TRIPLE PARALLELOGRAM ROBOTIC EDUCATIONAL MODEL

BOHDAN FETSO[1], MICHAL KELEMEN[1], IVAN VIRGALA[1], ĽUBICA MIKOVA[1], ERIK PRADA[1], MARTIN VARGA[1], PETER JAN SINCAK[1], LEO BRADA[1], ROBERT RAKAY[1]

[1]Technical University of Kosice, Faculty of Mechanical Engineering, Kosice, Slovakia

DOI: 10.17973/MMSJ.2025_10_2025067

michal.kelemen@tuke.sk

The article deals with the design of an educational robot consisting of three parallelogram mechanisms. Such a type of robot automatically ensures the parallelism of the end point position with the initial end point position. This is important, for example, when manipulating loose or liquid materials. A simulation model is solved and subsequently a hardware prototype is created for experimental work by students.

KEYWORDS

Robot, parallelogram, kinematics, simulation, manipulator

1 INTRODUCTION

The aim of this work is to develop a didactic model of a robot and also create a simulation model that can be applied in the educational process. In recent years, the application possibilities of robots have become increasingly wider, and we can encounter them more and more often in everyday areas of our lives. Robotics is becoming a part of processes not only in industry but also in construction, agriculture, restaurants, hospitals, services, but also in households [Kuric 2021, Lestach 2022, Liptak 2018, Mikova 2013, Mikova 2023, Oscadal 2020, Pavlasek 2018].

The educational process of students in study programs focused on automation, mechatronics and robotics is essential to support didactic models to improve the level of knowledge and practical experience and skills [Bezak 2014, Brada 2023, Fetso 2024, Hroncova 2023, Kelemen 2018, Malik 2025, Mikova 2014, Murcinkova 2013, Nguyen 2024, Pivarciova 2016, Romancik 2024, Vagas 2024, Virgala 2012 & 2014a,b].

2 ROBOT TOPOLOGY

The robot is made up of three mechanisms, which are called parallelograms (Fig. 1). A parallelogram mechanism is a four-joint mechanism made up of two pairs of parallel segments, in which both main parts rotate in the same direction and maintain mutual parallelism during movement. It is used, for example, in construction mechanisms of excavators and loaders, where it is necessary to manipulate objects so that they are in a parallel position with the ground (Fig. 1). For example, this involves manipulating loose or liquid materials or this movement is important for some process such as applying glue or paint. Its advantage is the simplicity of construction, stability and automatic maintenance of parallelism with the ground or other base. If it is necessary to create multiple degrees of freedom, it is possible to use several of these parallelogram mechanisms and thus create a structure for manipulation in a larger space (Fig. 1). In this work, the robot is created by combining three parallelogram mechanisms (Fig. 1) and at the end there is a gripper for gripping objects. If the manipulated object is a glass of water, then this type of robot will ensure trouble-free manipulation. The first two degrees of freedom of the planar mechanism are adjusted so that they are driven by servo motors located near the base. This ensures that the robot does not have to carry the weight of the servo motor (Fig. 1).

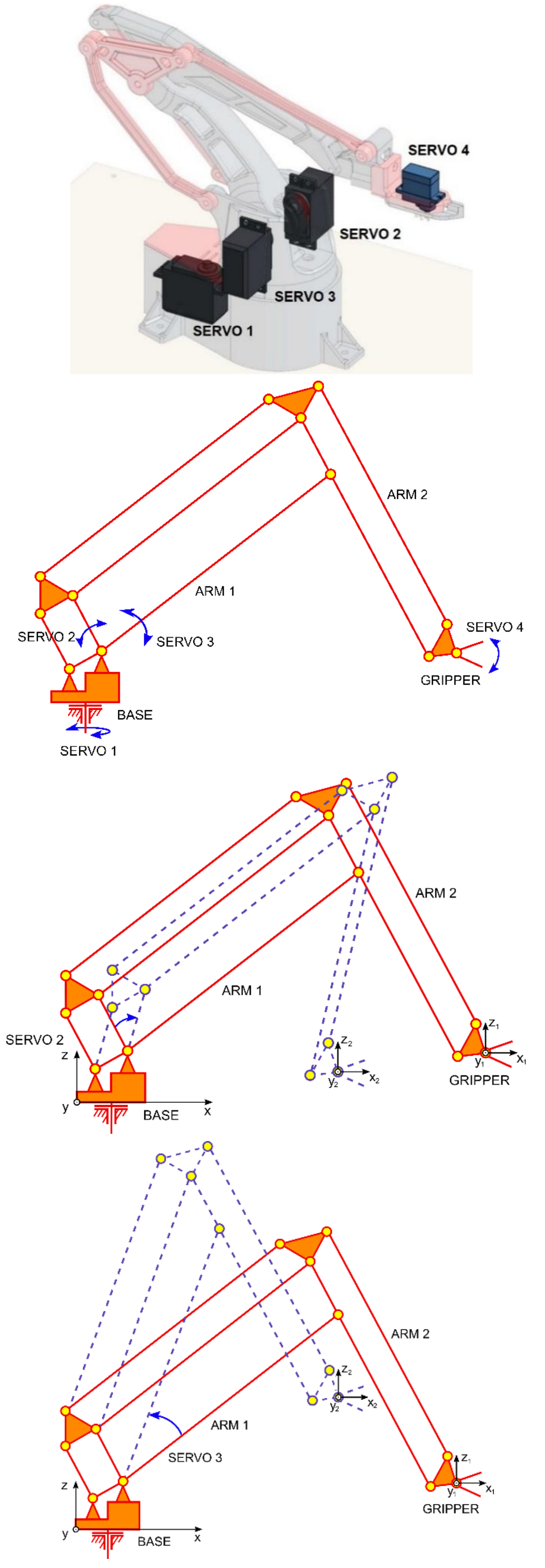

Figure 1. Hardware topology and kinematic scheme of a robotic system

At the end of this robot there is then only a gripper with its own smaller servo motor. So that this robot is not only planar, but also supplemented with another servo motor in the base, which allows the rotation of the robot around a vertical axis. This is how we achieved that this robot is spatial, and its workspace will be larger (Fig. 1).

This triple parallelogram robot is designed to be used as an educational model for students. The effort was to create both a simulation model and a real robot prototype for students' experimental work. Any control system capable of controlling servo drives can be used as a control system. It can be any microcontroller, but after adjusting the voltage levels of the signals, a programmable logic controller can also be used. Servo motors were designed for the drive, which are controlled using a pulse width modulated signal. The task is therefore to transform information about the desired position into information about a pulse width modulated signal, which is sent to the servo motor control units. To achieve the desired position, a signal must be generated for the servo motors. The servo motors used have position feedback and so control takes place at the lowest level. From these servo motors, it is also possible to obtain information about the current position of the output shaft.

3 ROBOT MOTION CONTROL

The control of the robot movement is solved by three activities: interpretation, path planning and trajectory generation. The control sequence is initialized by the operator who defines the required instructions. Programming of this robot can be done using various programming languages such as C or C++ or Python.

The movement of the robot can be along a straight line or along a selected curve. The required trajectory is supplemented with the speed at the individual nodal points of the trajectory, thus determining the time course of the path and the speed of the robot's endpoint formed by the gripper. Subsequently, a movement sequence is created, which the control unit is to execute via servo motor control.

4 ROBOT PATH PLANNER

All entered movements are calculated by the path planner, which determines the desired trajectory - path, i.e., determines the geometric properties of individual robot members in space. The path planner determines the length of the path between two points, the shape of the path, the orientation of the gripper at the target point and other robot settings. The goal of the path planner is to solve the robot movement according to the kinematic model. In the kinematic model, it is possible to calculate the direct and inverse kinematic task.

There are several approaches to path planning:

1. Grid-based planning

The environment is divided into squares or cubes (in 3D) that are marked as free or occupied. Algorithms such as:

- A* (A-star) – finds the shortest path with respect to cost,
- Dijkstra – similar, but without heuristics.

2. Sampling-based planning

Uses random sampling to find the path:

- RRT (Rapidly-Exploring Random Tree) – fast way to plan in complex spaces,
- PRM (Probabilistic Roadmap) – suitable for repeated planning in the same environment.

3. Potential fields

The robot moves like a particle in a force field – it is attracted to the goal and repelled by obstacles.

4. Trajectory optimization

Creates smooth and optimized trajectories (e.g., via methods such as CHOMP, STOMP, TrajOpt), often in combination with kinematics.

5 FORWARD KINEMATICS

Forward kinematics is a fundamental task in robotics, which is determined by determining the position and orientation of an end effector (e.g., a gripper) based on known values of joint variables (rotation angles or displacements of individual robot joints). In other words, when we know the configuration of the robot - that is, how the individual joints are determined (e.g., its angular rotation for rotary joints or longitudinal displacement for linear joints) - we use forward kinematics to calculate where in space the robot's endpoint is located and its orientation. This calculation is usually performed using transformation matrices that describe the relationship between the individual robot joints. The most commonly used is the Denavit-Hartenberg (DH) convention, which allows for the systematic construction of a kinematic model of the robot.

A coordinate system with a specific position and orientation in the workspace is used. In robotics, we distinguish four main frameworks:

- GCS (Global Coordinate system) - The global coordinate system determines the starting point of the whole space or "world". This structure is most useful when there are multiple robots in the workplace so they can know where they are within the framework.
- MCS (Machine Coordinate system) The robot's coordinate system is its local coordinate system, usually located where its stationary base is. The other parts of the robot take this point as a reference for calculating their position.
- TCS (Tool Coordinate system) The coordinate tool system is a frame located at the position of the end effector, of which the instruments can be a part.
- WCS (Workpiece coordinate system) The workpiece coordinate system is the starting point of the workspace in which the robot has to perform a specific task.

It is necessary to perform frame operations in kinematic equations. If the position of a point in a given frame is known and we want to find its position from another frame, we must use several transformations. The first and most straightforward way is linear feed, implemented as a simple
sum with offset. We denote the offset with D = $\{dx, dy, dz\}$, which determines the position of the
second frame relative to the first. The point in the first frame is denoted as P_0 = $[x_0, y_0, z_0]$ and P_1 is the position of P_0 from the second frame. Its calculation looks like this:

$$P_1 = \begin{cases} x_1 = x_0 + dx \\ y_1 = y_0 + dy \\ z_1 = z_0 + dz \end{cases} \tag{1}$$

If the frames rotate with each other, we must use a rotation nut created as a combination of rotations around the individual axes. The rotary nut has three rows and three columns. The following equations will show the shape of the rotating nut in the case of rotation around the individual axes, i.e., around the *x*, *y*, and *z* axes).

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{2}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

If a given operation requires both rotation and translation, as it is the case with most robotic axes, we can use a homogeneous transformation matrix that combines these two operations, see equation (5).

$$T = \begin{bmatrix} R & \Delta \\ 0 & 1 \end{bmatrix} \tag{5}$$

The homogeneous transformation matrix T combines the rotation matrix R and the delta translation (D) into one matrix. For example, the rotating nut in the case of rotation along the z axis would look like this from the following equation, see equation 6.

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

The translation, or delta offset, from equation 7 will be located next to the rotation matrix.

$$\Delta = \begin{cases} \delta_x \\ \delta_y \\ \delta_z \end{cases} \tag{7}$$

The homogeneous transformation matrix looks as follows (equation 8).

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & \delta_x \\ \sin\theta & \cos\theta & 0 & \delta_y \\ 0 & 0 & 1 & \delta_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

Combining these two matrices creates one matrix with four rows and four columns with zeros under the rotation matrix and the unit under translational displacement. Combining these two operations can perform both translation and frame rotation simultaneously. It is a simple multiplication of the current position P_0 with the homogeneous matrix T. The result is the desired position and orientation P_1.

$$P_1 = P_0 T \tag{9}$$

Returning to the introductory issue of positioning, where the position of X, Y, Z and the orientation A, B, C of the end effector are based on the position of the joints (J_1, J_2, … J_n) must be found utilising a direct transformation (Fig 2). This calculation is performed using a simple transformation between the two frames. It transforms from the basic framework, typically the MCS, to the end effector framework. What is needed to use here is a very general rule for solving any serial kinematic chain, no matter how many joints we have. The final transformation matrix represents a homogeneous transformation and depends on all the robot's joint positions and mechanical parameters. This matrix is accomplished by building a chain of frames, translations, and rotations from base to TCP. The endpoint P_1 can then be expressed as a multiple of point P_0 with a homogeneous frame transformation.

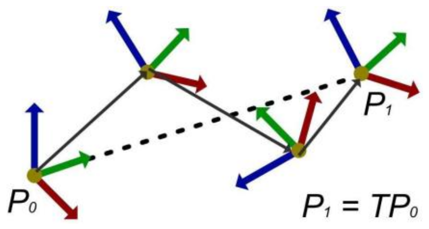

Figure 2. Chain of transformation frames

All rotations are caused by movements from the first joint to the n-joint in the system. These are not constant over time; they vary according to the movements of the motors. However, unlike the rotation parameters, the translation parameters are, in fact, constant because the mechanical properties of the robot do not change.

6 INVERSE KINEMATICS

Inverse Kinematics is a task in robotics that solves the opposite problem of direct kinematics [Trojanova 2021]. Instead of determining where the end effector is based on joint variables, we know the desired position and orientation of the end effector and want to find out what joint settings are needed to get the robot there (Fig. 3).

This is a very important task, for example, when controlling a robot in industrial production, where we need the end tool to reach the exact location - for example, when assembling, welding or manipulating objects.

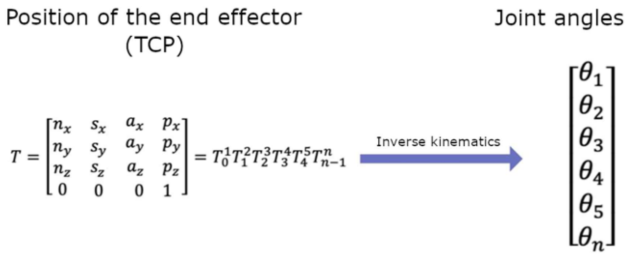
Position of the end effector (TCP)
Joint angles
$T = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_{n-1}^n$
Inverse kinematics
$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_n \end{bmatrix}$

Figure 3. Principle of inverse kinematics

The solution of inverse kinematics consists of solving a set of (often nonlinear) equations that express the relationship between the desired position/orientation and the joint variables. This problem is generally more challenging than direct kinematics because:

- it may have multiple solutions (the robot can reach the same position in multiple ways),
- some positions may be unreachable (outside the robot's workspace),
- there may be no solution if the goal is not physically feasible.

The position of the end effector is declared by a homogeneous matrix, which expresses its position from the robot's base. The angles of the individual joints (J_1, J_2, … J_n) are calculated using inverse kinematics.

We can always find out the current position of the axes from the motors, so the goal of inverse
kinematics is to work with future values based on where we want to move TCP. The robots are programmed by the operator in XYZ coordinates, which are known, but the motion software directly controls the motors that move the robot's joints. We need to determine how to calculate verbal commands from the route. It is similar to moving a human hand. If a person wants to catch something, he sees the target position in space but does not know the required angles of his joints (from the shoulder through the elbow to the wrist). For serial kinematics, the problem of solving inverse transformations is usually much more complex than direct transformations, but for most parallel kinematics robots, it is a less complex problem (this will be discussed in detail below).

There is no general approach to calculating the joint axis with the inverse kinematic problem. It depends on the mechanical structure of the robot, but there may be situations where there is no solution for a given TCP position, or there is more than one. In any case, the effort is to minimise the necessary movement and find the optimal solution for moving TCP to the most accurate possible position.

The end effector (TCP) coordinates are expressed concerning the basic framework. The joint calculation also starts from the first joint J1 to Jn. Every joint angle calculation is unique. Each calculation must consider the geometric structure of the robot, in which directions the joint can be bent and how its movement will affect the TCP position. The result of the calculation of one joint is the angular deviation from the zero position of the motor. In most cases, this task can be simplified to a two-dimensional task when calculating the axis, finding a triangle between the various joints. The following example shows how to solve the inverse kinematics problem for a simple robot model with only two joints (see figure 4). This issue will show the general principle of calculating joints using inverse kinematics.

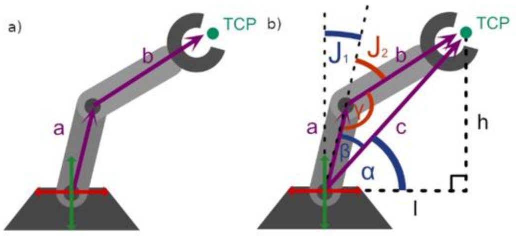

Figure 4. a) Simple robotic arm; b) Inverse kinematics robotic arm

7 GAZEBO SIMULATION

Simulation is a crucial phase in the design of any robot. It allows us to quickly test new designs and see how this implementation fits the design before starting the complicated work of designing a real robot. It can be risky for robots if the designer has designed a model that has not been tested in the simulation. Incorrect parameters can cause unpredictable behaviour of the robot.

Using the ROS control and simulation software, various software platforms have been developed to study spatial serial and parallel robots. Examples of such existing platforms are MATLAB/Simulink and ZeroSim. These platforms are flexible for the possibility of adding their serial and parallel robots of various types, and it is possible to extend them with other algorithms. However, Gazebo is the ROS-integrated simulation platform for studying any type of robot. It allows creating and analysing of any parallel robot with the ability to customise algorithms.

With Gazebo, it is possible to use different simulation engines. ODE is the default. It also supports BULLET but not all features currently. Therefore, using BULLET as an extension for Gazebo is better for kinematic calculation and simulation (Fig. 5).

The current version of the platform includes analytical tools for each of the following areas:

- Dynamics and control
- Motion and sensor control
- Kinematics
- Direct kinematics
- Inverse kinematics
- Workspace analysis
- Design optimisation

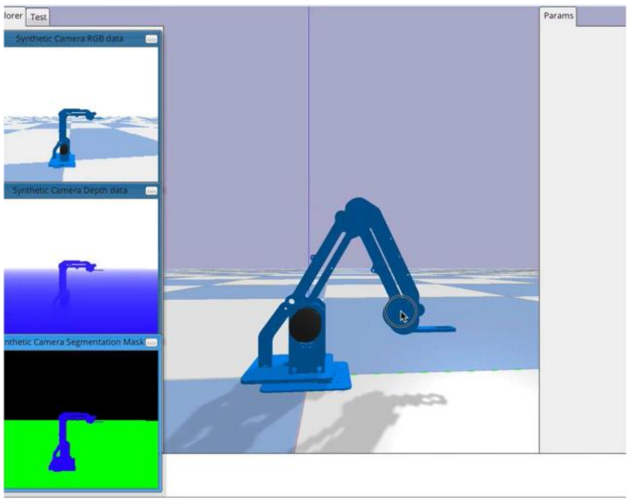
lorer
Test
Params
Synthetic Camera RGB data
Synthetic Camera Depth data
nthetic Camera Segmentation Mask

Figure 5. Gazebo simulator graphical interface

A new parallel robot model can be added to Gazebo by specifying it in semi-structured XML data files. There are four main files, robot.urdf, config.xacro, spawn.launch and empty.world. Robot.urdf defines the robot body, links, and joints. Config.xacro contains a set of link layouts, connection points of joints, and properties. Spawn.launch is a set of how files must load in Gazebo and Rviz, set properties for corresponding features of the robot through the files mentioned earlier and apply any additional configuration files. Finally, empty.world is a world that surrounds the robot. It is required for the robot to spawn on the ground underneath instead of endlessly falling in the simulation. The simulation can have more results. The most interesting result that can be used for hardware implementation is the generated collision boxes of the robot links. It allows to properly model robot parts and avoid possible jamming. Orientation of each DOF is generated using inverse kinematics. The simulator calculates the circular degree of its joints for each robot's position during the execution of the trajectory and saves them in a table in chronological order.

The Gazebo is primarily a simulation platform and does not offer a hardware implementation but leaves room for it to be extended. It offers source code methods and protocols developers can use to utilise simulation results for actual deployment to an existing robot. ROS is one of the best existing means of connecting robotic hardware, which provides a well-supported interface for expansion and integration with Gazebo. In this way, it is convenient to set up easy-to-use robotic hardware and take advantage of the flexibility and robustness of Gazebo.

8 ROBOT PROTOTYPE

According to the simulation results, a physical model of the robot was designed and constructed for didactic purposes (Fig. 6).

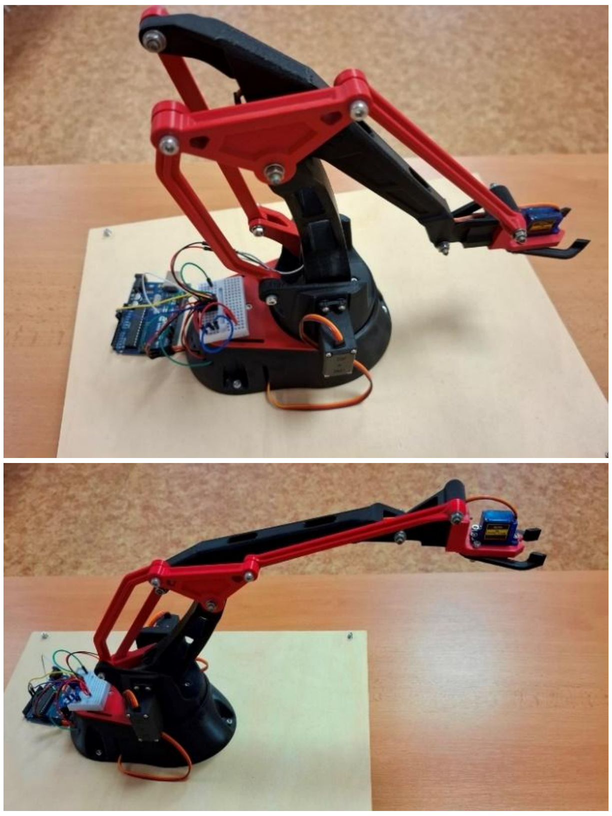

Figure 6. Robot prototype

The dimensions of the robot were designed so that the robot was easily portable and usable for experimental work by students. The dimensions of the robot are 310×124×257 mm and the maximum load capacity at the end point of the robot gripper is 500 grams. However, the gripper itself is not adapted to such a weight of the manipulated object and so the weight of the manipulated object is a maximum of 100 grams.

Servomotors 2 and 3 perform arm movements and thus must carry the largest load. The servomotors are mounted in the lowest part of the robot (Fig. 7). Commercial robots almost always have servomotors in the joint where they perform rotation.

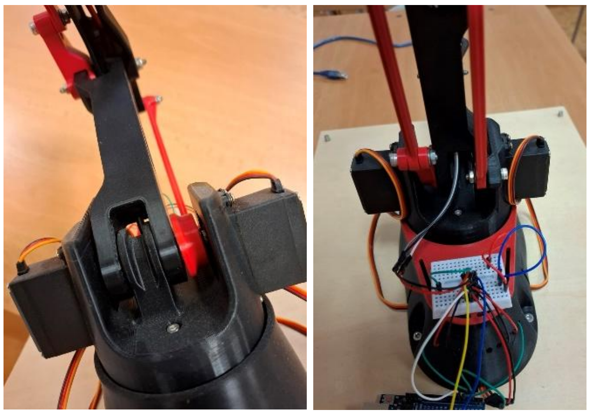
Figure 7. Attaching servo motor 2 and 3

For the first experiments, an Arduino Uno microcontroller was used as the control system. This low-cost platform allows for relatively simple control of this robot. A gripper is designed at the end point of the robot for gripping an object (Fig. 8).

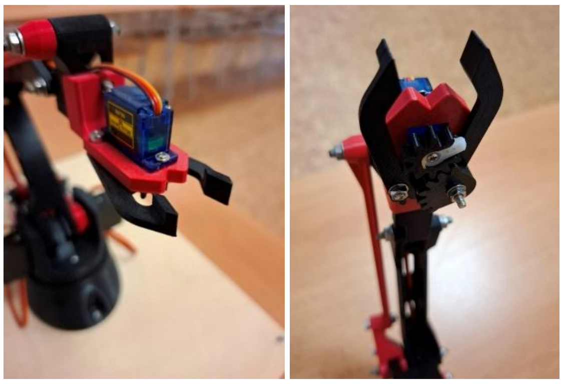
Figure 8. Robot gripper

The mechanical parts of the robot are created by 3D printing and commonly available servo motors for radio-controlled models are used. These are low-cost actuators that are controlled using a PWM signal.

9 CONCLUSIONS

Education of future industry professionals is a key condition for achieving quality products and services. The educational process for students must therefore meet the requirements of practice, so that students have important competencies such as creativity, innovation, creativity, analysis, synthesis, etc. Practical models of real devices should provide students with the opportunity to acquire these skills.

Similar tasks are also solved at other workplaces in various other areas [Bozek 2012, Bratan 2023, Kelemen 2012, Kelemen 2014, Koniar 2014, Saga 2018, Saga 2020, Tlach 2019, Vagas 2025, Zidek 2018].

ACKNOWLEDGMENTS

The authors would like to thank the Slovak Grant Agency - project VEGA 1/0436/22, KEGA 008TUKE-4/2024, VEGA 1/0409/25 and KEGA 016TUKE-4/2025.

REFERENCES

[Bezak 2014] Bezak, P., et al. Advanced Robotic Grasping System Using Deep Learning. Procedia Engineering, 2014, Vol. 96, pp. 10-20. https://doi.org/10.1016/j.proeng.2014.12.092.

[Bozek 2012] Bozek, P., Pivarciova, E. Registration of Holographic Images Based on Integral Transformation. Computing and Informatics, 2012, Vol. 31, No. 6, pp. 1369-1383.

[Brada 2023] Brada, L., et al. Conducting an Examination of the Trajectory and Workspace of the Manipulator within the Matlab Environment. AD Alta-Journal of Interdisciplinary Research, 2023, Vol. 13, Issue 2, pp. 353-356. https://www.doi.org/10.33543/1302.

[Bratan 2023] Bratan, S., Sagova, Z., Saga, M., Yakimovich, B., Kuric, I. New Calculation Methodology of the Operations Number of Cold Rolling Rolls Fine Grinding. Applied Sciences, 2023, Vol. 13, No. 6. DOI: 10.3390/app13063484.

[Fetso 2024] Fetso, B., et al. Educational Model of the Robot. MM Science Journal, 2024, No. November, pp. 7764-7771. DOI: 10.17973/MMSJ.2024_11_2024053.

[Hroncova 2023] Hroncova, D., et al. Inverse and Forward Kinematics and Dynamics of a Two Link Robot Arm. MM Science Journal, 2023, No. December, pp. 7085-7092. DOI: 10.17973/MMSJ.2023_12_2023067.

[Kelemen 2012] Kelemen, M., et al. Design and Development of Lift Didactic Model within Subjects of Mechatronics. Procedia Engineering, 2012, Vol. 48, pp. 280-286. DOI: 10.1016/J.PROENG.2012.09.515.

[Kelemen 2014] Kelemen, M., et al. Rapid Control Prototyping of Embedded Systems Based on Microcontroller. Procedia Engineering, 2014, Vol. 96, Issue 11, pp. 215-220. doi.org/10.1016/j.proeng.2014.12.146.

[Kelemen 2018] Kelemen, M., et al. A Novel Approach for a Inverse Kinematics Solution of a Redundant Manipulator. Applied Sciences, 2018, Vol. 8, Issue 11. https://doi.org/10.3390/app8112229.

[Koniar 2014] Koniar, D., et al. Virtual Instrumentation for Visual Inspection in Mechatronic Applications. Procedia Engineering, 2014, Vol. 96, pp. 227-234. DOI: 10.1016/j.proeng.2014.12.148.

[Kuric 2021] Kuric, I., et al. Analysis of Diagnostic Methods and Energy of Production Systems Drives. Processes, 2021, Vol. 9, 843. doi.org/10.3390/pr9050843.

[Lestach 2022] Lestach, L., et al. Two-legged Robot Concepts. MM Science Journal, 2022, No. October, pp. 5812-5818. DOI: 10.17973/MMSJ.2022_10_2022091.

[Liptak 2018] Liptak, T., et al. Modeling and control of two-link snake. International Journal of Advanced Robotic Systems, 2018, Vol. 15, Issue 2. DOI: 10.1177/1729881418760638.

[Malik 2025] Malik, M., et al. Optimization of a Robotic Cell in the Roboguide Environment. MM Science Journal, 2025, No. June, pp. 8276-8281. DOI: 10.17973/MMSJ.2025_06_2025021.

[Mikova 2014] Mikova, L., et al. Simulation Model of Manipulator for Model Based Design. Applied Mechanics and Materials, 2014, Vol. 611, No. 1, pp. 175-182. https://doi.org/10.4028/www.scientific.net/AMM.611.175.

[Mikova 2013] Mikova, L., et al. Concept of Locomotion Mobile Undercarriage Structure Control for the Path Tracking. Solid State Phenomena, 2013, Vol. 198, pp. 79-83. DOI: 10.4028/www.scientific.net/SSP.198.79.

[Mikova 2023] Mikova, L. et al. Non-Minimum Phase Systems. MM Science Journal, 2023, No. December, pp. 7143-7147. DOI: 10.17973/MMSJ.2023_12_2023133.

[Murcinkova 2013] Murcinkova, Z., Krenicky, T. Applications utilizing the damping of composite microstructures for mechanisms of production machines and manipulator devices. In: SGEM 2013: 13th Int. Multidisciplinary Sci. Geoconf., Vol. 1; 16-22 June 2013, Albena, Bulgaria. Sofia: STEF92 Technology, 2013, pp. 23-30. ISBN 978-954-91818-9-0.

[Nguyen 2024] Nguyen, T.P., et al. Design and implementation of a field robot using a parallelogram mechanism. Science Progress, 2024, Vol. 107, No. 4. DOI: 10.1177/00368504241291124.

[Oscadal 2020] Oscadal, P., et al. Improved Pose Estimation of Aruco Tags Using a Novel 3D Placement Strategy. Sensors, 2020, Vol. 20, Issue 17. DOI: 10.3390/S20174825.

[Pavlasek 2018] Pavlasek, P., Hargas, L., Koniar, et al. Flexible Education Environment: Learning Style Insights To Increase Engineering Students Key Competences. In: 10th Int. Conf. on Education and New Learning Technologies, 2-4 July 2018, Palma, Spain, 2018, pp. 10156-10165. DOI: 10.21125/edulearn.2018.2468.

[Pivarciova 2016] Pivarciova, E., Csongrady, T. Tracer robot with a proportional control. MM Science Journal, 2016, No. November, pp. 1277-1286. DOI: 10.17973/MMSJ.2016_11_201690.

[Romancik 2024] Romancik, J., et al. Design, Implementation, And Testing of a 3D printed Gripper Actuated by Nitinol Springs. MM Science Journal, 2024, No. June, pp. 7352-7356. DOI: 10.17973/MMSJ.2024_06_2024009.

[Saga 2018] Saga, M., et al. Effective algorithm for structural optimization subjected to fatigue damage and random excitation. Scientific Journal of Silesian University of Technology - Series Transport, 2018, Vol. 99, pp. 149-161. DOI: 10.20858/sjsutst.2018.99.14.

[Saga 2020] Saga, M., et al. Case study: Performance analysis and development of robotized screwing application with integrated vision sensing system for automotive industry. International Journal of Advanced Robotic Systems, 2020, Vol. 17, No. 3. https://doi.org/10.1177/1729881420923997.

[Tlach 2019] Tlach, V., et al. Collaborative assembly task realization using selected type of a human-robot interaction. Transportation Research Procedia, 2019, Vol. 40, pp. 541-547. https://doi.org/10.1016/j.trpro.2019.07.078.

[Trojanova 2021] Trojanova, M., Cakurda, T., Hosovsky, A., Krenicky, T. Estimation of Grey-Box Dynamic Model of 2-DOF Pneumatic Actuator Robotic Arm Using Gravity Tests. Applied Sciences, 2021, Vol. 11, No. 10, Art. No. 4490.

[Vagas 2024] Vagas, M., et al. Implementation of IO-Link Technology Into The Handling and Sorting Sub-Station of the Festo FMS 500 Automated Line. MM Science Journal, 2024, No. June, pp. 7348-7351. DOI: 10.17973/MMSJ.2024_06_2024008.

[Vagas 2025] Vagas, M., et al. Data Processing Approach Based on OPC UA Architecture Implementation and Bluebird Platform. IEEE Access, 2025, Vol. 13, pp. 51069-51084. DOI: 10.1109/ACCESS.2025.3552976.

[Virgala 2012] Virgala, I., et al. Manipulator End-Effector Position Control. Procedia Engineering, 2012, Vol. 48, pp. 684-692. DOI: 10.1016/j.proeng.2012.09.571.

[Virgala 2014a] Virgala, I., et al. Analyzing, Modeling and Simulation of Humanoid Robot Hand Motion. Procedia Engineering, 2014, Vol. 611, pp. 75-82. https://doi.org/10.1016/j.proeng.2014.12.121.

[Virgala 2014b] Virgala, I., et al. Inverse Kinematic Model of Humanoid Robot Hand. Applied Mechanics and Materials, 2014, Vol. 96, pp. 489-499. DOI: 10.4028/www.scientific.net/AMM.611.75.

[Zidek 2018] Zidek, K., et al. Auxiliary Device for Accurate Measurement by the Smartvision System. MM Science Journal, 2018, No. March, pp. 2136-2139. DOI: 10.17973/MMSJ.2018_03_201722.

CONTACTS:

Michal Kelemen, Prof. Ing. PhD.
Technical University of Kosice, Faculty of Mechanical Engineering
Institute of Automation, Mechatronics, Robotics and Production Techniques
Letna 9, 04200 Kosice, Slovak Republic
michal.kelemen@tuke.sk