyses of game theory (with players of competitive objectives. Instead of using a deterministic location, probabilistic estimate methods use the robot's location probability density function (PDF) over space,

which allows for coping with uncertainty. The goal is to maintain a PDF position for every robot's potential stance.
,mx xcmcm
As a result, the path was not smooth. Although, Fuzzy Markov decision processes (FDMPs) can yield smooth trajectories [51]. The Kalman filter employs the same iterative prediction-update procedure as Bayesian approaches, but it does not depend on the restrictive assumptions of Kalman filter [52]. The advantages include using nonlinear models for both sensing and arbitrary distribution and path planning rather than a Gaussian. However, this could result in a more computing cost than Kalman filters.

3.4 Mathematical programming

Mathematical programming-based obstacle avoidance systems apply a set of inequalities to the configuration parameters. Motion planning is then introduced as a mathematic problem optimization in which the goal is to identify the curve between the beginning and the goal configurations that minimize or maximize a particular goal function. For instance, energy consumption (or actuator effort), execution time, path travelling distance, and jerk, or maximizing smoothness (e.g., [53], [54], [55]). It creates a nonlinear problem optimization with multiple inequalities and differential constraints that must be addressed numerically. Furthermore, Pareto optimum solutions have been developed in the literature for multi-objective optimization issues ([56], [57]).

3.5 Approaches Comparison

The advantages and disadvantages of the various techniques are shown in Table 1. Completeness (whereas if the path is available, the path and trajectory are found). Optimality (the acquired plan is optimal in terms of specific parameters, not stuck in a local minimum). Efficiency (the computational cost of the algorithm, i.e., if it can change the queries and the world without repeating computing everything or re - planning from the start) are among the issues that have been addressed. Additional challenges include execution time (lower times are desirable), accuracy (high precision path tracking and controversies) (for the robot, its surroundings, and humans), and scalability (the problem grows effectively as the configuration space dimensions significantly rise). On the other hand, robot trajectory planning optimization approaches are continually evolving (e.g., [59], [60], [61]).

Table 1. Summary

Approach	**Pros**	Cons
Potential fields	Real-time, good scalability. It is simple, highly safe and computationally fast.	Not **complete, not efficient** world and queries update>, path not optimal (local minimum), potential field forces must be set
Cell decomposition	Complete, elegant, all cells are either full, which signifies that it is occupied completely by an obstacle, or it is empty, which signifies it is free. The union of the entirety of cells precisely represents the configuration space.	High computational cost, and **high execution time.** The availability of local minima.
Visibility graph	Complete and yields **minimal** length paths,optimal. The vertices retain the precise geographic characteristics of their mapping points. It is likely to computer the common measurements of space syntax after the creation of the graph. Among the measures, depth and	High computational cost, high execution time, bad scalability, not efficient world. It solves a subset of the motion planning problems. The polygonal assumption of obstacles is very restricted.

	connectivity are the most fundamental ones.	
Voronoi diagram	Complete and generates roadmap **with maximum distance, efficient worldandaue.ries updates. Allows quick recovery neighbourhood of information**	Possibly inefficient paths, time, bad scalability, path not **optimal.** The stored data suffers from redundancy, and it utilizes more space than needed. The data structure which is mostly used for storing Voronoi diagrams is the DCEL (double connect edges list)
Heuristic approaches	Low **execution** time, parallel **The heuristics approaches offer a quick solution that is straightforward to implement. Heuristic algorithms are practically used and serve as fast and feasiblely based short-term solutions to plan**	Not **complete, the** possibility of **providing smooth oaths.** Errors are bounded to occur in the final results, given that a large number of heuristic operations are subject to prejudice and biases. Shortages in data might make quick decisions via heuristic techniques. Inaccurate decisions and imprecise solutions might result.
Exact cell **decomposition**	Complete. Exact cell decomposition is elegant because of its precision and completeness.	High computational cost, high **execution time. Existence of local minima.**
Approximate cell **decomposition**	Robust and useful when only a **coarse representation of the workspace is available**	Not complete
Bug (bug I and bug 2).	Complete, easy **implementation, parameters eas:v to adiu$f: perfection in positioning and sensing.**	Long paths, high **execution** Time. Algorithms are time-consuming. Difficult to show Branching and Looping in Algorithms. Big tasks are difficult to put in Algorithms
A*	Complete, optimal grid.	Not **efficient,** bad scalability, not efficient world **andaue.ries updates**
Rapidly exploring random tree (RRT)	Complete, **semi-efficient world** and **queries** updates, good scalability	The path is not optimal. The whole random tree requires to be passed for searching the parent points in the processes of new point generations, which might be consuming a lot of computational time.
Probabilistic roadmap (PRLvls)	**Complete, semi-efficie.nt queries** updates, optimal graphs, and good scalability. Path planning is the most	**Not efficient world updates.**

	significant matter in robot path finding and trajectory finding. It is referred to as finding a geometrically based path from the current position of the arm to a goal position in a way that enables the algorithm to be avoiding hazards.	

4. SYSTEM MODELING of 6-DOF ROBOT

The robotic arm with 6 degrees of freedom robotic platform employed in this study is the ED7220C, which ED Korean Corporation built. The robot arm has been used widely in research, education, and development. It is essentially a series that manipulate all joints arranged in a revolution shape. As illustrated in Figure (4), the arm geometry consists of the elbow, shoulder, waist, and wrist by the human arm joints.

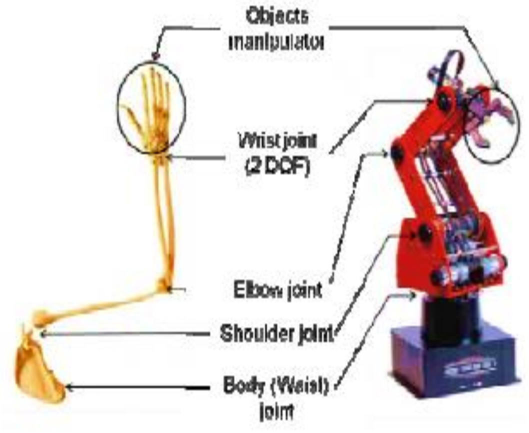


Figure (4) 6 Degree of Free[50]

Because the wrist may move the two planes (tone and roll), the end effector is more versatile in object handling.

The robot, which is built with vertical articulation, allows for the quick visual study of each joint of mechanical behaviour. A micro-servo motor drives the entire arm, with each DOF achieved via an optical encoder (DME 38B50G). A dual-state clutch with rubber pads serves as the last transponder.

In a control algorithm malfunction, built-in mechanical safety restrictions prevent joint movement. The key features of the ED7220C robotic arm are listed in Table 2.

Table 2 ED7220C robotic arm

Feature	Description
Position precision	**±0.5mm (approx.) Movement speed IOOmmfs (max.) Load <:-apacity I Kg**
\Veight	**33 Kg**
Wrist .Pitch:	**260°. Ro11:360°**
Range Of Motion	
(ROM)	
Elbow:	**172°**
Shoulder:	**90° Waist: 3J0°**

The general setup of the robot is depicted in the diagram below, with its control unit connected to a regular computer and an educational pendant. The controller function (ED) provides ports for the encoded motors, allowing the robot to be driven visually. The controller has over 100 cores with higher-level commands, allowing the platform to be used in various ways. The teaching pendant allows the controller to control any of the two. The necklace serves as a driving practice pad. A robotic arm is controlled manually. It is used to allow the robot to recognize any accessible coordinates. The control unit can save the points taught to perform a restart later. On the other hand, the robot must be educated each time the thing's location changes. In order to achieve this, the robotic arm is self-contained. [9] built and presented a robotic image-guided system. The general robot architecture is shown in Figure 5.

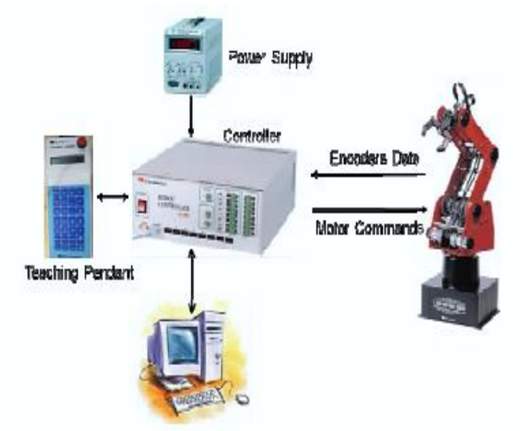


Figure (5) General System of The Robot[60]

- **A. Forward Kinematic Model**

Different methods can be used to investigate a robot's kinematic problem. Denavit-Hartenberg (DH) pascrew displacements are employed in two regularly used ways. Both methods are better suitable for simulating serial manipulators since they are methodical. Some researchers also regularly use geometric approaches for serial manipulators with elementary geometry. Because of its adaptability and suitability for simulating any number of serial manipulator joints and linkages, regardless of complication, the DH technique was utilized to construct the robot's kinematic model in this study. Figure (6) depicts a simplified kinematic model of the robotic arm in an inverted 'L' position. The first three joints move the tooltip to the desired position, while the last two joints change the orientation of the end lengths in Table (3).

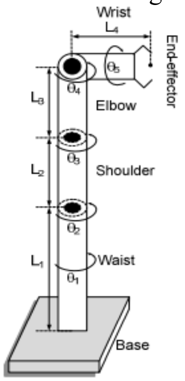


Figure (6) simplified kinematic model of the robotic arm[66]

Table (3) Two joints change the orientation of the end lengths

Joint	**Waist**	**Shoulder**	**Elbow**	**Wrist**
Symbol	L,			
Link Length [mm]	385	220	220	155

Link offset, link length, twist angle, and joint angle are all represented by quadruple d, ai-1, and i-1 in DH. As illustrated in Figure (7), the system of an orthonormal coordinate coupled with each manipulator link by DH standard. Table (4) lists the DH parameters for the robotic arm ED7220C.

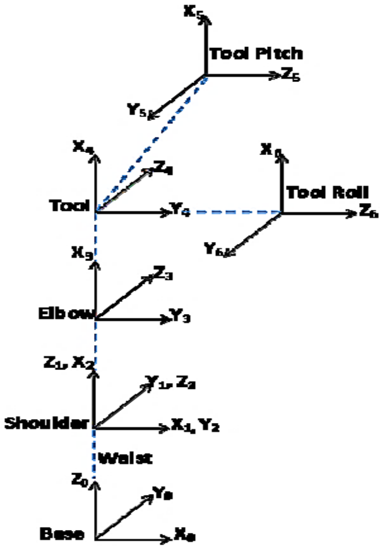


Figure (6) DH parameters for the robotic arm ED7220C [67]

Table (4) lists the DH parameters for the robotic arm ED7220C.

SYMBLE	**1**	**2**	**3**	**4**	**5**	**6**
$\boldsymbol{a_i - 1}$	0	-90	0	0	-90	0
$\boldsymbol{a_i - 1}$	0	0	L_2	L3	0	0
$\boldsymbol{d_i}$	L_1	0	0	0	0	L4
$\boldsymbol{\theta_i}$	θ_i	$\theta_2 - 90$	$\theta 3$			0

The corresponding transformation matrices for each link of the robotic arm have been generated using the general form of the transformation matrix for each link (expressing joint I in its preceding nearby joint i-1) derived in [11]. These different transformation matrices, when multiplied, provide the overall matrix reflecting the base's end terms, thanks to the compound transformation property (1).

$$ {}_6^0T = \begin{bmatrix} C_1C_5S_{234} + S_1S_5 & -C_1S_{234}S_5 + S_1C_5 & C_1C_{234} & C_1A \\ -S_1C_5C_{234} - C_1S_5 & S_1C_{234}C_5 + C_1C_5 & S_1C_{234} & S_1A \\ C_{234}C_5 & -C_{234}S_5 & -S_{234} & B \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

Where

$$ \mathrm{A} = L_2S_2 + L_3S_{23} + L_4C_{234} $$

$$ \mathrm{B} = L_1 + L_2C_2 + L_3C_{23} - L_4S_{234} $$

The rotation is represented by the 3X3 matrix, which consists of the first three rows and the first three columns, while the location (x, y, z) of the base's end-effector is represented in the last column.

- **Inverse Kinematic Model**

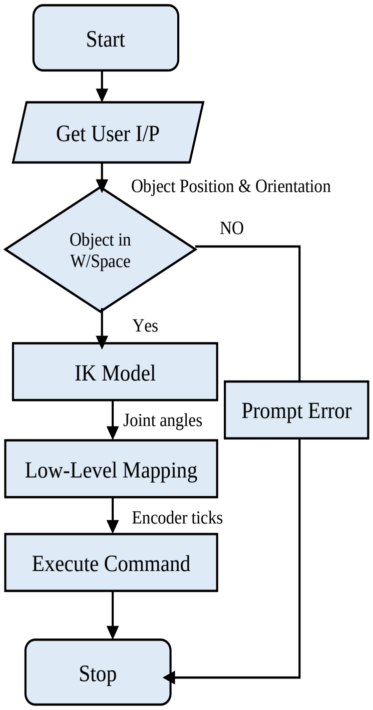
The inverse kinematic (IK) model has greater applicability in real-world robotic systems. IK is important not only in robotics but also in other industries, such as 3D games. IK, unlike forwarding kinematics, there is not a single solution to it. The systems that guarantee free-of-collision operation and joint with minimal motion are the best. The IK model of ED7220C was developed using an analytical technique. This method ensures that the model determines joint angle correctness for any object within the workspace of the robotic arm. The joint angles required to accomplish the specified position and orientation are computed using the IK model. The mechanical arm movement control and planning are based on inverse kinematic robotics analysis. The known end-effector, related to the reference coordinate system's position and linkage parameters, determines the value of each joint angle i. This method is used to calculate the first four joint angles: waist (_1), shoulder (_2), elbow (_3), and toll pitch (_4), whereas tool roll (_5) is directly determined by the desired orientation for object manipulation.

Because rotation and translation are involved in transformation, the usual form of the transformation matrix from tool to base is ().

$$ {}^{Base}_{Tool}T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

In an IK problem, the first 3x3 matrix and (px, py, and pz) indicate the translation and rotation of the end-effector w.r.t the base of the robot. After extensive mathematical computations, the established analytical IK model offers equations (), (), (), and () for the joint angles _1, _2, _3, and _4, respectively. The requisite joint angles are expressed as coefficients in these equations ().

$$T1-1*T = T2*T3*T4*T5*T6$$
$$\theta 1 = \mathrm{atan}\,2d\,(py, px)$$
$$d = squr(px^2 + py^2)$$



$$r_4 = d - a_4 * \mathrm{Cosd}\,(\theta 234)$$
$$z_4 = p_z - a_4 * \mathrm{Sind}\,(\theta 234)$$
$$\theta_3 = acosd(\frac{s^2 - a^{2^2} - a3^2}{2*a2*a3})$$
$$beta = atan2d(a3 * sind(\theta_3), a2 + a3 * cosd(\theta_3)$$
$$alpha = atan2d(z4 - a1, r4)$$
$$\theta 2 = alpha + beta$$

Figure **7.** The block diagram of time-optimal trajectory planning.

$$\theta 4 = \theta 234 - \theta 3 - \theta 3$$
$$\theta 5 = acosd(\frac{(sind(\theta 1)*px) - (cosd(\theta 1)*py)}{sind(\theta 4)*a5})$$
$$\theta 6 = atand\left(cosd(\theta 5) * cosd(\theta 4) * sind(\theta 2 + \theta 3) + sind(\theta 5) * cosd(\theta 2 + \theta 3) - sind(\theta 4) * \frac{cosd(\theta 5)}{cosd(\theta 5) + sind(\theta 4) + sind(\theta 2 + \theta 3)} \right)$$

• IK Model Implementation

The IK model has been implemented on the arm manipulator of the genuine robot. A known position and orientation have been assigned to an object. The created method initially evaluates if the object is inside the robot workspace using this known information from the user, as shown in IK. The algorithm finishes after prompting the user if the object is not inside the scope of the work. Otherwise, the IK model calculates the necessary joint angles for the end-effector to point in the correct position and orientation. The low-level encoder ticks are then transferred to these joint angles. Finally, the program's Kernel-based instructions carry out the commands for moving the robotic motors according to the mapped encoder ticks. The figure illustrates the flow chart implementation of the model. As shown in Figure (8), PH implementation of the model.

To execute the generated IK model on the robotic arm, the object (for example, a vehicle key with a key) must be within the robotic arm workspace. For this purpose, two blocks are used to raise the platform on which the object is put (in height z). The task at hand is shifting an object from one area to another. The position and orientation of both the source and destination have been provided as inputs. The robot moves according to computed joint angles based on the user's object coordinates (IK model). The gripper shuts when the robot reaches the target spot, and the object is grasped. Picking up the thing in order, The robot moves using the coordinates the user has also taught it. The point of the target must be inside the operational workspace of the robot. After arriving at that location, the robot drops it and returns to its original station.

5. Time-Optimal Trajectory Planning Algorithm

The running time of the trajectory, denoted by t, is used in this analysis as the control variable. As the values of the state variable, the joint angular velocity and the manipulator joint angular acceleration are used. In this paper, the min–max rule is utilized for the purpose of solving the issue of finding a local optimal solution when using time-search algorithms of discovering the trajectory minimal time throughout the trajectory planning procedure. If only the constraint conditions of the angular velocity of joint I am needed, then a time t could be found such that when the manipulator is running with the trajectory, the maximal angular velocity of joint I reaches the angular velocity constraint conditions of j. This is accomplished by finding Figure 9 presents a TOTP block diagram for your viewing pleasure. A temporal search was carried out, and the joint parameters that were determined following each iteration of trajectory planning and inverse kinematics were compared with the joint constraints.

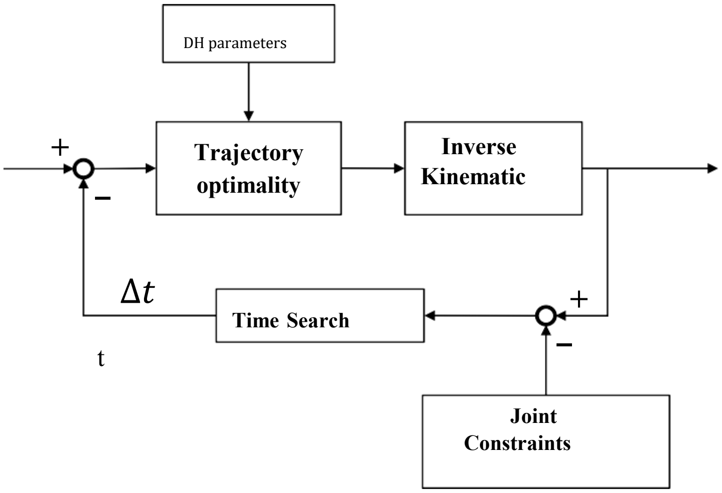


Figure (8) DH implementation of the model[68]
.

There were 12 constraints corresponding to the 12 shortest times. Using the minimum–maximum rule, the maximum value of the 12 shortest times is the shortest time of the trajectory.

6. Challenges, Recommendations, and Future Directions

The field of medical robotics objectives to produce applications and scientific self-recovery agents by proposing the ideas of the mechanism of control and implementation of medical robots. Moreover, the usability of medical robots is. Damaged agents have the ability to be self-reconfiguring and also have the ability to grasp, relying on their modular architecture and functional redundancy for adapting their behaviors to their tasks and environment. These fundamental attitudes let robot-based applications to be working in dangerous or anonymous environments.

The repair or replacement of damaged robots is cost-prohibited or impossible. The robot might be a great tool for doctors to perform operations in medicine, where it might be changed into several configurations if some parts were dysfunctional. How can one make resilient robots be delivering on their best potential? rapid design instruments are required besides designing recipes for less-cost but robust robots, novel approaches used to control behavior change, and self-reconfigurations or state change that account for fast damage recovery prerequisite. The availability of such requirements ought to be existing in critical situations. How can true potential be achieved? This depends on the failures, tasks and environment, as with different aspects of robotics. For domains that require strong resilience (for example, rescue tasks) or dealing with a large quantity of uncertainty, resilient robots can bring the capabilities of robots to new levels. where robot replacement or maintenance is impossible or prohibitively expensive. For instance, the robot could save lives during an earthquake by assuming various configurations if some of its components were compromised. But how do we reach the point where resilient robots perform to their utmost capacity? Self-reconfiguration and/or state changes that account for rapid damage recovery requirements, which are necessary in an emergency, are necessary. But how resilient must an automaton be in order to realize its full potential? As with numerous other aspects of robotics, this is dependent on the failures, the task, and the environment. For domains that require a high level of resilience (such as rescue tasks) or deal with a great deal of ambiguity (such as space), resilient robots can expand the capabilities of robots. For a given task, a robot has relatively greater resilience if it employs all three recovery strategies and recovers the original function in the shortest time possible. In addition, learning to predict what behaviour should be avoided [68], diagnosing failures, and tracing the failures' origins is a good way to increase a robot's resilience so that it can be applied in a variety of situations where a robot must autonomously adjust its behaviour. Enhanced hardware, including sensors, actuators, and communication, will also contribute to increased resilience. We anticipate a comprehensive measurement of robots' resilience [69] as well as the relationship between resilience and other system properties, such as cost and reliability, resulting in robust, resilient robots that can surmount more failures. In conclusion, the main future directions for this expanding field are as follows. Classifying robot failures and devising methods to predict and prevent them. Investigating the relationships between recovery strategies and failures, as there may be multiple strategies available when a failure occurs, and the optimal strategy must be determined.

7. CONCLUSION

This study gives an overview of the best autonomous robot trajectory planning methods. They address a wide variety of topics, including robot kinematics, achieving collision-free trajectories, and taking the robots' physical constraints into account. The various motion planning approaches are reviewed, and their benefits and drawbacks. As a result of these advantages and disadvantages, various methods may be employed for a wide range of robot applications.

References

[1] Banerjee, Hritwick, et al. "Highly Stretchable Flame-Retardant Skin for Soft Robotics with Hydrogel–Montmorillonite-Based Translucent Matrix." Soft Robotics 9.1 (2022): 98-118.

[2] C. Zhou and K. H. Low, “Design and locomotion control of a biomimetic underwater vehicle with fin propulsion,” IEEE/ASME Trans. Mechatron., vol. 17, no. 1, pp. 25–35, Feb. 2012

[3] Vidyarthi, Prabhat Kumar, Koena Mukherjee, and Binoy Krishna Roy. "Fish-Like Robots And Applications Of Sensor-A Review." 2022 4th International Conference on Energy, Power and Environment (ICEPE). IEEE, 2022.

[4] Zhao, Xiaojie, et al. "Trajectory Planning for 6-DOF Robotic Arm Based on Quintic Polynomial." *2017 2nd International Conference on Control, Automation and Artificial Intelligence (CAAI 2017)*. Atlantis Press, 2017.

[5] YuYang Fu,6R.Research on Structural Design and Motion Trajectory Planning of Industrial Robot [D].Master Thesis of Wuyi University.2013.

[6] Saeed B.Niku. Introduction to Robotics [M]. BEIJING: Publishing House of Electronics Industry, 2013.

[7] Wu, Lei, et al. "Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot." Expert Systems with Applications 215 (2023): 119410.

[8] Hou, Wenbin, et al. "Enhanced ant colony algorithm with communication mechanism for mobile robot path planning." Robotics and Autonomous Systems 148 (2022): 103949.

[9] Rafai, Anis Naema Atiyah, Noraziah Adzhar, and Nor Izzati Jaini. "A Review on Path Planning and Obstacle Avoidance Algorithms for Autonomous Mobile Robots." Journal of Robotics 2022 (2022).

[10] Ali, Zain Alabedeen, and Konstantin Yakovlev. "Safe Interval Path Planning With Kinodynamic Constraints." arXiv preprint arXiv:2302.00776 (2023).

[11] Hossain, Tagor, et al. "Local path planning for autonomous mobile robots by integrating modified dynamic-window approach and improved follow the gap method." Journal of Field Robotics 39.4 (2022): 371-386.

[12] Gugan, Gopi, and Anwar Haque. "Path Planning for Autonomous Drones: Challenges and Future Directions." Drones 7.3 (2023): 169.

[13] Koç, Okan, Guilherme Maeda, and Jan Peters. "Online optimal trajectory generation for robot table tennis." Robotics and Autonomous Systems 105 (2018): 121-137.

[14] Gigras, Yogita, and Kusum Gupta. "Metaheuristic algorithm for robotic path planning." International Journal of Computer Applications 85.3 (2014).

[15] Payedimarri, Anil Babu, et al. "Effectiveness of platform-based robot-assisted rehabilitation for musculoskeletal or neurologic injuries: a systematic review." Bioengineering 9.4 (2022): 129.

[16] Losey, Dylan P., and Marcia K. O'Malley. "Learning the correct robot trajectory in real-time from physical human interactions." ACM Transactions on Human-Robot Interaction (THRI) 9.1 (2019): 1-19.

[17] Wang, Ting, et al. "Optimal Trajectory Planning of Grinding Robot Based on Improved Whale Optimization Algorithm." Mathematical Problems in Engineering 2020 (2020).

[18] Li, Zhijun, Chenguang Yang, and Etinne Burdet. "Guest editorial an overview of biomedical robotics and bio-mechatronics systems and applications." IEEE Transactions on Systems, Man, and Cybernetics: Systems 46.7 (2016): 869-874.

[19] Gunantara, Nyoman, and I. Nurweda Putra. "The characteristics of metaheuristic method in selection of path pairs on multicriteria ad hoc networks." Journal of Computer Networks and Communications 2019 (2019).

[20] Reshamwala, Alpa, and Deepika P. Vinchurkar. "Robot path planning using an ant colony optimization approach: a survey." International Journal of Advanced Research in Artificial Intelligence 2.3 (2013): 65-71.

[21] Rath, Asita Kumar, et al. "Analysis and use of fuzzy intelligent technique for navigation of humanoid robot in obstacle prone zone." Defence technology 14.6 (2018): 677-682.

[22] Sahoo, Biswajit, Dayal R. Parhi, and B. K. Priyadarshi. "Analysis of path planning of humanoid robots using neural network methods and study of possible use of other AI techniques." Emerging trends in engineering, science and manufacturing,(ETESM-2018), IGIT, Sarang, India (2018).

[23] Bajcsy, Andrea, et al. "Learning robot objectives from physical human interaction." Conference on Robot Learning. PMLR, 2017.

[24] Chicco, Gianfranco, and Andrea Mazza. "Metaheuristic Optimization of Power and Energy Systems: Underlying Principles and Main Issues of the ‘Rush to Heuristics’." Energies 13.19 (2020): 5097.

[25] Wang, Ling, et al. "Feature selection based on meta-heuristics for biomedicine." Optimization Methods and Software 29.4 (2014): 703-719.

[26] Dai, Xiaolin, et al. "Mobile robot path planning based on ant colony algorithm with A* heuristic method." Frontiers in neurorobotics 13 (2019): 15.

[27] Yu, Jinglun, Yuancheng Su, and Yifan Liao. "The Path Planning of Mobile Robot by Neural Networks and Hierarchical Reinforcement Learning." Frontiers

in Neurorobotics 14 (2020).

[28] Santa, Fernando Martinez, Fredy H. Martinez Sarmiento, and Holman Montiel Ariza. "Path Optimization for Mobile Robots using Genetic Algorithms." International Journal of Advanced Computer Science and Applications 13.2 (2022).

[29] Ji, Wei, and Lihui Wang. "Industrial robotic machining: a review." *The International Journal of Advanced Manufacturing Technology* 103.1 (2019): 1239-1255.

[30] Wichapong, Kritsada, et al. "Trajectory planning of a 6D robot based on Meta Heuristic algorithms." *MATEC Web of Conferences*. Vol. 220. EDP Sciences, 2018.

[31] Parhi, D. R. "Advancement in navigational path planning of robots using various artificial and computing techniques." *Int Rob Auto J* 4.2 (2018): 133-136.

[32] Funes Lora, Miguel Angel, et al. "Metaheuristic techniques comparison to optimize robotic end-effector behaviour and its workspace." *International Journal of Advanced Robotic Systems* 15.5 (2018): 1729881418801132.

[33] Hosseininejad, Seyedhadi, and Chitra Dadkhah. "Mobile robot path planning in a dynamic environment based on the cuckoo optimization algorithm." *International Journal of Advanced Robotic Systems* 16.2 (2019): 1729881419839575.

[34] Kim, Yoon Jae, et al. "A study on a robot arm driven by three-dimensional trajectories predicted from non-invasive neural signals." *Biomedical engineering online* 14.1 (2015): 1-19.

[35] Suchi, MB Markus, and M. Vincze. "Meta-Heuristic search strategies for Local Path-Planning to find collision-free trajectories." *Proceedings of the Austrian Robotics Workshop (ARW-14)*. 2014.

[36] El Hayek, Dikra, et al. "Optimal trajectory planning for a spherical robot using evolutionary algorithms." *Procedia Manufacturing* 32 (2019): 960-968.

[37] Mohamed, Khaled, Hassan Elgamal, and Amr Elsharkawy. "Dynamic analysis with optimum trajectory planning of multiple degree-of-freedom surgical micro-robot." *Alexandria engineering journal* 57.4 (2018): 4103-4112.

[38] Zafar, Mohd Nayab, and J. C. Mohanta. "Methodology for path planning and optimization of mobile robots: A review." *Procedia computer science* 133 (2018): 141-152.

[39] Tiginyanu, Ion, Victor Sontea, and Serghei Railean, eds. 4th International Conference on Nanotechnologies and Biomedical Engineering: Proceedings of ICNBME-2019, September 18-21, 2019, Chisinau, Moldova. Vol. 77. Springer Nature, 2019.

[40] Zhang, Tie, Meihui Zhang, and Yanbiao Zou. "Time-optimal and smooth trajectory planning for robot manipulators." *International Journal of Control, Automation and Systems* 19.1 (2021): 521-531.

[41] El-Dosuky, M. A., et al. "Collaborating Robotics Using Nature-Inspired Meta-Heuristics." *arXiv preprint arXiv:1212.5777* (2012).

[42] Ajeil, Fatin H., et al. "Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm." *Applied Soft Computing* 89 (2020): 106076.

[43] Zhao, Ran. *Trajectory planning and control for robot manipulations*. Diss. Université Paul Sabatier-Toulouse III, 2015.

[44] Parque, Victor, and Tomoyuki Miyashita. "Smooth curve fitting of mobile robot trajectories using differential evolution." *IEEE Access* 8 (2020): 82855-82866.

[45] Sandakalum, Thushara, and Marcelo H. Ang Jr. "Motion planning for mobile manipulators—a systematic review." Machines 10.2 (2022): 97.

[46] Llopis-Albert, Carlos, Francisco Rubio, and Francisco Valero. "Optimization approaches for robot trajectory planning." *Multidisciplinary Journal for Education, Social and Technological Sciences* 5.1 (2018): 1-16.

[47] Benotsmane, Rabab, László Dudás, and György Kovács. "Trajectory Optimization of Industrial Robot Arms Using a Newly Elaborated “Whip-Lashing” Method." *Applied Sciences* 10.23 (2020): 8666.

[48] Doan, Ngoc Chi Nam, and Wei Lin. "Optimal robot placement with consideration of redundancy problem for wrist-partitioned 6R articulated robots." *Robotics and Computer-Integrated Manufacturing* 48 (2017): 233-242.

[49] Chicco, Gianfranco, and Andrea Mazza. "Metaheuristic Optimization of Power and Energy Systems: Underlying Principles and Main Issues of the ‘Rush to Heuristics’." *Energies* 13.19 (2020): 5097.

[50] Liang, Yanyang, et al. "Design and implementation of multi-axis real-time synchronous look-ahead trajectory planning algorithm." The International Journal of Advanced Manufacturing Technology (2022): 1-19.

[51] Ratiu, Mariana, and Mariana Adriana Prichici. "Industrial robot trajectory optimization-a review." *MATEC Web of Conferences*. Vol. 126. EDP Sciences, 2017.

[52] Zhao, Su-na, et al. "Teleoperation control of a wheeled mobile robot based on Brain-machine Interface." Mathematical Biosciences and Engineering 20.2 (2023): 3638-3660.

[53] Kelly, Matthew. "An introduction to trajectory optimization: How to do your own direct collocation." *SIAM Review* 59.4 (2017): 849-904.

[54] V. Lumelsky and A. Stepanov, Path planning

strategies for point mobile automaton moving amidst unknown obstacles of arbitrary shape. Algorithmica 2 (1987) 403–430.

[55] Omisore, Olatunji Mumini, et al. "Motion and Trajectory Constraints Control Modeling for Flexible Surgical Robotic Systems." Micromachines 11.4 (2020): 386.

[56] Xu, Shoujiang, Edmond SL Ho, and Hubert PH Shum. "A hybrid metaheuristic navigation algorithm for robot path rolling planning in an unknown environment." *Mechatronic Systems and Control* 47.4 (2019): 216-224.

[57] Liu, Xinmin, and Rodney D. Wiersma. "Optimization-based trajectory planning for real-time 6DoF robotic patient motion compensation systems." *PloS one* 14.1 (2019): e0210385.

[58] Zheng, Shengjie, et al. "The Brain-Inspired Cooperative Shared Control for Brain-Machine Interface." arXiv preprint arXiv:2210.09531 (2022).

[59] Duraklı, Zafer, and Vasif Nabiyev. "A new approach based on Bezier curves to solve path planning problems for mobile robots." Journal of Computational Science 58 (2022): 101540.

[60] Daniel, Bonilla Licea, Ghogho Mounir, and Saska Martin. "When Robotics Meets Wireless Communications: An Introductory Tutorial." arXiv preprint arXiv:2209.02021 (2022).

[61] Shi, Xinxin, Jiacai Huang, and Fangzheng Gao. "Fractional-Order Active Disturbance Rejection Controller for Motion Control of a Novel 6-DOF Parallel Robot." *Mathematical Problems in Engineering* 2020 (2020).

[62] Amini Azar, Wahab, Adel Akbarimajd, and Esmaeil Parvari. "Intelligent Control Method of a 6-DOF parallel robot Used for Rehabilitation Treatment in lower limbs." *Automatika: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije* 57.2 (2016): 466-476.

[63] Al Mashhadany, Yousif Ismail. "Design and Analysis of 7-DOF Human-Link Manipulator Based on Hybrid Intelligent Controller." *Информатика и автоматизация* 19.4 (2020): 774-802.

[64] Hao, Bing, et al. "A Path-Planning Approach Based on Potential and Dynamic Q-Learning for Mobile Robots in Unknown Environments." Computational Intelligence and Neuroscience 2022 (2022).

[65] Banyassady, Bahareh, et al. "Unlabeled Multi-Robot Motion Planning with Tighter Separation Bounds." arXiv preprint arXiv:2205.07777 (2022).

[66] Zinage, Vrushabh, et al. "3D-Online Generalized Sensed Shape Expansion: A Probabilistically Complete Motion Planner in Obstacle-Cluttered Unknown Environments." IEEE Robotics and Automation Letters (2023).

[67] Huo, Da, et al. "Collision-free Model Predictive Trajectory Tracking Control for UAVs in Obstacle Environment." IEEE Transactions on Aerospace and Electronic Systems (2022).

[68] Reza Zarrabi, Mohammad, and Nasrollah Moghaddam Charkari. "Single-Point And Triple-Point Queries Visibility Constrained Minimum Link Paths In Simple Polygons." The Computer Journal 66.2 (2023): 496-507.

[69] Moniruzzaman, M. D., et al. "Teleoperation methods and enhancement techniques for mobile robots: A comprehensive survey." Robotics and Autonomous Systems 150 (2022): 103973.

[70] Sampson III, Myles B. Discrete-to-Complete: The Fundamentals of Design Directed Robotics. Diss. Massachusetts Institute of Technology, 2022.

[72] Li, Fei-Fei, Yun Du, and Ke-Jin Jia. "Path planning and smoothing of mobile robot based on improved artificial fish swarm algorithm." Scientific Reports 12.1 (2022): 659.

[73] Dong, Jianquan, et al. "Sustainable landscape pattern: a landscape approach to serving spatial planning." Landscape Ecology (2022): 1-12.

[74] Atlantis, Panagiotis, Charalampos P. Bechlioulis, and Kostas J. Kyriakopoulos. "Mutli-Robot Cooperative Object Transportation with Guaranteed Safety and Convergence in Planar Obstacle Cluttered Workspaces via Configuration Space Decomposition." Robotics 11.6 (2022): 148.

[75] Gonzalez-Perez, Ali, et al. "Deep and Machine Learning Image Classification of Coastal Wetlands Using Unpiloted Aircraft System Multispectral Images and Lidar Datasets." Remote Sensing 14.16 (2022): 3937.

[78] Nantabut, Chinnawut, and Dirk Abel. "Prioritized Anytime Dynamic A* Path Planning for Multi-Agent Vehicles in Heterogeneous Traffic." Advances in Intelligent Traffic and Transportation Systems. IOS Press, 2023. 34-41.

[79] Lafmejani, Amir Salimi. Decentralized Motion Planning for Autonomous Multi-Agent Systems: Multi-Segment Manipulators and Mobile Robot Collectives. Diss. Arizona State University, 2022.

[80] Yue, Junchen. "Kinematic modelling and simulation of LeArm robot." 2022 6th International Conference on Robotics, Control and Automation (ICRCA). IEEE, 2022.

[81] De Paolis, Lucio Tommaso, et al. "An augmented reality platform with hand gestures-based navigation for applications in image-guided surgery: prospective concept evaluation by surgeons." International Journal of Human–Computer Interaction 38.2 (2022): 131-143.

[82] Mamidi, Teja Krishna, and Sandipan Bandyopadhyay. "A computational framework for the dynamic analyses of cable-driven parallel robots with feed and

retrieval of cables." Mechanism and Machine Theory 186 (2023): 105338.

[83] Singh, Gurjeet, and V. K. Banga. "Robots and their types for industrial applications." Materials Today: Proceedings 60 (2022): 1779-1786.