10.1515/acsc-2015-0033

Archives of Control Sciences
Volume 25(LXI), 2015
No. 4, *pages 513–527*

Reference trajectory tracking for a multi-DOF robot arm

RÓBERT KRASŇANSKÝ, PETER VALACH, DÁVID SOÓS, JAVAD ZARBAKHSH

This paper presents the problem of tracking the generated reference trajectory by the simulation model of a multi-DOF robot arm. The kinematic transformation between task space and joint configuration coordinates is nonlinear and configuration dependent. To obtain the solution of the forward kinematics problem, the homogeneous transformation matrix is used. A solution to the inverse kinematics is a vector of joint configuration coordinates calculated using of pseudoinverse Jacobian technique. These coordinates correspond to a set of task space coordinates. The algorithm is presented which uses iterative solution and is simplified by considering stepper motors in robot arm joints. The reference trajectory in Cartesian coordinate system is generated on-line by the signal generator previously developed in MS Excel. Dynamic Data Exchange communication protocol allows sharing data with Matlab-Simulink. These data represent the reference tracking trajectory of the end effector. Matlab-Simulink software is used to calculate the representative joint rotations. The proposed algorithm is demonstrated experimentally on the model of 7-DOF robot arm system.

Key words*:* inverse kinematics, real-time reference tracking, signal generator, multi-DOF, dynamic data exchange.

1. Introduction

Robotics represents one of the main disciplines in the industry. The synergy of robotics with other applications like car assembly operations, vision systems or artificial intelligence allows not only for innovations but also reduces the manufacture costs. In the kinematic analysis of a robot arm position, two separate problems have to be solved: forward kinematics and inverse kinematics. Forward kinematics includes solving the forward transformation equation to find the location of the hand in terms of the angles and displacements between the links of the robot arm.

R. Krasňanský, P. Valach and D. Soós are with Faculty of Electrical Engineering and IT, Slovak University of Technology in Bratislava, Ilkovicova 3, 81219 Bratislava, Slovak Republic. E-mails:(robert.krasnansky, peter.valach, david.soos)@stuba.sk. J. Zarbakhsh is with Carinthia University of Applied Science, Europastrasse 4, 9524 Villach, Austria. E-mail: zarbakhsh@cuas.at.

Received 16.07.2015.

This work has been supported by Grant N. 1/1241/12 of the Slovak Scientific Grant Agency.

In order to find the corresponding sets of joint angle given the desired position and orientation of the end effector it is necessary to solve the inverse kinematics problem. Inverse kinematics, on the other hand, includes solving the inverse transformation equations in order to find the relationships between the links of the robot arm from its location in space. There has been an increasing amount of work in recent years to derive the inverse kinematics of the robot arm [1], [9], [6].

Many schemes as well as profound comparative analysis for the inverse kinematic problem of redundant robot arm are proposed in previous robotics works.

Some schemes provide essential approach for manipulator geometries with unknown inverse kinematic functions. However, for a continuous path motion control problem, the inverse of the Jacobian matrix is required. A generalized inverse of Jacobian matrix, so-called pseudoinverse is widely applied for a redundant robot.

The present work is concerned with the inverse kinematics solution for a seven-DOF redundant manipulator. The trajectory planning of robot arms is a very active area since many tasks require special characteristics to be satisfied. Using of MS Excel and MS VBA (Visual Basic for Applications) software, the application for reference signal generation has been developed. It allows to determine arbitrary reference trajectory and to change it during runtime. By using of the Dynamic Data Exchange (DDE), the communication between the application and Matlab-Simulink is secured in real-time.

The paper is organized as follows. In section 2, the general robot kinematics as well as the inverse kinematics analysis is presented. In section 3, the kinematics model of the seven-DOF robot arm is presented and the inverse kinematics solution is proposed. Section 4 describes a real-time signal generator developed in MS Excel to generate an arbitrary reference signal for the end effector. Subsequently, computer simulation and experimental results of the proposed methodology are presented. Finally, the conclusions are drawn in Section 5.

2. Robot kinematics

2.1. Homogeneous transformation matrix

The analysis the motion, a robot arm can be arranged by assigning certain coordinate frames to each link, in order to obtain a transformation relating joint to space coordinates as the position and orientation of the end effector with respect to a reference frame [3]. The relative position and orientation of translation and rotation of the robot arm in 3D space can be represented by the 4x4 homogeneous transformation matrix in the following form:

$$\Theta = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}. \tag{1}$$

(Notation and symbols are given in the end of this paper).

Homogeneous transformation is used to calculate the new coordinate values for each robot part. The rotation for each axis in the space is represented by the 3x3 matrix, which may change with respect to rotation value:

$$\Theta = \left[\begin{array}{c|c} \begin{array}{c} 3x3 \\ \text{rotation matrix} \end{array} & \begin{array}{c} 3\text{x}1 \\ \text{translation} \end{array} \\ \hline \begin{array}{cc} 1\text{x}3 & \text{perspective} \end{array} & \text{global scale} \end{array}\right] = \left[\begin{array}{ccc|c} r_1 & r_2 & r_3 & \Delta x \\ r_4 & r_5 & r_6 & \Delta y \\ r_7 & r_8 & r_9 & \Delta z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]. \tag{2}$$

The resulting motion is generated by composition of elementary motions of each link with respect to the previous one. The joints are supposed to be controlled individually. The translation matrix has dimension 3x1 and introduces the changing value between the coordinate systems. Using rotation matrices brings a compact representation of the rotational movement of a rigid body.

Consider the rotation of a given reference frame with respect to the world reference frame (rotation about *z*-axis) as depicted in Fig. 1.

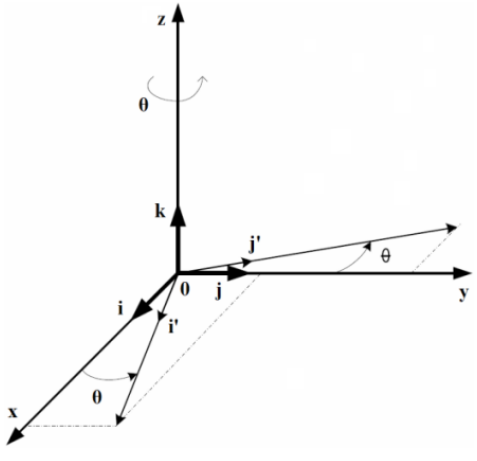

Figure 1. Rotation of frame (x', y', z') respect to the (x,y,z) reference frame

Since the both frames have the common origin, the relationship between them follows only from the rotation. Rotation is then represented by the following matrix:

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

In the same way, we can define other rotations with respect to the axis y and x as follows:

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}, \tag{4}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}. \tag{5}$$

The translation about x, y, and z axes of l is then in the form:

$$T_x = \begin{bmatrix} l \\ 0 \\ 0 \end{bmatrix}, \quad T_y = \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix}, \quad T_z = \begin{bmatrix} 0 \\ 0 \\ l \end{bmatrix}. \tag{6}$$

Consider the 2-DOF robot arm as depicted in Fig. 2.

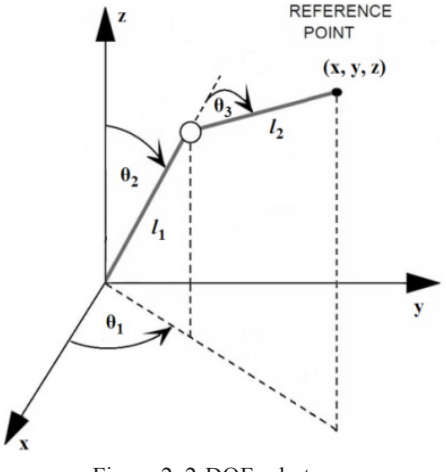

Figure 2. 2-DOF robot arm

Equation representing the dependence of the end effector (EE) coordinates on joint coordinates and vice versa can be obtained according to the basic trigonometry in the following way:

$$EE = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = R_z(\theta_1) R_y(\theta_2) T_z(l_1) R_y(\theta_3) \begin{bmatrix} 0 \\ 0 \\ l_2 \\ 1 \end{bmatrix} \tag{7}$$

where

$$R_z(\theta_1) = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

$$R_y(\theta_2) = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

$$T_z(l_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$R_y(\theta_3) = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_3) & 0 & \cos(\theta_3) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{11}$$

Resulting position of the end effector is described by the following equations

$$x = l_1 \cos\theta_1 \sin\theta_2 + l_2 \cos\theta_1 \sin(\theta_2 + \theta_3) \tag{12}$$

$$y = l_1 \sin\theta_1 \sin\theta_2 + l_2 \sin\theta_1 \sin(\theta_2 + \theta_3) \tag{13}$$

$$z = l_1 \cos\theta_2 + l_2 \cos(\theta_2 + \theta_3). \tag{14}$$

2.2. Inverse kinematics

Inverse kinematics uses the kinematics equations of a robot to determine the joint parameters providing a desired position of the end effector. For example, using these formulas allows calculation of the joint parameters which set a robot arm to pick up an object. Similar formulas might determine the positions of the skeleton of an animat-

ed character which is supposed to move in a particular way. However, the solution of the inverse kinematics problem is not always unique, since the same end effector pose might be reached within several configurations corresponding to distinct joint position vectors.

There are many methods of modeling and solving inverse kinematics problems. Besides of analytic methods, most flexible methods typically rely on iterative optimization in order to find an approximate solution. The main reasons are the difficulty of inverting the forward kinematics equation and the possibility of an empty solution space [5], [8], [2].

The inverse Jacobian technique is a simple yet effective way of implementing inverse kinematics. The velocities in joint space are being mapped to velocities in Cartesian space. It contains the first order partial derivatives of the joint angle parameters:

$$J = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \cdots & \frac{\partial x}{\partial \theta_n} \\ \frac{\partial y}{\partial \theta_1} & \cdots & \frac{\partial y}{\partial \theta_{n1}} \\ \frac{\partial z}{\partial \theta_1} & \cdots & \frac{\partial z}{\partial \theta_n} \end{bmatrix}. \tag{15}$$

Calculation the inverse Jacobian allows the inverse kinematics to be directly calculated for a small time step as follows:

$$\Delta\theta = J^{-1}(\theta)\Delta x. \tag{16}$$

According to (16), the real function can be linearly approximated for sufficiently small steps. For kinematically redundant robots with $n > 6$ DOF, the Jacobian matrix is non-square and its inverse can be obtained using a pseudoinverse J^{+} [4], [10]

$$J^{+} = (J^{T}J)^{-1}J^{T}. \tag{17}$$

However, by using the pseudoinverse, the problems of singularities in the Jacobian still remain. These singularities might occur for instance, when multiple links become aligned in the same direction and subsequently, identical derivatives for several joints of the robot arm are obtained. Inversion of nearly singular matrices results in excessively large velocities and causes unrealistic behavior with oscillations around a singular configuration. There are many alternative approaches which address the inverse kinematics problem, such as Jacobian transpose methods [13], quasi-Newton and conjugate gradient methods [12], [14] or Levenberg-Marquardt damped least squares methods [11], [7].

3. Modeling and control of multi-DOF robot arm

3.1. Model of 7-DOF robot arm

The 7-DOF robot arm is composed of four rotational joints and five links with lengths l_1 to l_5 as depicted in Fig. 3. We consider seven stepper motors for rotation and translation of individual links. Joints two and four can rotate about two axes. All the others can rotate just about one axis. We introduce the following notation:

θ_i – joint variable (generalized coordinates), $i = 1, 2,\ldots,7$;

l_i – length of i-th link, $i = 1, 2,\ldots,5$.

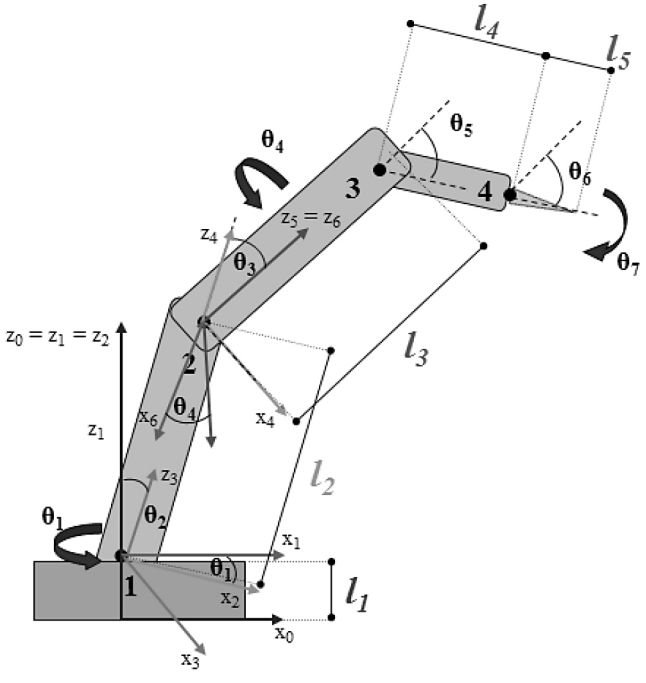

Figure 3. 7-DOF robot arm

3.2. Inverse kinematics analysis

The control task was to determine the joint parameters that provide a desired position of the end effector. This position is determined by the reference values of x, y, z coordinates and angles θ_6, θ_7, which determine the rotation settings of the end link. The end effector position was determined according to (7) in the following way:

$$EE = T_z(l_1)R_z(\theta_1)R_y(\theta_2)T_z(l_2)R_y(\theta_3)T_z(l_3)R_z(\theta_4)$$
$$R_y(\theta_5)T_z(l_4)R_y(\theta_6)R_x(\theta_7)\begin{bmatrix}0 & 0 & l_5 & 1\end{bmatrix}^T \tag{18}$$

where

$$T_z(l_1)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&l_1\\0&0&0&1\end{bmatrix},\qquad T_z(l_2)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&l_2\\0&0&0&1\end{bmatrix} \tag{19}$$

$$T_z(l_3)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&l_3\\0&0&0&1\end{bmatrix},\qquad T_z(l_4)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&l_4\\0&0&0&1\end{bmatrix} \tag{20}$$

$$R_z(\theta_1)=\begin{bmatrix}\cos(\theta_1)&-\sin(\theta_1)&0&0\\\sin(\theta_1)&\cos(\theta_1)&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix},\qquad R_z(\theta_4)=\begin{bmatrix}\cos(\theta_4)&-\sin(\theta_4)&0&0\\\sin(\theta_4)&\cos(\theta_4)&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix} \tag{21}$$

$$R_y(\theta_2)=\begin{bmatrix}\cos(\theta_2)&0&\sin(\theta_2)&0\\0&1&0&0\\-\sin(\theta_2)&0&\cos(\theta_2)&0\\0&0&0&1\end{bmatrix},\qquad R_y(\theta_3)=\begin{bmatrix}\cos(\theta_3)&0&\sin(\theta_3)&0\\0&1&0&0\\-\sin(\theta_3)&0&\cos(\theta_3)&0\\0&0&0&1\end{bmatrix} \tag{22}$$

$$R_y(\theta_5)=\begin{bmatrix}\cos(\theta_5)&0&\sin(\theta_5)&0\\0&1&0&0\\-\sin(\theta_5)&0&\cos(\theta_5)&0\\0&0&0&1\end{bmatrix},\qquad R_y(\theta_6)=\begin{bmatrix}\cos(\theta_6)&0&\sin(\theta_6)&0\\0&1&0&0\\-\sin(\theta_6)&0&\cos(\theta_6)&0\\0&0&0&1\end{bmatrix} \tag{23}$$

$$R_x(\theta_7)=\begin{bmatrix}1&0&0\\0&\cos(\theta_7)&-\sin(\theta_7)\\0&\sin(\theta_7)&\cos(\theta_7)\end{bmatrix}. \tag{24}$$

Since the rotation of the end link of the robot arm is determined by the reference trajectory, it stays in determined position until the reference values of angles θ_6, θ_7 will change. The calculation of joint parameters is realized according to Algorithm 3.1. Resulting Jacobian matrix has a size of 3 x 5.

Algorithm 3.1 Inverse kinematics

Inputs: length of links $l_1 - l_5$, reference coordinates x, y, z, reference angles θ_6, θ_7.
Outputs: joints parameters $\Delta\theta_i$.
Step 1: Determine the position of the end effector's according to (18) – (24)
Step 2: Determine the Jacobian matrix according to (15)
Step 3: **for** $k = 1$: n **do**
1: Evaluate Jacobian with actual parameters θ_i and the inverse Jacobian using pseudoinverse J^+ (17);
2: Obtain the vector $\Delta\theta$ according to (16);
3: Compute the joints parameters $\theta_i(k+1) = \Delta\theta_i + \theta_i(k)$;
4: Check the position error of the end effector according to reference trajectory point err = $x_{ref} - x_i$; if err satisfy certain condition, go to step 4, otherwise go back to step 3;
end for
Step 4: Update $\theta_i(k) \leftarrow \theta_i(k+1)$.

In this way, we obtained suitable joint rotations in order to achieve the desired position of the end effector.

3.3. Real-time reference trajectory generator

In the motion control we consider the trajectory of the end effector which is updated through the time, so these information can be stored in a table or in a text file. Consequently, they are sent to Matlab-Simulink software in real-time. In Matlab-Simulink we are able to compute exact joint rotation to achieve required reference trajectory by using the inverse kinematics. In this paper, we consider the trajectory data obtained from table stored in Microsoft Excel (MS Excel), where the so-called signal generator (SG) is developed. Thus, the signals for the end effector motion in x, y, z coordinates as well as the exact rotation of the end effector using the angles θ_6 and θ_7 (Fig. 3) are generated in real-time.

The communication between Microsoft Excel and Matlab-Simulink needed for data exchange is established by using DDE (Dynamic Data Exchange) communication developed by Microsoft. Thus, the goal of using Microsoft Excel is to generate analogue processing variables in real-time, which are used to generate trajectory of the end effector.

In order to generate different kinds of signals in real-time, it was needed to develop the automatic generator on-line. By using appropriate functions in MS Excel, it was possible to obtain the value of real system time. In order to update data sheet continuously, it was needed to create a macro using Visual Basic software, which would per-

form these updates in the shortest time interval. The macro can be run by using the start/stop button, as can be seen in Fig. 4 (only a part of the main window of the application is shown in Fig. 4).

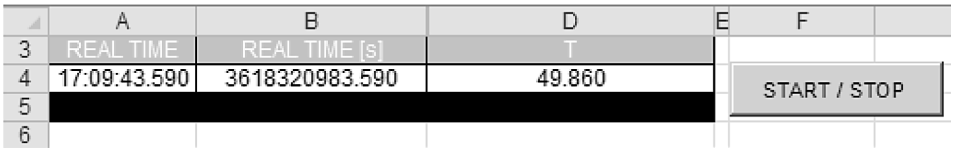

Figure 4. Real-time signal generator

The task of the macro is to run the procedure containing infinite update cycles in MS Excel data sheets. In Fig. 4 it can be also seen how the cells are generated in the real-time. In order to create required trajectory of the motion of the end effector, different kinds of mathematical functions and operations for the coordinates and angles were used and stored in other cells of the data sheet.

In this way, we obtained the vector of desired reference values in the following form:

$$r = \left(x, y, z, \theta_6, \theta_7\right) \tag{25}$$

The vector r is updated continuously in MS Excel (Fig. 4). The values of the vector r are consequently sent via DDE to Simulink software, where the inverse kinematics problem is solved according to (15)–(17) and the resulting joint parameters are obtained. From these parameters the position of each robot link are calculated according to (18) for simulation purposes. The calculated position of the end effector is sent for plotting as depicted in Fig. 5.

Obtained data can be used to control of the real robotic system by simple interconnection of Arduino platform and Matlab–Simulink software. In the next section, the simulation results of the motion of individual robot links are shown.

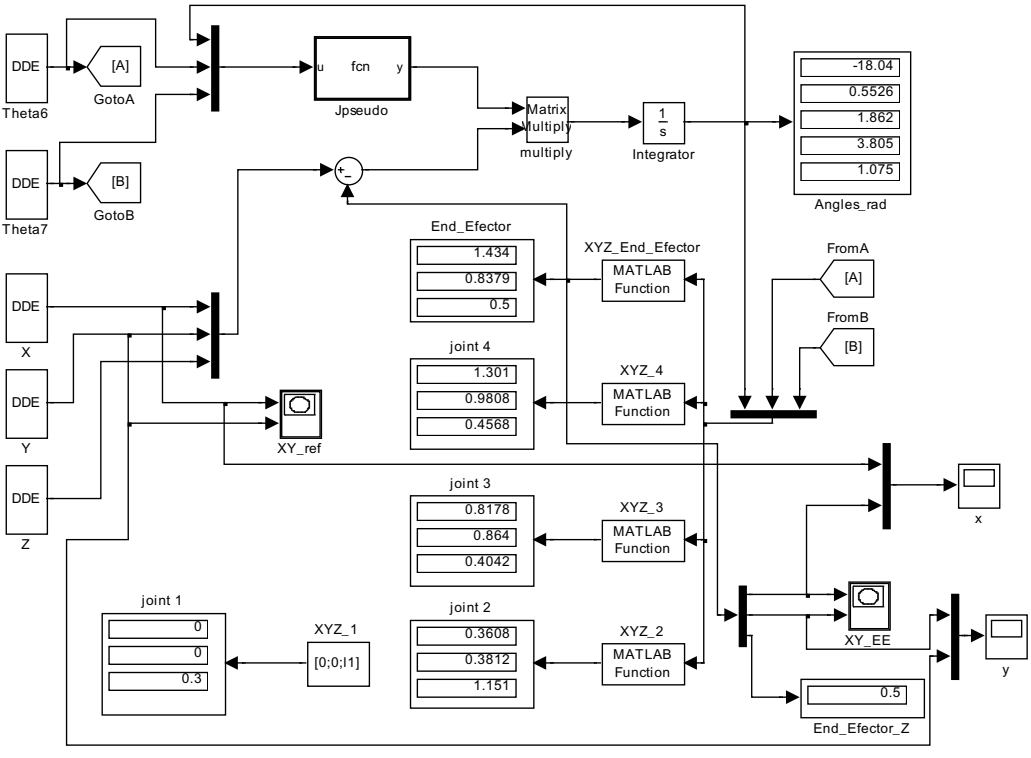

Figure 5. Real-time control scheme in Matlab-Simulink

4. Simulation results

The simulations were carried out in the Matlab–Simulink software. The link length parameters of the 7-DOF robot arm are as follows: $l_1 = 0.3$ m, $l_2 = 1$ m, $l_3 = 1$ m, $l_4 = 0.5$ m, $l_5 = 0.2$ m. The rotational range of motion of individual joints is as follows: $\theta_1 \in <0,\ 270°>$, $\theta_2 \in <-60°,\ 120°>$, $\theta_3 \in <-120°,\ 150°>$, $\theta_4 \in <-180°,\ 180°>$, $\theta_5 \in <-90°,\ 90°>$, $\theta_6 \in <-90°,\ 90°>$, $\theta_7 \in <-90°,\ 90°>$, where the values of the angles θ_6 and θ_7 are determined by the user via signal generator. According to these parameters we were able to evaluate and plot the workspace of the end effector in 3D space as depicted in Fig. 6.

The simulation results of the robot arm tracking the desired circle-shape trajectory generated in real-time via signal generator in MS Excel are presented in Figs 6–8. Each link is represented by a unique color.

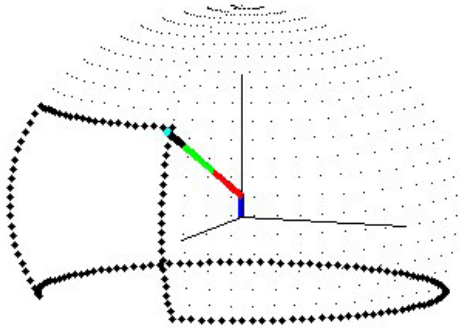

Figure 6. Workspace of the end effector in 3D space

The circle with the radius of 0.5 m is situated in the workspace of the end effector considering the midpoint coordinates: $(x, y, z) = (1, 1, 1)$. Simulated trajectories projected on the XY and XZ planes are shown in Fig. 7.

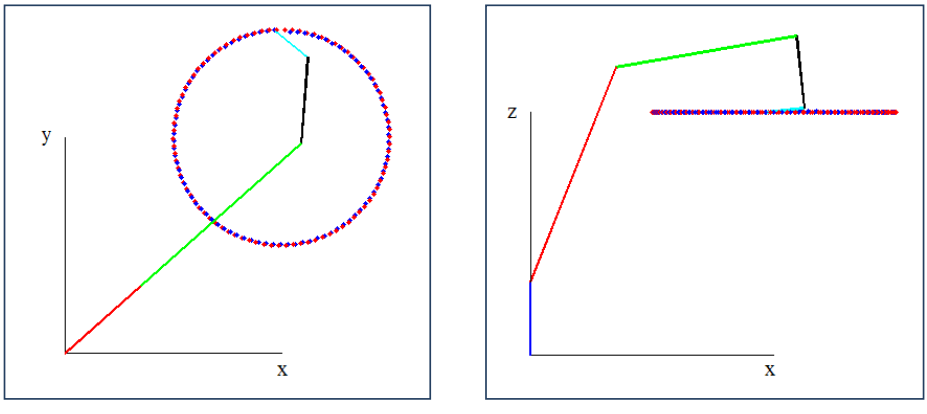

Figure 7. Trajectory of the end effector‘s motion (blue dots) and reference trajectory (red dots) in A) XY plane B) XZ plane

The robot arm′s configuration during the simulation is depicted in Fig. 8. For better illustration of the motion of each link, every sixth step of motion is considered.

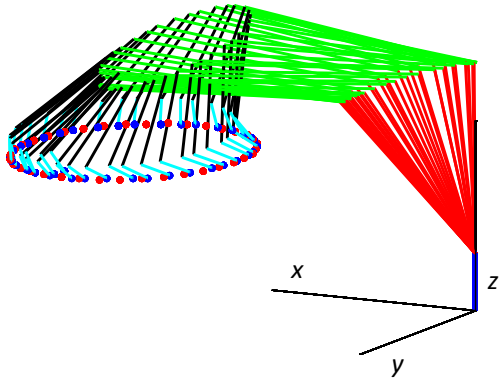

Figure 8. Robot arm configuration

The time responses of the end effector′s position in X and Y axis are shown in Fig. 9. These results verify the sufficient tracking of the reference trajectory by the end effector.

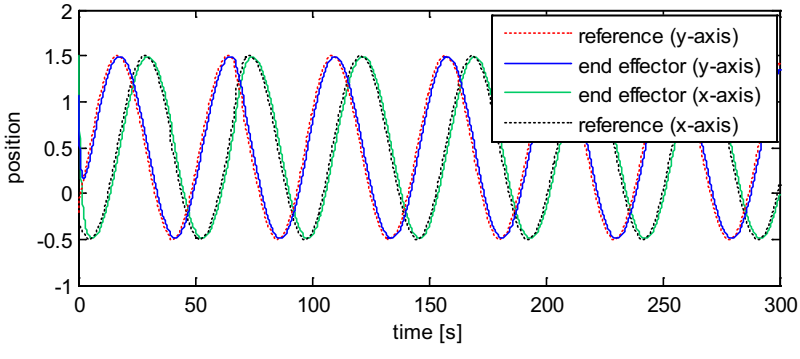

Figure 9. Time response of the end effector position in X, Y axes

As can be seen from the figures, the simulation experiments have shown desired results.

5. Conclusions

This paper presents a complete solution to the inverse kinematics of a 7-DOF robot arm. In order to obtain the solution, the iterative procedure based on calculating of pseudoinverse Jacobian was proposed. The procedure requires that the Jacobian is computed numerically. The main contribution of this work has been the development of the signal generator in order to generate the reference trajectory in real-time. This tool has been developed using macros in MS Excel and it enables modulation of the shape of the reference trajectory continuously during runtime. The simulation results prove the

efficiency of the proposed solution with the developed tool for the real-time reference trajectory tracking by the robot arm.

Notation and symbols

DOF	degree of freedom
DDE	dynamic data exchange
VBA	visual basic for applications
EE	end effector
SG	signal generator
k	time instant
R	rotation matrix
T	translation matrix
Θ	homogeneous transformation matrix
q	rotation angle between the link and the respective reference frame
l	link of the robot arm
J	Jacobian
J^+	pseudoinverse of Jacobian
J^T	transpose of Jacobian
x, y, z	coordinates in Cartesian coordinate system
r	desired reference value
Δ	time derivative

References

[1] J. ANGELES, F. RANJBARAN and R.V. PATEL: On the design of the kinematic structure of seven-axes redundant manipulators for maximum conditioning. *Proc. of the IEEE Int. Conf. Robotics and Automation*, Nice, France,(1992), 494-499.

[2] J.G.P. BARNES: An algorithm for solving non-linear equations based on the secont method. *Computer J.,* **8**, (1965), 66-72.

[3] J. DENAVIT and R.S. HARTENBERG: A kinematic notation for lower-pair mechanisms based on matrices. *Journal of Applied Mechanics,* (1955), 215-221.

[4] R.G. FENTON, B. BENHABIB and A.A. GOLDENBEREG: Optimal point-to-point motion control of robots with redundant degrees of freedom. *J. of Manufacturing Science and Engineering,* **108**(2), 1986, 120-126.

[5] R. FLETCHER: Generalized inverse methods for the best least squares solution of systems of non-linear equations: *Computer J.,* **10** (1968), 392-399.

[6] C.A. KLEIN and C.H. HUNG: Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Trans. on Systems, Man, and Cybernetics*, **13**, 245-250.

[7] Y. NAKAMURA and H. HANAFUSA: Inverse kinematics solutions with singularity robustness for robot manipulator control. *J. of Dynamic Systems, Measurement, and Control*, **108** (1986), 163-171.

[8] P. RABINOWITZ: Numerical Methods for Non-linear Algebraic Equations. New York: Gordon and Breach, 1970, 87-114.

[9] J. DE SCHUTTER, T. DE LAET and J. RUTGEERTS: Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *The Int. J. Robotics Research*, **26**(5), (2007), 433-455.

[10] B. SICILIANO: Kinematic control of redundant robot manipulators: A tutorial. *J. of Intelligent and Robotic Systems*, **3**(3), (1990), 201-212.

[11] C.W. WAMPLER: Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods. *IEEE Trans. on Systems*, *Man and Cybernetics*, **16** (1986), 93-101.

[12] L.C.T. WANG and C.C. CHEN: A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Trans. on Robotics and Automation*, **7** (1991), 489-499.

[13] W. A. WOLOVICHAND and H. ELLIOT: A computational technique for inverse kinematics. *23rd IEEE Conf. on Decision and Control*, (1984).

[14] J. ZHAO and N. I. BADLER: Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Trans. on Graphics*, **13** (1994), 313-336.