VIRTUAL BISCARA ROBOT INTEGRATED WITH OPEN-ARCHITECTURE CONTROL SYSTEM

Conference Paper · September 2021

5 authors, including:

Nikola Slavković
University of Belgrade
35 PUBLICATIONS **271** CITATIONS
SEE PROFILE

Vorkapic Nikola
University of Belgrade
16 PUBLICATIONS **40** CITATIONS
SEE PROFILE

Sasa Zivanovic
University of Belgrade
82 PUBLICATIONS **450** CITATIONS
SEE PROFILE

Zoran Dimic
Lola Institut
46 PUBLICATIONS **206** CITATIONS
SEE PROFILE

MMA 2021
FLEXIBLE TECHNOLOGIES

14th INTERNATIONAL SCIENTIFIC CONFERENCE
NOVI SAD, SERBIA, SEPTEMBER 23-25, 2021

Slavković, N., Vorkapić, N., Živanović, S., Dimić, Z., Kokotović, B.

VIRTUAL BISCARA ROBOT INTEGRATED WITH OPEN-ARCHITECTURE CONTROL SYSTEM

Abstract: *The paper presents the developed control system with open architecture for the BiSCARA robot, based on the robot kinematic model. The control system is realized in the LinuxCNC software environment and includes the virtual robot model configured using several predefined Python classes and OpenGL. Presented methodology for configuring virtual robots could be used for any other robot or machine tool with parallel kinematic. The verification of the robot control system and robot kinematic model has been performed through several examples of drawing of contour on the configured virtual robot.*

Key words: *Kinematic modelling, Virtual robot, Control system, LinuxCNC, Simulation*

1. INTRODUCTION

The first five-bar robot, i.e. pantograph robot, shown in Fig. 1a, was described in a US patent in 1934 [1, 2]. Much later, in 1978, Prof. Hiroshi Makino invented the well-known SCARA robot [3]. Then, in 1985, Donald C. Fyler came up with the idea of using a five-bar mechanism, shown in Fig. 1b, as a robot [1,4].

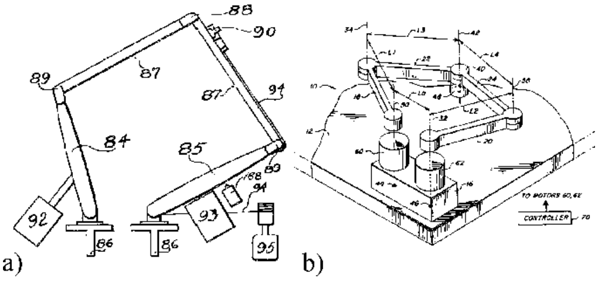

Fig. 1. Concept of five-bar, i.e. dual SCARA or BiSCARA, robot [1]

The first company that construct and commercialize the dual SCARA or BiSCARA robot, known as MELFA RP-1A, was Mitsubishi Electric, Fig. 2a, in 1998 [1].

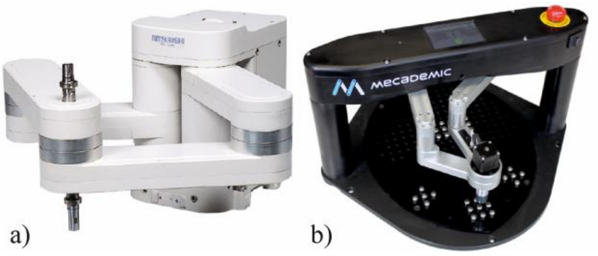

Fig. 2. Industrial BiSCARA robots [1]

An example of a developed industrial prototype, based on such a mechanism, is the DexTAR robot (Dextrous Twin-Arm Robot), Fig. 2b, which is developed for manipulation purposes [5].

The BiSCARA robot is very popular and is suitable for education and experimental work. Also, it is useful as a base mechanism that can be improved with the additional translatory and rotary axis in order to develop four or five-axis robots.

This paper presents the developed control system with open architecture for the adopted model of the BiSCARA robot, according to the kinematic and CAD models [6]. The control system is realized in the LinuxCNC software environment and includes the virtual robot model configured using several predefined Python classes and OpenGL.

2. KINEMATIC MODELLING

Parallel BiSCARA robot can be viewed as a planar manipulator with two degrees of freedom, Fig. 1. The kinematic modelling, which is described in detail in [6], is realized in order to develop an open-architecture control system. Figure 3 shows the kinematic model of the BiSCARA robot. The robot consists of a base, a platform, and two kinematic chains with struts lengths l_1 and l_2. All elements of the mechanism are connected by a joint with one rotary degree of freedom. The frame {B} represents the base frame, while the platform is represented by point P because the struts of length l_2 are connected at the point P.

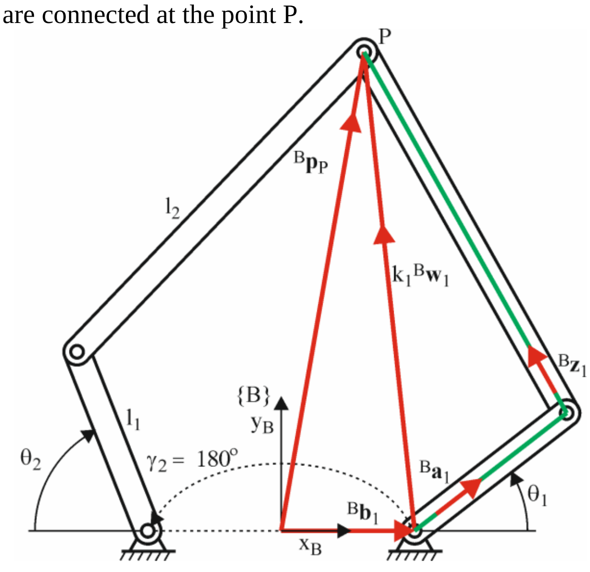

Fig. 3. Kinematic model of BiSCARA robot

As it can be seen from Fig. 3 the world coordinate vector can be represented as

$$ {}^{B}\boldsymbol{p}_P = [x_P \quad y_P]^T \tag{1} $$

while joint coordinate vector can be defined as

$$ \boldsymbol{\theta} = [\theta_1 \quad \theta_2]^T \tag{2} $$

As it is said in [6], the vectors defined by the parameters of the mechanism are

- position vectors of the join centers at the base ${}^{B}\boldsymbol{b}_i = [b_{ix} \quad 0]^T$,
- unit vectors ${}^{B}\boldsymbol{a}_i$ and ${}^{B}\boldsymbol{z}_i$ along struts with lengths l_1 and l_2, and
- unit vectors ${}^{B}\boldsymbol{w}_i$.

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

From Fig. 3 it is obvious that unit vector ${}^{B}\boldsymbol{a}_i$ can be defined as

$$ {}^{B}\boldsymbol{a}_i = \begin{bmatrix} \cos(\theta_i)\cos(\gamma_i) \\ \sin(\theta_i) \end{bmatrix} \tag{3} $$

where the angle γ_i represents the angle that defines arrangement of kinematic chains and was introduced in order to generalize the solution of the inverse kinematic problem and parallel determination of joint coordinates during the solving an inverse kinematic problem.

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

$$ \begin{aligned} {}^{B}\boldsymbol{p}_P &= {}^{B}\boldsymbol{b}_i + k_i\,{}^{B}\boldsymbol{w}_i \\ k_i\,{}^{B}\boldsymbol{w}_i &= l_1\,{}^{B}\boldsymbol{a}_i + l_2\,{}^{B}\boldsymbol{z}_i \end{aligned} \tag{4} $$

from which inverse and direct kinematic problems can be solved in analytic form.

Complete solution of inverse and direct kinematic problems, as the determination of Jacobian matrix and analysis of workspace is represented in authors' previous work [6]. Here is presented only the solutions of inverse kinematic problem that is crucial for development of virtual robot integrated with control system.

The joint coordinates can be determined using following equations

$$ \theta_i = 2Atan(t_{1/2}) \tag{5} $$

where

$$ t_{1/2} = \frac{y_P \pm \sqrt{A_i^2 + y_P^2 - B_i^2}}{(A_i + B_i)} \tag{6} $$

and

$$ \begin{aligned} A_i &= (x_P - b_{ix})\cos(\gamma_i) \\ B_i &= \frac{l_1^2 - l_2^2 + (x_P - b_{ix})^2 + y_P^2}{2l_1} \end{aligned} \tag{7} $$

From equation (6), it is obvious that there are two solutions of the inverse kinematic problem whereby the appropriate solution is adopted according to the part of the workspace necessary to robot perform the task.

3. VIRTUAL ROBOT INTEGRATED WITH CONTROL SYSTEM

The open architecture control system is based on the LinuxCNC software system and is a real-time control system for machine tools and robots, whose code can be freely used, modified, and distribute [7, 8, 9]. Software LinuxCNC [10] enables the programming of machine tools and robots according to the RS-274 or ISO 6983 standard.

The LinuxCNC software structure is shown on Fig. 4. It consists of four basic modules: motion controller (EMCMOT), discrete I/O controller (EMCIO), task coordinating module (EMCTASK), and graphical user interface (GUI). Of these four modules, only EMCMOT is a real-time module.

EMCMOT module performs trajectory planning, direct and inverse kinematics calculations, and computation of desired outputs to motor drivers. All I/O functions that are not directly related to the actual motions of machine axes are handled within the EMCIO module. EMCTASK module is a task-level command handler and program interpreter for the RS-274 NGC machine tool programming language, commonly referred to as a G-code [8].

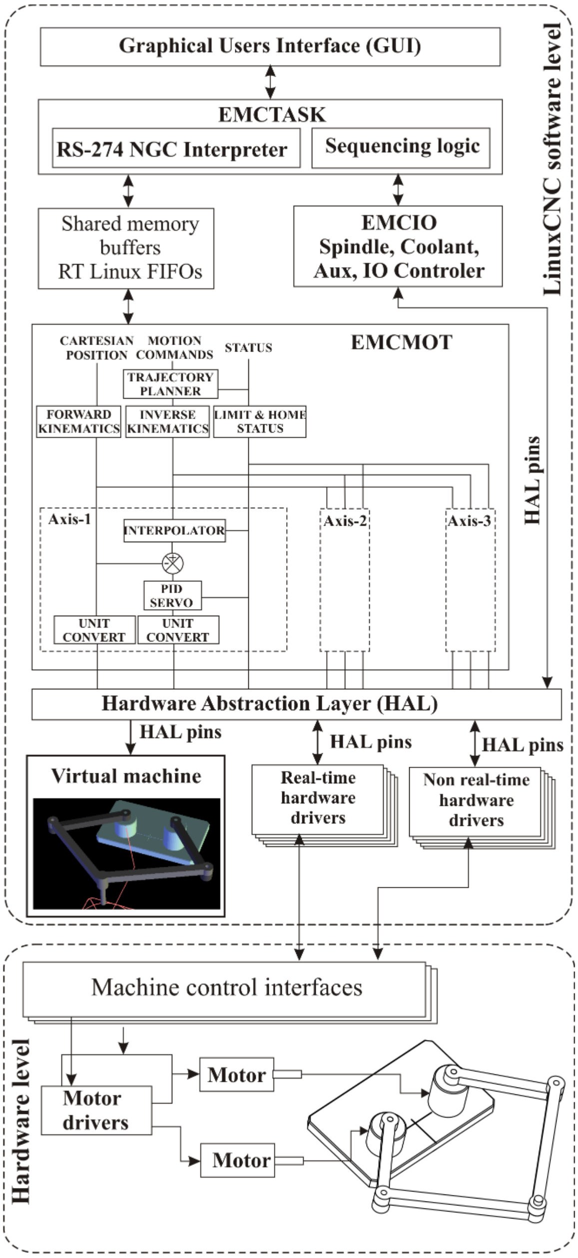

Fig. 4. Software structure of control system based on LinuxCNC

Several user interfaces have been developed for EMC2 software system. AXIS is the most advanced GUI, featuring an interactive G-code previewer. It is expanded to specific application needs of the proposed robotic machining system.

Based on the equations of inverse and direct kinematics, defined and derived from kinematic model of the BiSCARA robot, the kinematic module is programmed in C language and is integrated into the LinuxCNC software system in the EMCMOT module.

The virtual robot is configured in a Python 3D environment. Python is a programming language that can be used to program graphical user interface and allows programming and connection geometric primitives, as well as their integration with the EMC Axis GUI environment [9].

For virtual machine simulation set of Python functions Vismach is used, which can be used to create and animate the machine tools or robot models. The Vismach shows the machine or robot model in a 3D environment while moving parts of the robot are virtually actuated and they move with changes in the corresponding values signals received via the HAL pin connection.

The basic flow of activities in configuring the virtual robots in a Python 3D environment is to: (1) create HAL connections that control movement and that perform actuated of moving axes; (2) create basic components of robots, which make up its structure. Components can be programmed in the Python languages environment itself and grouped into collections or can be load as finished components; (3) create moving robot elements; (4) create animated components of robots; (5) assembling the robot model by loading and by positioning the components in the appropriate place [9].

Configuring virtual robots, directly in Python language, is practically the programming of the coordinates of geometric primitives, in order to define virtual robot assemblies. The job is getting easier by modeling a simplified model of a machine in some CAD system, from where could get the required coordinates, and then the programming of virtual robot components are approached in the Python program language. Virtual robot components can be significant simplified and described by elementary geometric primitive (Box, Cylinder, Sphere ...). The position of the primitive is programmed relative to the given reference coordinate system. Primitives, which make up one whole, are grouped into collections. Moving elements are connected by appropriate rotary or translatory connections. All parameters of the virtual robot are needed to be correctly placed as on a real robot, and the directions of the axes are set according to the defined kinematic model. During the programming, the errors are noticed immediately, corrected, and then the next component is defined.

The second way allows getting more faithful copies of real robots in the virtual world and consists of loading complete subassemblies of the robot, prepared in a CAD/CAM environment. Components are prepared in Ascii STL or Ascii OBJ format, which Python can load directly into the reference coordinate system, and then the component has to be positioned and oriented appropriately.

The result of any described procedures is a virtual robot or machine in a Python 3D environment that is integrated with a graphical interface Axis. The virtual robot runs in a Python 3D environment and allows the axis of the robot to move which results in the drawing of the toolpath. The simulation was generated as a result of the execution of G-code in real-time, in the same way as to control a real machine. That way it is possible to complete and verify the control system before the completion of real robots.

According to the second described procedure the virtual BiSCARA robot is generated and integrated with a developed open architecture control system, Fig. 4.

4. EXPERIMENTS

Development of virtual environments, for programming and robots or machine tools simulation is important for several reasons: (1) the need to verify the program, (2) verification of control system during the development of a new generation of robots or machine tools with complex kinematics, and (3) training and education for programming of robots or machine tools [9].

Verification of the developed virtual prototype integrated into the control system was performed through several examples of drawing tool paths. The programming was done by generating a G-code in the CAD/CAM system PTC Creo 5.0 [11].

A CAD virtual BiSCARA robot has also been prepared within a CAD/CAM programming system, Fig.5.

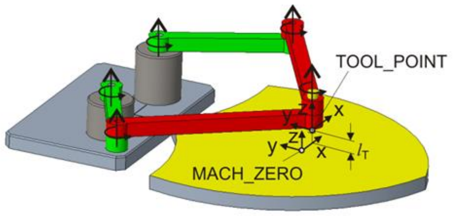

Fig. 5. CAD virtual model of BiSCARA robot [6]

This virtual robot works based on loaded toolpaths (CLF) and precedes the postprocessing of the program, i.e. generating the G-code, and is described in detail in authors' previous work [6].

The developed virtual two-axis mechanism, integrated with control system, is suitable for drawing of tool paths and laser engraving, so the programs generated in the CAD/CAM system are based on the strategy of trajectory milling. The programming itself is completely conventional and the same as for all other machine tools designed to perform these types of tasks.

Figure 6 shows one of the experiments, on virtual robot integrated with control system, which represents the drawing of the programmed tool path in the shape of the Serbian coat of arms. Figure 6a shows the developed virtual robot while drawing a programmed path that is as detail presented in Fig. 6c. Figure 6b shows the environment of the LinuxCNC software system and the loaded programmed path, which is essentially also one of the verifications of the generated G-code.

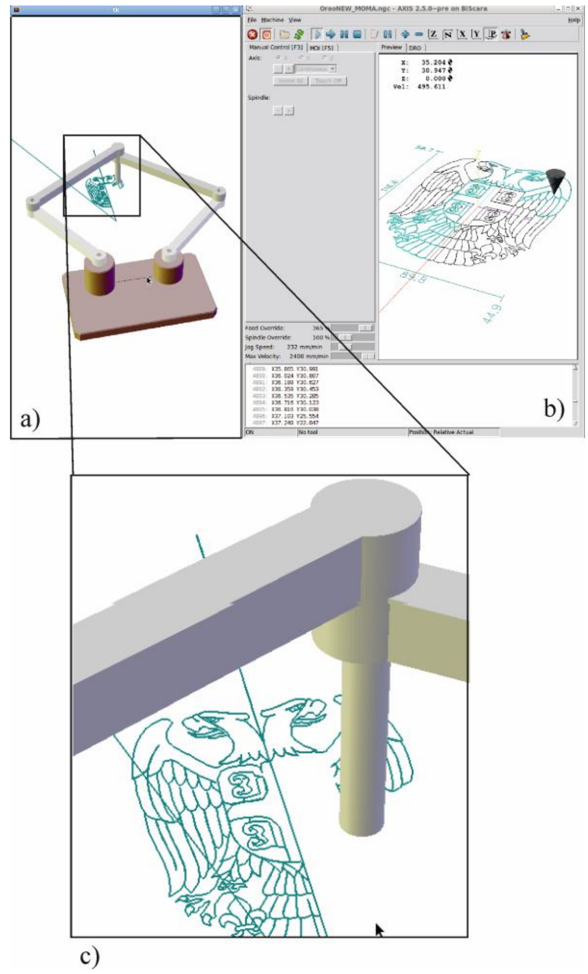

Fig. 6. Drawing of programmed tool path on virtual BiSCARA robot

5. CONCLUSION

Configuring of virtual robots or machine tools, among other things is important to verify control systems during the development of a new generation of robots or machine tools with complex kinematics.

The paper presents a developed open-architecture control system for BiSCARA robots. The open architecture control system is based on the LinuxCNC software system and includes the integrated virtual robot in a Python graphical environment.

The virtual prototype of the robot is configured based on the developed complete kinematic model of the robot, and also the CAD model of the robot based on the adopted parameters of the mechanism.

The realization of the laboratory prototype will be covered by further research, which will also include the development of a 4-axis robot by adding on described base mechanism one translatory and one rotary axis.

6. 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] Joubair, A., Slamani, M., Bonev, I. A: *Kinematic calibration of a five-bar planar parallel robot using all working modes*, Robotics and Computer-Integrated Manufacturing, vol. 29, pp. 15–25, 2013.

[6] Slavkovic, N., Zivanovic, S., Vorkapic, N.: *Configuring a virtual prototype of a BiSCARA robot*, TEHNIKA, Union of Engineers and Technicians of Serbia, vol. 70, no.3, pp. 311-317, 2021. (in Serbian)

[7] Živanović, S., Dimić, Z., Vasilić, G., Kokotović, B.: *Configuring a virtual reconfigurable two-axis machine with parallel kinematics integrated with an open architecture CNC system based on EMC2 software*, TEHNIKA, Union of Engineers and Technicians of Serbia, vol.67, no.4, pp. 519-526, 2018. (in Serbian)

[8] 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.

[9] Živanović, S., Dimić, Z.: *Virtual five-axis machine tool integrated with programming and control system*, TEHNIKA, Union of Engineers and Technicians of Serbia, vol.68, no.3, pp. 397-404, 2019. (in Serbian)

[10] *Linux CNC*, Enhanced Machine Control - EMC2, webpage, http://www.linuxcnc.org/

[11] *PTC Creo*, webpage, https://www.ptc.com/

***Authors:* Assoc. Prof. Nikola Slavković, Teaching Assistant Nikola Vorkapić, Full Prof. Saša Živanović, Research Associate Dr. Zoran Dimic, Assist. Prof. Branko Kokotović** University of Belgrade, Faculty of Mechanical Engineering, Kraljice Marije 16, 11120 Belgrade, Serbia, Phone: +381 11 33-02-423, Fax: +381 11 33-70-364

Lola Institute, Kneza Višeslava 70a, 11030 Belgrade, Serbia, Phone: +381 11 35-72-990, Fax: +381 11 25-44-096

E-mail: nslavkovic@mas.bg.ac.rs; nvorkapic@mas.bg.ac.rs; szivanovic@mas.bg.ac.rs; zoran.dimic@li.rs; bkokotovic@mas.bg.ac.rs

ACKNOWLEDGMENTS: 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. 451-03-9/2021-14/200105), and by the Science Fund of the Republic of Serbia, grant No. 6523109, AI - MISSION4.0, 2020-2022.