Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal 2024, 18(3), 314–332
https://doi.org/10.12913/22998624/186361
ISSN 2299-8624, License CC-BY 4.0

Received: 2024.02.26
Accepted: 2024.03.21
Published: 2024.04.20

New Approach to Planning the Complex Movements of 6-DOF Industrial Robot Subjected to Acceleration Constraints

Grzegorz Pająk[1]

[1] Institute of Mechanical Engineering, University of Zielona Góra, ul. Licealna 9, 65-417 Zielona Góra, Poland
E-mail: g.pajak@iim.uz.zgora.pl

ABSTRACT

A method of trajectory planning with regards to joint velocity and acceleration constraints for industrial 6DOF manipulator is presented. The task of the robot is to move to specified location in the workspace passing through intermediate waypoints. The proposed algorithm can be used to plan the task of the robot by autonomous systems in smart factories eliminating human participation in the robot programing process. Opposite to similar approaches it does not assume the type of function describing the motion of the robot. The trajectories generated using the proposed approach are smooth and provide smooth velocities and continuous joint accelerations. The motion is planned in such a way to fulfill joint velocity and acceleration constraints. Fulfillment of velocity limitations is accomplished by perturbing the manipulator motion close to velocity limits. To satisfy acceleration constraints a trajectory scaling approach carried out in limited periods of time is used. The results of the research are illustrated by simulations and experiments, in which an analysis of the method of performing robot tasks carried out using built-in algorithms and presented methods are performed.

Keywords: trajectory planning, waypoints, velocity and acceleration constraints, industrial robot.

INTRODUCTION

The dynamic changes taking place in the organization of production process and the challenges posed by the fourth industrial revolution, known as Industry 4.0, force a change in the approach to planning the tasks of industrial robots [1, 2]. There are many works presenting concepts for implementing robotic systems, such as the use of lean robotics [3], which allows for the elimination of inefficient activities, or the implementation of modern robotic systems (called cobots) to carry out selected activities in human-robot teams [4]. The introduction of the idea of Industry 4.0 also requires the development of algorithms enabling the autonomous operation of robots performing certain technological operations. Many robotics problems related to the need for precise positioning of details can be solved with the support of vision systems. An example of such an application was discussed in the work [5] presenting the possibilities of cooperation between a human and a robot at an assembly station. Equally important is ensuring the possibility of automatic planning of robot motions. The idea of smart factories and autonomous systems that quickly adapt to new processes excludes human participation in creating programs for robots and makes it necessary to develop new methods that will allow planning complex tasks of industrial robots in a fully automatic manner. In recent years there has been intensive progress of hardware, but there is a stagnation in the development of software for industrial robots. Despite the fact, that there are known many theoretical studies of advance motion planning methods in the literature, they are not used in practice and typical industrial robots sill use classical algorithms for planning movements.

Contemporary works on robotics often concern redundant manipulators and advanced mechanisms such as humanoid robots, mobile or free-flying space robots. The designed motion planning algorithms take into account a number of conditions that are usually not considered during programing a classical industrial robot. One of the common problems are collision detection and collision-free motion. The proposition of solution of such problem for redundant manipulator using artificial potential fields was presented by Palmieri and Scoccia in [6]. An example of heuristic approach utilizing A* algorithm was presented by Gurui et al. in the work [7]. Another frequently considered issue is multi-robot collaboration. The problem of cooperative transportation task by nonholonomic mobile redundant manipulators was considered in the work [8], while algorithm for dual stationary robot cooperative system realizing assembly task was discussed in [9]. Solutions taking into account manipulability of the robot and avoiding singular configurations were considered by Pardi et al. in [10]. Problems resulting from the uncertainty of the model and the effects of external disturbances are also currently the subject of many studies. The proposed solutions use, for example, sliding control [11] or neural controllers [12]. Wide overview of recent trend in robot motion planning was prepared by Tamizi et al. in [13]. Most of the presented solutions is still the subject of scientific research and due to the limited capabilities of modern robots usually cannot be directly used in industrial applications.

In the case of practical motion planning algorithms for industrial robots researchers often use analytical methods that do not require large computational effort. The algorithm of the trajectory planning in the task with multiple waypoints using combination of 4^{th} and 5^{th} order polynomial functions taking into account velocity, acceleration and jerk limits was proposed by Boscariol et al. in the work [14]. Similar solution using 5^{th} order B-spline interpolation technique was presented in [15] by Huang et al. Jerk limited trajectory algorithm based on trapezoidal acceleration model with a 7^{th} degree polynomial was introduced by Lin et al. in [16]. Another approach to description of the trajectory was proposed by Scheiderer et al. in the work [17] in which Bezier curves were used to plan the smooth trajectory C^1 class. To solve practical problems of trajectory planning of industrial robots there are also attempts to use artificial intelligence methods. In the work [18] authors utilized convolutional neural network in order to learn continuous motion behavior. An example of usage of fuzzy control to solve trajectory planning problem was presented by He and Huang in [19]. The overview of path and trajectory planning algorithms was prepared by Gasparetto et al. in [20].

In this paper the motion planning method for 6-DOF industrial manipulator is presented. The proposed algorithm is based on previous author works developed for both stationary and mobile redundant manipulators [21, 22]. The proposed method allows for automatic determination of the robot trajectory for a task with intermediate waypoints and can be used both when the robot should stop at intermediate points and when it should move smoothly between them. Opposite to similar approaches it does not assume the type of function describing the motion of the robot. The trajectory of the manipulator is planned at the acceleration level with regards to joint velocity and acceleration constraints. Fulfillment of velocity limitations is accomplished by perturbing the manipulator motion close to velocity limits. To satisfy acceleration constraints a trajectory scaling approach is used. The scaling procedure is carried out only in limited periods of time, when accelerations are close to constraints. Proposed approach leads to accelerations which satisfy constraints and are continuous function of time. The proposed solution to the problem of motion planning can be used to control a real robot and is an alternative to algorithms commonly used in controllers of modern industrial robots. Unlike classical methods, the proposed approach generates a smooth trajectory, ensures that velocity and acceleration limits are satisfied at every moment of movement and guarantees the continuity of accelerations. In a consequence, application of the proposed algorithm leads to significant reduction of undesirable vibrations of the manipulator arm and smooth robot motion. The effectiveness of the proposed method is confirmed both in the simulations and in the experiments involving Universal Robot UR10e industrial manipulator.

PROBLEM FORMULATION

The typical task of industrial manipulator is to move its Tool Center Point (TCP) from location resulting from the actual configuration of the robot to final location specified by the operator. Such motion can be performed directly to this location or it may lead through a series of waypoints. In the second case the task of the manipulator can be considered as a sequence of movements between waypoints where the final robot configuration achieved in the previous step becomes the initial configuration in the next task. Therefore, the elementary task of the robot is to move from some initial configuration and displacement its TCP to certain specified location in the workspace.

In order to formulate such task it is assumed that the manipulator is described by n-elemental vector of generalized coordinates q and location of its TCP, described by m-elemental vector p, can be determined from kinematic equation of the robot as follows

$$p = k(q), \tag{1}$$

where: k: $\Re^n \rightarrow \Re^m$ denotes m-dimmensional mapping describing the position and orientation of robot TCP, $q \in \Re^n$ is a vector containing joints angles for revolute joints and link offsets for prismatic joints, $p = [x^T, \phi^T]^T \in \Re^m$ is a vector composed of m_x-elemental vector x describing position and m_ϕ-elemental vector ϕ describing orientation of the TCP in the workspace, $m = m_x + m_\phi$.

The elementary task of the manipulator is to find the trajectory $q(t)$ ensuring reaching, at the final time instant $t = T$, configuration in which TCP is placed at the specified final location p_f. Using dependency (1) such task can be formulated as

$$k\big(q(T)\big) - p_f = 0. \tag{2}$$

Additionally, from practical point of view, following boundary conditions imposed on the beginning and the end of the trajectory $q(t)$ should be taken into account:

$$\begin{aligned} \dot{q}(0) &= 0, \\ \dot{q}(T) &= 0. \end{aligned} \tag{3}$$

Moreover, during the execution of the task, the manipulator should not exceed velocity and acceleration limits specified by the operator, i.e. it should maintain constraints expressed in the following form

$$\forall t \in [0, T]\ \dot{q}^i_{min} \leq\ \dot{q}_i \leq \dot{q}^i_{max}, i = 1, \dots, n, \tag{4}$$

$$\forall t \in [0, T]\ \ddot{q}^i_{min} \leq\ \ddot{q}_i \leq \ddot{q}^i_{max}, i = 1, \dots, n, \tag{5}$$

It is worth noting, that introducing constraints (4) and (5) to the robot task requires different approaches. The method presented in this paper provides the solution at the acceleration level, so the first one is a state constraint and it is possible to use one of typical solution based on penalty function approach. The acceleration constraint need to design the new method, using trajectory scaling technique, based on concept developed and presented by the author in [22] and [23].

The solution of the elementary robot task formulated in the above manner, can be used directly to perform typical industrial robot task, when the manipulator moves between several waypoints and it stops at each of them. When the task of the manipulator is to move through the waypoints without stopping it is necessary to introduce certain modification of the solution in the neighborhood of the waypoints. Such case will be detailed discussed in dedicated section below.

TRAJECTORY PLANNING

Elementary robot task

To solve elementary robot task taking into account the boundary conditions (3) and velocity constraints (4) the approach based on the method developed by the author for stationary and mobile redundant manipulator was applied [21, 22]. According to that concept, in order to eliminate known from the literature the drift effect, resulting in the trajectory execution error increasing over time, an algorithm with a feedback loop was proposed. For this purpose TCP tracking error $e(t)$, describing distance between current and final location of TCP, is introduced in the following form

$$e(q(t)) = k(q) - p_f. \tag{6}$$

Using the error (6) the elementary task of the robot can be described in general form as follows

$$\lim_{t\to\infty} e(q(t)) = 0, \tag{7}$$
$$\lim_{t\to\infty} \dot{e}(q(t)) = 0. \tag{8}$$

It is worth noting that the above formulation of the task allows reaching the desired TCP location, in a finite time, with any ε accuracy defined by the needs of the realized process. Applying the approach presented for mobile redundant manipulators in [21], the dependencies specifying the trajectory of the manipulator from the initial configuration $q(0)$ to the final location p_f, satisfying boundary constraints (3) are put forward as the following differential equation

$$\ddot{e}(q,\dot{q},\ddot{q}) + \Lambda_V \dot{e}(q,\dot{q}) + \Lambda_P e(q) = 0, \tag{9}$$

where: $\dot{e}(q,\dot{q}) = \frac{d}{dt}e(q)$, $\ddot{e}(q,\dot{q},\ddot{q}) = \frac{d}{dt}\dot{e}(q,\dot{q})$, $\Lambda_V = diag\{\Lambda_V^1, \dots, \Lambda_V^m\}$ and $\Lambda_P = diag\{\Lambda_P^1, \dots, \Lambda_P^m\}$ are matrices with positive gain coefficients ensuring the stability of differential equation (9) and determining the convergence rate of the solution. Time derivatives of error function $e(q)$ can be determined as follows

$$\dot{e}(q,\dot{q}) = \frac{d}{dt}\big(k(q) - p_f\big) = \frac{d}{dt}k(q) = \frac{\partial k(q)}{\partial q}\dot{q} = J(q)\dot{q},$$
$$\ddot{e}(q,\dot{q},\ddot{q}) = \frac{d}{dt}(J(q)\dot{q}) = \frac{d}{dt}J(q)\dot{q} + J(q)\ddot{q} = \dot{J}(q)\dot{q} + J(q)\ddot{q},$$

where: $J(q)$ – an analytical Jacobian of the manipulator.

The Equation 9 is a second order vector homogeneous differential equation with constant coefficients and to find its solution $2m$ consistent dependencies should be given. In the case under consideration these dependencies may be determined using initial conditions:

$$\begin{aligned} &e\big(q(0)\big) = k\big(q(0)\big) - p_f, \\ &\dot{e}\big(q(0),\dot{q}(0)\big) = J\big(q(0)\big)\dot{q}(0) = 0. \end{aligned} \tag{10}$$

Using Lyapunov stability theory it can be shown that the solution of differential equation (9) is asymptotically stable for positive gain coefficients Λ_V and Λ_P. The proof of stability for nonholonomic redundant mobile manipulator was shown in the work [21]. That case was more general, so its conclusion can be applied to stationary manipulator considered in this paper. The property of asymptotic stability implies fulfillment of the dependencies (7) and (8) i.e. TCP of the manipulator can achieve the final location p_f reducing velocity to zero with precision required by the task conditions. Moreover, as it was shown in [21], selecting gain coefficients in such a way as to satisfy condition

$$\Lambda_V^i > 2\sqrt{\Lambda_P^i}, i = 1, \dots, m \tag{11}$$

Solution of Equation 9 is strictly monotonic function. The fulfillment of inequality (11) guarantees that tracking error (6) will not increase, so TCP will not move away from the final location, what prevents unnecessary increasing of the task execution time. Finally, in order to determine the trajectory of the manipulator $q(t)$ the dependency (9) should be written in the explicit form. After substitution of $\ddot{e}(q,\dot{q},\ddot{q})$ and $\dot{e}(q,\dot{q})$ it can be expressed as follows

$$\dot{J}(q)\dot{q} + J(q)\ddot{q} + \Lambda_V J(q)\dot{q} + \Lambda_p\big(k(q) - p_f\big) = 0. \tag{12}$$

Equation 12 allows to determine vector of generalized accelerations of the manipulator. The transformation of this equation depends on the type of the robot. In the case of redundant manipulator ($n > m$) pseudoinverse or extended Jacobian approach can be used. In the case of most common non-redundant industrial manipulator ($n = m$), considered in this work, the equation can be transformed using inverse of the Jacobian as follows

$$\ddot{q} = -J^{-1}(q)\Big(\dot{J}(q)\dot{q} + \Lambda_V J(q)\dot{q} + \Lambda_p\big(k(q) - p_f\big)\Big). \tag{13}$$

Dependency (13) can be interpreted as a solution of the inverse kinematic problem at the acceleration level. It allows to determine the trajectory for elementary robot task leading TCP from the initial position to the final location p_f. Moreover, the proposed form of equation (9) causes that its solution given by dependency (13) fulfills boundary conditions (3). The reaching zero generalized velocities at the end of the task (second boundary condition (3)) is a consequence of reducing TCP velocity to zero and full rank of Jacobian matrix. Moreover, as it was detailed discussed in [24], for more general solution involving mobile redundant manipulator, the TCP path obtained utilizing equation (13) strictly depends on the choice of gain coefficient Λ_V, Λ_P. If $\Lambda_V^i = \Lambda_V^j$ and $\Lambda_P^i = \Lambda_P^j$ for $i, j = 1, \dots, m_x$ then TCP moves along the line section path, if all coefficients in Λ_V and Λ_P are equal both position and orientation of TCP change linearly. It should be noted that a necessary condition for the existence of a solution is a correct formulation of the task, i.e. in such a way that the motion proceeds away from the singular configurations. In the presented work, the problem of robot Jacobian singularity is not considered in detail, but it is worth noting that the proposed algorithm can be supplemented using an approach similar to the one used in the next section to fulfill velocity limitations. In such case, a disturbance maintaining high manipulability of the arm, should be added to dependency (13), to prevent the loss Jacobian rank. The proposed concept is based on the solution used in the work [25], which considers the planning of movements of a humanoid robot, and will be the subject of further research.

Velocity limitations

In order to fulfill condition (4), i.e. joint velocity limitations, the interior penalty function increasing to infinity when the i-th joint velocity approaching limits $\dot{q}_{min}^i$ and $\dot{q}_{max}^i$ is introduced as follows

$$\kappa(\dot{q}_i) = \begin{cases} \rho\left(\frac{1}{d(\dot{q}_i)} - \frac{1}{\vartheta}\right) & \text{for } d(\dot{q}_i) < \vartheta \\ 0 & \text{otherwise} \end{cases}, \tag{14}$$

where: $d(\dot{q}_i) = \min\left(\dot{q}_i - \dot{q}_{min}^i, \dot{q}_{max}^i - \dot{q}_i\right)/\left(\dot{q}_{max}^i - \dot{q}_{min}^i\right)$ denotes normalized distance of i-th joint velocity from its limits, ϑ is a constant positive coefficient determining the threshold value which activates constraints of i-th joint, ρ is a positive coefficient specifying the strength of penalty.

It should be noted, that above form of penalty affects the trajectory only when velocities are approaching their limits. It is important from practical point of view, because the penalty is activated only when it is needed, in this way, unlike the global penalty, it slightly increases the time of task execution. For taking into account condition (4) motion of the manipulator is disturbed using perturbation $\Delta\ddot{q}$ in the following form

$$\Delta\ddot{q} = -\sum_{i=1}^{n} \frac{\partial\kappa(\dot{q}_i)}{\partial\dot{q}}. \tag{15}$$

Using Lyapunov stability theory it is possible to show that perturbation (15) reduces joint velocities to values determined by threshold ϑ. For this purpose Lyapunov function, active only near the constraints, is taken in the form

$$V(\dot{q}) = \sum_{i=1}^{n} \kappa(\dot{q}_i).$$

It follows from definition of penalty function (14) that $\kappa(\dot{q}_i) > 0$ when $d(\dot{q}_i) < \vartheta$, so it is easy to see that the function $V(\dot{q})$ is positive definite. Differentiating Lyapunov function with respect to time

$$\dot{V}(\dot{q}) = \left\langle \sum_{i=1}^{n} \frac{\partial\kappa(\dot{q}_i)}{\partial\dot{q}}, \ddot{q} \right\rangle$$

and substituting perturbation (15), after simple algebra, the derivative of function $V(\dot{q})$ is obtained as

$$\dot{V}(\dot{q}) = -\left\langle \sum_{i=1}^{n} \frac{\partial\kappa(\dot{q}_i)}{\partial\dot{q}}, \sum_{i=1}^{n} \frac{\partial\kappa(\dot{q}_i)}{\partial\dot{q}} \right\rangle.$$

As it can be seen $\kappa(\dot{q}_i) > 0$ for $d(\dot{q}_i) < \vartheta$, so in a consequence $\dot{V}(\dot{q}) < 0$ when joint velocities exceed the values determined by the threshold ϑ, therefore the perturbation (15) forces velocities reduction when approaching the limits. Finally, the trajectory satisfying condition (4) can be obtained from (13) extended by perturbation (15) as follows

$$\ddot{q} = -J^{-1}(q)\left(\dot{J}(q)\dot{q} + \Lambda_V J(q)\dot{q} + \Lambda_p\big(k(q) - p_f\big)\right) + \Delta\ddot{q}. \tag{16}$$

Acceleration limitations

Taking into account condition (5) and determination the trajectory fulfilling joint acceleration limits requires a different approach than in the previous section. In the presented paper the method of trajectory scaling based on idea used in [23] was proposed. The solution presented in the mentioned work involved trajectory planning with control constraints, however, as it is shown below, such approach is also suitable for fulfilling acceleration limits. In order to consider condition (5) following notations are used

$$a(q,\dot{q}) = -J^{-1}(q)\dot{J}(q)\dot{q},$$
$$b(q,\dot{q}) = -J^{-1}(q)\Lambda_V J(q)\dot{q} - J^{-1}(q)\Lambda_p\big(k(q) - p_f\big) + \Delta\ddot{q}$$

and a scaling coefficient $c(t)$ is introduced into trajectory (16) as follows

$$\ddot{q} = a(q,\dot{q}) + c(t)b(q,\dot{q}). \tag{17}$$

It is worth to note that joint accelerations of the manipulator are linear function of scaling coefficient $c(t)$, which means that changes in this coefficient cause proportional changes in accelerations. The general idea of the proposed approach is assumption that scaling coefficient is equal to 1 if condition (5) is fulfilled and it is reduced if i-th acceleration exceeded its limit, i.e.

$$\ddot{q}_i < \ddot{q}^i_{min} \text{ or } \ddot{q}_i > \ddot{q}^i_{max}.$$

In this case, for each $\ddot{q}_i$ exceeding limit, it is necessary to find such a value of $\hat{c}_i$ that the inequality is satisfied

$$\ddot{q}^i_{min} \le a_i + \hat{c}_i b_i \le \ddot{q}^i_{max}$$

and next out of all $\hat{c}_i$ choose one $\hat{c}$ suitable for all accelerations at a given time instant. For each $\ddot{q}_i$ that violates constrains it is assumed that coefficient b_i is non-zero and four cases should be considered:

(1) $\ddot{q}_i > \ddot{q}^i_{max}$ and $b_i > 0$: $\hat{c}_i \le \frac{\ddot{q}^i_{max} - a_i}{b_i}$

(2) $\ddot{q}_i < \ddot{q}^i_{min}$ and $b_i < 0$: $\hat{c}_i \le \frac{\ddot{q}^i_{min} - a_i}{b_i}$

(3) $\ddot{q}_i > \ddot{q}^i_{max}$ and $b_i < 0$: $\hat{c}_i \ge \frac{\ddot{q}^i_{max} - a_i}{b_i}$

(4) $\ddot{q}_i < \ddot{q}^i_{min}$ and $b_i > 0$: $\hat{c}_i \ge \frac{\ddot{q}^i_{min} - a_i}{b_i}$

If at a certain time instant only one acceleration exceeds its limit then value of $\hat{c}$ takes the value equal to the right side of corresponding inequality (1-4). If more than one acceleration does not meet the constraints three cases need to be taken into account:

(a) if there are only cases 1 or 2 then

$$\hat{c} = \hat{c}_{max} = \min_{i,j}\left\{\frac{\ddot{q}^i_{max} - a_i}{b_i}, \frac{\ddot{q}^j_{min} - a_j}{b_j}\right\},$$

(b) if there are only cases (3) or (4) then

$$\hat{c} = \hat{c}_{min} = \max_{i,j}\left\{\frac{\ddot{q}^i_{max} - a_i}{b_i}, \frac{\ddot{q}^j_{min} - a_j}{b_j}\right\},$$

(c) if cases 1 or 2 and 3 or 4 occur simultaneously $\hat{c}$ should be selected in such a way that the inequality is satisfied

$$\hat{c}_{min} \le \hat{c} \le \hat{c}_{max}.$$

In this case, it seems reasonable to choose $\hat{c}$ as the value closest to the value of this coefficient determined at the previous time instant.

It is worth noting that if there are only cases (1), (2) or (3), (4) trajectory scaling coefficient is always determinable, otherwise fulfillment of inequality formulated in case (c) is a condition for the existence of a solution, i.e. if there is no $\hat{c}$ satisfying this inequality then it is not possible to find trajectory satisfying condition (5).

The method presented above describes the choice of value of scaling coefficient $c(t)$ if at certain time instant the accelerations exceed their limits. In such case $c(t) = \hat{c}$. After the accelerations return to the allowable values, the coefficient $c(t)$ should be increased to 1 in order to prevent the extension of the task execution. In practical point of view, continuity of acceleration is desirable, so the change of coefficient $c(t)$ must be continuous. For this purpose it is assumed that value $c(t)$ asymptotically increases according to the following dependency

$$c(t) = (\hat{c}_0 - 1)e^{-(t-t_0)} + 1, \tag{18}$$

where: t_0 specifies the time instant at which the accelerations were last changed according to the method described above and $\hat{c}_0$ is value of scaling coefficient at time t_0.

Choosing an appropriate value of scaling coefficient at the initial moment of motion and utilization the dependency (18) can additionally ensure a smooth start of the manipulator at the beginning of the task. It seems reasonably to take the initial value $t_0 = 0$ and $\hat{c}_0 = 0$, which consequently gives $c(0) = 0$. It is easy to see, that in such case $\ddot{q}(0) = 0$ (due to the first boundary condition (3)), so the fulfillment of acceleration condition (5) is guaranteed and dependency (18) provides smooth accelerations changes up to the maximum values.

Task with waypoints

The method presented in previous sections allows to solve the elementary robot task, i.e. determine the trajectory of the manipulator from given initial configuration $q(0)$ to specified final location p_f taking into account boundary conditions (3) as well as joint velocity and acceleration limits (4) and (5). However, in a typical industrial application the robot passes through a w-elemental set of waypoints

$$P = \{p_1, p_2, \dots, p_w\},$$

where: p_w is a final location achieved by the robot at the end of the task.

Proposed solution of elementary robot task can be used directly to move the manipulator between the waypoints, when it stops at each of them. In this case the motions between waypoints are planned using dependency (17) taking subsequent waypoints as final location and the configuration achieved in the previous stage as an initial configuration. When the task of the manipulator is to move TCP through the waypoints without stopping the solution has to be adapted. For this purpose, in the presented paper, the approach of modifying error function (6) in the ϵ-neighborhood of the i-th waypoint is proposed. The new form of error function is introduced as follows

$$e(t) = \begin{cases} k(q) - p_i & \text{for } \delta \geq \epsilon \text{ or } i = w \\ k(q) - (\sigma p_{i+1} + (1-\sigma)p_i & \text{for } \delta < \epsilon \text{ and } i < w, \end{cases} \tag{19}$$

where: p_i is a current waypoint towards which TCP of the manipulator is moving or in whose neighborhood it is located, p_{i+1} is a next waypoint from the set P, δ denotes distance between current position of TCP and waypoint p_i, $\sigma \in [0, 1]$ is a coefficient increasing its value when TCP moves through the neighborhood of the i-th waypoint in such a way that it takes value 0 when TCP enters and 1 when it leaves the neighborhood.

As it can be seen the derivatives of the original tracking error (6) and the modified error (19) are the same, so the change of function $e(t)$ does not affect the other terms of the differential equation (9). Hence, the proposed solution of the elementary robot task (17) with new form of error function $e(t)$ can be also used when the task of the manipulator is to move TCP through the waypoints without stopping.

EXPERIMENTAL SETUP AND RESULTS

In order test the effectiveness of the proposed solution series of simulations and experiments involving 6-DOF industrial robot were performed. In all cases the UR10e manipulator from Universal Robots was used. The task of the robot was to move TCP from initial pose through series of intermediate waypoints (without stopping) to final location. Three simulations were performed: in the first one inequality conditions (4), (5) were not considered, in the second case only condition (4) was taken into account

and in the last case all conditions were taken into consideration. Finally, two experiments using real robot were conducted. In the first case the task of the robot was realized using original UR10e software, in the second experiment the manipulator accomplished the trajectory prepared using the presented method, based on dependency (17).

Model and task of the manipulator

The parameters of the robot according to modified Denavit-Hartenberg notation, based on producer specification [26], are collected in Table 1.

Table 1. D-H parameters of the UR10e manipulator

i	$\alpha_{i-1}\,[rad]$	$a_{i-1}\,[m]$	$d_i\,[m]$	$\theta_i\,[rad]$
1	0	0	0.1807	θ_1
2	$\pi/2$	0	0	θ_2
3	0	-0.6127	0	θ_3
4	0	-0.57155	0.17415	θ_4
5	$\pi/2$	0	0.11985	θ_5
6	$-\pi/2$	0	0.11655	θ_6

Using Z-Y-X Euler angles to describe TCP orientation (notation utilized in producer software), kinematic equation of the manipulator, corresponding to D-H parameters presented in Table 1, can be determined by dependency

$$k(q) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \phi_X \\ \phi_Y \\ \phi_Z \end{bmatrix} = \begin{bmatrix} d_6 r_{13} + d_5 c_1 s_{234} + a_3 c_1 c_{23} + a_2 c_1 c_2 + d_4 s_1 \\ d_6 r_{23} + d_5 s_1 s_{234} + a_3 s_1 c_{23} + a_2 s_1 c_2 - d_4 c_1 \\ d_6 r_{33} - d_5 c_{234} + a_3 s_{23} + a_2 s_2 + d_1 \\ atan2(r_{32}, r_{33}) \\ atan2(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}) \\ atan2(r_{21}, r_{11}) \end{bmatrix}$$

where $c_1 = \cos(\theta_1)$, $s_1 = \sin(\theta_1)$, $c_2 = \cos(\theta_2)$, $s_2 = \sin(\theta_2)$, $c_{23} = \cos(\theta_2 + \theta_3)$, $s_{23} = \sin(\theta_2 + \theta_3)$, $c_{234} = \cos(\theta_2 + \theta_3 + \theta_4)$, $s_{234} = \sin(\theta_2 + \theta_3 + \theta_4)$, $c_5 = \cos(\theta_5)$, $s_5 = \sin(\theta_5)$, $c_6 = \cos(\theta_6)$, $s_6 = \sin(\theta_6)$, $r_{11} = c_1 c_{234} c_5 c_6 - c_1 s_{234} s_6 + s_1 s_5 c_6$, $r_{12} = -s_1 s_5 s_6 - c_1 c_{234} c_5 s_6 - c_1 s_{234} c_6$, $r_{13} = s_1 c_5 - c_1 c_{234} s_5$, $r_{21} = -c_1 s_5 c_6 + s_1 c_{234} c_5 c_6 - s_1 s_{234} s_6$, $r_{22} = c_1 s_5 s_6 - s_1 c_{234} c_5 s_6 - s_1 s_{234} c_6$, $r_{23} = -c_1 c_5 - s_1 c_{234} s_5$, $r_{31} = c_{234} s_6 + s_{234} c_5 c_6$, $r_{32} = c_{234} c_6 - s_{234} c_5 s_6$, $r_{33} = -s_{234} s_5$.

Analytical Jacobian of the manipulator $J(q)$ can be computed from the geometric Jacobian $J_g(q)$ using known transformation relating TCP angular velocity with changes of Z-Y-X Euler angles as follows

$$J = \begin{bmatrix} I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & B^{-1}(\phi) \end{bmatrix} J_g(q),$$

where $I_{3\times3}$, $0_{3\times3}$ denote 3×3 identity and zero matrices,

$$B(\phi) = \begin{bmatrix} 0 & -s_Z & c_Y c_Z \\ 0 & c_Z & c_Y s_Z \\ 1 & 0 & -s_Y \end{bmatrix}, s_Y = \sin(\phi_Y), c_Y = \cos(\phi_Y), s_Z = \sin(\phi_Z), c_Z = \cos(\phi_Z),$$

$$J_g(q) = \begin{bmatrix} -x_2 & -c_1(x_3 - d_1) & c_1(v_1 - a_3 s_{23}) & c_1 v_1 & -d_6(s_1 s_5 + c_1 c_{234} c_5) & 0 \\ x & -s_1(x_3 - d_1) & s_1(v_1 - a_3 s_{23}) & s_1 v_1 & d_6(c_1 s_5 - s_1 c_{234} c_5) & 0 \\ 0 & s_1 x_2 + c_1 x_1 & v_2 + a_3 c_{23} & v_2 & -d_6 s_{234} c_5 & 0 \\ 0 & s_1 & s_1 & s_1 & c_1 s_{234} & r_{13} \\ 0 & -c_1 & -c_1 & -c_1 & s_1 s_{234} & r_{23} \\ 1 & 0 & 0 & 0 & -c_{234} & r_{33} \end{bmatrix},$$

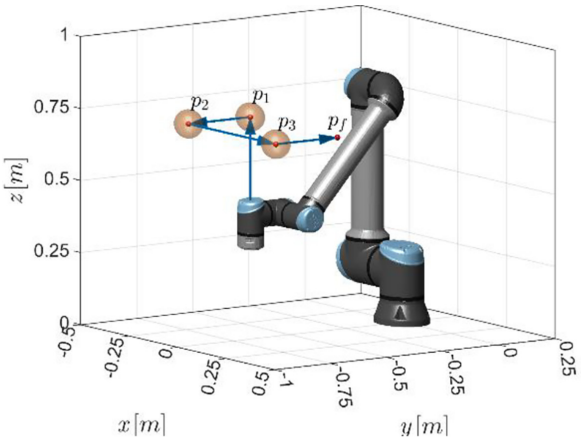

Figure 1. The manipulator and the task

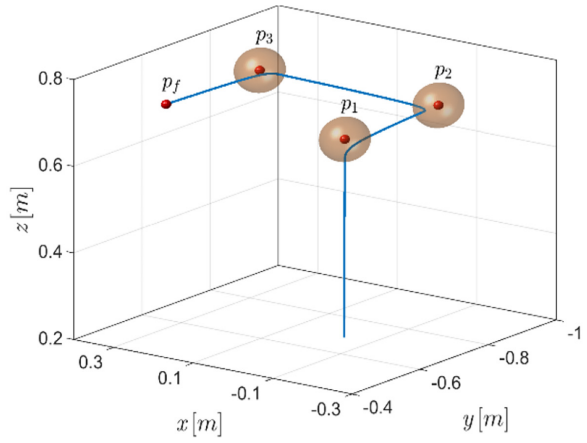

Figure 2. The path of TCP in the first simulation

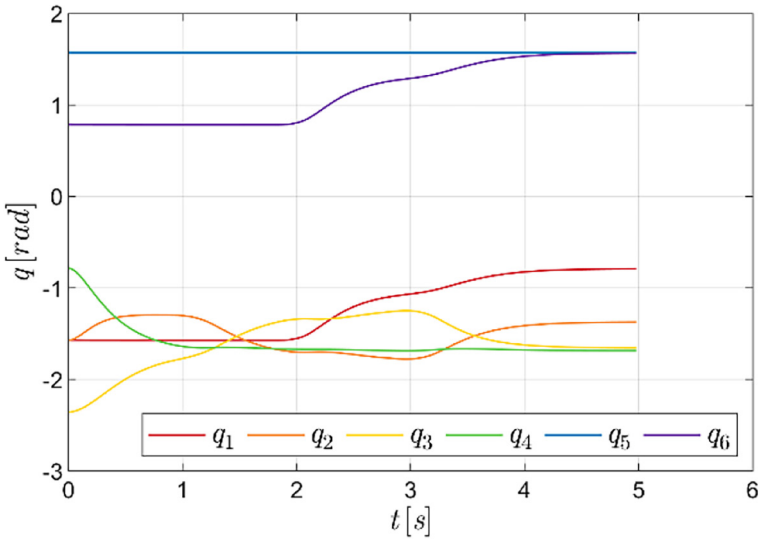

Figure 3. Generalized coordinates obtained in the first simulation

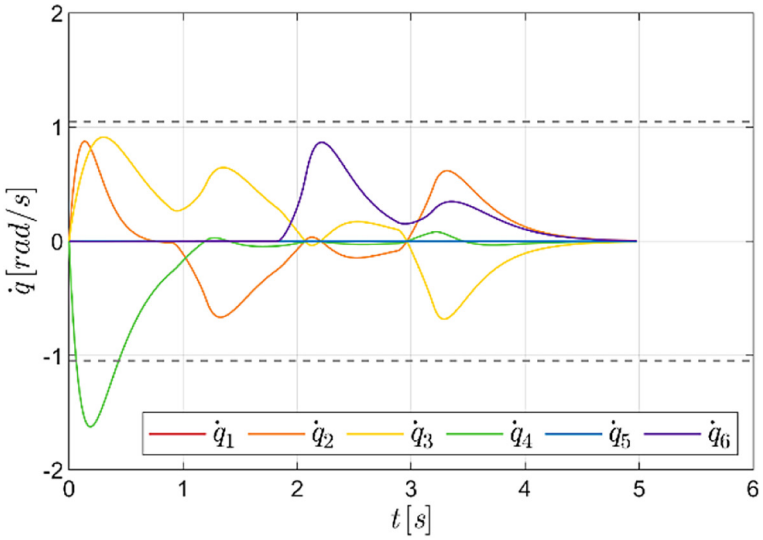

Figure 4. Generalized velocities obtained in the first simulation

$v_1 = d_6 s_{234} s_5 + d_5 c_{234}, v_2 = -d_6 c_{234} s_5 + d_5 s_{234}.$

The velocity and acceleration limits were the same for all joints and they were adopted based on the default parameters set by the producer in its software as

$$\dot{q}^i_{max} = -\dot{q}^i_{min} = 60\ [°/s] \approx 1.05\ [rad/s] \qquad i = 1, \dots, 6,$$
$$\ddot{q}^i_{max} = -\ddot{q}^i_{min} = 80\ [°/s^2] \approx 1.40\ [rad/s^2] \quad i = 1, \dots, 6.$$

At the initial moment the manipulator was in the configuration

$$q(0) = [\,-\pi/2\,, -\pi/2\,, -3\pi/4\,, -\pi/4\,, \pi/2\,, \pi/4\,]^T,$$

which corresponded to the following initial position and orientation of TCP

$$p_0 = [x_0^T, \phi_0^T]^T = [\,-0.174, -0.524, 0.273, \pi, 0, 3\pi/4\,]^T.$$

The task of the robot is to move its TCP to the final location

$$p_f = \left[x_f^T, \phi_f^T\right]^T = [\,0.275, -0.525, 0.730, \pi, 0, 3\pi/4\,]^T$$

passing three intermediate waypoints

$$p_1 = [x_1^T, \phi_1^T]^T = [\,-0.175, -0.525, 0.730, \pi, 0, 3\pi/4\,]^T,$$
$$p_2 = [x_2^T, \phi_2^T]^T = [\,-0.175, -0.800\ 0.730, \pi, 0, 3\pi/4\,]^T,$$
$$p_3 = [x_3^T, \phi_3^T]^T = [\,0.275, -0.800, 0.730, \pi, 0, 3\pi/4\,]^T.$$

The manipulator in the initial configuration and its task is shown in Figure 1. In all performed simulations and experiments the following algorithm parameters were assumed:

- gain coefficients of dependency (16)
$$\Lambda_P^i = 20, \Lambda_V^i = 9.392, i = 1, \dots, 6,$$
- accuracy of the task performance (i.e. TCP positioning precision in final location)
$$\varepsilon = 0.001\ [m],$$
- size of the neighborhoods around the waypoints
$$\epsilon = 0.05\ [m],$$
- size of velocity safety margins
$$\vartheta = 0.1.$$

Simulations

The trajectory planning algorithm designed based on the method presented in this paper were written end tested in MATLAB R2023a environment. The obtained results were visualized using the library codeveloped by the author and available online The Robot Toolbox for Matlab 2.0 [27]. Three simulations were performed in order to show the effectiveness of the proposed method for trajectory planning with intermediate waypoints and taking into account the constraints on the velocity and acceleration of the robot.

In the first simulation the conditions (4) and (5) were not taken into consideration and the manipulator performed the motion through intermediate waypoints without stopping, according to the modified tracking error approach (19). TCP path, changes of generalized coordinates, velocities and accelerations of the robot, obtained in this simulation, are shown in Figures 2–5 respectively.

As it can be seen, in this case the final time of task execution was equal to 4.98 s. The algorithm generated smooth trajectory (Fig. 3) and TCP moved between waypoints smoothly changing the direction of motion in their neighborhoods (marked with orange spheres in Fig. 2). It is worth to note, that velocity and acceleration limitations (marked in Figures 4 and 5 with a gray dashed lines) were exceed due to the fact, that the corresponding constraints were not taken into account.

In the second simulation velocity limitation (4) was taken into consideration and trajectory was generated using dependency (16). The way of performing the task is presented similarly to the first simulation in the Figures 4–5.

As it is shown perturbation (15) did not significantly affect the trajectory of the manipulator (Fig. 7) and path of TCP (Fig. 6), but allowed to satisfy velocity limitations (Fig. 8), slightly extending the time of completing this task to 5.22 s

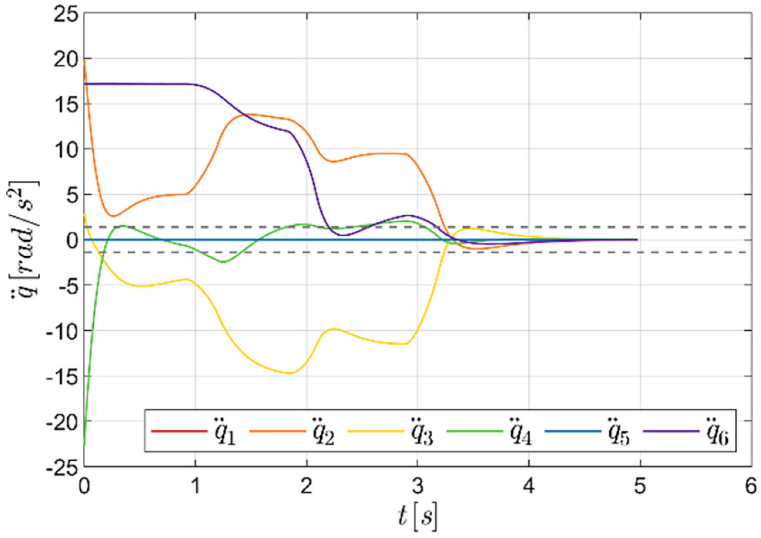

Figure 5. Generalized accelerations obtained in the first simulation

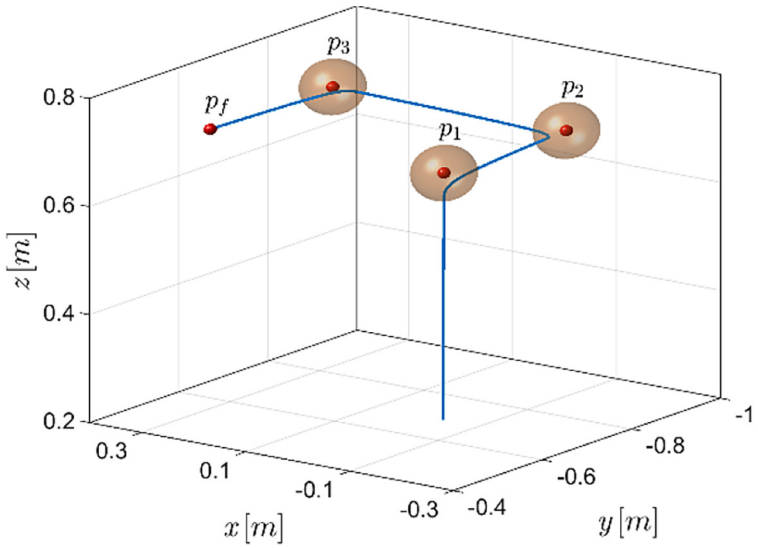

Figure 6. The path of TCP in the second simulation

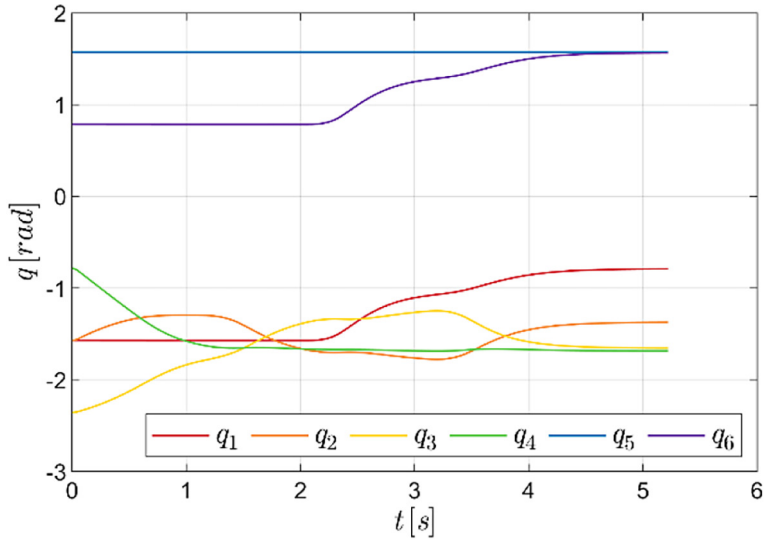

Figure 7. Generalized coordinates obtained in the second simulation

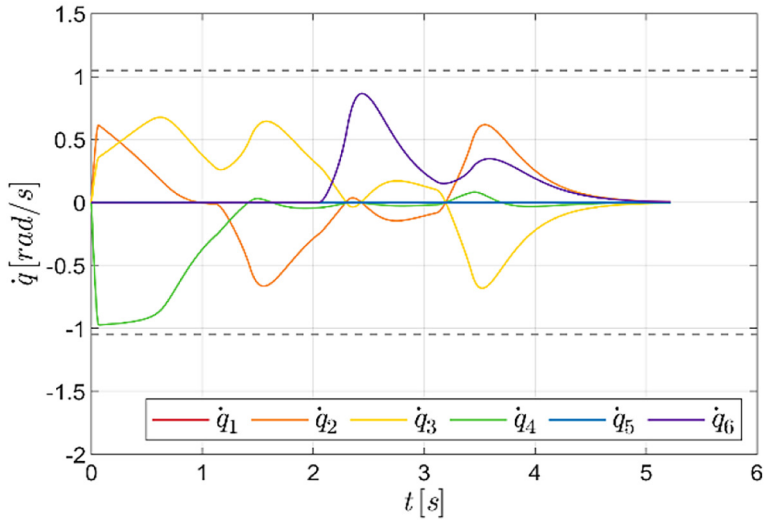

Figure 8. Generalized velocities obtained in the second simulation

(an increase of about 4.8%). The obtained acceleration values (Fig. 8), as in the first simulations, significantly exceeded the assumed limits.

The last simulation took into account both velocity and acceleration constraints and trajectory of the robot was prepared using dependency (17), i.e. utilizing perturbation (15) as well as trajectory scaling approach presented in acceleration limitations section (Fig. 9). As in previous simulations, the obtained results are presented in the Figures 10–13, additionally changes of trajectory scaling coefficient $c(t)$ are presented in Figure 14.

As it can be seen in Figure 13 the proposed approach is effective and any acceleration did not exceed its limit. Moreover, the trajectory scaling method preserves continuous accelerations, which is guaranteed by continuous changes of trajectory scaling coefficient (Fig. 14) and in a consequence it generates smooth changes of generalized coordinates (Fig. 11) as well as joint velocities (Fig. 12). Additionally, as it is seen in Figure 10, TCP correctly moves between the waypoints, passing through their neighborhoods. It is worth emphasizing the high efficiency of the strategy of selection the coefficient $c(t)$, which scales the trajectory only in limited time intervals and starts to grow as soon as possible. Despite a significant reduction in acceleration, such approach allowed to find the trajectory avoiding a significant increase in the time execution, which in this case was equal to 5.32 s (an increase of about 1.9% compared to the previous simulation). Moreover, as expected, taking the initial values of scaling coefficient $c(0) = 0$ ensured a smooth start of the manipulator at the beginning of the task, what can be seen by comparing Figure 12 and 8.

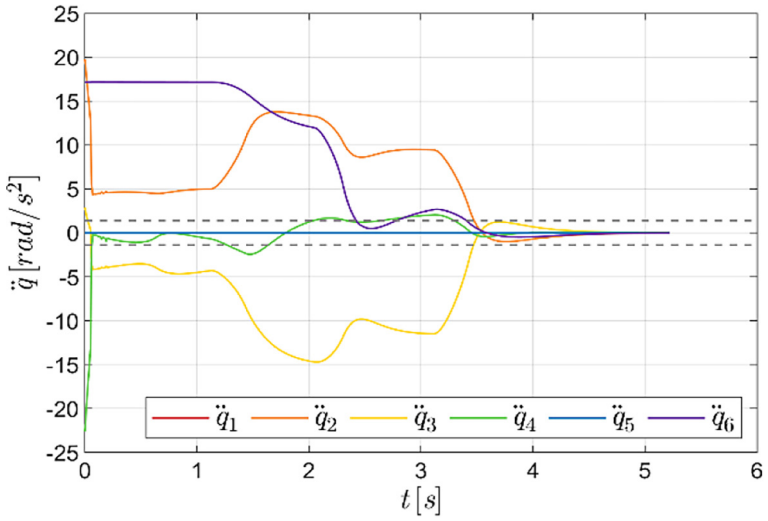

Figure 9. Generalized accelerations obtained in the second simulation

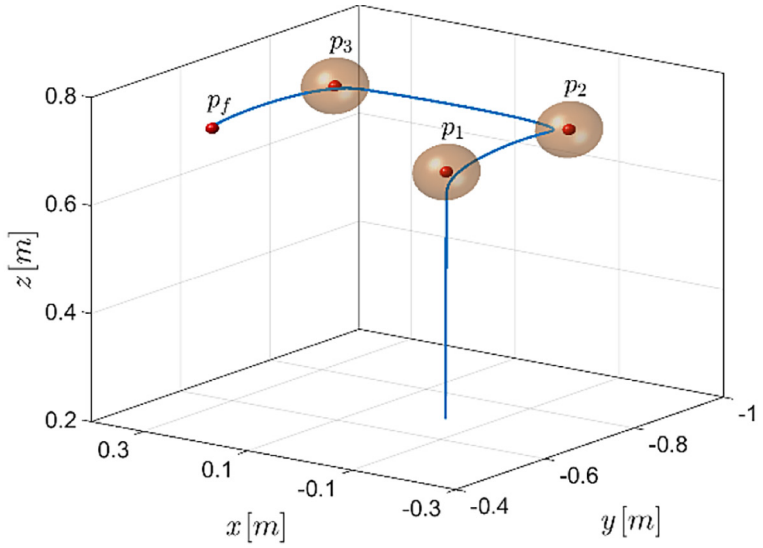

Figure 10. The path of TCP in the third simulation

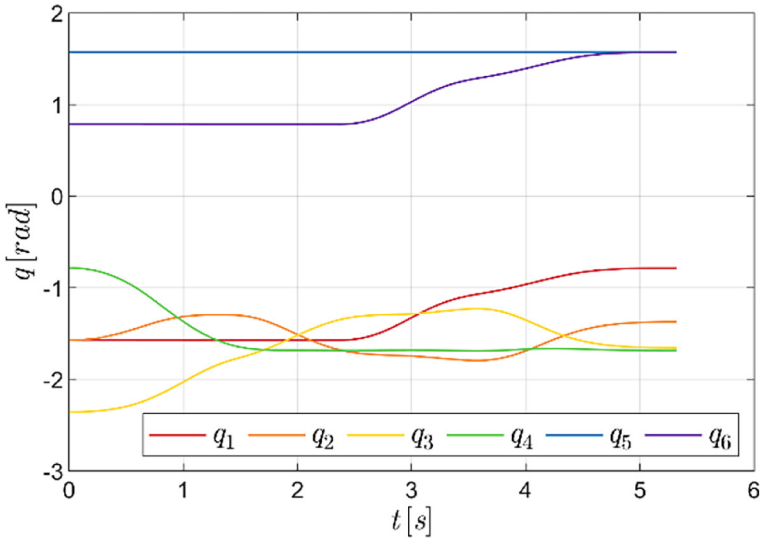

Figure 11. Generalized coordinates obtained in the third simulation

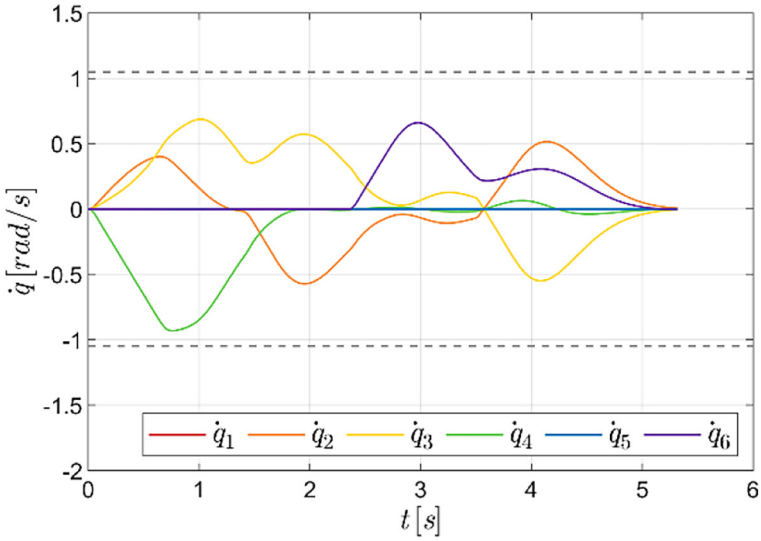

Figure 12. Generalized velocities obtained in the third simulation

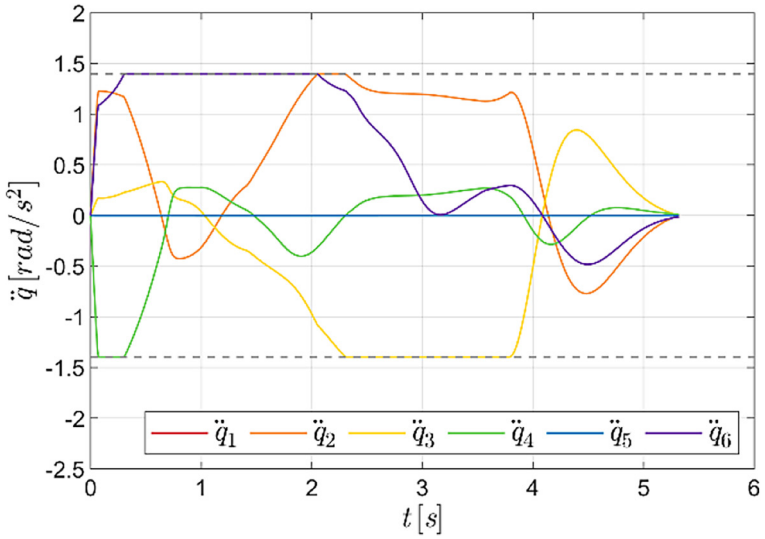

Figure 13. Generalized accelerations obtained in the third simulation

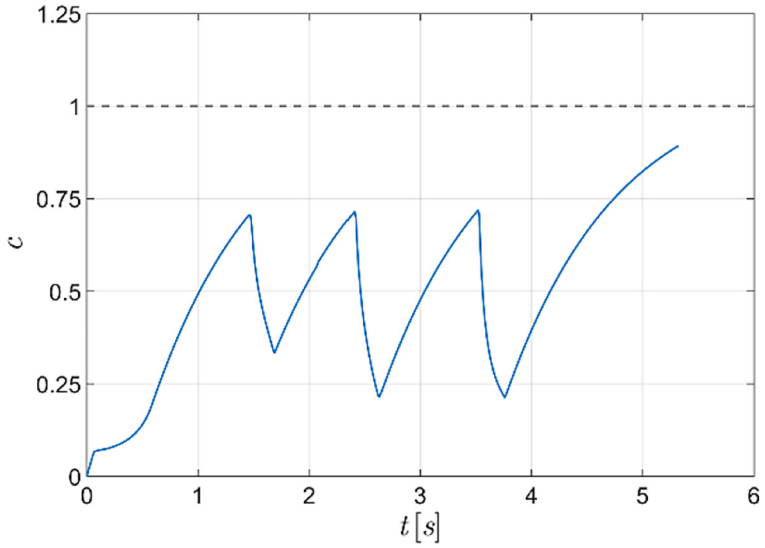

Figure 14. Changes of trajectory scaling coefficient

The key results obtained in the conducted simulations are summarized in the Table 2. The following columns present the task execution time resulting from the adopted task parameters, the minimum and maximum of joints velocities and accelerations and the final value of error function, i.e. positioning accuracy at the final location.

Experiments

In order to verify the effectiveness of the proposed method in real case two experiments were conducted. In the first one the robot task presented at the beginning of this section was performed utilizing original UR10e software, in the second one the robot traced trajectory generated

Table 2. The key results obtained in the simulations

Simulation	T $[s]$	$max(\dot{q}_i)$ $[rad/s]$	$min(\dot{q}_i)$ $[rad/s]$	$max(\ddot{q}_i)$ $[rad/s^2]$	$min(\ddot{q}_i)$ $[rad/s^2]$	$e(q(T))$ $[m]$
1	4.98	0.91*	−1.62*	19.77*	−22.63*	$5.81e-04$
2	5.22	0.86	−0.97	19.77*	−22.63*	$5.68e-04$
3	5.32	0.69	−0.93	1.40	−1.40	$6.26e-04$

Note: * limitations not taken into consideration.

in the third simulation. In both cases the way of performing the task (i.e. actual configurations and velocities of robot joints) was recorded in real-time at a frequency of 100 Hz using UR Log Viewer software provided by Universal Robots. Experimental setup is presented in Figure 15 and videos documenting both experiments are available online at https://staff.uz.zgora.pl/gpajak/rtoolbox/astrj2024.

The first experiment was carried out in order to analyze the execution of the task by the original robot controller. For this purpose the manipulator realized the program in which TCP moved between waypoints, using "MoveJ" command. The default constraints on velocity and acceleration of the robot joints were adopted, which correspond to those used in simulations (60% and 80% respectively). In order to avoid the robot stopping at the waypoints technique available in the controller named "radius blending" [28] with the radius equal to size of neighborhoods (0.05 m) was used. TCP path, general coordinates and velocities recorded from robot sensors are presented in Figures 16–18. It should be noted, that robot software does not provide information about joint accelerations obtained by the robot during the task execution and the only available data are accelerations generated by the controller shown in Figure 19.

According to the manual of the robot trajectory is planned by the controller using trapezoidal velocity profile, which is confirmed by obtained experimental results. Such approach allowed to achieve correct TCP path (Fig. 16) and smooth trajectories (Fig. 17). It also guarantees the continuity of the velocity and fulfillment of the limitations, but does not ensure its smoothness (Fig. 18). As it can be seen in Figure 19 at the certain time instant

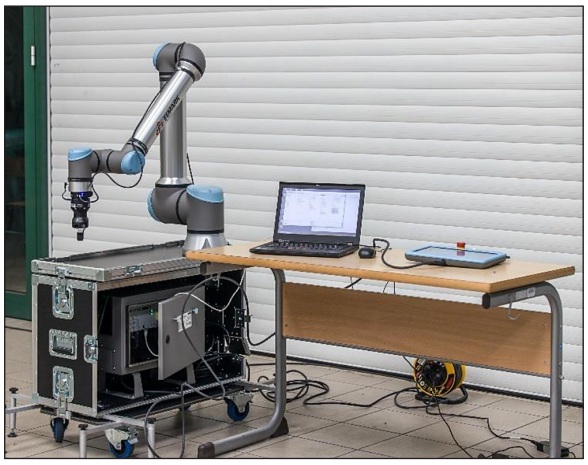

Figure 15. Experimental setup

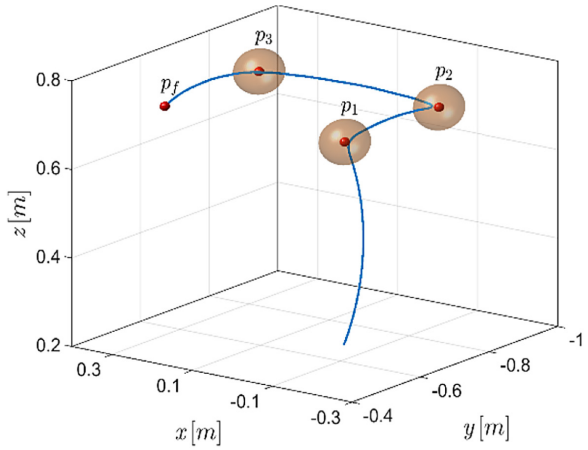

Figure 16. The path of TCP in the first experiment

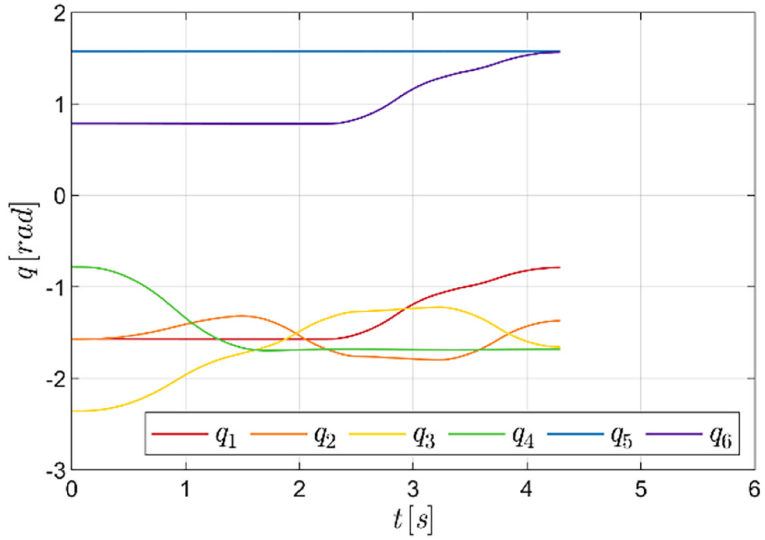

Figure 17. Generalized coordinates recorded in the first experiment

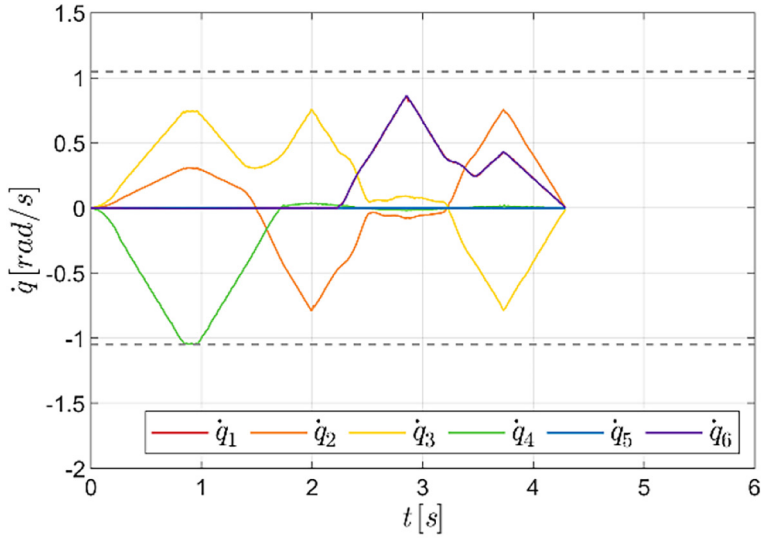

Figure 18. Generalized velocities recorded in the first experiment

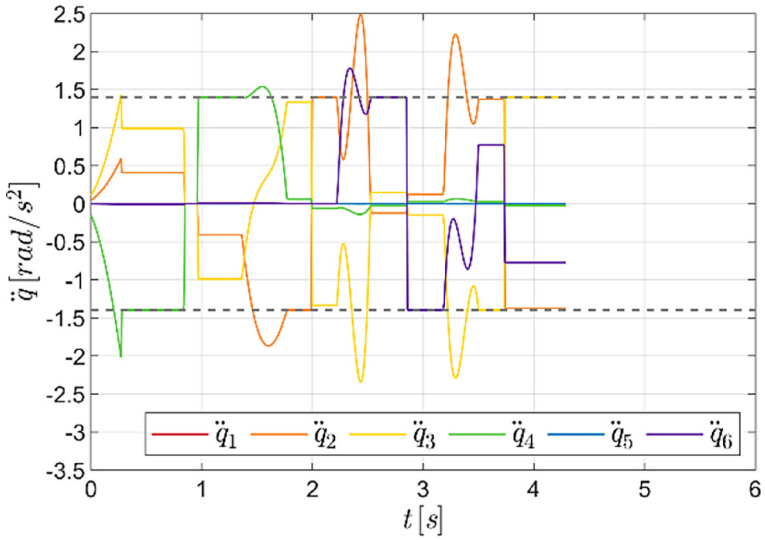

Figure 19. Generalized accelerations generated by controller in the first experiment

there are rapid changes in accelerations results in rapid changes in direction of movement affecting the undesirable vibration of the robot arm. In the experimental setup the base of the robot was not rigid connected to the laboratory floor, so mentioned effect was clear and it can be seen in the video documenting this experiment. It is worth noting that the task was performed using default velocity and acceleration limitations, far from the available maximum values, so this undesirable effect will be stronger when increasing the constraints. Moreover, as it is seen in Figure 19 the robot controller planning the trajectory took into account the joint acceleration limits on sections between waypoints, however they were significantly exceeded in the neighborhoods of the waypoints when the blending technique was used.

In the second experiment the manipulator traced trajectory generated in the third simulation. For this purpose trajectory points generated by proposed algorithm were sent to the robot using TCP/IP connection and synchronized by Real-Time Data Exchange (RTDE) interface. The data were received by application running on the robot controller and the trajectory was traced using "ServoJ" command. Registered TCP path, general coordinates and velocities of the robot are presented in Figures 2022. As it is mentioned above the robot controller does not provide information about current accelerations, so they cannot be shown.

As it can be seen the robot followed the planned trajectory precisely, so the changes of generalized coordinates (Fig. 21) as well as TCP path (Fig. 20) collected during the task execution

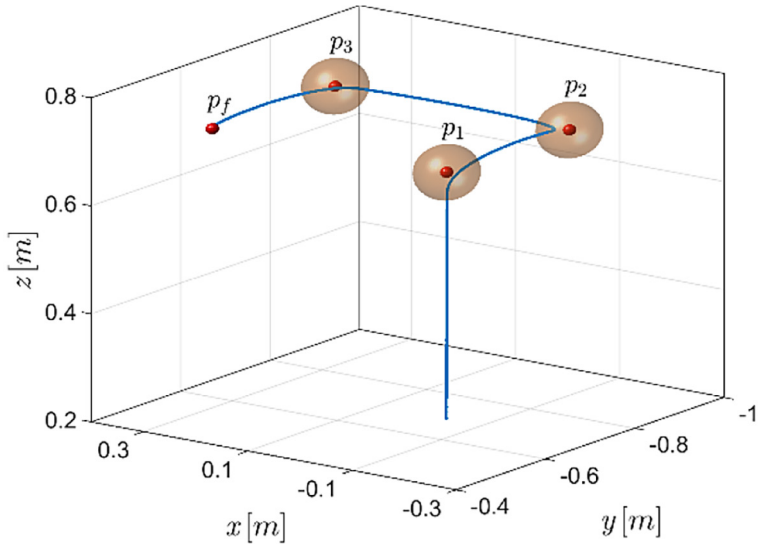

Figure 20. The path of TCP in the second experiment

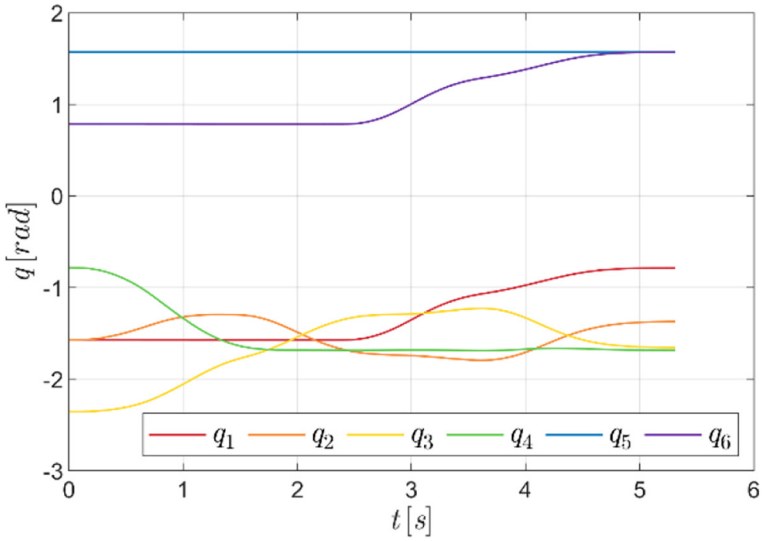

Figure 21. Generalized coordinates recorded in the second experiment

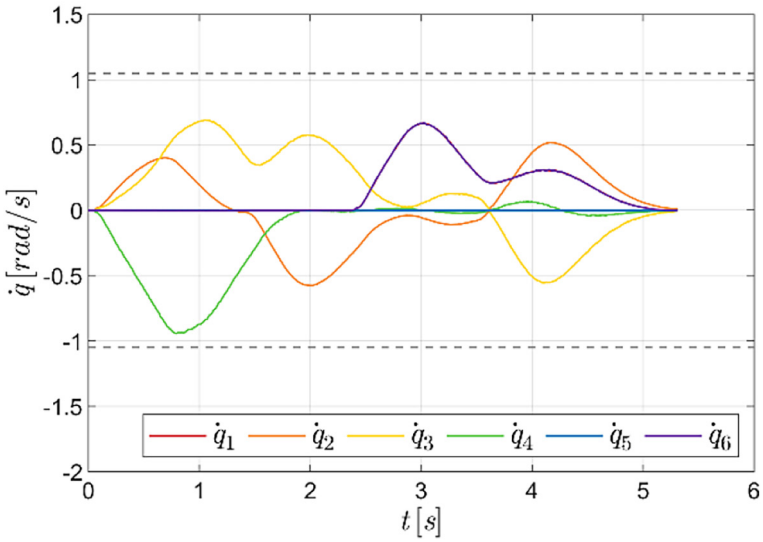

Figure 22. Generalized velocities recorded in the second experiment

Table 3. The key results obtained in the experiments

Experiment	T $[s]$	$max(\dot{q}_i)$ $[rad/s]$	$min(\dot{q}_i)$ $[rad/s]$	$max(\ddot{q}_i)$ $[rad/s^2]$	$min(\ddot{q}_i)$ $[rad/s^2]$	$e(q(T))$ $[m]$
1	4.29	0.86	-1.05	2.48	-2.34	$5.64e-05$
2	5.31	0.69	-0.94	1.40	-1.40	$9.76e-04$

by the real robot are the same as in the third simulation (Figs 11 and 10 respectively). Slight disturbances can be seen in the velocity profiles (Fig. 22), but they are probably due to problems with RTDE interface or incorrect tuning of the ServoJ command (the issues connected to synchronizations of external applications with robot controller will be investigated further). However, as in is shown in Figure 22 registered velocities were continuous, smooth and satisfied the assumed constraints. Due to the lack of relevant data, it is not possible to compare the real accelerations obtained in both experiments, but it is possible to compare accelerations generated by the robot controller and proposed algorithm. It should be emphasize that the proposed method, unlike the robot controller, preserved acceleration constraint for the entire duration of the task (Fig. 13 vs. 19). Neglecting this constraint may result in a shorter task execution time for the first experiment (4.29 s vs. 5.32 s), but it is worth noting that during performing the trajectory planned according the approach proposed in this paper undesirable vibration of the robot arm were significantly reduced, which can be seen in the video documenting this experiment. Moreover, the parameters of the proposed algorithm allow for trajectory tunning and gives hope for better results in future research.

The key results obtained in the experiments are summarized in the Table 3. The subsequent columns correspond to Table 2.

CONCLUSIONS

In this paper a method of trajectory planning for 6-DOF industrial manipulator performing motion through intermediate waypoints was presented. The proposed approach allows the robot to complete the task both when it is necessary to stop at intermediate points and also when it is not needed. The trajectory of the robot is planned on the acceleration level in such a way that both joint velocity and acceleration constraints are taken into account. Moreover, the obtained velocity profiles, unlike the trapezoidal profiles generated by the robot controller, are smooth, which significantly reduces undesirable vibrations of the manipulator arm and leads to smooth robot motion. Additionally, the proposed algorithm can be used in autonomous systems to generate complex trajectories without human participation.

The presented results are the first step to apply previously developed methods to real industrial robots. Based on the conducted research it seems that proposed method is effective, but it can be improved. The further research will focus on considering manipulability of the robot, collision avoidance issues, obtaining smooth accelerations as well as improving the synchronization of the robot controller with an external application.

Acknowledgements

This work was partially supported by a program of the Polish Ministry of Science under the title ‘Regional Excellence Initiative’, project no. RID/SP/0050/2024/1.

REFERENCES

1. Wang S.Y., Wan J.F., Li D., Zhang C.H. Implementing smart factory of industrie 4.0: An outlook. International Journal of Distributed Sensor Networks 2016; 12(1): 3159805, DOI: 10.1155/2016/3159805.
2. Sajjad A., Ahmad W., Hussain S. Decision making process development for industry 4.0 transformation. Advances in Science and Technology Research Journal 2022; 16(3): 1–11, DOI: 10.12913/22998624/147237.
3. Sobaszek Ł. A lean robotics approach to the scheduling of robotic adhesive dispensing process. Advances in Science and Technology Research Journal 2022; 16(5): 136–146, DOI: 10.12913/22998624/152332.
4. Pizoń, J., Cioch, M., Kanski, L., García, E.S. Cobots implementation in the era of industry 5.0 using modern business and management solutions. Advances in Science and Technology Research Journal 2022, 16(6): 166–178, DOI: 10.12913/22998624/156222.
5. Bochen A., Ambrożkiewicz B. The influence of light intensity on the operation of vision system in collaborative robot. Advances in Science and

Technology Research Journal 2023, 17(4): 206–214, DOI: 10.12913/22998624/169884.

6. Palmieri G., Scoccia C. Motion planning and control of redundant manipulators for dynamical obstacle avoidance. Machines 2021; 9(6): 121, DOI: 10.3390/machines9060121.
7. Guruji A.K., Agarwal H., Parsediya D. Time-efcient A* algorithm for robot path planning. Procedia Technol. 2016; 23: 144–149, DOI: 10.1016/j.protcy.2016.03.010.
8. Pajak I. Real-time trajectory generation methods for cooperating mobile manipulators subject to state and control constraints. Journal of Intelligent and Robotic Systems 2019; 93(3–4, SI): 649–668, DOI: 10.1007/s10846-018-0878-5.
9. Chen X.Y., You X.J., Jiang J.C., Ye J.H., Wu H.B. Trajectory planning of dual-robot cooperative assembly. Machines 2022; 10(8): 689, DOI: 10.3390/machines10080689.
10. Pardi T., Ortenzi V., Fairbairn C., Pipe T., Esfahani A.M.G., Stolkin R. Planning Maximum-Manipulability Cutting Paths. IEEE Robotics and Automation Letters 2020; 5(2): 1999–2006, DOI10.1109/LRA.2020.2970949.
11. Chen N., Song F., Li G., Sun X., Ai C. An adaptive sliding mode backstepping control for the mobile manipulator with nonholonomic constraints. Communications in Nonlinear Science and Numerical Simulation 2013; 18(10): 2885–2899, DOI: 10.1016/j.cnsns.2013.02.002.
12. Dexu B., Wei S., Hongshan Y., Cong W., Hui Z. Adaptive robust control based on rbf neural networks for duct cleaning robot. International Journal of Control Automation and Systems 2015; 13(2): 475–487, DOI: 10.1007/s12555-012-0447-9.
13. Tamizi M.G., Yaghoubi M., Najjaran, H. A review of recent trend in motion planning of industrial robots. International Journal of Intelligent Robotics and Applications 2023; 7(2): 253–274, DOI: 10.1007/s41315-023-00274-2.
14. Boscariol P., Gasparetto A., Vidoni R. Planning continuous-jerk trajectories for industrial manipulators. In: Proc of 11th ASME Biennial Conference on Engineering Systems Design and Analysis, (ESDA 2012), Nantes, France 2012, 127–136, DOI: 10.1115/ESDA2012-82103.
15. Huang J.S., Hu P.F., Wu K.Y., Zeng M. Optimal time-jerk trajectory planning for industrial robots. Mechanism and Machine Theory 2018; 121: 530–544, DOI: 10.1016/j.mechmachtheory.2017.11.006.
16. Lin J.J., Rickert M., Knoll A. Parameterizable and jerk-limited trajectories with blending for robot motion planning and spherical cartesian waypoints. In: Proc of IEEE International Conference on Robotics and Automation ICRA 2021, Xian, China 2021, 13982–13988, DOI: 10.1109/ICRA48506.2021.9561601.
17. Scheiderer C., Thun T., Meisen T. Bezier curve based continuous and smooth motion planning for self-learning industrial robots. In: Proc of 29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM) - Beyond Industry 4.0 - Industrial Advances, Engineering Education and Intelligent Manufacturing, Limerick, Ireland 2019, 423–430, DOI: 10.1016/j.promfg.2020.01.054.
18. Meyes R., Scheiderer C., Meisen T. Continuous motion planning for industrial robots based on direct sensory input. In: Proc of 51st CIRP Conference on Manufacturing Systems, Stockholm, Sweden 2018, 291–296, DOI: 10.1016/j.procir.2018.03.067.
19. He F.F., Huang Q.J. Time-optimal trajectory planning of 6-DOF manipulator based on fuzzy control. Actuators 2022; 11(11), DOI: 10.3390/act11110332.
20. Gasparetto A., Boscariol P., Lanzutti A., Vidoni R. Path planning and trajectory planning algorithms: A general overview. Motion and Operation Planning of Robotic Systems: Background and Practical Approaches, Mechanisms and Machine Science, Springer 2015, DOI: 10.1007/978-3-319-14705-5_1.
21. Pajak G., Pajak I. Point-to-point collision-free trajectory planning for mobile manipulators. Journal of Intelligent and Robotic Systems 2017; 85(3-4, SI): 523-538, DOI: 10.1007/s10846-016-0390-8.
22. Pajak G., Pajak I. Planning of a point to point collision-free trajectory for mobile manipulators. In: Proc of 10th International Workshop on Robot Motion and Control (RoMoCo), Poznan, Poland 2015, 142–147.
23. Pajak G. Trajectory planning for mobile manipulators subject to control constraints. In: Proc of 11th International Workshop on Robot Motion and Control (RoMoCo) 2017, 117–122.
24. Pajak G., Pajak I. Collision-free trajectory planning for mobile manipulators subject to control constraints. Archive of Mechanical Engineering 2014; 61(1): 35–55, DOI: 10.2478/meceng-2014-0002.
25. Pajak I., Pajak G. Motion planning for a mobile humanoid manipulator working in an industrial environment. Applied Sciences-Basel 2021; 11(13), DOI: 10.3390/app11136209.
26. DH parameters for calculations of kinematics and dynamics. Available online: https://www.universal-robots.com/articles/ur/application-installation/dh-parameters-for-calculations-of-kinematics-and-dynamics
27. The Robot Toolbox for Matlab 2.0 (Pajak G., Pajak I). Available online: http://staff.uz.zgora.pl/gpajak/rtoolbox (accessed on 23 July 2023).
28. Universal Robots e-Series User Manual. Available online: https://www.universal-robots.com/download/manuals-e-series/user/ur3e/512/user-manual-ur3e-e-series-sw-512-english-international-en/