FACULTY OF MECHANICAL AND CIVIL ENGINEERING
IN KRALJEVO
UNIVERSITY OF KRAGUJEVAC

XI TRIENNIAL
INTERNATIONAL CONFERENCE

HEAVY MACHINERY HM 2023

Proceedings

VRNJAČKA BANJA, SERBIA
June 21– June 24, 2023

PLENARY SESSION

SESSION A: EARTH-MOVING AND TRANSPORTATION MACHINERY

Development of a Domestic 4-axis SCARA Robot

Zoran Miljković[1*], Nikola Slavković[1], Bogdan Momčilović[1], Đorđe Milićević[2]
[1]Faculty of Mechanical Engineering, Department for Production Engineering, University of Belgrade, Belgrade (Serbia)
[2] Smurfit Kappa Paper Factory Belgrade LLC, Belgrade (Serbia)

The global manufacturing industry has been demanding a steady increase in active industrial robots worldwide for years. The fields and technological tasks in which industrial robots are applied are rapidly expanding with a constant demand for improvement of their functions, technical characteristics as well as control and programming systems. One of the goals of the current research in the Laboratory for Robotics & AI is development of a domestic industrial robot with the possibility of automated programming based on information obtained from the camera. The paper presents the first part of the research developing a 4-axis SCARA industrial robot with the control system integrated camera. Professor Hiroshi Makino from Yamanashi University designed SCARA (Selective Compliance Assembly Robot Arm), and this robot is the most famous robot configuration originated at the universities. This part of the research includes the design of the mechanical structure, preliminary CAD/CAM testing, development of control and programming systems, virtual robot simulation, and robot production that were parts of two Master theses done in 2022. The realization of the robot control system starts from a well-known SCARA robot kinematic model. The open architecture control system realized in the LinuxCNC software allows the possibility of further development and full camera integration. The control system includes the integrated virtual robot model configured using several predefined Python classes and OpenGL as a digital shadow of the developed SCARA robot. Several successfully done examples of technological tasks of laser engraving have shown the verification of the complete robotic system.

Keywords: Industrial robot, Control and programming, Virtual robot

1. INTRODUCTION

One of the outcomes of Industry 4.0 is highly powerful, safe, and cost-effective intelligent factories developed with advanced robotics, cloud computing, the Internet of Things, and other advanced technological developments. Industrial robotics is an essential technology of Industry 4.0, which provides extensive capabilities in the field of manufacturing [1]. Industrial robots have a crucial role in modern factories with tasks such as picking and placing, inspection, assembly, machining, additive manufacturing, etc. Robot arms are complex enough to perform these tasks. However, the fields and technological demands in which industrial robots are applied are rapidly expanding with the constant requirement to improve their functions, technical characteristics as well as control and programming systems. This results in the robot arm being constantly modified with additional subsystems that result in advanced capabilities of its functionality. In other words, many industrial robots operating in intelligent factories use artificial intelligence to perform the high-level tasks. Today, they can also decide and learn from the experience in various ongoing situations [1].

Robot off-line programming and simulation through differently realized digital models are crucially important for some complex manufacturing tasks. As it is known, one of the essential Industry 4.0 technologies is a digital twin technology. The digital twin presents a virtual representation of the process, system or other physical entity using virtual and augmented reality, 3D visualization, and data modelling. Off-line programming can be generally categorized into computer-aided design-based (CAD-based) and vision-based approaches. These two off-line programming approaches, as a hybrid approach, have been widely applied in robotic systems to realize many manufacturing tasks [2].

The complexity of industrial robot programming in the manufacturing tasks such as cutting, laser engraving, 3D printing, etc., lies in the fact that there is no unique robot native language [3, 4]. Unlike robots, standard ISO 6983 (G-code) is used to program the machine tools. For this reason, it is difficult to involve widely available machine tools' CAD/CAM systems for programming machining robots. One way to solve this problem is by using the appropriate specialized CAM systems for robot programming or the developed translators to convert toolpaths for machining tasks into the corresponding robotic languages [4]. Robots' CAM systems or translators are not cost-effective and could be developed. Another way presents the open architecture control system development that allows programming in G-code [3].

One of the main goals of the current research in the Laboratory for Robotics & AI, Department of the Production Engineering at the Faculty of Mechanical Engineering within the University of Belgrade, is development of a domestic industrial robot and its digital twin with the possibility of automated programming based on the information obtained from the camera and AI techniques.

The selected robot for this research is SCARA (Selective Compliance Assembly Robot Arm) most famous robot configuration that originated at universities. The report [5] recounts mainly the first stage of the development of the SCARA. As stated in [5], the SCARA was invented by Professor Hiroshi Makino of the University of Yamanashi, Japan, the author of the report [5], and developed by him in collaboration with his colleagues and industrial partners. The SCARA is an industrial robot typical for those widely used in assembly processes. The prototype of the SCARA robot was built in 1978. Fundamental studies were done on the characteristics and

*Corresponding author: Kraljice Marije 16, 11120 Belgrade 35, Serbia, zmiljkovic@mas.bg.ac.rs

usability of this prototype, and the second was built in 1980. In 1981, some industrial partners began to market their versions of the SCARA. These models were called SCARA-type robots.

2. OUTLINE OF THE CONCEPT

The development of a domestic 4-axis SCARA industrial robot with programming in G-code and its digital twin with the possibility of a hybrid approach for automated programming based on information obtained from the camera and AI techniques consists of two parts.

The first part of the research includes the design of the mechanical structure, preliminary CAD/CAM testing, development of control and programming systems, virtual robot simulation, and robot production that were parts of two Master theses done in 2022 [6, 7]. This part is presented in the paper. The realization of the robot control system starts from a well-known SCARA robot kinematic model. The open architecture control system realized in the LinuxCNC software, which enables programming in G-code, allows further development and full camera integration. The control system includes the integrated virtual robot model configured by using the several predefined Python classes and OpenGL as a digital shadow of the developed SCARA robot. Several successfully done examples of technological tasks of laser engraving have shown the verification of the complete robotic system.

The highlights of the further research (the second part) will include: (1) the development of analytical and machine learning methods based on artificial neural networks to realize the accurate robot position based on camera information, (2) the applying artificial intelligence techniques and information obtained from the camera to generate the path of the robot end-effector for a selected class of tasks in a technological environment, and (3) the development of the experimental environment to achieve the software-hardware integration of the control system and visual detection based on the camera for automatic hybrid robot programming.

3. KINEMATIC MODELLING AND DESIGN

The kinematic modelling necessary for development of the control and programming system starts from well-known kinematic equations of the SCARA robots [8, 9]. The kinematic modelling of the robot includes: solutions for inverse and direct kinematic problems and analysis of the workspace.

3.1. Kinematic modelling of the 4-axis SCARA robot

The direct kinematic problem of serial manipulators can be easily solved by using Denavit-Hartenberg parameters and homogenous transformation.

The geometric model of the SCARA robot, its planar part including the first two links, is presented in Figure 1a. The world coordinate vector, Figure 1a and Figure 2a, is defined as:

$$\boldsymbol{x} = [p_x \quad p_y \quad p_z \quad \phi]^T \tag{1}$$

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

$$\boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad d_3 \quad \theta_4]^T \tag{2}$$

From Figure 1a, it could be derived equation (3) as follows:

$${p_x}^2 + {p_y}^2 = {a_1}^2 + {a_2}^2 + 2a_1a_2c\theta_2 \tag{3}$$

By expressing in terms of the unknown variable θ_2 the equation (4) is derived as:

$$c\theta_2 = \frac{{p_x}^2+{p_y}^2-{a_1}^2-{a_2}^2}{2a_1a_2} \tag{4}$$

Now using the expression $s\theta_2 = \pm\sqrt{1-(c\theta_2)^2}$ the joint coordinate θ_2 can be determined as:

$$\theta_2 = Atan2(s\theta_2, c\theta_2) \tag{5}$$

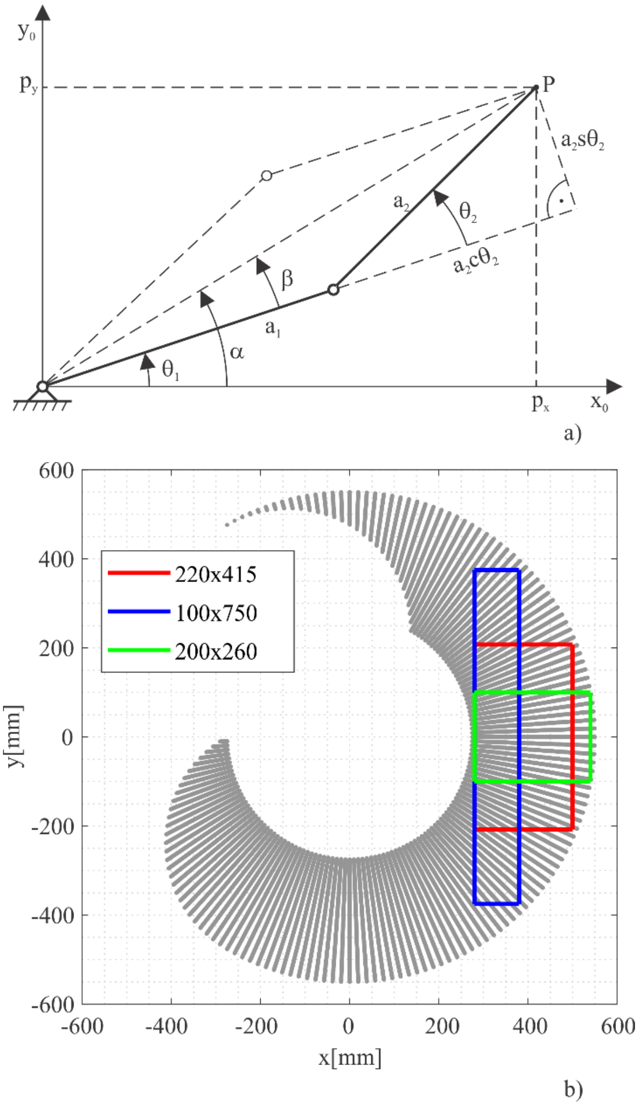


Figure 1: Kinematic model of SCARA robot

Joint coordinate θ_1 can be derived according to the value of angles α and β as:

$$\theta_1 = \alpha - \beta \tag{6}$$

$$\theta_1 = Atan2(p_y, p_x) - Atan2(a_2s\theta_2, a_1 + a_2c\theta_2) \tag{7}$$

Based on Figure 2a, the remaining two joint coordinates can be calculated simply as:

$$d_3 = d_1 - p_z \tag{8}$$

$$\theta_4 = \phi - \theta_1 - \theta_2 \tag{9}$$

Another characteristic of the robot structure for developing the virtual model and the physical prototype is the robot workspace. The method used to determine workspace considers whether the point defined in Cartesian space is reachable, according to the limits in the joints, based on solutions of the inverse kinematic problem. Designing a new prototype implies that the working space analysis is an iterative procedure. It involves the mechanism parameters being changed and, in the end, adopted based on the satisfactory dimensions of the workspace [10]. The robot parameters are $a_1 = a_2 = 275\,mm$, and joint coordinates θ_1 and θ_2 have limits $\pm 120°$. Figure 1b presents the shape and dimension of the workspace of the considered SCARA 4-axis robot in the xy plane. The third dimension of the workspace depends on joint coordinate d_3, Figure 2a. According to the robot structure, it is evident that the shape and dimensions of the workspace in the xy plane are the same along the z axis direction. The operators could use all portions of the workspace with irregular shape or reduced workspace to appropriate parallelepiped according to the task.

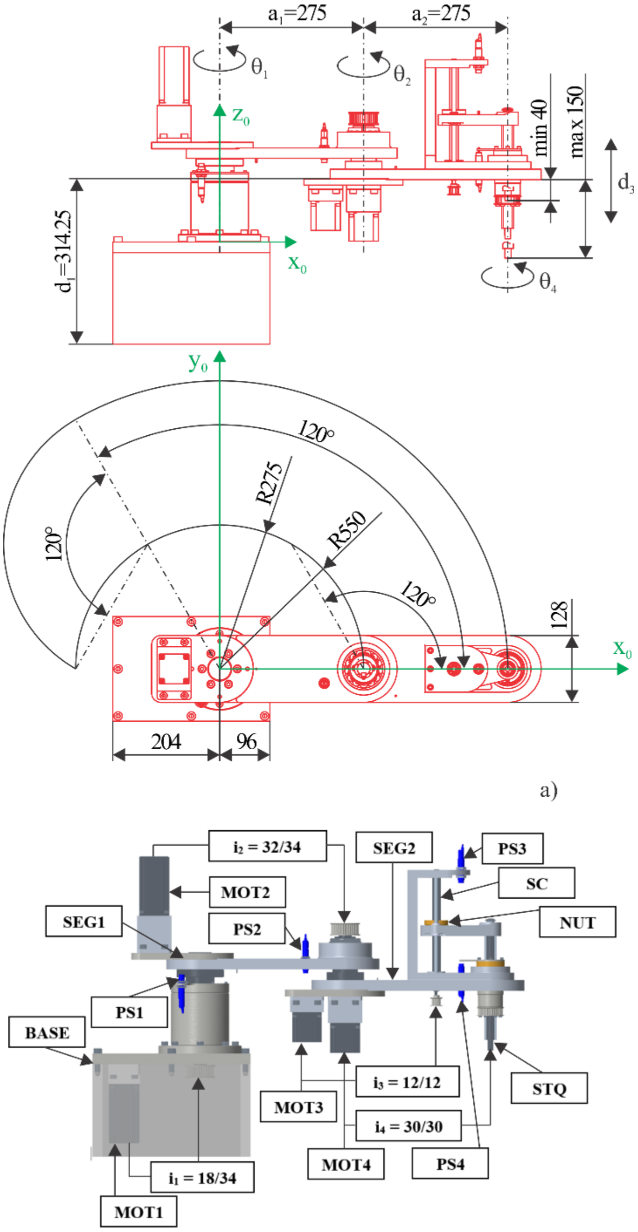
3.2. Designing the considered SCARA robot

Figure 2 shows the project of the SCARA robot, which is created within the CAD/CAM system PTC Creo, with its elements and gear ratios of the actuated axes. As it is said, the robot has four degrees of freedom. As with any robot of the SCARA configuration, the end-effector position in the horizontal plane is ensured by the coupled movement of the two segments by two rotary joints. The kinematic pair of a screw spindle and nut enables the position in the vertical plane. The fourth actuated rotary axis achieves orientation of the end-effector around its axis.

All axes are actuated by a stepper motor, which ensures the precise positioning of the moving elements of the robot. The first two rotating axes use NEMA 23, 57x112, 3.0A, 2.8Nm stepper motors, while the remaining two axes use, NEMA 23, 57x56, 2.8A, 1.17Nm. The torque transmission from the motor to the shafts of the actuated axis is achieved by using a toothed belt pair, HTD-5M/15. As we can see in Figure 2b, the gear ratio of the first actuated axis is $i_1 = 18/34$, the second $i_2 = 32/34$, the third $i_3 = 12/12$, and the fourth $i_4 = 30/30$. Since stepper motors are used while no encoders measure the joint position, the robot has four inductive sensors (one for each axis). These sensors are used in the initialization step when the physically determined zero-position of the robot has to be found.

The CAD/CAM environment in which the robot is configured can also be used as a system for programming and simulating the machining task of the industrial robots. To include the virtual prototype of the robot in the simulation of the programmed path, the robot must be configured with appropriate kinematic links, which was previously done by applying the rotary joints (Pin) and translatory joints (Slider). It is necessary to define the local coordinate systems of the robot and end-effector. The robot (machine) coordinate system is defined in the boundaries of the workspace as MACH_ZERO and on the workpiece. The coordinate system of the end-effector is defined as the TOOL_POINT. This coordinate system is automatically generated on the tool when it's selected. By matching these coordinate systems on a virtual model prototype, it is possible to start the robot simulation along the given toolpath.

During this research, the CAD model of the robot is also statically analysed by using the primary functions of the FEM method available in the CAD environment. One of the results of this analysis is the deformation of the robot in the most unfavourable configuration when the robot is in the most extended position loaded only by the gravity force.



a)

SEG1 – *first segment*

SEG2 – *second segment*

SC – *screw*

STQ – *shaft for torque transmission and achieving desired orientation of the end-effector*

MOT(1-4) – *motors of the first, second, third an fourth axis, respectively*

PS(1-4) – *proximity sensors for initialization of the first, second, third and fourth axis, respectively*

$i_{(1\text{-}4)}$ – *gear ratios of the first, second, third and fourth axis, respectively*

b)

Figure 2: CAD model of SCARA robot

The deformation is 0.07mm, which is acceptable for the robot laser engraving tasks. The analysis also includes the deformation analysis for different masses considered deformation of the robot model for the manipulation tasks.

4. DEVELOPMENT OF CONTROL AND PROGRAMMING SYSTEM

One of the features of this research is that the robot is applied directly by CNC machine tool programmers using the existing CAD/CAM systems and programming in G-code. The open architecture control system could enable this approach. Such control system can allow correctly implemented applications in it to work smoothly on different platforms of different suppliers and provide those applications cooperation with other system applications enabling the same communication with the user.

4.1. Open architecture control system software analysis

Various open architecture control systems are available today, which can be applied as a control system for the realized SCARA robot.

GRBL represents an open architecture control system with a built-in G-code interpreter (according to the RS274D standard) and a high-performance CNC (Computer Numerical Control) controller written in the optimized C programming language. Such control system can run on a microcontroller development environment called Arduino. The Atmega328p used on the Arduino platform provides the real-time positioning and asynchronous operations. Also, it can be used with various modifications to control milling machines, 3D printers, lathes, and SCARA configuration robots. However, the limitation refers to the control of three-axis only, so for the consideration of the SCARA robot, this control system can only be used when three axes are needed to perform the task. Figure 3a shows the position of GRBL in an open architecture CNC control system. GRBL can be seen as a firmware, which needs to be implemented in the Arduino to control the stepper motors. It is important to note that this system provides the control of machines that use stepper motors. The system uses low-cost stepper motor drivers current up to 2A. However, it can control high-current stepper motors where the current is provided by directly connecting the driver to the Arduino development environment (without Arduino CNC Shield). One of the advantages of such system is the low price of the hardware for the implementation of satisfactory control performance, as well as the fact that there is no need for the computer to have a Parallel Port, so a laptop computer is usually used. Figure 3b shows the connection diagram of the hardware components needed to realize the control of the SCARA robot with only two rotary axes, which ensure positioning in the horizontal plane. This example refers to the technical task of two-axis laser engraving, which the robot needs to perform. The micro switches are used to set the axes in the reference position, but the inductive sensors can be used identically with interface relays. With such a system, communication with the user can be achieved using different graphical user interfaces (GUI - Graphical User Interface). Such interfaces allow the user to send commands to the machine related to start the program, sending commands in manual mode, sending all axes to the reference position, etc. The frequently used GUI is the so-called UGS – Universal G-code Sender. It is possible to read each axis position at any time, as well as the programmed feed rate and spindle speed, which in the case of laser engraving would represent a percentage of the power of the laser beam.

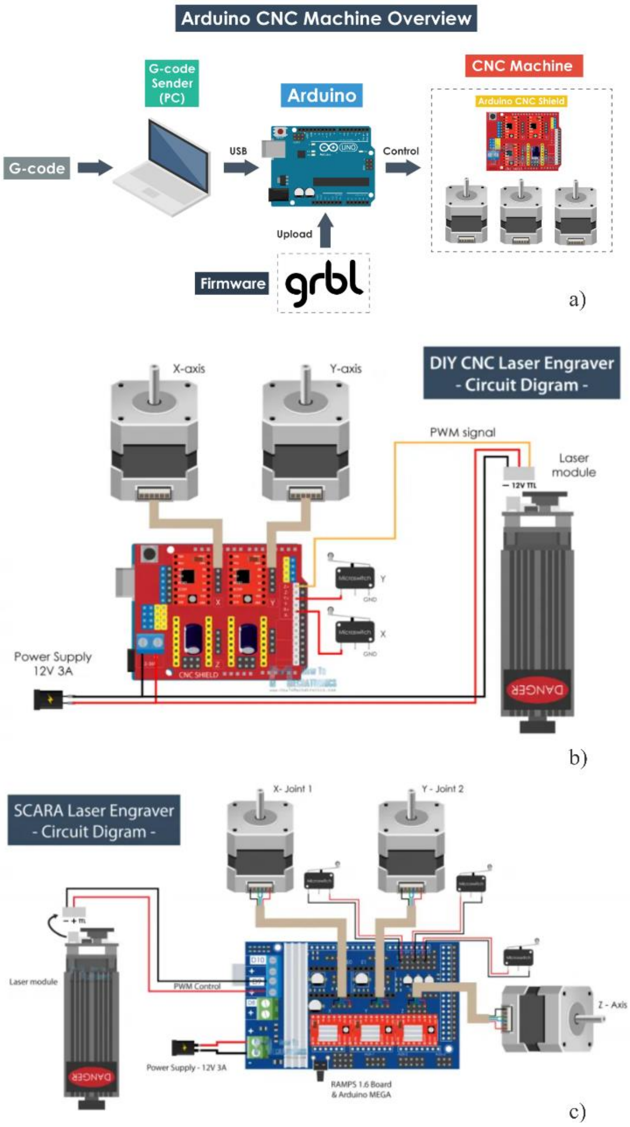


Figure 3: Open architecture control systems [11]

Marlin Firmware is an open architecture control system most commonly used for 3D printer control purposes. The firmware can also control CNC machines and laser engraving machines, Figure 3c. The firmware is used on a microcontroller development environment called Arduino Mega2560 and a board called RAMPS 1.6. Its main feature is reconfigurability and adaptability with different configurations, and accordingly, it supports Cartesian, DELTA, or SCARA kinematics. The firmware enables the execution of all machine activities in real-time, including the 3D printer heaters control, stepper motors, sensors, LCD screens, and everything else needed. Similar to the previously described GRBL, this firmware is implemented on the microcontroller, which should ensure the operation of the machine. For communication with machine users, there are several options in this case. The graphical user interface called Repetier-Host is popular in 3D printing. This interface is suitable when the SCARA robot is used as a 3D printer or a laser engraving machine.

The developed open architecture control system is based on the LinuxCNC software system. It is a real-time control system for machine tools and robots where code can be freely used, modified, and distribute [3]. Software LinuxCNC enables machine tools and robot programming according to the RS-274 or ISO 6983 standard. Configuring the control system includes configuring both hardware and software.

4.2. Development of control system based on LinuxCNC

The LinuxCNC software structure is shown in Figure 4. It consists of four basic modules: motion controller (EMCMOT), discrete I/O controller (EMCIO), task coordinating module (EMCTSK), and graphical user interface (GUI).

The GUI (Graphical User Interface) is an external program that communicates with the EMC by sending commands such as: turning on the machine, switching to automatic mode, starting the program, switching on, etc. The user interfaces can also send commands initiated by the operator, such as moving the machine axes into the manual mode or sending all axes to the reference position. The AXIS interface is often used because it looks like several numerically controlled machine tools. It is expanded to the specific application needs of the proposed robotic machining system [3, 12].

EMCIO (Discrete I/O Controller) is a module that performs all communications not related to motion control. This module has child modules for the main spindle, tool change, coolant and lubricant, auxiliary functions all stop, lubrication, etc.

EMCTSK (Task Coordinating Module - process controller) is a module that distributes commands on the machine. Performs program interpretation in G-code, according to the RS 274 NGC standard.

Of the 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.

The HAL (Hardware Abstraction Layer) is designed as a flexible interface between the motion controller on the one hand and everything needed to interface with the user and the machine on the other. It represents a method for activating or switching on all the necessary modules so a numerical control system is completed which can be used adequately in industrial robots to perform certain groups of tasks.

LinuxCNC enables the control of machines and robots with complex (non-trivial) kinematics. Such system is controlled based on the solved inverse and direct kinematic problems, which are programmed in the C language and implemented instead of the existing trivial kinematics. As the SCARA configuration represents a machine with serial but non-trivial kinematics, the inverse and direct kinematics were translated into the C programming language format and incorporated into the machine kinematics file. The kinematics file of the machine must then be compiled so that when starting, the robot works according to the defined kinematics.

When configuring the hardware for the robot control system, it is necessary to determine the electronic components needed to perform the task, Figure 5.

As said, the robot uses stepper motors. The stepper motor drivers, DM542, 20-50VDC, Max 4.2A, and TB6600, 9-40VDC, Max 4.5A are used. Both types of drivers (DM542 and TB6600) have current and micro-stepping adjustments via DIP micro-switches, according to the motors and the application for which they are used.

For the initialization step, PNP NO (normally open) inductive sensors with an operating voltage of 24VDC are used. These sensors are connected via an interface relay to the control panel.

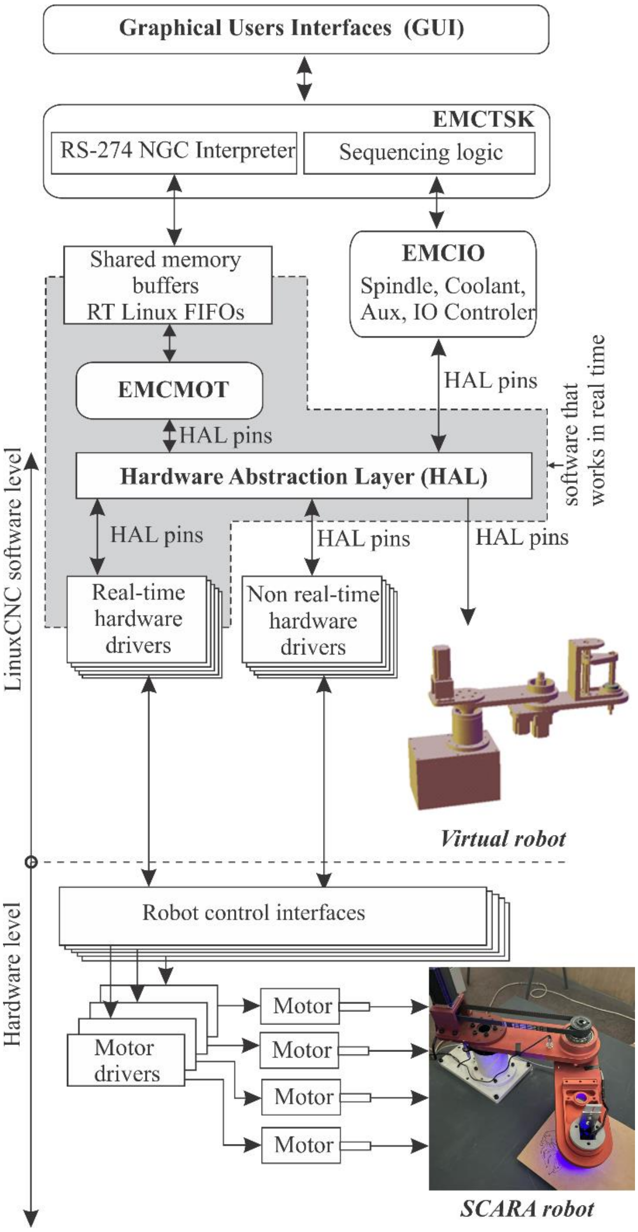


Figure 4: Developed robot control system

The laser module is necessary to perform the technological task of laser engraving, which emits a laser beam with sufficient optical power for engraving on wood. A laser module with an output optical power of 15W is used with a suitable driver that regulates the output optical power and fan speed, as well as a power supply whose input is alternating current with voltage 220VAC, while the output is direct current with voltage 12VDC and 5A current (module input power is 60W). The driver of the laser module can control the PWM signal.

As LinuxCNC communicates with the peripherals of the PC through the parallel port, the Mach3 CNC Controller board is used to connect with the necessary components. This board can control up to five axes, which are actuated by stepper motors.

Only the crucial electronic components are listed here. All selected electronic components are arranged and circuited according to an electrical diagram, Figure 5.

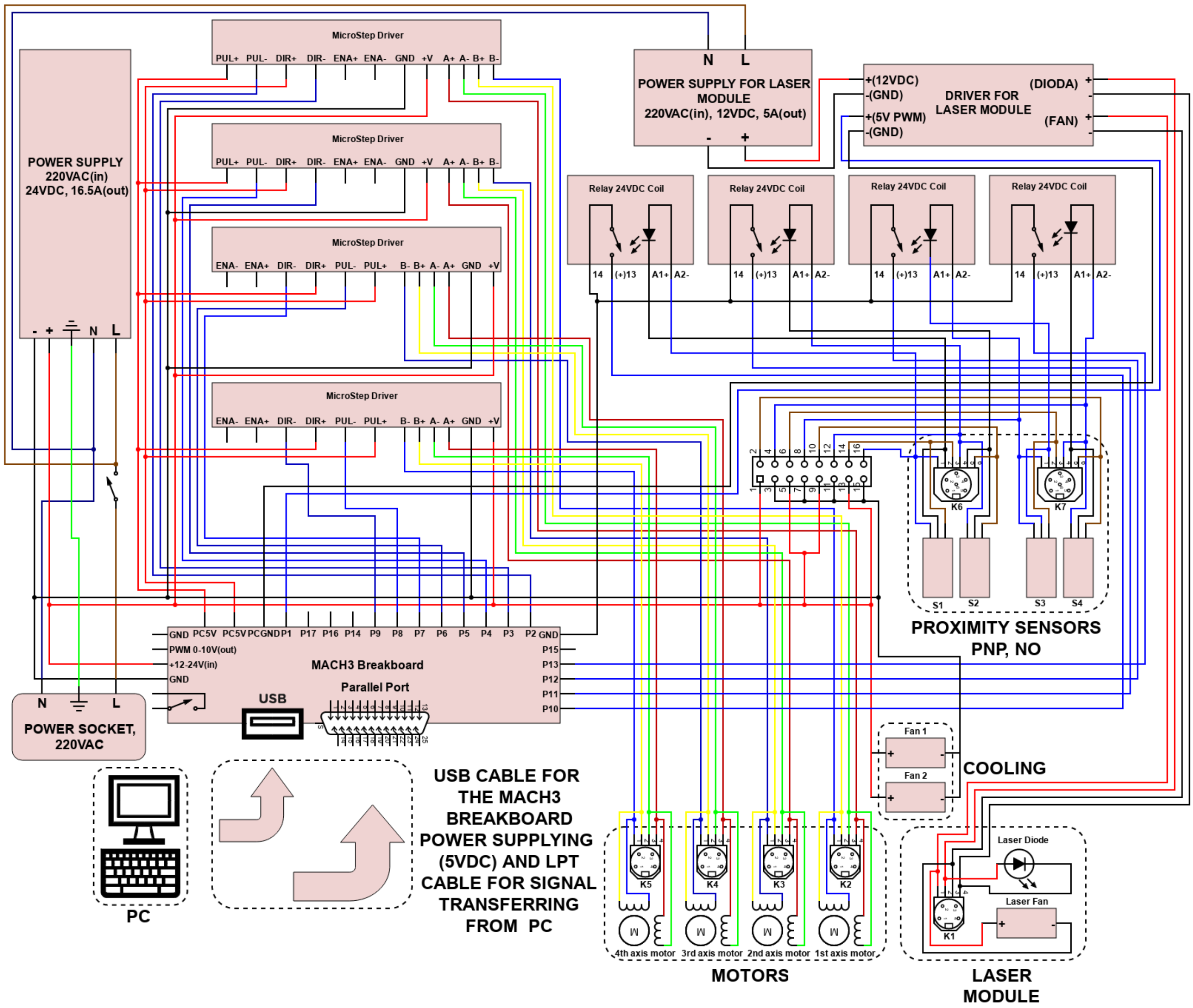


Figure 5: The electrical circuit of electronic components

4.3. Virtual robot configuring

The virtual robot is configured in the Python 3D environment. Python is a programming language that can be used to program graphical user interfaces allowing the programming and connecting of geometric primitives and their integration with the LinuxCNC AXIS GUI environment.

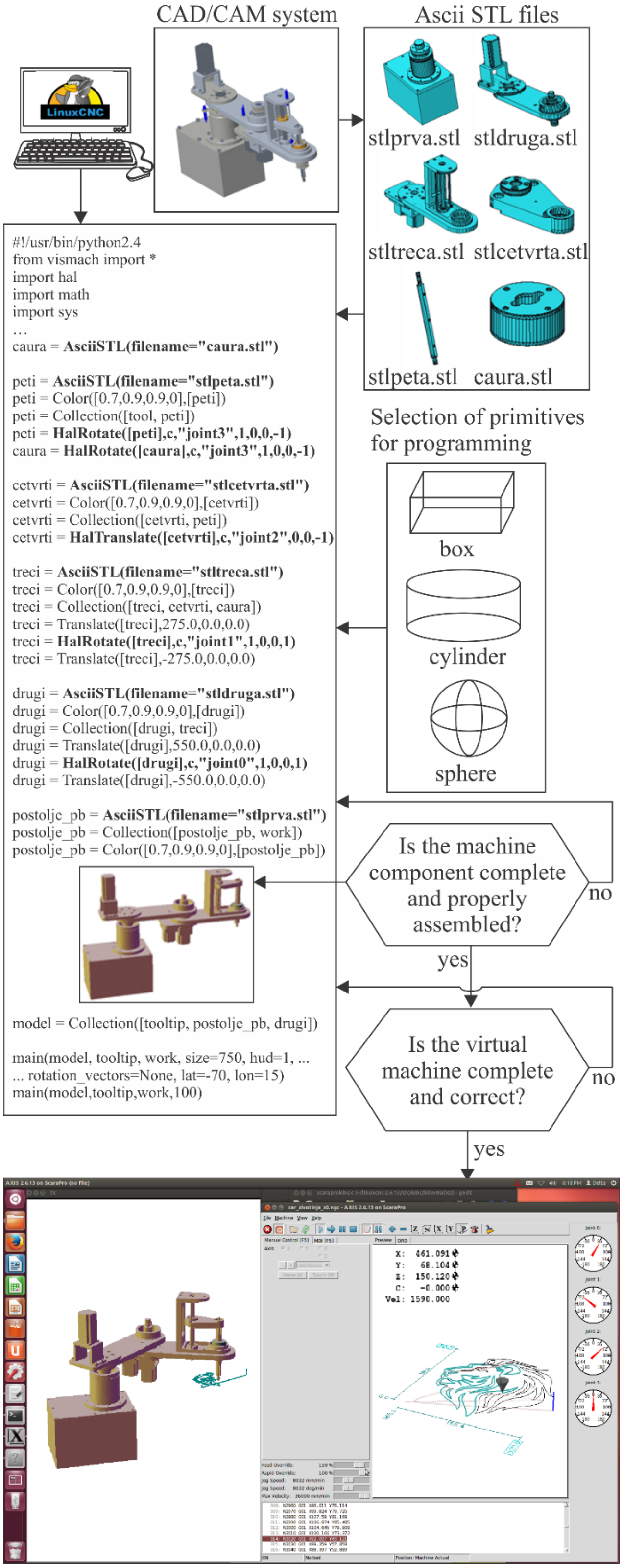
Python functions can create and animate machine tools or robot models. The machine or robot model is displayed in a 3D environment with the robot moving parts virtually actuated and moved with changes in the corresponding signal values received via the HAL pin connection.

The basic flow of activities in configuring virtual robots in the Python 3D environment is [12]:

- create HAL connections that control movement and actuated moving axes,
- create components that make up the robot structure. It can be programmed in the Python environment itself and grouped into collections, or it can be loaded as finished components,
- creation of moving robotic elements,
- creation of animated robot components, and
- assembling the robot model by loading and positioning the components in the appropriate place.

Configuring virtual robots directly in Python is practically programming the coordinates of the geometric primitives that define the virtual robot models. It is easier to model a simplified machine model in a CAD system, where the required coordinates can be obtained. Then the programming of the virtual robot components is approached in the Python programming language, Figure 6. The virtual robot components can be significantly simplified and described by an elementary geometric primitive (box, cylinder, sphere, ...). The position of the primitive is programmed relative to a given reference coordinate system. Primitives, which form a whole, are grouped into collections. Moving elements are connected by using the appropriate rotary or translatory connections. The virtual robot parameters are set correctly, as on the real robot, and the axis directions are set according to the defined kinematic model. During programming, errors are immediately detected and corrected. Then the next component is defined.

The second way allows obtaining more realistic copies of real robots in the virtual world and consists of loading robot subassemblies prepared in a CAD/CAM environment. Components are in ASCII STL or ASCII OBJ format, which Python can load directly into the reference coordinate system, and then it must be positioned and oriented appropriately.



```
#!/usr/bin/python2.4
from vismach import *
import hal
import math
import sys
…
caura = AsciiSTL(filename="caura.stl")

peti = AsciiSTL(filename="stlpeta.stl")
peti = Color([0.7,0.9,0.9,0],[peti])
peti = Collection([tool, peti])
peti = HalRotate([peti],c,"joint3",1,0,0,-1)
caura = HalRotate([caura],c,"joint3",1,0,0,-1)

cetvrti = AsciiSTL(filename="stlcetvrta.stl")
cetvrti = Color([0.7,0.9,0.9,0],[cetvrti])
cetvrti = Collection([cetvrti, peti])
cetvrti = HalTranslate([cetvrti],c,"joint2",0,0,-1)

treci = AsciiSTL(filename="stltreca.stl")
treci = Color([0.7,0.9,0.9,0],[treci])
treci = Collection([treci, cetvrti, caura])
treci = Translate([treci],275.0,0.0,0.0)
treci = HalRotate([treci],c,"joint1",1,0,0,1)
treci = Translate([treci],-275.0,0.0,0.0)

drugi = AsciiSTL(filename="stldruga.stl")
drugi = Color([0.7,0.9,0.9,0],[drugi])
drugi = Collection([drugi, treci])
drugi = Translate([drugi],550.0,0.0,0.0)
drugi = HalRotate([drugi],c,"joint0",1,0,0,1)
drugi = Translate([drugi],-550.0,0.0,0.0)

postolje_pb = AsciiSTL(filename="stlprva.stl")
postolje_pb = Collection([postolje_pb, work])
postolje_pb = Color([0.7,0.9,0.9,0],[postolje_pb])

model = Collection([tooltip, postolje_pb, drugi])

main(model, tooltip, work, size=750, hud=1, ...
... rotation_vectors=None, lat=-70, lon=15)
main(model,tooltip,work,100)
```

```
  X:   461.091
  Y:    68.104
  Z:   150.120
  C:    -0.000
Vel:  1590.000
```

Figure 6: Virtual robot configuring in LinuxCNC

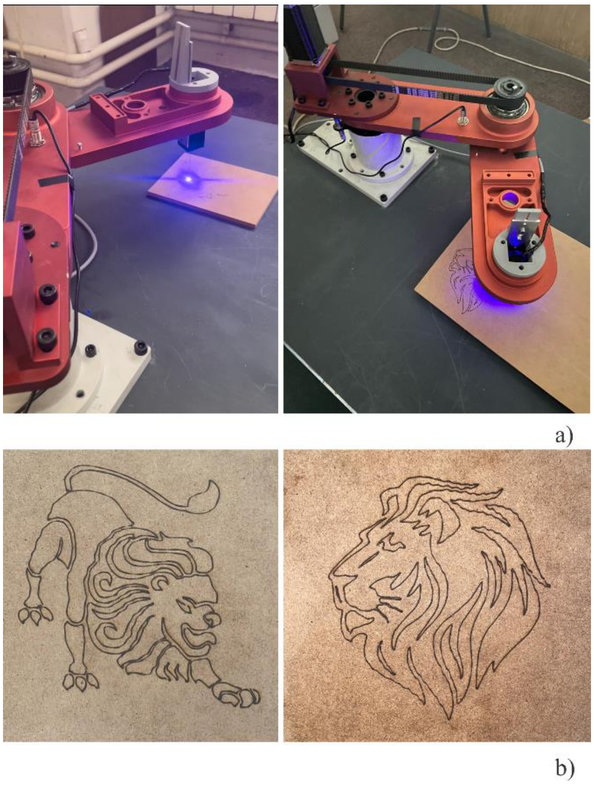
The result of any described procedure is a virtual robot or machine in the Python 3D environment integrated with the AXIS graphical interface. The virtual robot runs in the Python 3D environment and allows the robot axis movement through the toolpath simulation. The simulation is generated as a result of executing the G-code in real-time, in the same way as if a real machine were being controlled. In this way, it is possible to complete and verify the control system before the completion of the actual robots. According to the second described procedure, a virtual SCARA robot is generated and integrated with the developed open architecture control system, Figure 6.

5. EXPERIMENTAL VERIFICATION

The programming method is conventional and starts from the CAD/CAM environment. After verifying the toolpath (CL file), the procedure follows post-processing and obtaining the G-code. Thus the robot program in G-code is loaded into the control software and then verified on a virtual robot in real-time, Figure 6. Then the control signals can safely be directed to a real robot, Figure 7a.

When starting the control and programming system, a choice is made between controlling a virtual or a real robot. Usually, the virtual robot starts, initializes, and sets it up first. Then testing the program in two ways is realized. First, loading the G-code into the AXIS interface of the LinuxCNC software on the screen shows the programmed toolpath, and second, the final program verification on a virtual robot.

This is important for checking the toolpath, as well as for checking the placement of the workpiece within the limits of the working space and the correctness of determining the zero point.


a)

b)

Figure 7: Robot laser engraving

Several successfully done examples of technological tasks of laser engraving have shown the verification of the complete robotic system. Figure 7b shows some realized examples of robot laser engraving of contours on a wood plate. The examples shown are laser engraving of the silhouette of the lion's head and its complete figure.

For both examples, G-codes were generated in the CAD/CAM environment based on the DXF file.

After switching on the robot, the robot's physical position is unknown to the control system due to the stepper motor uses without an encoder. Before starting the robot task, an initialization sequence must be performed to LinuxCNC calculates the reference position for counting motor pulses. The prepared program is generated concerning the zero point of the workpiece, marked with the G55 function, the workspace position has to be determined in the robot workspace and forwarded to the control system.

After the program is generated and loaded, the initialization is done. Another step describes the procedure for testing on a virtual robot. Then, the laser engraving tasks for selected examples are done, Figure 7.

6. CONCLUSION

Industrial robot applications are rapidly expanding with a permanent improvement of their functions, technical characteristics as well as control and programming systems. The presented research includes the development of a domestic 4-axis SCARA industrial robot with the possibility of automated programming based on information obtained from the camera and AI techniques. The first part of this research covered the production of the selected robot. This part includes the design of the mechanical structure, preliminary CAD/CAM testing, development of control and programming systems, and virtual robot simulation.

Among several open architecture control software, the LinuxCNC system is selected. Software LinuxCNC enables machine tools and robot programming according to the RS-274 or ISO 6983 standard. It is a real-time control system for machine tools and robots whose code can be freely used, modified, and distributed. This system enables the use of widely available machine tools' CAD/CAM systems for programming machining robots. The control system includes the integrated virtual robot model configured using several predefined Python classes and OpenGL. The virtual robot prototype is configured based on the developed complete kinematic model of the robot and the robot CAD model. This virtual model is a digital shadow of the developed SCARA robot.

Further advanced research will include applying the artificial intelligence techniques and information obtained from the fully integrated camera to extract the toolpath of the robot for automatic hybrid robot programming.

ACKNOWLEDGEMENTS

This work has been financially supported by the Ministry of Science, Technological Development and Innovation 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. Javaid, A. Haleem, R.P. Singh and R. Suman, “Substantial capabilities of robotics in enhancing industry 4.0 implementation”, Cognitive Robotics, Vol. 1, pp. 58-75, (2021)

[2] C. Zheng, Y. An, Z. Wang, H. Wu, X. Qin, B. Eynard and Y. Zhang, “Hybrid offline programming method for robotic welding systems”, Robotics and Computer-Integrated Manufacturing, Vol. 73, 102238, (2022)

[3] D. Milutinovic, M. Glavonjic, N. Slavkovic, Z. Dimic, S. Zivanovic, B. Kokotovic and Lj. Tanovic, “Reconfigurable robotic machining system controlled and programmed in a machine tool manner”, International Journal of Advanced Manufacturing Technology, Vol. 53(9-12), pp. 1217-1229, (2011)

[4] N. Slavkovic, S. Zivanovic and D. Milutinovic, “An indirect method of industrial robot programming for machining tasks based on STEP-NC”, International Journal of Computer Integrated Manufacturing, Vol. 32(1), pp. 43–57, (2019)

[5] H. Makino, “Development of the SCARA”, Journal of Robotics and Mechatronics, Vol. 26(1), pp. 5-8, (2014)

[6] B. Momcilovic, “Development of the control system of the 4-axis SCARA robot”, M.Sc. Thesis, University of Belgrade, Faculty of Mechanical Engineering (Serbia), 2022

[7] Dj. Milicevic, “Designing the prototype of the 4-axis SCARA industrial robot”, M.Sc. Thesis, University of Belgrade, Faculty of Mechanical Engineering (Serbia), 2022

[8] J. Fang and W. Li, “Four degrees of freedom SCARA robot kinematics modeling and simulation analysis”, International Journal of Computer, Consumer and Control, Vol. 2(4), pp. 20-27, (2013)

[9] N.R. Slavković, S.T. Živanović and N.M. Vorkapić, “Configuring a virtual prototype of a BiSCARA robot”, Tehnika, Vol. 76(3), pp. 311-317, (2021)

[10] N. Slavkovic, S. Zivanovic, N. Vorkapic and Z. Dimic, “Development of the Programming and Simulation System of 4-axis Robot with Hybrid Kinematic”, FME Transactions, Vol. 50(3), pp. 403-411, (2022)

[11] https://howtomechatronics.com/category/projects/

[12] N. Slavković, N. Vorkapić, S. Živanović, Z. Dimić and B. Kokotović, “Virtual BiSCARA robot integrated with open-architecture control system”, Proceedings of 14th International Scientific Conference “Flexible Technologies MMA 2021”, Novi Sad (Serbia), 23-25 September 2021, pp. 63-66, (2021)