

# Development of the Programming and Simulation System of 4-axis Robot with Hybrid Kinematic

**Nikola Slavkovic**

Associate Professor  
University of Belgrade  
Faculty of Mechanical Engineering

**Sasa Zivanovic**

Full Professor  
University of Belgrade  
Faculty of Mechanical Engineering

**Nikola Vorkapic**

Teaching Assistant  
University of Belgrade  
Faculty of Mechanical Engineering

**Zoran Dimic**

Research Associate  
LOLA Institute Belgrade

*This paper presents an approach for developing the programming and off-line simulation systems for low-cost industrial robots in the MatLab/Simulink environment. The approach is presented in the example of a virtual model of a 4-axis robot with hybrid kinematics intended for manipulation tasks. The industrial robot with hybrid kinematics consists of the well-known 5R planar parallel mechanism to which two serial axes have been added. The programming system developed in a MatLab environment involves generating G-code programs based on given pick and place points. The virtual model included in the simulation system is configured in the Simulink environment based on the CAD model of the robot and its kinematic structure. The kinematic model and the inverse kinematic problem have to be included in the virtual model to realize the motion of the virtual robot. The system of programming and simulation has been verified through several examples that include object manipulation to perform various tasks.*

**Keywords:** kinematics, programming, virtual model, MatLab/Simulink, CAD system.

## 1. INTRODUCTION

The paper discusses the programming and simulation of a 4-axis robot with hybrid kinematics at the level of its virtual model. The considered virtual model is based on a robot with hybrid kinematics consisting of the well-known planar parallel mechanism (five-bar planar mechanism, BiSCARA, or pantograph) [1], to which two serial axes (translatory and rotary) have been added. The development of a five-bar mechanism started from a US patent in 1934 [1,2]. In 1978, Prof. Makino patented the well-known SCARA robot [3]. After that, Donald C. Fyler came up with the idea to use this five-bar mechanism as a robot [1,4].

Such robots are widespread in academic institutions because they are suitable for education and experimental work. A numerical approach for the position, velocity, and acceleration analysis for a five-bar parallel manipulator is described in the paper [5]. The paper [6] presents the design of a five-bar robot, analyzing optimal workspace and dexterity. The paper [7], besides the analytical kinematic model of the five-bar robot, demonstrates the use of the bond graph model to solve kinematic and dynamic problems with planar manipulators. The complete kinematic model of the mechanism presented in the paper [8] is formulated by considering the uncertainties included in the model. The kinematic performance with the uncertainties is computed using the Monte Carlo method. Then, the kinematic performance atlases based on workspace size and kinematic dexterity are analyzed, including the

uncertainties. A complete mechatronic approach to designing and developing a five-bar robotic arm for handling paper pot seedlings in a trans-planter is presented in the paper [9].

The hybrid kinematics allows this robot to use three degrees of freedom to reach a position and one degree of freedom to reach orientation around the vertical axis, making it suitable for manipulation tasks. The significant advantage of the used parallel robot is that two rotary motors are fixed at the base, resulting in lighter moving parts [1].

Most industrial robots are programmed either by teaching (on-line) or by a programming language (off-line). Industrial robots are successfully programmed for some of today's tasks using their programming languages and adequate software for simulation. The off-line robot programming complexity lies in that each robot manufacturer uses its robot programming language. This fact can be solved by using specialized CAM software for programming robots that generates directly native robot language using appropriate postprocessors for robots. There are also solutions for industrial robots that use G-code as a programming language [10,11].

The subject of this paper will be the development of the programming and simulation system for the virtual robot model when the G-code is used as a program. The programming and simulation system is developed in the MatLab/Simulink environment. In this way, before the pre-planned realization of the physical prototype, an adequate environment for programming and simulation is realized to verify the control programs. The G-code was selected as a programming language because of an open-architecture control system's pre-planned development based on the LinuxCNC software [12,13]. The open-architecture control system enables the imple-

Received: February 2022, Accepted: May 2022

Correspondence to: Prof. Nikola Slavkovic  
Faculty of Mechanical Engineering,  
Kraljice Marije 16, 11120 Belgrade 35, Serbia  
E-mail: nslavkovic@mas.bg.ac.rs

doi:10.5937/fme2203403S

© Faculty of Mechanical Engineering, Belgrade. All rights reserved

FME Transactions (2022) 50, 403-411 403

mentation of kinematic models of the different robots and is suitable for education in robot control system development.

Working in a virtual environment allows the verification of a program prepared for the real robot, i.e., it is possible to prevent errors that occur during the programming process, which would lead to a collision between the robot segments themselves or robot elements and the environment [14]. The virtual robot programmed in G-code can be implemented within the CAD/CAM system or open-architecture control system of the robot itself.

The off-line robot simulation and programming software include almost all robotic arms on the market to easily implement them 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 MatLab/Simulink environment [15-21].

In [15] is developed a system that integrates kinematics and motion control simulation using MatLab/Simulink, which can be connected with a considered robot to verify the simulation results. The paper [16] derived the theoretical model of the kinematics analysis of the Gough Stewart mechanism built into the Simulink/MatLab package to obtain the manipulators' lengths, position, and orientation at any manipulator time of motion. In paper [17], a DELTA parallel robot is designed to realize the seed sorting, simplifying the FANUC parallel robot. The CAD model of the robots is converted into a SimMechanics model. The simple motion simulation experiments are carried out to verify the correctness of the model. The paper [18] presents the model-based motion planning of a DELTA parallel robot in the Simulink/Simscape environment. A model was developed and simulated for a motion to solve the direct kinematics of the parallel manipulator and to check its efficacy. In [19] is proposed software to solve the problem of insufficient robots in schools. The authors combined MatLab/Simulink and AutoCAD to establish forward/inverse kinematics and trajectory planning.

The paper [20] presents the design, simulation, and control of a hexapod robot using tools available in MatLab software and the implementation of a dynamic model using the Simscape toolbox. The SimMechanics toolbox of MatLab is used to establish the mechanism model. The paper uses the kinematics mathematical model to simulate the robot motion control in a MatLab environment [21].

MatLab/Simulink environment is also useful in simulating machine tools with serial or parallel kinematics [22-24].

The main aim of the paper [22] was to design a simulation model of mechanism hexapod in the software MatLab/Simulink. The developed model comprises individual function blocks arranged to the block schema and mutually connected by suitable links. Besides the kinematic structure of the mechanism, the model contains the control system, PID controller, and interface for trajectory data input.

The paper [23] includes the general architecture of CNC machine tools developed in the Simulink environment. In contrast, the paper [24] presents the virtual machining system simulator containing a graphical user interface designed in Visual Basic and a multi-body simulation developed in Simulink.

Configuring the virtual robots in the Simulink/Simscape environment for programming and simulation of low-cost robots, i.e., a new prototype of robots as a part of the presented research, includes a complete kinematic model.

## 2. OUTLINE OF THE CONCEPT

A virtual model of the considered robot is a software implementation of its structure and kinematic model, which can execute generated G-code programs in the developed Simulink environment. The approach considered in this paper for developing industrial robots programming and simulation systems in MatLab/Simulink environment is presented in Fig. 1.

The considered approach for the development of a simulation system for low-cost robots consists of three modules:

- creating a file that defines pick and place points significant for the manipulation tasks,
- the development of the program for the joint space trajectory generation in the MatLab environment, and
- configuring the virtual robot in the Simulink environment.

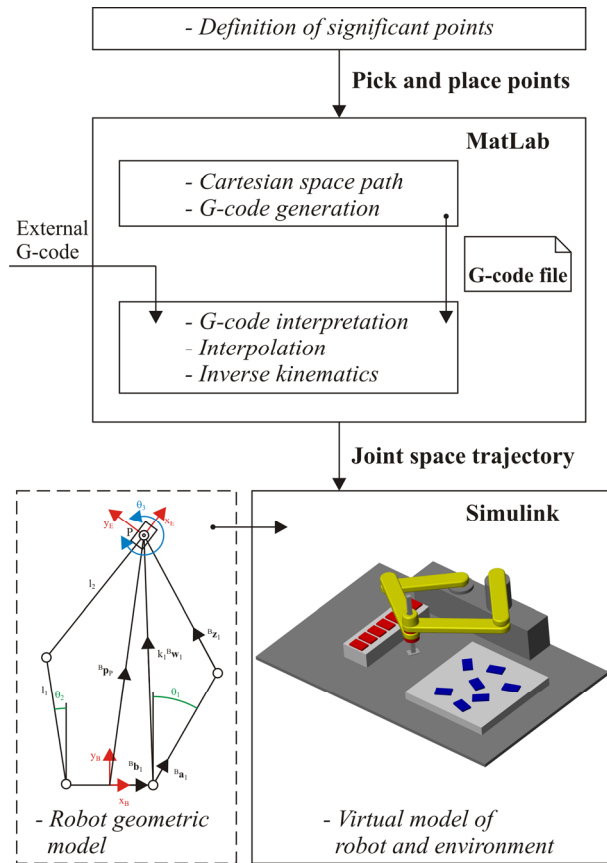
The first module is simple and involves the definition of a file that includes vectors of the points at which the robot performs the tasks, i.e., the pick and place points.

The module developed in the MatLab environment consists of two parts. One refers to developing functions for generating a G-code program based on a defined file of significant points. The other one refers to the development of MatLab functions to generate the joint space trajectories based on generated or loaded G-code. A joint space trajectory is generated based on the inverse kinematic problem[25], a function included in this module.

The main idea of this paper is to present the complete approach that includes both of these parts. The reason for this lies in the fact that there is almost no software that can be used for programming robot manipulation tasks by G-code. The first part does not have to be involved, and the second could interpret the external G-code generated in any available market CAD system to simulate any other robot applications, such as robot laser engraving or 3-axis milling. On the other hand, these two parts are independent, enabling the developed system to be used in different robot applications. Here is highlighted laser engraving, 3-axis milling, or 3D printing but the use of the developed system refers to any tasks that can be performed to consider the robot according to its configuration and workspace.

The third module, developed in the Simulink/Simscape environment, covers the implementation of

robot elements in STEP format, generated in the CAD system, in the Simscape model or \*.slx file [26].

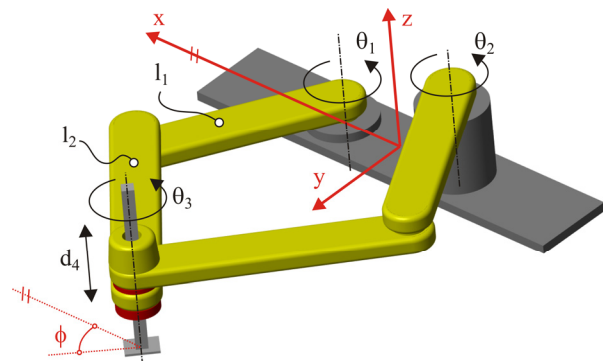


**Figure 1. Developed system for the robot programming and simulation**

The commercial PTC Creo software is used [27] in this research. The configuration of a virtual robot is done according to its kinematic structure, including kinematic connections (joints). It is necessary to create a geometric model of a robot to transfer the physical robot structure in the Simscape model, Fig. 1, because of the kinematic chains, the direction of robot joint axes, robot joint limits, reference robot frame, etc. A robot's geometric model can be found in Section 3 in detail.

### 3. KINEMATIC MODELING

As an approach to developing programming and simulation systems, the 4-axis robot with hybrid kinematics was chosen to show the application. Figure 2 shows the virtual model of the considered robot.



**Figure 2. The 4-axis BiSCARA robot**

The parallel robot structure consists of a base, a platform, and two kinematic chains with two struts with lengths  $l_1$  and  $l_2$ . The parallel mechanism allows the motion of end-effectors in the XY plane, while two added serial axes (translatory and rotary) enable the move in the Z direction and orientation of the end-effector about a vertical axis. Rotary joints connect all elements of the parallel mechanism with one degree of freedom.

Since the robot is based on a mechanism with hybrid kinematics, the inverse kinematics problem (IKP) for this robot could be divided into two parts [28]:

- IKP for parallel mechanism, and
- IKP for serial components of the mechanism.

#### 2.1 Inverse kinematic problem

The geometric model of the robot is established, Fig. 3, to solve IKP, The world coordinate vector, Fig 3a, is defined as

$$x = {}^B p_E = [x_E \ y_E \ z_E \ \phi]^T \quad (1)$$

In the same manner, the joint coordinate vector is derived as

$$q = [\theta_1 \ \theta_2 \ \theta_3 \ d_4]^T \quad (2)$$

The IKP for the parallel mechanism, Fig. 3b, is presented in [12,29], except that the robot is in the initial position in a different configuration, which leads to different equations for the first two joint coordinates.

The vectors necessary for derivation of the solution of angles  $\theta_1$  and  $\theta_2$  are:

- position vectors of the joint center at the base  ${}^B b_i = [b_{ix} \ 0]^T$ ,
- unit vectors  ${}^B a_i$  and  ${}^B z_i$  along struts with lengths  $l_1$  and  $l_2$ , and
- unit vectors  ${}^B w_i$

where  $i=1,2$  represents the number of the kinematic chains. The unit vector  ${}^B a_i$  is defined as

$${}^B a_i = \begin{bmatrix} -\sin(\theta_i) \\ \cos(\theta_i) \end{bmatrix}^T \quad (3)$$

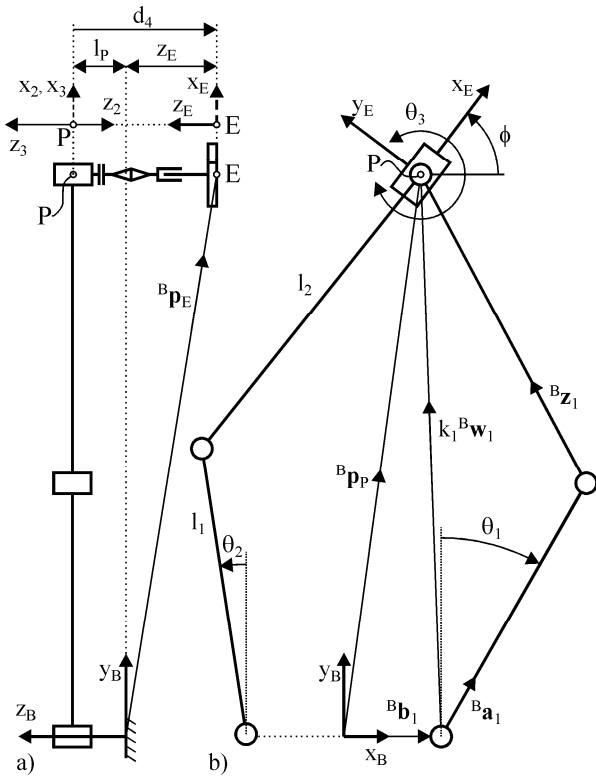
Based on the defined vectors, according to Fig. 3b, observing one kinematic chain, the following vector equations can be derived.

$${}^B p_p = {}^B b_i + k_i {}^B w_i k_i {}^B w_i = l_1 {}^B a_i + l_2 {}^B z_i \quad (4)$$

where vector  ${}^B p_p$  represents the position of the moving platform (point P) in the XY plane and is defined as

$${}^B p_p = [x_p \ y_p]^T = [x_E \ y_E]^T \quad (5)$$

From the system of equations (4), squaring the second equation and determining the vector  ${}^B w_i$  from the first equation, the inverse kinematic problem of parallel mechanisms is solved in a closed-form solution.



**Figure 3. Geometric model of the robot**

In this way, the well-known trigonometric equation is derived

$$A_i \cos(\theta_i) + B_i \sin(\theta_i) = c_i \quad (6)$$

Where

$$\begin{aligned} A_i &= -y_p \\ B_i &= x_p - b_{ix} \\ C_i &= \frac{l_1^2 - l_2^2 - x_p^2 - y_p^2 - b_{ix}^2 + 2x_p b_{ix}}{2l_1} \end{aligned} \quad (7)$$

By introducing the expression  $t = \tan\left(\frac{\theta_i}{2}\right)$  in equation (6) a well-known quadratic equation is obtained whose solutions are

$$t_{1/2} = \frac{B_i \pm \sqrt{A_i^2 + B_i^2 - C_i^2}}{A_i + C_i} \quad (8)$$

From equations (8), the joint coordinates  $\theta_1$  and  $\theta_2$  are determined as

$$\theta_i = 2A \tan(t_{1/2}) \quad (9)$$

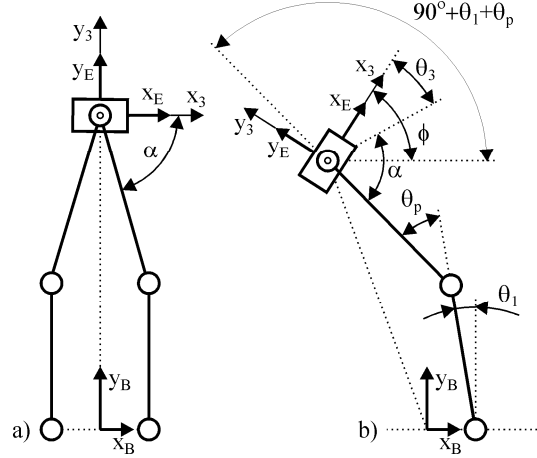
where  $i=1,2$  represents the number of the kinematic chains.

There are two solutions to the inverse kinematic problem for the parallel mechanism. The appropriate solution must be adapted to the part of the workspace necessary for the robot to perform the task.

The IKP of added two axes, i.e., the serial part of the mechanism covers the solution for the remaining two joint coordinates  $\theta_3$  and  $d_4$ . Figure 4 describes the procedure for solving joint coordinate  $\theta_3$ . To solve joint

coordinate  $\theta_3$  it is necessary to determine angle  $\alpha$ , Fig. 4a. The angle  $\alpha$  is calculated using the equation

$$\alpha = A \tan 2\left(\sqrt{l_2^2 - b_{1x}^2}, b_{1x}\right) \quad (10)$$



**Figure 4. Determination of the orientation angle**

Another important angle  $\theta_p$ , Fig. 4b, has to be calculated by the equation

$$\theta_p = A \tan 2\left(s\theta_p, c\theta_p\right) \quad (11)$$

Where

$$s\theta_p = \sqrt{1 - c\theta_p^2} \quad (12)$$

And

$$c\theta_p = \frac{(b_{1x} - x_p)^2 + y_p^2 - l_1^2 - l_2^2}{2l_1 l_2} \quad (13)$$

Now using equations (10) – (13), the joint coordinates  $\theta_3$ , Fig. 4b, could be calculated as

$$\theta_3 = \phi - \theta_1 - \theta_p - \alpha + 90^\circ \quad (14)$$

It is obvious from Fig. 3a that joint coordinates  $d_4$  can be calculated from the equation

$$d_4 = -z_E + l_p \quad (15)$$

The equations (9), (14), and (15) represent the solution of the inverse kinematic problem of a considered 4-axis robot with hybrid kinematics. These equations are required for the realization of the virtual model.

## 2.2 Workspace analysis

Another characteristic of the robot structure that is not crucially important for developing the virtual model is the robot workspace. It is necessary to determine the dimensions and shape of the robot workspace to program and simulate robot tasks correctly.

The method presented in [29] can be used to determine the workspace, which determines if the point defined in Cartesian space is reachable or not, according to the limits in the joints, based on solutions of the inverse kinematic problem. Determining the dimensions and shape of the workspace, in the case of designing a

new prototype, is an iterative procedure. It involves the mechanism parameters being changed and, in the end, adopted based on the satisfactory dimensions of the workspace.

The adopted parameters of the robot are  $l_1 = 210$  mm and  $l_2 = 270$  mm. Figure 5 presents the shape and dimension of the workspace of the considered 4-axis robot in the XY plane. The third dimension of the workspace depends on joint coordinate  $d_4$ , which is adapted to have a motion range of 100mm. According to robot structure, it is evident that the shape and dimensions of the workspace in the XY plane are the same along the Z direction, i.e., adopted 100mm.

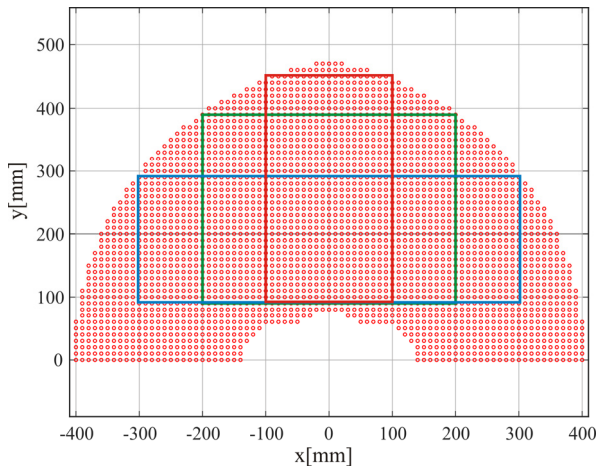


Figure 5. Workspace of considered robot

For the application of robot tasks, the operators could use all portions of the workspace with irregular shapes or reduced workspace to appropriate parallelepiped according to tasks, Fig. 5.

#### 4. PROGRAMMING AND SIMULATION SYSTEM

Briefly, the developed system for programming and off-line simulation for new low-cost laboratory prototype robots starts from the generation of programs in a MatLab environment. The G-code program is interpreted, and a joint space trajectory is determined. The joint space trajectory is then loaded to a virtual robot configured in the Simulink/Simscape environment. The virtual robot simulates the motion of programmed tasks using joints trajectory. Appropriate functions have been developed for each of these steps.

##### MatLab programming system

The MatLab programming system is developed only for manipulation tasks in G-code. The virtual environment can use G-code generated in some available CAD systems for other purposes.

The input parameter for the programming system is the file that consists of the initial position of robot configuration and significant points in which the robot performs the task. This file is formatted as presented in Fig. 6. The initial point is defined with vector  $p_0$ . The points of object picking are represented with vectors  $p_{1i}$ ,  $p_{2i}$ , etc., while the points of an object placing are represented with vectors  $p_{1o}$ ,  $p_{2o}$ , etc. All these vectors are defined in the robot reference coordinate system.

Based on created file, the MatLab function *trajectoryWorld* generates the robot trajectory in Cartesian space, Fig. 6. The generated trajectory is similar to the well-known CLF (Cutter Location File). Since the programming system is developed for manipulation tasks, the trajectory is generated based on the programmed rule that the gripper goes first to the point above the manipulation object, then to the point of picking the object, and at the end again to the point above the object. The same rule applies to the generation of the trajectory part when the gripper places the object.

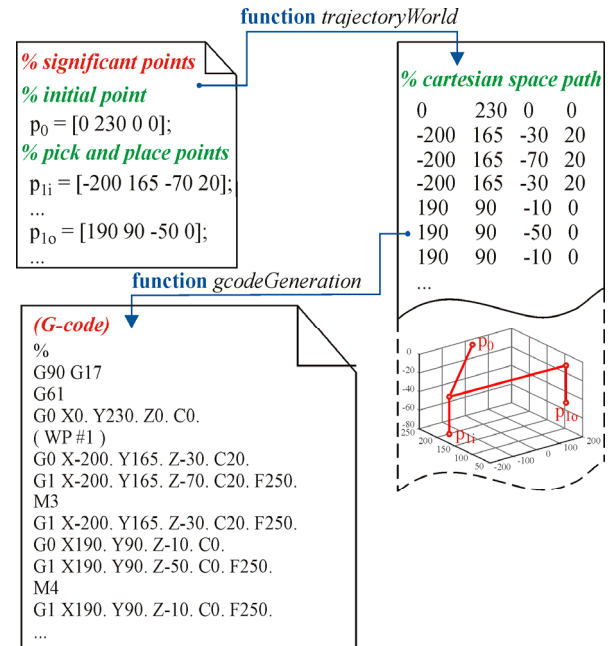


Figure 6. Developed programming system

The file in which the world space trajectory is written contains information on the feed rate and whether the gripper is moving at a rapid or not. It also indicates the points of picking and placing objects. The file is post-processed using the function *gcode Generation* to obtain the G-code.

Figure 6 shows part of the G-code from the example shown in Section 5. Besides the standard beginning of the program that defines programming in absolute coordinates and interpolation in the XY plane, parts of the code related to object manipulation can also be seen. In these parts, crucial commands are M3 and M4 that activate the gripper to pick or place an object. These auxiliary functions can be remade in LinuxCNC software [11].

##### Joint space trajectory generation

The functions for joint space trajectory generation starts from G-code are generated in a developed module or market CAD systems. This module is programmed to be completely independent due to the possibility of simulating some other tasks on a virtual robot. However, it uses some of the information defined in the previous module when generating G-code for manipulation tasks. First, the function *gcodeInterpreter* reads and parses one by one line of G-code. Its result is an inter-file that consists of the points defined in the



reference frame, speed, etc. In other words, the file defines the segments of the trajectory and the end-effector feed rate on those segments, Fig. 7.

Then, based on the inter-file, the function *interpolation* divided all segments using the rule of linear interpolation in Cartesian space resulting in the interpolated trajectory. This rule is programmed according to segment length, speed, and time increment.

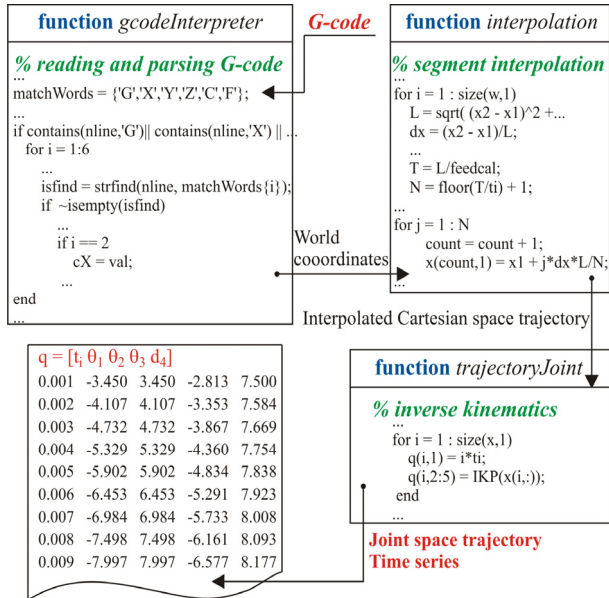


Figure 7. Generating the joint space trajectory

After the linear interpolation of trajectory in Cartesian space, the function *trajectoryJoint* involving the inverse kinematic solution, equations (9), (14), and (15), generates the joint space trajectory with an added vector of interpolated time as the first column. The result presents the time-series of joint coordinates necessary for the simulation of generated Simscape virtual robot.

### Simulink model of 4-axis robot

Configuring a virtual robot involves implementing a kinematic structure, which involves kinematic connections between segments, in a software environment. The CAD model of the considered robot, necessary for realizing virtual robot in Simulink environment, is implemented in PTC Creo software. From CAD software, the STEP files of robot segments are exported, and they present the input in the Simulink environment to configure the virtual robot. Figure 8 shows the structure of the considered robot realized in the Simulink environment taking into account the direction of robot joint axes, robot joint limits, reference robot frame, etc.

It is helpful to have a completely virtual environment, not only a virtual robot, to simulate robot tasks. Figure 8a presents the complete structure (robot and environment) in Simulink for the example described in Section 5. It is divided into three subsystems in the Simulink to change the robot environment for simulating other tasks easily. These subsystems are:

- joint time-series in MatLab environment,
- model of robot, and
- model of the environment.

The generation of joint time-series is presented in previous sections and uses only *From Workspace* element in Simulink. The environment model can be easily modeled in a CAD environment and implemented by appropriate connection in the Simulink model. These connections are the coordinate frames created in CAD or *Brick Solid* element in the Simulink.

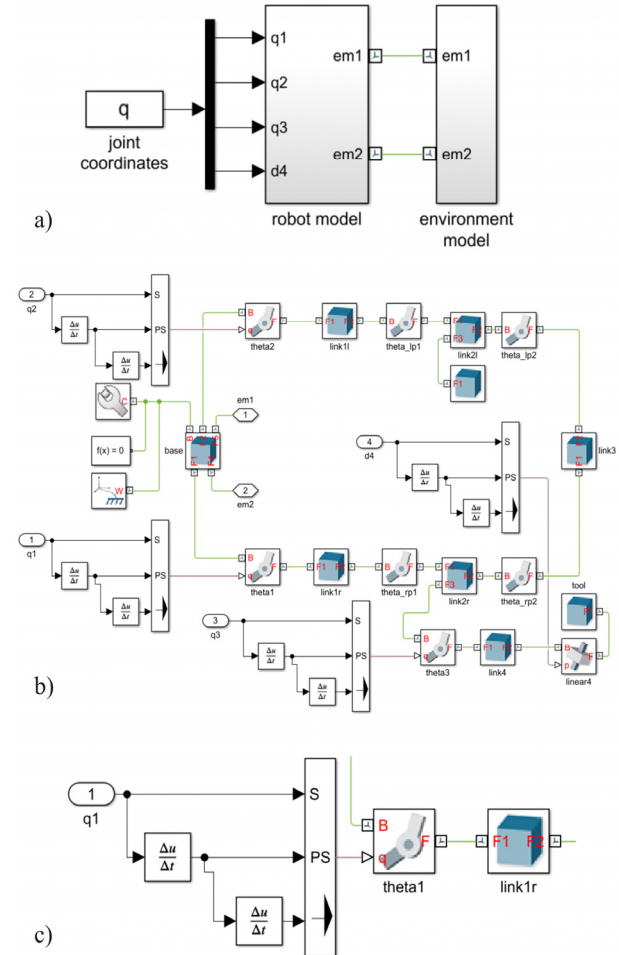


Figure 8. Simulink model of the robot

The structure of configured virtual robot in Simulink is presented in Fig. 8b. It consists of a base and three kinematic chains. The base is represented with the *Brick Solid* element connected with *World Frame*, *Solver Configuration*, and *Mechanism Configuration* elements. Two of the three kinematic chains present the parallel part of the considered mechanism, while the third present the serial part. The first two kinematic chains are interconnected with joint connections, and each of them is connected to the base. The third (serial) chain is connected to one of the first two, depending on kinematic calculation. Any kinematic chain consists of *Brick Solid* elements and appropriate *Revolute Joint* and/or *Prismatic Joint* elements.

The active joints in these kinematics chains have to be actuated by joint time series generated in MatLab while passive joints numerical compute their position. The actuated revolute joint is presented in Fig. 8c. It is actuated with joint angle, velocity, and acceleration, Fig. 8c, provided by an appropriate connection to receive the signal from the joint coordinates subsystem, Fig. 8a.

## 5. SYSTEM VERIFICATION

The developed programming and offline simulation system have been verified through several examples, including object manipulation to perform various tasks.

One example of an objects manipulation experiment is shown in Fig. 9a. The planned experiment is implied to take rectangular cross-section parts (blue parallele-pipeds) of arbitrary position and orientation and place them (red parallelepipeds) in a precisely defined position on a pallet site. The placing site can also be the conveyor belt, which can be modeled in a CAD environment and placed by appropriate connection in a virtual environment.

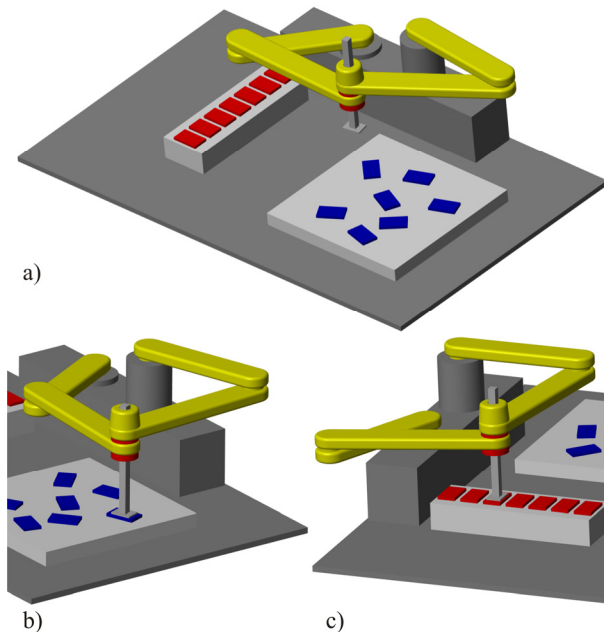


Figure 9. An example of the system verification

The start position and orientation of the objects are defined in the file at the beginning of experiments, Fig. 1. When a real robot performs a task, these positions could be obtained from a camera mounted on the robot itself. The virtual robot executes a program in G-code, which allows a simulation of robot motion according to the generated trajectory in joint space. Before the simulation is started, the G-code program is generated based on the created file. It is important to simulate the robot's approach to each position in the appropriate orientation for taking objects, Fig. 9a, as well as display the gripper position and orientation at the time of placing the object, Fig 9b.

The simulation covers all the robot's movements but not the display of the gripper picking, manipulating, and placing the objects. It is not important for verifying the accuracy of the program because it is planned to attach the vacuum or electromagnetic gripper to the robot plate.

After reviewing the complete simulation, it can be concluded that the virtual robot works correctly according to the generated G-code program and approaches all positions with appropriate orientations for picking and placing objects.

## 6. CONCLUSION

The main advantages of using virtual robots are the ability to check the end-effector's motion along the

programmed path, taking into account the limitations in the joints motions, and visual detection of collisions between segments and end-effector with the environment.

This paper presents an approach for developing the programming and simulation system of a 4-axis virtual robot with hybrid kinematics in the MatLab/Simulink environment. A virtual model of the considered robot is a software implementation of its CAD and kinematic models in the Simulink environment, which can execute generated G-code. The inverse kinematic problem is implemented in the virtual model to realize the motion of the virtual robot, as the real robot does.

The developed program in a Matlab environment based on implemented functions generates a G-code for the robot manipulation task. The developed virtual robot can simulate any other robot task such as laser engraving, 3-axis milling, or 3D printing, i.e., the virtual robot can use the G-code generated in some other CAD system. The developed system for programming and simulation of the configured virtual robot has been verified through several examples, including object manipulation to perform various tasks.

The presented approach of developing the programming and off-line simulation systems is crucial for the new laboratory prototypes of low-cost industrial robots that are not implemented in market software for simulation. The further research direction will cover the optimization of robot parameters and the development of a prototype of a considered 4-axis robot with hybrid kinematics.

## ACKNOWLEDGMENT

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. 451-03-68/2022-14/ 200105) and by the Science Fund of the Republic of Serbia, Grant No. 6523109, AI - MISSION4.0, 2020-2022.

## REFERENCES

- [1] DexTAR, User's Manual, Version 1.0, by Mecademic Inc., 2014–2015.
- [2] Pollard Jr., W.L.G.: Spray Painting Machine, US Patent 2,213,108, filed October 29, 1934, issued August 27, 1940.
- [3] Makino, H., Kato, A., and Yamazaki, Y.: Research and commercialization of SCARA robot, International Journal of Automation Technology, Vol. 1, No. 1, pp. 61–62, 2007.
- [4] Fyler, D.C.: Control Arm Assembly, US Patent 4,712,971, filed February 13, 1985, issued December 15, 1987.
- [5] Sosa-López, E.D., Arias-Montiel, M., Lugo-González, E.: A numerical approach for the inverse and forward kinematic analysis of 5R parallel manipulator, in: Proceedings of 2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), IEEE, pp. 1-6, October, 2017.

- [6] Huang, M.Z.: Design of a planar parallel robot for optimal workspace and dexterity, *International Journal of Advanced Robotic Systems*, Vol. 8, No. 4, pp. 176–183, 2011.
- [7] Jian, S., Yin, C., Rolland, L., James, L.: Five bar planar manipulator simulations and analysis by bond graph, in: *Proceedings of the ASME 2014 International Mechanical Engineering Congress and Exposition*, November 14-20, 2014, paper IMECE2014-37602.
- [8] Lara-Molina, F.A., Koroishi, E.H., Steffen, V., Martins, L.A.: Kinematic performance of planar 5R symmetrical parallel mechanism subjected to clearances and uncertainties, *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, Vol. 40, No. 4, pp. 1-15, 2018.
- [9] Rahul, K., Raheman, H., Paradkar, V.: Design and development of a 5R 2DOF parallel robot arm for handling paper pot seedlings in a vegetable transplanter, *Computers and Electronics in Agriculture*, 166, p. 105014, 2019.
- [10] Milutinovic, D., Glavonjic, M., Slavkovic, N., Dimic, Z., Zivanovic, S., Kokotovic, B., Tanovic, Lj.: Reconfigurable robotic machining system controlled and programmed in a machine tool manner, *International Journal of Advanced Manufacturing Technology*, Vol. 53, No. 9-12, pp. 1217-1229, 2011.
- [11] Milutinovic, D., Slavkovic, N., Kokotovic, B., Milutinovic, M., Zivanovic, S., Dimic, Z.: Kinematic modeling of reconfigurable parallel robots based on DELTA concept, *Journal of Production Engineering*, Vol.15, No.2, pp. 71-74, 2012.
- [12] Slavkovic, N., Vorkapic, N., Zivanovic, S., Dimic, Z., Kokotović, B.: Virtual Biscara robot integrated with open-architecture control system, in: *Proceedings of the 14th International Scientific Conference MMA 2021 – Flexible Technologies*, 23-25.09.2021, Novi Sad, pp. 63-66.
- [13] LinuxCNC, from <http://linuxcnc.org/>, accessed on 2021-05-07.
- [14] Vokapic, N., Zivanovic, S., Dimic, Z., Kokotovic, B., Slavkovic, N.: Virtual Horizontal Machining Center LOLA HBG 80 for Program Verification and Monitoring, *FME Transactions*, Vol. 49, No. 3, pp. 696-703, 2021.
- [15] Lee, W.-c., Kuo, S.-a.: Simulation and Control of a Robotic Arm Using MATLAB, Simulink and TwinCAT, in: *Proceedings of 2020 International Conference on Advanced Robotics and Intelligent Systems (ARIS) IEEE*, 2020, pp. 1-5.
- [16] Alwan, H.M., Sarhan, R.A.: 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.
- [17] Li, M., Bi, D., Xiao, Z.: Mechanism simulation and experiment of 3-DOF parallel robot based on MATLAB, in: *Proceedings of the 2015 International Power, Electronics and Materials Engineering Conference*, May 2015, pp. 489-494.
- [18] Makwana, M.A., Patolia, H.P.: Model-based motion simulation of delta parallel robot, *Journal of Physics: Conference Series*, Vol. 2115, No. 1, p. 012002, 2021.
- [19] Qassem, M.A., Abuhadrous, I., Elaydi, H.: Modeling and Simulation of 5 DOF educational robot arm, in: *Proceedings of 2010 2nd International Conference on Advanced Computer Control IEEE*, March 2010, pp. 569-574.
- [20] Urrea, C., Valenzuela, L., Kern, J.: Design, Simulation, and Control of a Hexapod Robot in Simscape Multibody, *Applications from Engineering with MATLAB Concepts*, pp. 126-137, 2016.
- [21] Yang, X., Dong, Y., Yang, H.: D2 delta robot structural design and kinematics analysis, in: *Proceedings of IOP Conference Series: Materials Science and Engineering*, Vol. 274, No. 1, p. 012009, IOP Publishing, December, 2017.
- [22] Bulej, V., Uriček, J., Eberth, M., Kuric, I., Stanček, J.: Modelling and simulation of machine tool prototype with 6DOF parallel mechanism in Matlab/Simulink, in: *Proceedings of MATEC Web of Conferences*, Vol. 254, p. 03002, EDP Sciences, 2019.
- [23] Cugnon, F., Berglind, L., Plakhotnik, D., Armendia, M., Ozturk, E.: Modeling of machining operations based on the Virtual Machine Tool concept, in: *Proceedings of 5th Joint International Conference on Multibody System Dynamics*, June 2018.
- [24] Theissen, N., Laspas, T., Szipka, K., Archenti, A.: Virtual machining system simulator: analysis of machine tool accuracy, *Procedia Manufacturing*, Vol. 25, pp. 338-343, 2018.
- [25] Cong, V.D.: Industrial robot arm controller based on programmable System-on-Chip device, *FME Transactions*, Vol. 49, No. 4, pp. 1025-1034, 2021.
- [26] Stevanović, I., Rašuo, B.: Development of a miniature robot based on experience inspired by nature, *FME Transactions*, Vol. 45, No. 1, pp. 189-197, 2017.
- [27] PTC Creo, webpage, from <https://www.ptc.com/>, accessed on 2022-02-01.
- [28] Milutinovic M., Slavkovic N., Milutinovic D.: Kinematic Modeling of Hybrid Parallel-Serial Five-Axis Machine Tool, *FME Transactions*, Vol.41, No.1, pp. 1-10, 2013.
- [29] Slavkovic, N., Zivanovic, S., Vorkapic, N.: Configuring a virtual prototype of a BiSCARA robot (in Serbian), *ТЕХНИКА*, Vol. 70, No.3, pp. 311-317, 2021.

---

**РАЗВОЈ СИСТЕМА ЗА ПРОГРАМИРАЊЕ И  
СИМУЛАЦИЈУ 4-ОСНОГ РОБОТА СА  
ХИБРИДНОМ КИНЕМАТИКОМ**



**Н. Славковић, С. Живановић, Н. Воркапић,  
З. Димић**

У раду је приказан приступ развоју система за програмирање и *off-line* симулацију развијених прототипова робота, који нису имплементирани у постојеће софтвере за роботе, у *MatLab/Simulink* окружењу. Приступ је приказан на примеру развоја виртуелног 4-осног робота са хибридном кинематиком, намењеног за задатке манипулације. Разматрани индустријски робот са хибридном кинематиком се састоји од добро познатог  $5R$  раванског механизма са паралелном кинема-

тиком, коме су додате две серијске осе. Систем програмирања развијен у *MatLab* окружењу обухвата генерисање  $G$ -код програма на основу задатих тачака у којима робот извршава задатак. Виртуелни модел је конфигурисан у *Simulink* окружењу на основу *CAD* модела робота и његове кинематичке структуре. Кинематичко моделирање и инверзни кинематички проблем су решени у циљу реализације кретања робота према генерисаном програму. Развијени систем за програмирање и симулацију је верификован кроз неколико примера који укључују манипулацију објектима при обављању различитих задатака.