https://doi.org/10.26552/com.C.2014.3A.154-160

Juraj Uricek - Tibor Galbavy - Vladimir Bulej - Peter Durec *

THE CALCULATION OF INVERSE KINEMATICS FOR 6DOF SERIAL ROBOT

This article deals with the inverse kinematic model of robot devices which was implemented into the simulation software RoboSim designed at the Department of Automation and Production Systems – Faculty of Mechanical Engineering, University of Zilina. Concretely there is shown the technique how to develop the system of equations for serial kinematic mechanism with 6 degrees of freedom (DOFs). Further this software can be used as universal platform for simulation of mechanisms with serial, parallel as well as hybrid kinematic structure. This flexibility can be considered as a main advantage of the software RoboSim. Its specific feature is that there are used two different methods for calculation of inverse kinematics: heuristic and vector method as well. The versatility, openness and access to the source code create the predisposition for its deployment under the laboratory conditions for controlling of robotic devices developed in authors' workplace within the last few years.

Keywords: *Vector method, robot inverse kinematics, matrix equations.*

1. Introduction

The main aim of this article is off-line programming of robotic systems as well as their computer simulation. It contains the basic information about the development process of universal software for simulation of automated workplaces equipped with one or more robots. This software is developed during last few years in authors' workplace and is called RoboSim. The article is focused on one specific part of its development, concretely on calculation of kinematic model for specified type of robots.

Kinematics is the study of possible motion and configuration of a mechanism and it is related to the geometry of solved system. To understand how the system will move in given circumstance requires knowledge of speed, forces, torque, inertia, energy, etc. We need to know the position and orientation of the last linkage or the end-effector in terms of robot joint variables. This method is called forward kinematics [1]. The basic idea is to find matrices corresponding to the motion of each joint. We can use rotation and transformation matrices for this task. Combining these matrices in the correct order will give us the basic transformation equations for the final linkage as a function of the joint variables [2]. The position and orientation of any point rigidly attached to the gripper can be found if the joint angles are known. Inverse kinematics tells us how to do it in a backward way. The angles for each joint are calculated separately which must be set when the given position and orientation of the gripper want to be reached. This is one of the basic aims in robotics, since whenever we specify the motion of the robot's gripper we need to know the corresponding joint motion.

There are currently a number of programs used for simulation of robots and complete robotic workstations on the market. These programs are used for visualization, testing, debugging and repairing of programs for real robots. Debugging option is therefore very advantageous in pre-production stage because the standing time of robots or whole line can be reduced to a minimum. These programs also find their application in production planning and training of operators in the field of robotics.

Requirements for simulation programs are:

- simple operation,
- good quality results,
- low initial and total costs
- quality of interface (the ability to import / export existing data)
- correct selection of simulation tools,
- processing and evaluation of results,
- visualization and animation.

Simulation programs can be divided into two main categories - programs developed by manufacturers of industrial robots (ABB, Kuka, Fanuc, etc.) and systems developed by software producers.

* **Juraj Uricek, Tibor Galbavy, Vladimir Bulej, Peter Durec**
Department of Automation and Production Systems, Faculty of Mechanical Engineering, University of Zilina, Slovakia
Email: juraj.uricek@fstroj.utc.sk

A specific feature of the software RoboSim is that there are used two different methods for calculation of inverse kinematics: heuristic and vector method as well. Vector method is applicable to only one type of kinematic structure (in our case just the serial robot with six degrees of freedom and rotation constraints) but it is very fast and accurate. On the other hand, heuristic method allows us to find a solution for more degrees of freedom than the number of known parameters. This method is especially useful when the robot moved fast from point to point when there is not needed very high precision of orientation (just high precision of position).

The versatility, openness and access to the source code created the predisposition for its deployment under the laboratory conditions for controlling of robotic devices developed in authors' workplace within the last few years. Some of them are based on parallel or hybrid kinematic structure and the simulation software can be used for creating and testing of control programs for these machines as well. Designed simulation software is based on modular principle what enable to modify the software according to our application (modules can be changed).

2. Basic transform equations within the robot workspace

The task is to formulate the robot with transform equations. These equations arise when the manipulated object as well as the robot is described with respect to the same reference system. In Fig. 1 is shown the automated cell based on robot which is described by *WRD-matrix* (world) within the global reference system. The robot end link is described by *UCS-matrix* and *T6-matrix* as well, while the end-effector is described by *GRP-matrix* (gripper) relatively to the robot's end link. All matrixes could be defined as dynamics, but only *T6-matrix* is controlled by inverse kinematics. At the opposite side in Fig. 1 is shown a manipulated object described by relative transformation *CONV-matrix* (conveyor), *OBJ-matrix* (object) and *PICK-matrix* (place where it is possible to catch the object).

Then, for the task "grasping the object by robot" we can mathematically formulate this equation [robot gripper coordinate system must be identified to the gripping (welding, touching, picking) position of a moveable object]:

$$WRD * UCS * T6 * GRP = WRD * CONV * OBJ * PICK \quad (1)$$

Then for the robot position we can write general equation:

$$T6 = CONV * OBJ * PICK * UCS^{-1} * GRP^{-1} \quad (2)$$

This is the basic procedure how it is possible to find *T6-matrix* for selected robot. Using inverse kinematics we can find unknown joint variables of the robot arm.

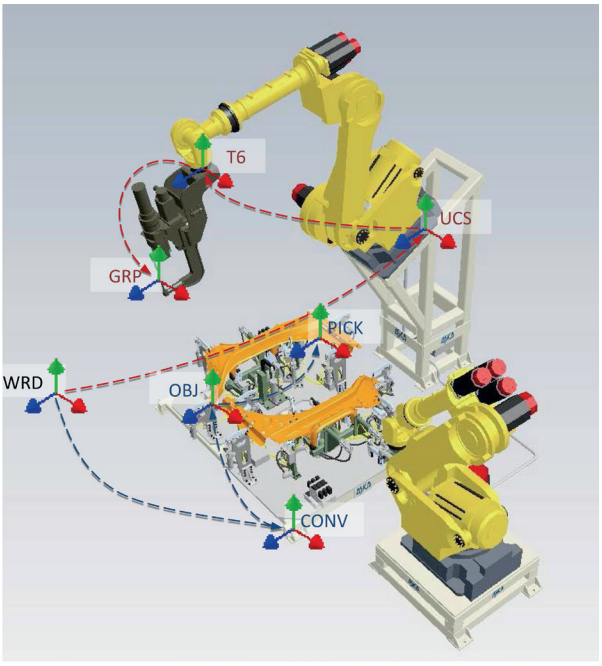


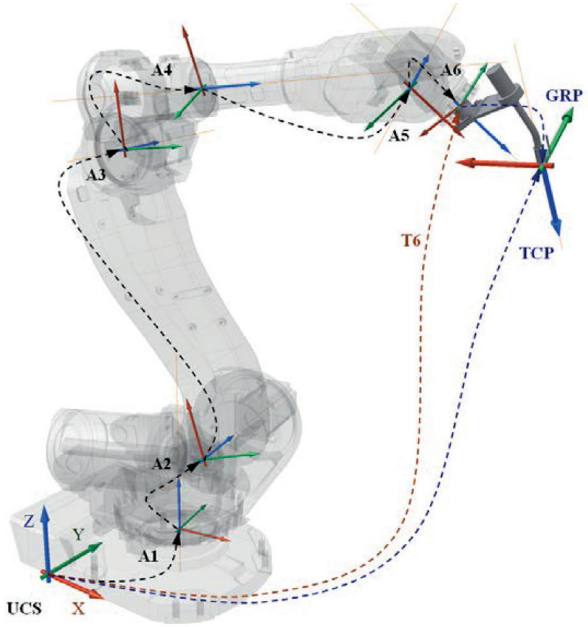
Fig. 1 Coordination system of working space and robots

3. General inverse kinematics calculation

Any robot can be taken as a group of links connected by joints. To each link of manipulator we can connect one coordinate system. Using homogeneous transformations, the relative position and orientation between the link coordinate matrixes can be described (Fig. 2). The *A-matrix* is a simple homogeneous transformation describing the relative displacement (translation and rotation) between the coordinate systems of two neighboring links. The *A1-matrix* describes the position and orientation (variable *U*) of the first link, *A2-matrix* the second link with respect to the first one. All *A-matrixes* can be composed of two other matrixes, *B-matrix* and *C-matrix*. The *B-matrix* represents translation and rotation to another link and the *C-matrix* represents only setup angle *U*. For a robot with six degrees of freedom, we can write:

$$T6 = A1 * A2 * A3 * A4 * A5 * A6 \quad (3)$$

$$A1=B1 * C1, \quad A2= B2 * C2 \quad \ldots \quad A6=B6 * C6 \quad (4)$$



UCS - *robot coordinate system, could be placed everywhere*
WRD - *basic world coordinate system (not shown)*
TCP - *position and orientation of end-effector relatively to UCS*
GRP - *position and orientation of end-effector relatively to T6*
WRS - *wrist center (orange axis)*

Fig. 2 Kinematic structure and coordinate system of serial links

Then, we can write:

$$WRD * UCS * TCP = WRD * UCS * T6 * GRP \quad (5)$$

And after modification and substitution:

$$A1 * A2 * A3 * A4 * A5 * A6 = TCP * GRP^{1} \quad (6)$$

The *T6-matrix* represents the desired position and orientation of the end-effector. The task is to find all values for joint variables inside *A-matrixes*. For standard manipulator it is supposed that the desired position and orientation of the final frame are given by:

$$T_6 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & O_x \\ r_{21} & r_{22} & r_{23} & O_y \\ r_{31} & r_{32} & r_{33} & O_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

For example, to find the corresponding joint variables *U1, U2, U3, U4, U5* and *U6* we must solve the following simultaneous set of nonlinear trigonometric equations:

$$r_{11} = c_1\left[c_2\left(c_4c_5c_6 - s_4s_6\right) - s_2s_5c_6\right] - s_1\left(s_4c_5c_6 + c_4s_6\right) \quad (8)$$

$$r_{21} = s_1\left[c_2\left(c_4c_5c_6 - s_4s_6\right) - s_2s_5c_6\right] - c_1\left(s_4c_5c_6 + c_4s_6\right) \quad (9)$$

$$r_{31} = -s_2\left(c_4c_5c_6 - s_4s_6\right) - c_2s_5s_6 \quad (10)$$

$$r_{12} = c_1\left[-c_2\left(c_4c_5c_6 + s_4s_6\right) + s_2s_5c_6\right] - s_1\left(-s_4c_5c_6 + c_4s_6\right) \quad (11)$$

$$r_{22} = s_1\left[-c_2\left(c_4c_5c_6 + s_4s_6\right) + s_2s_5c_6\right] - c_1\left(-s_4c_5c_6 + c_4s_6\right) \quad (12)$$

$$r_{32} = s_2\left(c_4c_5c_6 + s_4c_6\right) - c_2s_5s_6 \quad (13)$$

$$r_{13} = c_1\left(c_2c_4s_5 + s_2c_5\right) - s_1s_4s_5 \quad (14)$$

$$r_{23} = s_1\left(c_2c_4s_5 + s_2c_5\right) - c_1s_4s_5 \quad (15)$$

$$r_{13} = -s_2c_4s_5 + c_2c_5 \quad (16)$$

$$O_x = c_1s_2d_3 - s_1d_2 + d_6\left(c_1c_2c_4s_5 + c_1c_5s_2 - s_1s_4s_5\right) \quad (17)$$

$$O_y = s_1s_2d_3 - c_1d_2 + d_6\left(c_1s_4s_5 + c_2c_4s_1s_5 - c_5s_1s_2\right) \quad (18)$$

$$O_z = c_2d_3 + d_6\left(c_2c_5 - c_4s_2s_5\right) \quad (19)$$

These equations are too difficult to solve directly in a closed form. This is the case for most robot arms. Therefore, it is needed to develop efficient and systematic techniques that exploit the particular kinematic structure of the manipulator. Whereas the forward kinematics problem always has a unique solution that can be obtained simply by evaluating the forward equations, the inverse kinematics problem may or may not have a solution. Even if a solution exists, it may or may not be unique. Furthermore, because these forward kinematic equations are in general complicated nonlinear functions of the joint variables, the solutions may be difficult to obtain even when they exist.

Solving the inverse kinematics problem is most interesting in finding a closed form solution of the equations rather than a numerical solution.

Closed form solutions are preferable for two reasons:

- In certain applications, such as tracking a welding seam whose location is provided by a vision system, the inverse kinematic equations must be solved at a rapid rate, say every 20 milliseconds, and having closed form expressions rather than an iterative search is a practical necessity.

- The kinematic equations in general have multiple solutions. Having closed form solutions allows one to develop rules for choosing a particular solution among several.

The practical question whether the solution of inverse kinematics problem exists or not depends on engineering as well as mathematical considerations. For example, the motion of the revolute joints may be restricted to less than full 360 degrees of rotation so that not all mathematical solutions of the kinematic equations will correspond to physically realizable configurations of the manipulator.

4. Kinematics decomposition

The position and orientation of a robot's end-effector are derived from the joint positions by means of a geometric model of the robot arm. For serial robots, the mapping from joint positions to end-effector pose is easy whereas the inverse mapping is more difficult [3]. Therefore, most industrial robots have special designs that reduce the complexity of the inverse mapping. The most popular designs involve a spherical wrist.

Although the general problem of inverse kinematics is quite difficult, it turns out that for manipulators having six joints (with the last three joints intersecting at a point) it is possible to decouple the inverse kinematics into two separate and simpler problems (Fig. 3). They are known as **inverse position kinematics**, and **inverse orientation kinematics**. For a six-DOFs manipulator with a spherical wrist, the inverse kinematics problem may be separated into two simpler problems, namely first finding the position of the intersection of the wrist axes, called the wrist center, and then finding the orientation of the wrist (matrix WRS).

$$WRS = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \tag{20}$$

where P and R are the desired position and orientation of the tool frame, expressed with respect to the world coordinate system. For given P and R the inverse kinematics problem is to calculate parameters *U1-U6*. The important point of this assumption for the inverse kinematics is that the motion of the final three links about these axes will not change the position of WRS. The position of the wrist center is only a function of the first three joint variables (*U1, U2 and U3*). But orientation of the wrist center is a function of all joint variables (*U1-U6*). This is the reason why it is preferable to find the position of wrist as first.

Next important step is defining coding system for position and orientation of end-effector. Here is the big variability. It is possible to use: a coding system based by *Euler angles, rotation Roll Pitch Yaw, Cardan angles* or many others [4]. The best way for robot programing is to use a coding system based on rotating of robot wrist center around axis *Rz, Rx, Ry* (exactly

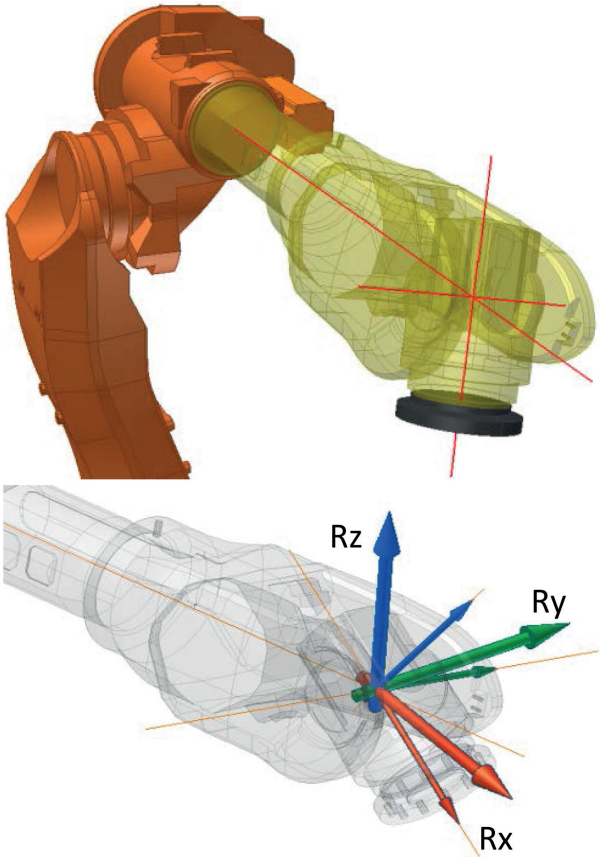


Fig. 3 The relative position of the wrist axis and coding system of wrist

in that order), where the direction of Z-axis is parallel with axis of the end-effector (direction of Y-axis goes left for right angle coding system). Finally, the coding system can be *X, Y, Z, Rz, Rx, Ry*, which is also user-friendly for robot programing. Coding system doesn't have any influence on calculation either for the controlling of robot. Coding system is important only for user comfort and for preventing errors and mistake.

5. General equations for robot links

The first task for joint variables computing is finding position and orientation for the center point of wrist, then:

$$WRS = TCP * GRP^{-1} * B6^{-1} \tag{21}$$

Matrix of wrist has a form:

$$WRS = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{22}$$

Vector *p* is determined by the position of the wrist. Three unit vectors *a, o, n* describe the orientation of the gripper as follows:

- *a* is the vector which defines the direction of closing up the wrist to the object
- *o* vector called orientation, determines the orientation of the hand
- *n* is the normal vector and is calculated as the vector product

Then, the position of wrist center is easy to find from *WRS* matrix:

$$P(p_x,p_y,p_z) \tag{23}$$

To find orientation it is good idea to put the position vector to zero because we know that the position doesn't have influence on orientation. Then:

$$WRS = R_z(U_z) \cdot R_y(U_y) \cdot R_x(U_x) \tag{24}$$

$$Rz\,(Uz)^{-1} \cdot WRS = Ry(Uy) \cdot Rx(Ux) \tag{25}$$

Having multiplied the matrix:

$$\begin{bmatrix} \cos Uz & -\sin Uz & 0 & 0 \\ \sin Uz & \cos Uz & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos Ux & -\sin Ux & 0 \\ 0 & \sin Ux & \cos Ux & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos Uy & 0 & \sin Uy & 0 \\ 0 & 1 & 0 & 0 \\ -\sin Uy & 0 & \cos Uy & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{26}$$

$$\begin{bmatrix} \cos Uz \cdot n_x + \sin Uz \cdot n_y & \cos Uz \cdot o_x + \sin Uz \cdot o_y & \cos Uz \cdot a_x + \sin Uz \cdot a_y & 0 \\ -\sin Uz \cdot n_x + \cos Uz \cdot n_y & -\sin Uz \cdot 0_x + \cos Uz \cdot o_y & -\sin Uz \cdot a_x + \cos Uz \cdot a_y & 0 \\ n_z & o_z & a_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos Uy & 0 & \sin Uy & 0 \\ \sin Ux \cdot \sin Uy & \cos Ux & -\sin Ux \cdot \cos y & 0 \\ -\cos Ux \cdot \sin Uy & \sin Ux & \cos Ux \cdot \cos Uy & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{27}$$

Result of equation from matrix:

$$Uz = arctg\frac{o_x}{o_y} \quad Ux = arctg\frac{o_z}{-\sin Uz \cdot o_x + \cos Uz \cdot n_y}$$
$$Uy = arctg\frac{\cos Uz \cdot a_x + \cos Uz \cdot a_y}{\cos Uz \cdot n_x + \cos Uz \cdot n_y} \tag{28}$$

It should be remembered that the calculated rotation angles are wrist angles (*WRS-matrix*) based on the original coordinate system (*UCS-matrix*). These angles will be needed later to calculate the real angles of the robot wrist *U4, U5, U6*. These angles are already affected by the position, not only the orientation of the wrist.

5.1 Inverse position

For the common kinematic arrangements, we can use a geometric approach to find the variables *U1, U2, U3* corresponding to the right center position of wrist. The complexity of the inverse kinematics increases with the number of nonzero link parameters in general. For most manipulators, many of the parameters of *B-matrix* and *C-matrix* are zero, or the angles included inside are 0 or ± π/2, etc. In these cases especially a geometric approach is the simplest and most natural way.

For calculation of *U1, U2, U3* it is easy to define this angle from the robot triangle, known links length and known center of wrist *P* (Fig. 4).

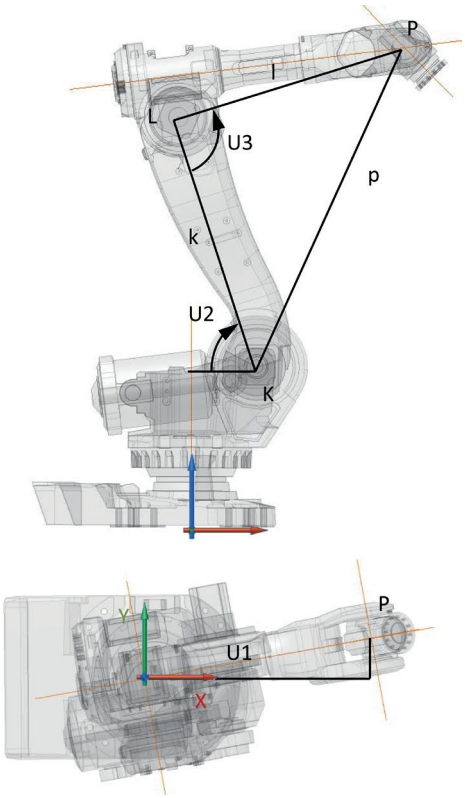


Fig. 4 Motion equation in a graphic form

The position of points is defined in the known matrix:

$$K[x,z] = B2\,[x,z] \tag{29}$$
$$P[x,z] = (WRS * A1^{-1})\,[x,z] \tag{30}$$

Then, it is possible to write for rectangle lengths:

$$k = B3z \tag{31}$$

$$l = \sqrt{B4x^2 + (B4z + B5z)^2} \tag{32}$$

$$p = \sqrt{(Px - Kx)^2 + (Pz - Kz)^2} \tag{33}$$

For angle U3 and U2 from cosine theorem are valid:

$$U3 = arcos\left(\frac{k^2 + l^2 - m^2}{2kl}\right) \tag{34}$$

$$U2 = 180 - \left(arcos\left(\frac{k^2 - l^2 + m^2}{2km}\right)\right) - \left(arcsin\frac{Pz - Kz}{m}\right) \tag{35}$$

And from top view for U1:

$$U1 = arcsin\frac{Py}{Px} \tag{36}$$

5.2 Inverse orientation

The inverse orientation problem is now one of finding the values of the final three joint variables *U4, U5, U6* corresponding to a given orientation with respect to the dimensions (all *B-matrixes*) and set-up of frame (join *U1, U2, U3*). For a spherical wrist, this can be interpreted as the problem of finding a set of *Rz, Rx, Ry* angles corresponding to a given rotation matrix *R*. Recall that equation shows that the rotation matrix obtained for the spherical wrist has the same form as the rotation matrix for the *Rz, Rx, Ry* transformation given in the previous chapter.

For last joint angles (*U4, U5* and *U6*) it is necessary to define differential matrix *DIF* like a difference between angles at the end of *T5-matrix* and requested set up angles *Rz, Rx, Ry*:

$DIF = WRS * T5^{-1}$

Also matrix *DIF* has a known form and it is possible to use the analogously mathematical principle like for finding angles from matrix *WRS*, then:

$$DIF = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_y & o_y & a_y & p_y \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{37}$$

And for finding angles *U4, U5, U6* are valid:

$$\begin{aligned} U4 &= arctg\frac{o_x}{o_y} \\ U5 &= arctg\frac{o_z}{-\sin U4 \cdot o_x + \cos U4 \cdot o_y} \\ U6 &= arctg\frac{\cos U4 \cdot a_x + \cos U4 \cdot a_y}{\cos U4 \cdot n_x + \cos U4 \cdot n_y} \end{aligned} \tag{38}$$

Now we have all unknown variable *U1-U6* like a function of TCP coordinate (*X, Y, Z, Rz, Rx, Ry*).

6. Conclusion

Industrial robots have a distinct set of capabilities that allows them to perform in industrial environments while also distinguishing them from other specialized robots. Manufacturing companies are seeking ways to ramp up production in a flexible way, as European economies start to recover from the recession [5]. One option is to use industrial robots, which are now more cost-effective both to purchase and to implement [6 and 7], thanks largely to programming software that is more engineer-friendly.

One of the basic tasks which need to be solved in robotics is so-called inverse kinematics. The vector method of inverse kinematics may not use Denavit-Hartenberg coordinates placement rule for links [8] but it is advantageous in terms of automated building transformation equations. This method is used most frequently for real time continuous path control due to easy programming as well as very fast calculation of results. It is, however, necessary to treat certain position programmatically links of mechanism when triangles and associated trigonometric equation, generate calculation errors.

Nowadays the investigation of system properties based on activity simulation in its virtual model belongs to the commonly used scientific methods for solving any technical problems. Simulation has substantial economic gains. Its usage can reduce pre-production stages. Thanks to that it can help us to examine several alternative solutions, find the optimal one as well as to avoid possible collisions.

The main aim of this article is original and special software *RoboSim* developed at authors' workplace which enables the simulation of automated manufacturing systems equipped with one or more robots. There is described more detailed one specific part of its development process – design of the mathematical model used for motion simulation of robots. Output equations are used for calculation of all joint coordinates which are necessary for motion control of robot devices. They make it possible to carry out collision detection between the robot and its environment in real time as well. Proposed software is based on a modular principle, which allows the system continuously to innovate and extend according to our requirements. The versatility, openness and access to the source code created the predisposition for its deployment under the laboratory conditions. The software development will continue in next period.

References

[1] SKARUPA, J., MOSTYN, B.: *Theory of Industrial Robots (in Czech)*. Vienala : Kosice, 2000, ISBN 80-88922-35-6, p. 146

[2] LUNG-WEN, T: Robot Analysis: *The Mechanics of Serial and Parallel Manipulators*. Interscience, 1999, 520 p., ISBN 04-7132-593-7.

[3] POPPEOVA, V., et al: *Design of Anti-collision System for Robotics.* Intern. Proc. of Computer Science and Information Technology: Mechanical Engineering, Robotics and Aerospace - ICMERA 2011, Singapore: IACSIT Press, editor: Jiang Xiaolan, vol. 4, 298-302, ISSN 2010-460X.

[4] BULEJ, V., POPPEOVA, V., et al: *Actual State of Development in Field of Parallel Kinematic Structures and Mobile Robots at the Department of Automation and Production Systems in Zilina (in Slovak)*. Technological forum 2011, 93-101, Praha, ISBN 978-80-01-04852-8.

[5] BULEJ, V., STOIANOVICI, G.-V., POPPEOVA, V.: *Material Flow Improvement in Automated Assembly Lines Using Lean Logistics.* Annals of DAAAM for 2011 & Proc. of 22nd intern. DAAAM Symposium, Vienna : DAAAM International, 2011, editor: B. Katalinic, vol. 22, No. 1, 253-254, ISSN 1726-9679.

[6] KURIC, I.: New Methods and Trends in Product Development and Process Planning. *Academic J. of Manufacturing Engineering. Editura Politehnica: Scientific Papers,* vol. 9, No. 1, 2011, Cluj-Napoca, 83-88. ISSN 1583-7904

[7] SAGA M., HANDRIK M., KOPAS P.: Contribution to Computer Simulation of Induction Bending of Large Diameter Pipes. *Metalurgija (Metallurgy),* vol. 49, No. 2, 2010, 498-502, ISSN 0543-5846.

[8] SAPIETOVA, A., SAGA, M., NOVAK, P.: Multi-software Platform for Solving of Multibody Systems Synthesis. *Communications - Scientific Letters of the University of Zilina,* vol. 14, No. 3, 2012, Zilina, pp. 43-48, ISSN 1335-4205.