Society of Production
Engineering

SPMS 2023
39. Savetovanje proizvodnog mašinstva Srbije
ICPES 2023
39th International Conference on Production Engineering of Serbia

Faculty of Technical
Sciences
University of Novi Sad

Novi Sad, Serbia, 26. – 27. October 2023

DEVELOPMENT OF THE DELTA ROBOT SIMULATION SYSTEM

Bogdan MOMCILOVIC[1, *], **Nikola SLAVKOVIC**[1]
[1]University of Belgrade, Faculty of Mechanical Engineering, Belgrade, Serbia
*Corresponding author: bmomcilovic@mas.bg.ac.rs

Abstract: *In the last few decades, alongside the development of serial industrial robots, parallel robots have attracted the attention of many industries and researchers. The DELTA robot is one of the most famous parallel kinematic robots. This paper presents the DELTA robot's complete kinematic modelling and simulation system development. The developed kinematic model includes the solution of the inverse and direct kinematic problem and the determination of the Jacobian matrix. Determining the robot's kinematic parameters in the iterative procedure enabled the analysis of the workspace and singular configurations. The kinematic model and both, the direct and inverse kinematic problems are included in the simulation model to realize the motion of the virtual (wireframe) robot. The virtual robot is developed in a MatLab environment. Using the direct kinematic problem, the positions of all actuated and non-actuated joints are calculated. The simulation system, besides others, includes two developed functions for G-code interpretation and Cartesian space linear interpolation. The developed simulation model can provide information about possible collisions of robot elements. In addition, the calculated joint coordinate vector provides information on whether all movements of the robot's end-effector can be executed according to the joint limits and the given G-code program. Verification of the developed robot simulation system and kinematic model was performed through several examples of the end-effector movement according to the program generated in a CAD/CAM environment.*

Keywords: *industrial robot, DELTA, kinematics, virtual model, simulations*

1. INTRODUCTION

In the last few decades, the development of robots with parallel kinematics has attracted the attention of many industries and researchers. This is because of their higher precision, rigidity, dynamic performance, and loading than the serial robots. To reduce or eliminate the disadvantages of serial kinematics machine tools or robots, parallel kinematic mechanisms are developed and produced by companies and used in many applications [1].

One of the most famous robots with parallel kinematics is the DELTA robot. The DELTA robot is usually a 3 DoF translational manipulator that consists of a fixed base linked to a mobile platform by three arms. The first model of the DELTA robot was invented in 1987 by Reymond Clavel as a suitable structure for high-speed and high-acceleration tasks. It is used for pick and place operations, packaging, sorting, precision positioning, and other applications [2, 3]. The parallel mechanism of the DELTA robot was the base of development for many other robots with parallel kinematics.

This paper presents the DELTA robot's simulation system development in the MatLab environment. The main advantages of using simulation systems (virtual environments) of machine tools and robots in programming tasks are (1) the ability to check the movement of the tool along the programmed path, taking into account limitations in joint ranges (joint coordinate) and axis movement speed, (2) visual detection of collisions between segments, as well as tools with workpiece and fixture, and (3) checking whether the workpiece is correctly positioned within the workspace. The existing robot off-line simulation and programming software includes almost all robotic arms with serial and parallel kinematics on the market. Using such software, it could be easy to implement an available robot structure in a simulation environment. To resolve the issues for low-cost robots, i.e., simulation and programming of a new laboratory prototype of robots, many researchers developed a system that integrated kinematics and motion control simulation using the MatLab/Simulink environments [4 - 6].

The simulation system presented in this paper is developed according to the complete robot kinematics model. The direct and inverse kinematic problems are included in the simulation model to realize the motion of the virtual (wireframe) robot. Direct kinematics allows the calculation of the positions of all actuated and non-actuated joints. The simulation system also includes two developed functions for G-code interpretation and Cartesian space linear interpolation. Verification of the developed robot simulation system and kinematic model was performed through several examples of the end-effector movement according to the program generated in a CAD/CAM environment.

2. KINEMATIC MODELING OF DELTA ROBOT

To realize the virtual robot, this Section presents the well-known DELTA robot kinematic modelling based on a minimal number of parameters [7]. The kinematic model of the DELTA robot with rotary actuated joints is shown in Figure 1. Joint parallelograms of the mechanism are represented as a unique rod.

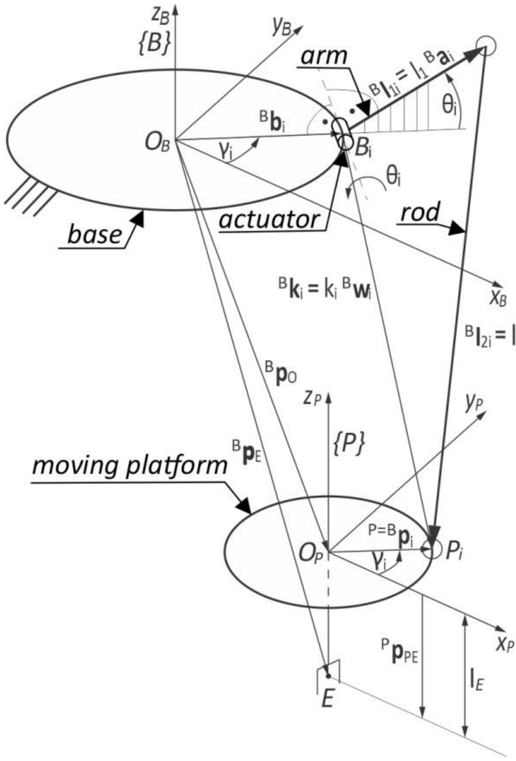


Figure 1. Kinematic model of the DELTA robot

The previously shown kinematic model can be described with the following parameters: radius of the base *R,* radius of the moving platform *r*, length of the arm l_1, length of the rod l_2, and length of the end-effector l_E.

To derive the DELTA robot's kinematic equations, some vectors are necessary. Position vectors, defined in frame {P}, of spherical joint centres (midpoints P_i) located on the moving platform on the circle of radius *r* are:

$$ {}^P\mathbf{p}_i = [r \cdot c\gamma_i \quad r \cdot s\gamma_i \quad 0]^T, i = 1,2,3 \qquad (1) $$

where joints angular positions are defined by $\gamma_i = 2\pi \cdot (i-1)/3$.

Position vector of the end-effector tip defined in coordinate frame {P} is:

$$ {}^{P}\mathbf{p}_{PE} = [0 \quad 0 \quad -l_E]^T \tag{2} $$

Position vectors of midpoints of rotary actuated joints B_{i}, defined in coordinate frame {B} are:

$$ {}^{B}\mathbf{b}_{\mathrm{i}} = [R \cdot \mathrm{c}\gamma_{\mathrm{i}} \quad R \cdot \mathrm{s}\gamma_{\mathrm{i}} \quad 0]^T, \mathrm{i} = 1,2,3 \tag{3} $$

World coordinate vector **x** that represents only the position of the moving platform in the base frame {B} and which will be further considered is:

$$ \mathbf{x} = {}^{B}\mathbf{p}_{OP} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} x_E \\ y_E \\ z_E - l_E \end{bmatrix} \tag{4} $$

The joint coordinate vector for the considered 3 DoF DELTA robot model is defined by:

$$ \mathbf{q} = \left[\theta_1 \quad \theta_2 \quad \theta_3\right]^T \tag{5} $$

Unit vectors ${}^{B}\mathbf{a}_{\mathrm{i}}$ that define vectors ${}^{B}\mathbf{l}_{1\mathrm{i}}$ as ${}^{B}\mathbf{l}_{1\mathrm{i}} = l_{1\mathrm{i}} \cdot {}^{B}\mathbf{a}_{\mathrm{i}}$ are consisted of joint coordinates and defined by:

$$ {}^{B}\mathbf{a}_{\mathrm{i}} = [\mathrm{c}\gamma_{\mathrm{i}} \cdot \mathrm{c}\theta_{\mathrm{i}} \quad \mathrm{s}\gamma_{\mathrm{i}} \cdot \mathrm{c}\theta_{\mathrm{i}} \quad \mathrm{s}\theta_{\mathrm{i}}]^T, \mathrm{i} = 1,2,3 \tag{6} $$

Other vectors and parameters are defined as shown in Figure 1.

Based on the relations shown in Figure 1, the following equations can be derived:

$$ k_{\mathrm{i}} \cdot {}^{B}\mathbf{w}_{\mathrm{i}} = {}^{B}\mathbf{p}_{OP} + {}^{B}\mathbf{p}_{\mathrm{i}} - {}^{B}\mathbf{b}_{\mathrm{i}} \tag{7} $$

$$ k_{\mathrm{i}} \cdot {}^{B}\mathbf{w}_{\mathrm{i}} = l_1 \cdot {}^{B}\mathbf{a}_{\mathrm{i}} + l_2 \cdot {}^{B}\mathbf{z}_{\mathrm{i}} \tag{8} $$

By taking the square of both sides of equation (8), equation (9) is derived:

$$ l_2^2 = k_{\mathrm{i}}^2 - 2 \cdot l_1 \cdot ({}^{B}\mathbf{a}_{\mathrm{i}} \cdot k_{\mathrm{i}} \cdot {}^{B}\mathbf{w}_{\mathrm{i}}) + l_1^2 \tag{9} $$

Using equations (1), (3), and (4) vectors $k_{\mathrm{i}} \cdot {}^{B}\mathbf{w}_{\mathrm{i}}$ can be obtained as:

$$ k_{\mathrm{i}} \cdot {}^{B}\mathbf{w}_{\mathrm{i}} = \begin{bmatrix} k_{wx\mathrm{i}} \\ k_{wy\mathrm{i}} \\ k_{wz\mathrm{i}} \end{bmatrix} = \begin{bmatrix} p_x - a \cdot \mathrm{c}\gamma_{\mathrm{i}} \\ p_y - a \cdot \mathrm{s}\gamma_{\mathrm{i}} \\ p_z \end{bmatrix} \tag{10} $$

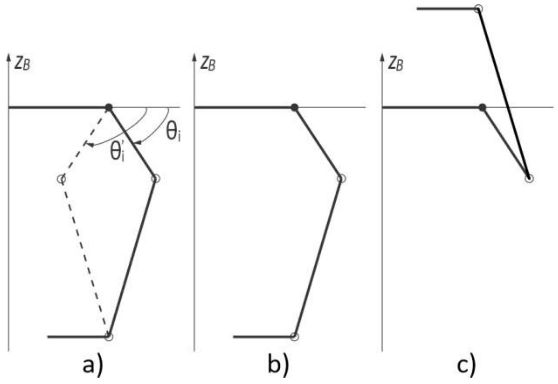
where $a = R - r$. Now, from equation (9) inverse and direct kinematics for the DELTA robot can be solved.

2.1 Inverse and direct kinematics

By substituting expressions (6) and (10) into equation (9) the well-known type of trigonometric equation can be obtained as:

$$ \mathrm{c}\theta_{\mathrm{i}} \cdot (\mathrm{c}\gamma_{\mathrm{i}} \cdot k_{wx\mathrm{i}} + \mathrm{s}\gamma_{\mathrm{i}} \cdot k_{wy\mathrm{i}}) + \mathrm{s}\theta_{\mathrm{i}} \cdot k_{wz\mathrm{i}} = \frac{l_1^2 - l_2^2 + k_{\mathrm{i}}^2}{2 \cdot l_1}, \mathrm{i} = 1,2,3 \tag{11} $$

From equation (11) joint coordinates θ_1, θ_2, and θ_3 are obtained and inverse kinematics is solved. There are two solutions of inverse kinematics, but one of the solutions has to be chosen because of singularities, Figure 2a.



a) b) c)

Figure 2. Solutions for inverse and direct kinematics

For direct kinematics, world coordinates p_x, p_y, and p_z can be obtained from equations (11). Substituting the corresponding vectors in equations (11), the system of three equations is derived to solve the direct kinematic problem. Two solutions of direct kinematics are presented in Figures 2b and 2c, but only one is physically possible, Figure 2b.

2.2 Jacobian matrix

The workspace analysis and analysis of the singularities needs the Jacobian matrix. The Jacobian matrix is obtained as a derivative of implicit joint and world coordinate function $f(\boldsymbol{x},\boldsymbol{q}) = 0$ concerning time:

$$J_x \cdot \dot{\mathbf{x}} = J_q \cdot \dot{\mathbf{q}} \tag{12}$$

where $J_x = \frac{\partial(\mathbf{x},\mathbf{q})}{\partial \mathbf{x}}$ is the Jacobian matrix of direct kinematics, and $J_q = -\frac{\partial(\mathbf{x},\mathbf{q})}{\partial \mathbf{q}}$ is the Jacobian matrix of inverse kinematics. The Jacobian matrix of DELTA mechanism is derived by:

$$J = J_q^{-1} \cdot J_x \tag{13}$$

The Jacobian matrix of parallel mechanisms maps the velocities of joints with the end-effector velocities.

2.3 Workspace analysis

Using the solved direct kinematics equations and joint limits, the workspace of the DELTA robot can be obtained. The position of the rotary actuated joints is divided within the range of limits with defined increments. According to joint incremental value combinations, end-effector positions are calculated using direct kinematics. Figure 3 shows the DELTA robot's obtained workspace by this method.

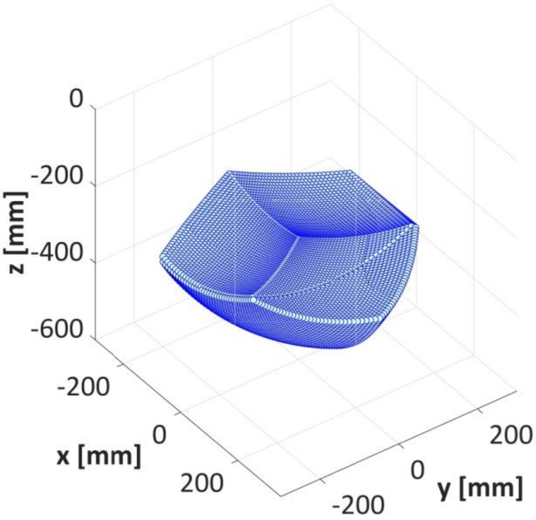


Figure 3. DELTA robot's workspace

Within the obtained workspace, workspaces with regular geometric shapes are selected, Figure 4. This is useful for programmers and operators of the robot.

Using a described method for workspace determination, the robot parameters are adopted in an iterative procedure according to the desired workspace dimensions.

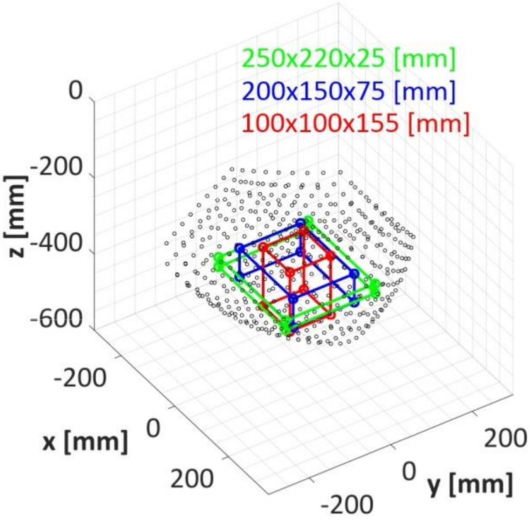


Figure 4. Selected portion of workspace

Adopted parameters of the considered DELTA mechanism are: radius of the base and moving platform R = 100 mm and r = 40 mm, respectively, arm length l_1 = 175 mm, rod (joint parallelogram) length l_2 = 475 mm, and length of the end-effector l_E = 100 mm.

Also, for the selected portion of the workspace, the distribution of determinants of the Jacobian matrix is calculated, Figure 5.

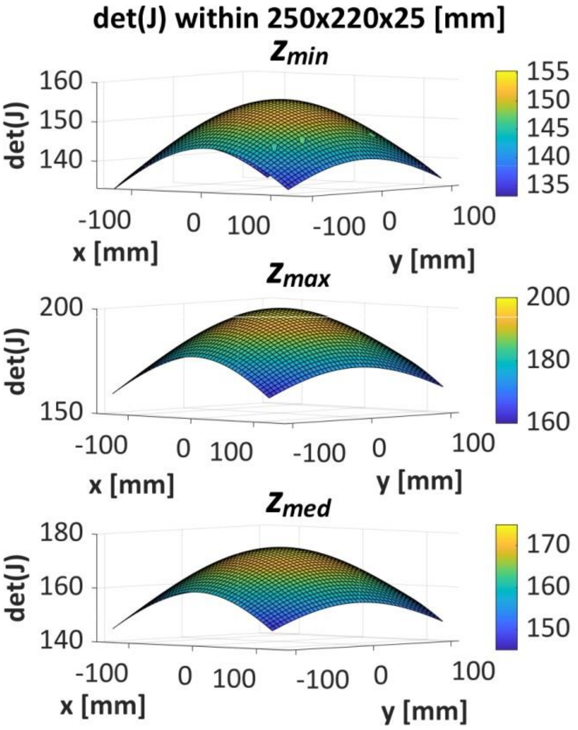


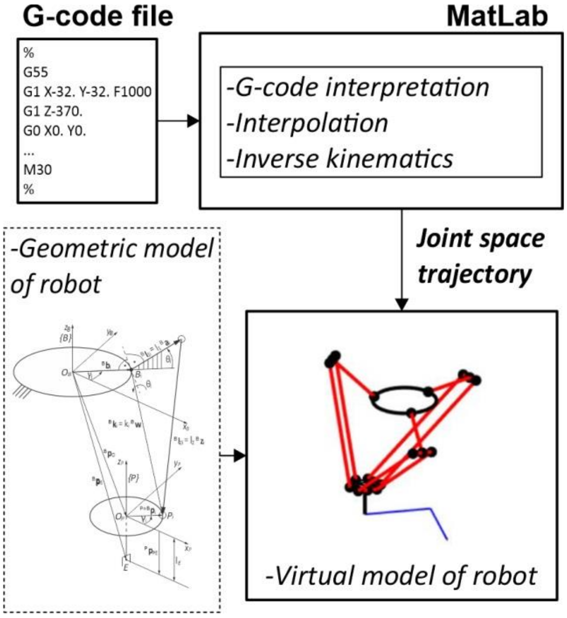
Figure 5. Distribution of determinants of the Jacobian matrix

The selected workspace's *x* and *y* coordinates are variated within the limits, while *z* coordinates have three values (minimum, medium, and maximum). By using that combination of the end-effector's positions, joint coordinates are calculated with the solved inverse kinematics. With known joint and world coordinates, the determinant of the Jacobian matrix can be obtained. Determinants of the Jacobian matrix for one of the selected workspaces are shown in Figure 5. Similar results are obtained for the other portions of the workspaces.

According to Figure 5, the values of the determinants aren't equal to 0 or infinity, and there are no singular positions within the selected workspaces.

3. SIMULATION SYSTEM

The developed DELTA robot simulation system consists of two modules: (1) the module for generating joint space trajectory based on the given G-code, and (2) the configured virtual (wireframe) robot in the MatLab environment, Figure 6.



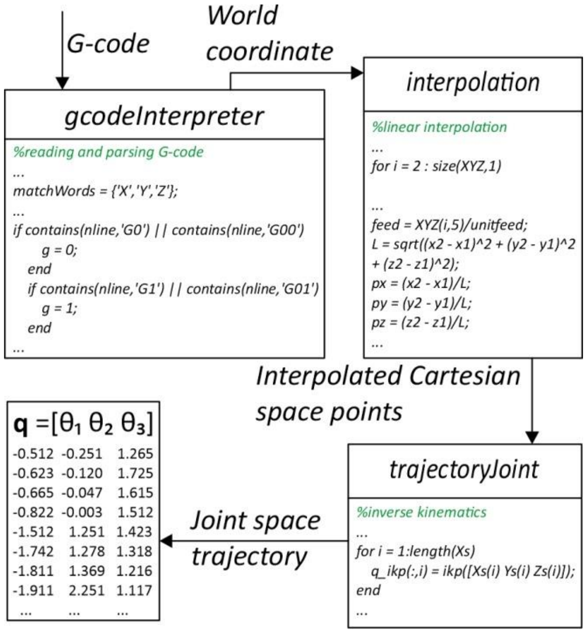
```
%
G55
G1 X-32. Y-32. F1000
G1 Z-370.
G0 X0. Y0.
...
M30
%
```

Figure 6. Developed simulation system

The first module consists of developed functions in MatLab for joint space trajectory generation based on the solved inverse kinematics of the DELTA robot. World coordinate vectors for the inverse kinematic problem are obtained from the G-code program. The second module includes the configured virtual DELTA robot using its geometric model. Based on a joint coordinate vector, parameters of the mechanism, and the solved direct kinematic problem, the positions of all actuated and non-actuated joints can be calculated.

3.1 Joint space trajectory generation module

This module is developed to simulate the various tasks on a virtual robot, such as picking and placing objects or laser engraving. The input for this module is a previously generated G-code program with limitations referring to only linear programmed movements of the end-effector. The module consists of three parts: (1) G-code interpreter, (2) interpolation, and (3) generating joint space trajectory, Figure 7. All of these three parts are realized as functions in the MatLab environment.



```
%reading and parsing G-code
...
matchWords = {'X','Y','Z'};
...
if contains(nline,'G0') || contains(nline,'G00')
     g = 0;
  end
  if contains(nline,'G1') || contains(nline,'G01')
     g = 1;
  end
...
```

```
%linear interpolation
...
for i = 2 : size(XYZ,1)


...
feed = XYZ(i,5)/unitfeed;
L = sqrt((x2 - x1)^2 + (y2 - y1)^2
+ (z2 - z1)^2);
px = (x2 - x1)/L;
py = (y2 - y1)/L;
pz = (z2 - z1)/L;
...
```

```
%inverse kinematics
...
for i = 1:length(Xs)
   q_ikp(:,i) = ikp([Xs(i) Ys(i) Zs(i)]);
end
...
```

```
-0.512 -0.251  1.265
-0.623 -0.120  1.725
-0.665 -0.047  1.615
-0.822 -0.003  1.512
-1.512  1.251  1.423
-1.742  1.278  1.318
-1.811  1.369  1.216
-1.911  2.251  1.117
  ...    ...    ...
```

Figure 7. The joint space trajectory generation

The first part of the module reads and parses one by one line of the given G-code program. The output of this part defines the segments of the programmed trajectory and the feed rate of the end-effector on those segments.

Then, the second part divides all trajectory segments using the rule of linear interpolation in Cartesian space. The output is a series of positions the end-effector must pass to achieve its programmed trajectory. The interpolation rule is programmed according to trajectory segment length and programmed speed.

After obtaining the series of end-effector positions, the third part of the module generates a joint space trajectory using the inverse kinematic solution.

3.2 Configuring the virtual robot

The DELTA robot's geometric model, the mechanism's parameters, and the obtained joint space trajectory are inputs for configuring the virtual DELTA robot, Figure 8. First, the positions of all actuated and non-actuated joints have to be calculated on the robot's desired joint space trajectory. For those calculations besides the joint coordinate vector, the world coordinate vector has to be calculated using the direct kinematic solution.

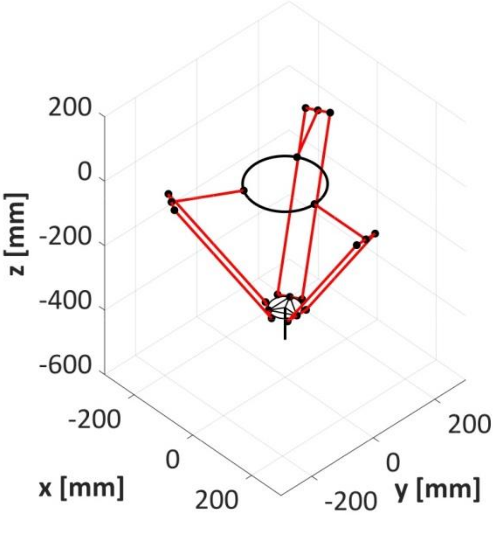


Figure 8. Virtual DELTA robot

Actuated joints on the mechanism's base are located on the circle of radius *R*, where angular positions are defined by $\gamma_i = 2\pi \cdot (i-1)/3$. Positions of the spherical joint centres on the mechanism's moving platform are calculated by using the geometric relations and obtained world coordinate vector. Positions of the joint parallelograms' spherical joint centres are calculated using the joint coordinate vector (the connection between arms and parallelograms) and the world coordinate vector (the connection between parallelograms and the moving platform).

All of the joints are represented as points with the known coordinates. It is possible to connect them, resulting in the configured virtual DELTA robot.

4. SYSTEM VERIFICATION

The developed DELTA robot simulation system has been verified through several examples of performing the technological task of laser engraving contours that are composed of linear parts, Figure 9. Testing programs were previously prepared in a CAD/CAM environment.

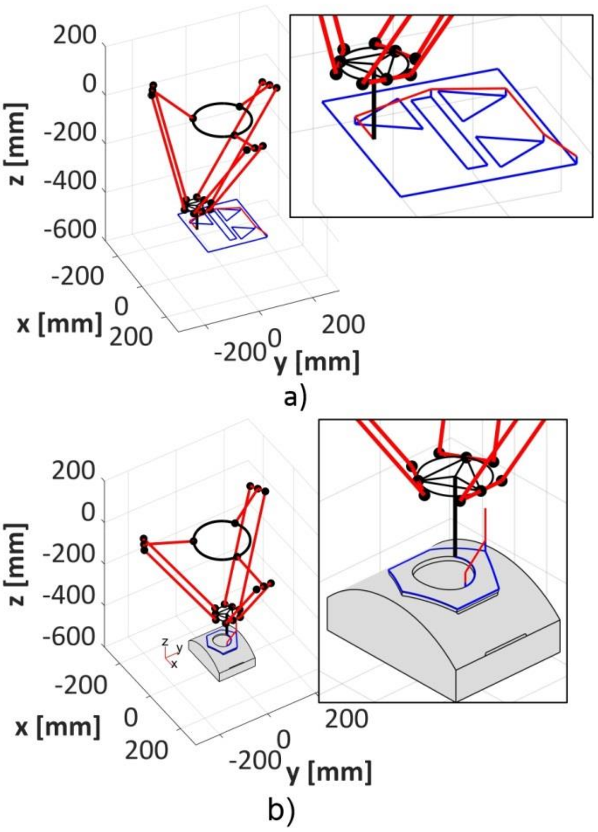


Figure 9. The examples of the system verification

One of the examples, Figure 9a, is the 2-axis laser engraving simple contours. The simulation system draws the trajectory of the end-effector tip (the blue colour is for the feed rate and the red colour is for the rapid move). In the simulation, the end-effector tip passes through all the positions that are obtained by the joint space trajectory generator. During the simulation, it can be visually checked if some of the robot elements are close to each other or if there’s a possibility of collision. After the simulation is done, all the trajectory positions can be checked with the pointer.

The second example is a 3-axis laser engraving complex contour on a calotte, Figure 9b. This example shows the simulation system’s possibility to work with 3-axis programs. Also, a CAD model of the workpiece is imported into the simulation. In this case, the end-effector tip has to follow a contour on the imported CAD model.

After completing the graphic view of the simulations, calculations are checked such as values in the actuated joints. Also, the inverse and direct kinematics solutions are checked through the simulation. Based on the obtained results, it can be concluded that the configured virtual robot works correctly according to the generated G-code program.

Such the developed virtual robot can be used for checking the inverse and direct kinematics equations before their implementation in some of the open architecture control systems such as LinuxCNC [8].

5. CONCLUSION

This paper presents an approach for developing the simulation system of a 3-axis DELTA robot in the MatLab environment. The developed simulation system can check programs for various robots’ tasks.

The simulation system includes solved inverse and direct kinematics as the real robot does. The system can be used for checking the inverse and direct kinematics equations before their implementation in some of the open architecture control systems. For the adopted mechanisms’ parameters, the workspace of the robot is checked with determinants of the Jacobian matrix and proved that there are no singular positions in the selected workspaces. Developed module for the joint space trajectory generation and its parts can be easily reconfigured to work with machines with different kinematics. The developed simulation system has been verified through several examples of performing the technological task of laser engraving, but it also can be used for 2-axis or 3-axis milling, 3D printing, etc.

The main advantage of using this type of virtual robot, i.e., the presented procedure for configuring such a virtual robot, is to develop a virtual robot for own developed low-cost robots and simulation and programming of a new laboratory prototype of robots with specific kinematics. The further research direction will cover the kinematic modelling of the 5-axis DELTA robot with hybrid kinematics and the development and implementation of the MatLab function for circular interpolation in the simulation system. This will enable the development of a complete methodology in the MatLab environment for the virtual robot laboratory prototype system realization with only changes in kinematics equations.

ACKNOWLEDGEMENT

This work has been financially supported by the Ministry of Education, Science and Technological Development of the Serbian Government through the project Integrated research in macro, micro, and nano mechanical engineering (contract No. 451-03-47/2023-01/200105).

REFERENCES

[1] M. Ratiu, D. M. Anton: A brief overview of parallel robots and parallel kinematic machines, in: *IOP Conference Series: Materials Science and Engineering*, Vol. 898, No. 1, pp. 012007, IOP Publishing, 2020.

[2] N. Cretescu, M. Neagoe, R. Saulescu: Dynamic Analysis of a Delta Parallel Robot with Flexible Links and Joint Clearances, Applied Sciences, Vol. 13, No. 11, pp. 6693, 2023.

[3] R. Clavel: Dispositif Pour le Déplacement et le Positionnement d’un Elément dans L’espace, Google Patents WO1987003528A1, 18 June 1987.

[4] H.M. Alwan, R.A. Sarhan: Kinematics Simulation of Gough-Stewart Parallel Manipulator by Using Simulink Package in MatLab Software, Journal of University of

Babylon for Engineering Sciences, Vol. 27, No, 2, pp. 10-20, 2019.

[5] M. Li, D. Bi, Z. Xiao: Mechanism simulation and experiment of 3-DOF parallel robot based on MATLAB, in: *Proceedings of the 2015 International Power, Electronics and Materials Engineering Conference*, Atlantis Press, pp. 489-494, 2015.

[6] N. Slavković, S. Živanović, N. Vorkapić, Z. Dimić: Development of the programming and simulation system of 4-axis robot with hybrid kinematic, FME Transactions, Vol. 50, No. 3, pp. 403-411, 2022.

[7] D. Milutinović, N. Slavković, B. Kokotović, M. Milutinović, S. Živanović, Z. Dimić: Kinematic modeling of reconfigurable parallel robots based on DELTA concept, in: *Proceedings of the 11th International Scientific Conference MMA 2012*, Novi Sad, Serbia, pp. 259-262, 2012.

[8] N. Slavković, N. Vorkapić, S. Živanović, Z. Dimić, B. Kokotović: Virtual BiSCARA robot integrated with open-architecture control system, in *Proceedings of the 14th International Scientific Conference MMA 2021*, Novi Sad, Serbia, pp. 63-66, 2021.