Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2025, 19(6), 330–341
https://doi.org/10.12913/22998624/202594
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.01.21
Accepted: 2025.04.15
Published: 2025.05.01

A hardware-software compatibility in robotic cyber-physical systems – an application based approach

Paweł Penar[1], Mateusz Szeremeta[1], Arkadiusz Gola[2*]

[1] Department of Applied Mechanics and Robotics, Rzeszow University of Technology, Aleja Powstańców Warszawy 8, 35-029 Rzeszów, Poland
[2] Faculty of Mechanical Engineering, Lublin University of Technology, ul. Nadbystrzycka 36, 20-618 Lublin, Poland
* Corresponding author's e-mail: a.gola@pollub.pl

ABSTRACT

One of the key technologies of Industry 4.0 and Industry 5.0 are cyber-physical systems (CPS), and the construction of CPSs remains the domain of various fields of engineering - mechanics, control theory or computer science. The communication problem occurring during the implementation of control of single CPS objects may concern the exchange of data within the CPS or between different objects. In this area, problems of software and hardware compatibility appear. This article presents a project of a modular application for the exchange of process data between mechatronic devices, such as robots, which is a solution to this problem. The solution presented in the paper is illustrated on the example of communication between a mobile wheeled robot with Macanum wheels and a quadruped robot. It was assumed that the control of the wheeled robot would be implemented from the perspective of realizing the basic robotics task of driving to the target, while the quadruped robot, constituting a moving target, together with the motion capture system, constituted a CPS whose outputs are incompatible in the software-hardware sense. By selecting this example, special attention was paid to the importance of multidisciplinarity in the context of the communication problem and also provided an application that allows for a modular solution to the software-hardware compatibility problem.

Keywords: cyber physical systems, robot communication, wheeled mobile robots, motion capture.

INTRODUCTION

While the development of industry in the 20th century was marked by breakthrough discoveries such as the steam engine, electricity and numerical control of technological machines, the beginning of the 21st century will be associated with cyber-physical systems and robotization and digitization of production [1]. An important element in this matter was the presentation in 2011 at the Hannover Fair of the key challenges regarding the development of industry in the 21st century, which were referred to as Industry 4.0. and although in 2021 the European Commission "corrected" these assumptions by putting humans at the center of production systems and emphasizing the importance of sustainable development and stability of supply chains, all of the paradigms of Industry 4.0 remain relevant, and importantly, they emphasize the relevance of correct and effective communication in machine-machine, machine-robot and robot-robot systems [2, 3]. Although the concept of Industry 4.0 has become widely used, there is still no agreed definition of this term, which poses serious limitations in building the theory of Industry 4.0 and comparing research in this area [4]. As part of the literature review conducted by Coulot et al., where almost 100 definitions of Industry 4.0 and related concepts were analyzed, similarities and differences between the definitions were indicated based on the adopted categories [5]. The multitude of definitions - although on the one hand it may be problematic, on the other hand it shows how current, important and widely discussed this

topic is in the area of emerging scientific work. The authors themselves indicate that their work is meant to be a contribution to further discussion and the adopted categorization is meant to organize future research on the approach to Industry 4.0 in its many aspects [5]. When analyzing the assumptions of Industry 4.0, it should be clearly stated that it is characterized by highly developed automation processes, in which the achievements of electronics and computer science were used [6]. From the perspective of production and service management, Industry 4.0 focuses on the creation of intelligent and communicating systems, such as machine-to-machine (M2M) and human-machine interaction (HMI), capable of communicating with other intelligent and distributed systems. Nine pillars of Industry 4.0 are described in the literature. These are: Industrial Internet of Things (IIoT), cloud computing, Big Data, simulations, augmented reality, additive manufacturing, horizontal and vertical system integration, autonomous robots and cybersecurity [7–10]. In particular, the concept of Industry 4.0 focuses on cyber-physical systems, which are a combination of embedded systems and physical objects [11, 12]. Embedded (computational) systems control physical processes, usually with feedback loops, in which physical processes affect computation and vice versa [13, 14]. The paper [13] presents the concepts and characteristics of CPS and further research perspectives in this area are indicated. The authors indicate that research on CPS is only just beginning and has a multidisciplinary character. This is due to the fact that CPS are heterogeneous systems, without a unified global model. Therefore, research on CPS is conducted by experts from various fields, focusing on system architecture, information processing and software design. This is confirmed, among others, in the work [14], which discusses what models should be used in CPS, since they combine various scientific and engineering disciplines. Their number will expand to include disciplines that study the environmental and social aspects of applying Industry 4.0, which, as it develops, emphasizes the human aspect, making it Industry 5.0 [15]. An important element is also the fact that CPS are an integral system with the IoT, and enable the creation of a network of cyber-physical CPS systems, one of which is perceived in the area of mechatronics, and the other in the area of computing.

In line with the research on Industry 4.0 and its implementation in the form of CPS, this work focuses on the issues of communication between hardware and software incompatible CPS components and between CPSs themselves. The aim of the paper is to create an application that solves the software-hardware compatibility problem by showing its use in the field of robotics, highlighting the multidisciplinary nature of the proposed solution.

In particular, the article presents an original, modular application for data processing and an example of its use in mobile robotics - in the task of reaching a moving target, which is another robot. Communication between CPSs, i.e. a mobile robot and a quadruped robot, is realized using incompatible protocols and a motion capture system, which provides the quadruped robot's positions and also tracks (in order to verify the odometric measurement) the mobile robot's positions. The next section will present the assumptions of the proposed application and selected technical details. Its use in an environment consisting of two robots will constitute the content of the next section. The article will end with a presentation of the results of the conducted research and a summary.

COMMUNICATION COMPATIBILITY

The main technical factors that underpin Industry 4.0 or are directly related to it come from information and communication technologies [16,17]. These technologies enable communication between CPSs and between CPS components. In Figure 1(a) shows a diagram of a simple

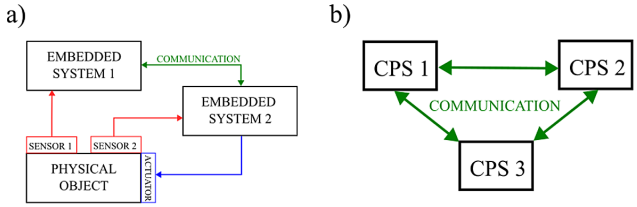

Figure 1. CPS structure (a); communication between the three CPSs (b)

CPS composed of two embedded systems that interact with a physical object via sensors and actuators and with each other via a specific communication method.

Figure 1(b) schematically shows the three CPSs interacting with each other. When communicating within a CPS structure as well as between them, a hardware-software compatibility problem may arise. The proposed application is intended to act as an agent, by introducing a hardware-software solution that will agree on the communication within or between the CPSs. Such a situation is shown in Figures 2(a) and 2(b) for communication between components of a CPS and between several CPSs.

The proposed approach to implement an agent is an application with an input-output architecture, as it is assumed that the agent can transfer data from multiple inputs to a single output. The adoption of such an architecture determines the unidirectional nature of the communication. However, this problem is eliminated by another assumption about the agent: inputs and outputs are to be configurable on the basis of interchangeable hardware-software modules. Bidirectional communication then requires only a duplication of the number of agents operating unidirectionally.

According to the adopted assumptions, the proposed agent should meet several objectives:

- it should be capable of transferring data between different interfaces in the hardware/software sense in one direction,
- should be able to support various protocols, through software and hardware modules,
- should be configurable, i.e. it allows the type of input and output to be specified,
- should allow the transmission of data from multiple inputs in a sequential or parallel manner,
- should ensure the time independence of the handling of a given input and output,
- should be multiplatform.Given the assumptions made, the following sections describe selected implementation details of the proposed agent, RoboDataLink.py, and an example of its use.

AGENT – ROBODATALINK.PY APPLICATION

As indicated earlier, communication between different systems or their components is one of the challenges in the application of the Industry 4.0 concept, of which CPS is a key solution.

As a first step in the development of an application connecting incompatible devices, LeicaConnector.py was used to communicate with the Leica laser tracker. As shown in it is possible to connect the Leica AT960 laser tracker to any device that communicates over Ethernet and does not have direct support for the tracker API. The program, which was realized in Python, had two threads. One of them communicated with the tracker based on the API provided by the manufacturer, while the other distributed measurement data based on the TCP/IP standard. Data exchange between threads was based on a shared buffer, implemented as an array.

The intermediary application described here, which has been given the name RoboDataLink.py, is a generalization of the LeicaConnector.py program. As the name suggests, it was written in Python due to the advantages of this language, i.e. popularity, multi-platform and wide availability of libraries. For the same reasons, the [18] paper's authors presented REDUCE, a Python module designed to minimize inconsistencies in multiplicative pairwise comparisons (PC), a fundamental technique in multi-criteria decision making (MCDM).

RoboDataLink.py (source code is available: github.com/ppenar/robo-data-link) assumes (Figure 3) that there can be many different inputs distributing data, the configuration of which is variable. From a program perspective, each is

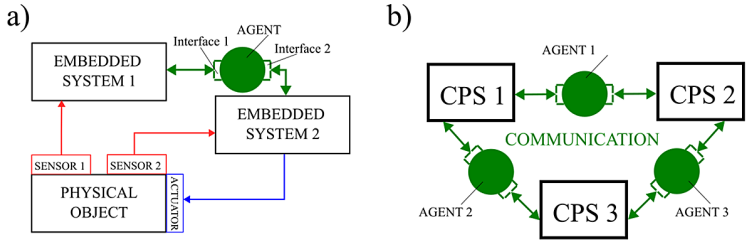

Figure 2. Structure of a CPS with an agent (a); communication between three CPSs with agents (b)

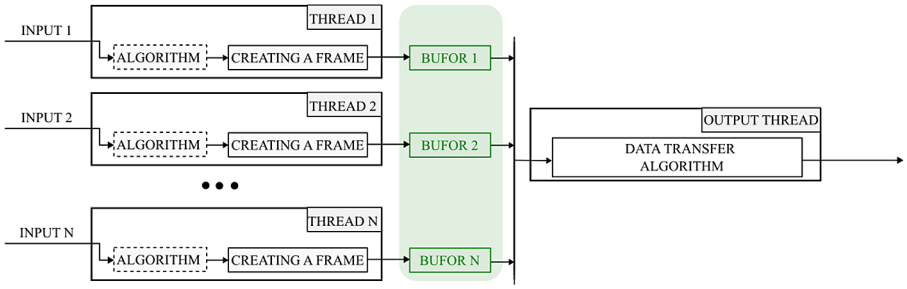

Figure 3. Schematic diagram of the RoboDataLink.py program

an independent module (programmatically they are threads) communicating with a physical device/program/data source. A module is created by implementing specific methods in a class whose instance is instantiated as a thread. The implementation of the module stores the data to be passed to the output in a buffer, which is implemented using an array of numbers of type int16. It is a shared element between input and output and defines the data frame. The output is another thread. It should be noted that the implementation of the input module may include an algorithm for processing the data before writing it to the shared buffer.

This approach to the application architecture allows for time independence of each input and between inputs and outputs, making it possible to integrate data exchange standards operating at different speeds.

Like inputs, the output of RoboDataLink.py can support different communication standards. This requires the implementation of specific methods in the class whose instance is run as a thread (Figure 3). In addition, the output must specify how to retrieve data from the buffer, which can be passed to the output sequentially or in parallel by combining buffers (frames) from multiple inputs.

So far, several input modules have been implemented, i.e. to support the Motion Capture Vicon system, the Lidar RPLIDAR S1 and a test module that generates random numbers. The data received by the input modules (after optional processing) is stored in a buffer. This is necessary due to the time independence of the inputs and outputs, but this approach can lead to delays in updating the RoboDataLink output.

The output from the RoboDataLink application, can be implemented based on serial communication in the RS232 standard or the Ethernet standard. The choice of the output type and its configuration is also specified in the configuration file.

TESTS OF THE ROBODATALINK.PY AGENT

The program described in the previous section, which acts as an agent for communication between incompatible CPSs, was tested in a laboratory environment. The tested system consisted of two CPSs (Fig. 4). The first is the four-wheeled mobile robot Panther with Mecanum wheels from Husarion [19] (physical object) together with a

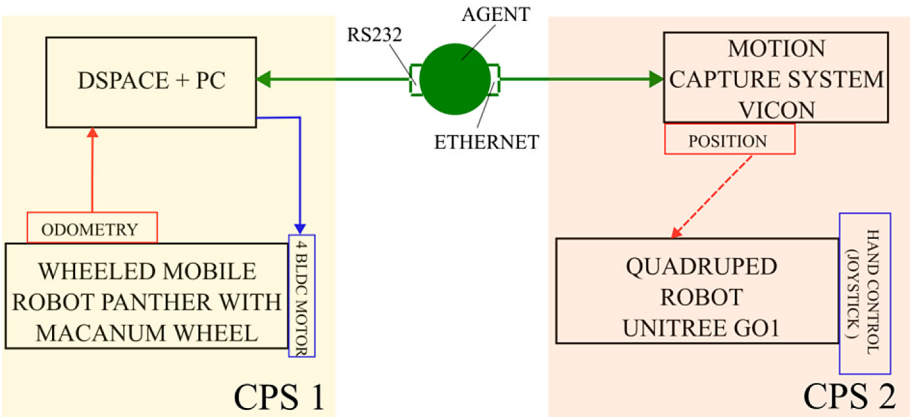

Figure 4. Schematic of test bench

Dspace DS1103 control and measurement card (embedded system) operated by a computer PC. The second CPS system is a Unitree GO1 quadruped robot (physical system) together with a motion capture system from VICON determining the robot's positions.

Research stand

The Panther robot (Figure 5a) is equipped with four Mecanum wheels with a radius of 85 [mm]. The dimensions of the robot are: 805 × 840 × 290 mm (length and width of the robot and height of the platform). The weight of the vehicle is 55 kg and its maximum payload is 80 kg. The robot has four BLDC 80PMB800K.80RBL motors (rated at 473 W each) with planetary gears and incremental encoders. The vehicle includes an internal control computer: Raspberry Pi 4B with Broadcom BCM2711 processor and a power source in the form of 36 V lithium-ion batteries. The maximum speed of the robot is: 2 m/s and the torque rating of each drive module is 34.5 Nm.

For this experiment, a Dspace DS1103 control and measurement card was used to control the wheeled robot. The card allows the measurement and acquisition of data coming from encoders and distributed via the RS232 standard, and the real-time generation of control signals for the motors (PWM signal) that drive the Mecanum wheels. The control algorithm is generated in the Matlab/Simulink environment. The program is then compiled to the level of optimized code in C using the RTW (Real Time Workshop) package. Its implementation into the Dspace board is made possible using the RTI (Real Time Interface) package [20].

The Unitree GO1 quadruped robot (Figure 5b) has 12 degrees of freedom (consisting of 12 servos). The manufacturer indicates that the robot has the ability to control the position of each joint. This makes it possible to realize force control of the whole robot [21]. The GO1 control itself can be realized manually using the supplied joystick or programmatically, using an API in C or Python language [22]. The program that implements the robot's movement is run on an external PC or on one of the embedded systems. These include a Raspberry PI computer and three Nvidia Jetson chips. The latter are connected to five cameras that form part of the robot's sensorics. Software access to the images from the cameras can be realized through a provided API. The same software layer allows the use of other sensorics components: joint-related encoders, an IMU chip and distance sensors.

The Vicon motion capture measurement system (Figure 6) used in the research allows accurate measurement of the tracked object based on a set of cameras tracking the position of markers placed on the tracked object. The entire system is equipped with 10 Vicon Vero v2.2 cameras, and it is not necessary for all of them to be used to carry out the measurements [23]. Each camera has a Vicon 6–12 mm zoom lens, the camera resolution is 2.2 MP (2048 × 1088) and the frame rate is 330 Hz. The minimum viewing angle of the camera for the telephoto lens is: 44.1° × 22.6° (horizontal to vertical), and for a wide-angle lens: 98.1° × 50.1°. The camera latency is 3.6 ms and the weight is approximately 0.5 kg. The cameras are equipped with an accelerometer to support the calibration process and a temperature sensor. Connectivity to the cameras is possible using an RJ45 connector.

The signals from all cameras go to the PoE Switch, which is connected to the computer. Then, using the Tracker 3.9 program, it is possible to track, in real time, the position of the object with an accuracy of a fraction of a millimeter. As can be seen from the diagram in Figure 4, the agent in the example under consideration enables

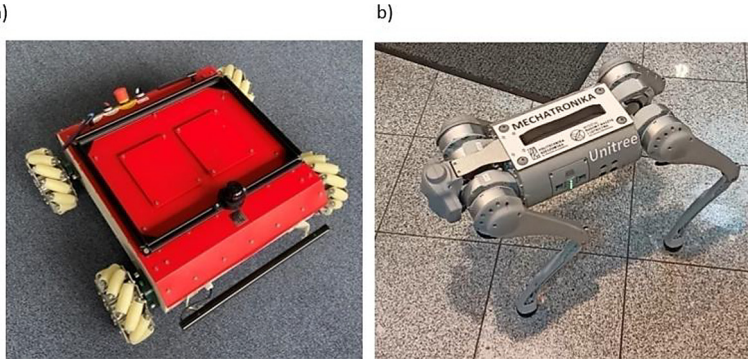

Figure 5. Mobile Robot Husarion Panther with Mecanum wheels (a); Unitree GO1 robot (b)

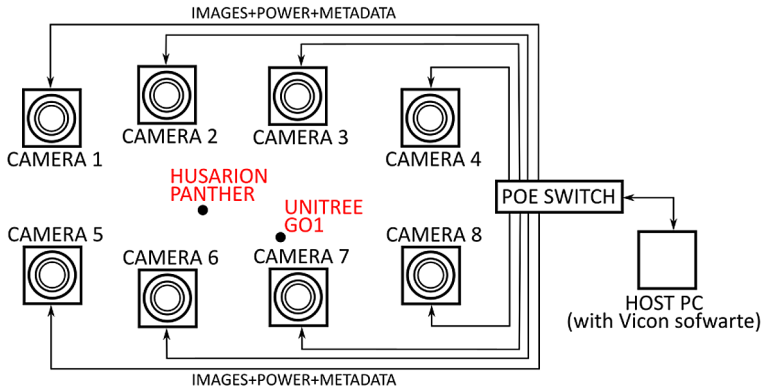

Figure 6. Vicon motion capture system scheme

the transmission of object position data tracked by Vicon's motion capture system. In this configuration (Figure 7a), the input to the agent is the motion capture system and the thread associated with it processes the data from Vicon based on the UDP standard. The Vicon system tracked the position of the quadruped robot and the mobile robot (this measurement was only used to verify the measurement from odometry). In the Tracker program, dictated to the Vicon system, the IDs of the tracked robots were specified as 1pies and 2hus. The output of the RoboDataLink.py program is a frame that is transmitted via the RS232 standard to the Dspace card. Its structure is shown in Figure 7(b). In the first three fields of the frame, the translation (in millimeters) of the object relative to the underlying coordinate system is transmitted. The next three elements of the frame are the rotation (in radians) multiplied by one hundred and projected to ensure accuracy when projected onto integers. It should be noted that one frame contains position and orientation information for only one object, which means that the position information of the robots was delivered to the Dspace measurement card alternately. Which robot the frame refers to is derived from the values of half 7 and 8 of the frame, i.e. 100+ID, where ID corresponds to the digit preceding the name of the tracked robot. e.g. for object 1dog, the value of fields 7 and 8 is 101. The last field of the frame is the checksum calculated as the arithmetic mean of the translations projected to integer values (i.e. round((transX+transY+transZ)/3)).

As the processing times of the input and output threads (implemented programmatically as a delay function in an infinite loop) can be different, the link between the aforementioned threads is the buffer into which the frame is written.

Control algorithm for a mobile wheeled robot

The control algorithm of the Mecanum mobile wheeled robot performed the task of following a moving target, which was a quadruped mobile robot. In robotics, this task is called following a moving target.

In the literature, this task is often implemented together with an obstacle avoidance task to enable autonomous vehicle movement in an unknown environment. Most often, the task of following a target is implemented based on methods: machine learning [24–27], fuzzy logic [24, 28] artificial potential fields [29, 30], predictive control [31] or behavioral [25, 29, 32]. As part of the research, a behavioral, hierarchical robot control system was developed, taking into account

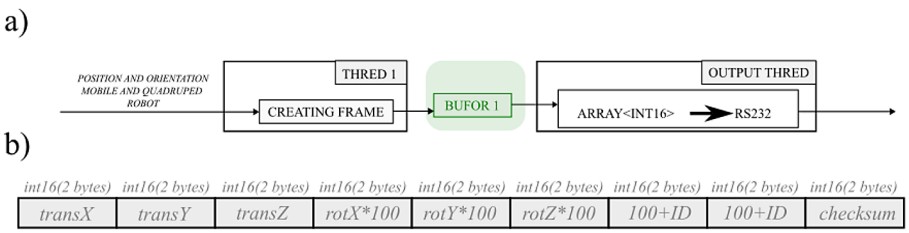

Figure 7. The structure of RoboDataLink.py for the application under consideration (a); the structure of the frame containing object position data from the motion capture system (b)

the holonomy of the robot with Mecanum wheels and the time-varying position of the target. This system is an extension of the solutions presented in the papers [27, 28, 33].

The control system consists of two layers: an outer and an inner layer (Figure 8). The outer layer enables the generation of the robot's motion trajectory, which fulfils the task of following a preset moving target. The inner layer generates a control signal that enables the robot to follow the motion trajectory generated in the outer layer.

The outer layer uses a behavioral algorithm to allow the task: "follow the target" (Figure 9). To fulfil this task, distances were first determined between the position of the target, which is the quadruped robot, and the position of the mobile robot:

$$\begin{aligned} d_x &= x_G - x_S \\ d_y &= y_G - y_S \end{aligned} \tag{1}$$

where: x_s, y_s – the position of the centre of gravity of the Mecanum wheeled mobile robot relative to a stationary *xy* reference system, x_G, y_G – position of the target, which is the characteristic point belonging to the mobile robot relative to the stationary xy reference system.

The position of the mobile robot was determined by reading the angular velocities of the robot wheels from the encoder sensors, while the positions of the target were read out based on the motion capture system used.

Based on the determined distance of the robot from the target (1), robot speed control signals were generated:

$$\begin{bmatrix} u_{Vx} \\ u_{Vy} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} d_x \\ d_y \end{bmatrix} \tag{2}$$

The control signal value u_{vx}, u_{vy} has been scaled so that: $|u_{vx}| \leq 1$, $|u_{vx}| \leq 1|$ while maintaining the direction of the total vector of the speed control signal: $\bar{u}_v = \bar{u}_{vx} + \bar{u}_{vy}$

In the next step, the motion trajectory of the mobile robot with Mecanum wheels was generated. The velocity of the robot's center of gravity was determined V_{xs}, V_{ys} relative to a stationary xy coordinate system was determined:

$$\begin{bmatrix} V_{xs} \\ V_{ys} \end{bmatrix} = \begin{bmatrix} V_{smax} & 0 \\ 0 & V_{smax} \end{bmatrix} \begin{bmatrix} u_{Vx} \\ u_{Vy} \end{bmatrix} \tag{3}$$

where: V_{smax} – maximum linear velocity, which is given by the formula:

$$V_{Smax} = \mathrm{V_{max}} \frac{1}{1+e^{-c \cdot t}} \ [\mathrm{m/s}].$$

Based on the kinematics model of the four-wheeled mobile robot with Mecanum wheels [34] and Equation 3, the inverse kinematics was solved by generating the angular velocities of the Mecanum wheels:

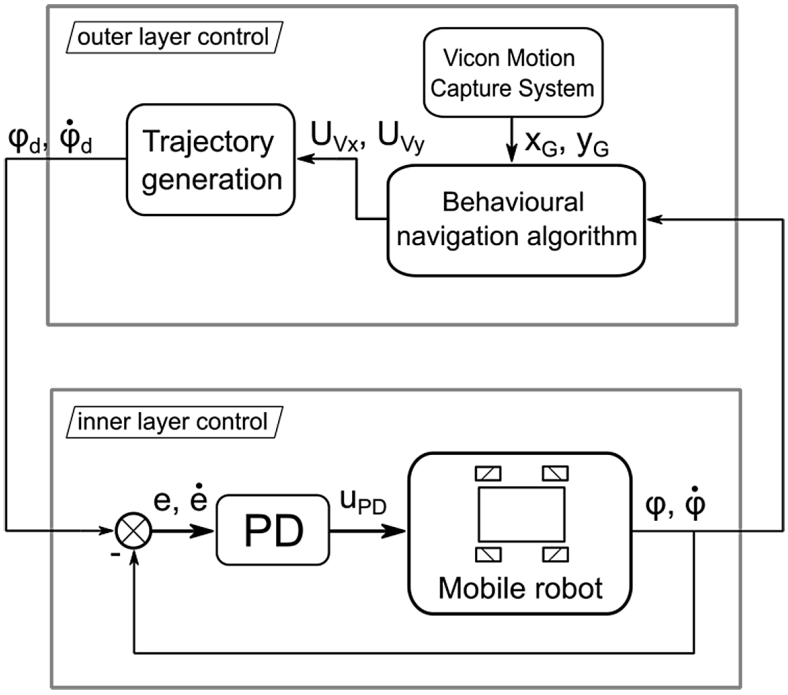

Figure 8. Hierarchical control algorithm for a mobile robot

$$\begin{aligned}
\dot{\varphi}_1 &= \frac{1}{(R+r)}\left[\begin{matrix} V_{xs}(\cos\beta+\sin\beta)+ \\ +V_{ys}(-\cos\beta+\sin\beta)-\dot{\beta}(s_x+s_y)\end{matrix}\right] \\
\dot{\varphi}_2 &= \frac{1}{(R+r)}\left[\begin{matrix} V_{xs}(\cos\beta-\sin\beta)+ \\ +V_{ys}(\cos\beta+\sin\beta)+\dot{\beta}(s_x+s_y)\end{matrix}\right] \\
\dot{\varphi}_3 &= \frac{1}{(R+r)}\left[\begin{matrix} V_{xs}(\cos\beta-\sin\beta)+ \\ +V_{ys}(\cos\beta+\sin\beta)-\dot{\beta}(s_x+s_y)\end{matrix}\right] \\
\dot{\varphi}_4 &= \frac{1}{(R+r)}\left[\begin{matrix} V_{xs}(\cos\beta+\sin\beta)+ \\ +V_{ys}(-\cos\beta+\sin\beta)+\dot{\beta}(s_x+s_y)\end{matrix}\right]
\end{aligned} \tag{4}$$

where: φ_1, φ_2, φ_3, φ_4 – angular velocities of the Mecanum wheels, β – angle of orientation of the robot frame, $\dot{\beta}$ – angular velocity of the robot frame, $2s_y$ – width of the robot platform, s_x – distance between the characteristic point S being the robot's center of gravity and the front and rear axes of the robot, R – radius of the Mecanum wheel, r – radius of the roller.

Due to the Mecanum wheeled robot's classification as a holonomic object, it is able to achieve any given linear velocity in two-dimensional space without having to change the orientation angle of the vehicle frame. Therefore, the rotation angle of the robot frame was assumed to be constant at zero: β = comst. = 0. The angular velocities of the wheels obtained from Equation 4 will be used as set motion parameters in the internal control layer.

In the inner layer of the control system, a tracking control task was performed, in which the angular parameters of the Mecanum wheels are supposed to follow the set angular parameters generated from the outer layer. In order to perform this task and to demonstrate the effectiveness of the developed control algorithm, one of the simplest control algorithms was used in the form of a PD controller, described by the formula:

$$u_{PD} = K_D s \tag{5}$$

where: K_D – diagonal matrix of the differential gain. The generalised error s and tracking error e, are defined by the relations:

$$s = \dot{e} + \Lambda e, e = \varphi_d - \varphi \tag{6}$$

where: φ_d – the vector of set angles of rotation of the Mecanum wheels, φ – vector of realised angles of rotation of the Mecanum wheels, Λ – diagonal matrix with positive elements $\lambda_{i,i}$.

Based on realised rotation angle of the Mecanum wheels read from the encoders and on the robot's movement trajectory generated in the outer layer, it is possible to control the mobile robot so that it carries out the task of following a moving target.

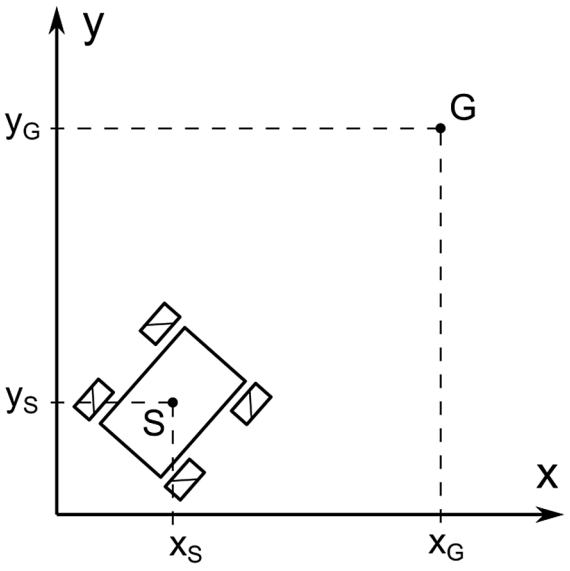

Figure 9. Position of mobile robot and target

RESULTS

In the experimental study, the target, which was a quadruped robot, was controlled manually. Together with the motion capture system Vicon, which determined its positions, it constituted a single CPS. The Panther mobile robot, which was tasked with autonomously following the moving target based on the reading of its position from encoder sensors placed on the motor shaft driving the Mecanum wheels, constituted a second CPS. Data exchange, i.e. the position reading of the Unitree GO1 robot in the Dspace card, was possible through the use of an agent, i.e. the RoboDataLink.py application.

Figures 10–13 show the experimental results of how the task of following moving target by the mobile robot is achieved by using the outer layer of the control system (Figures 10, 12) as well as the inner layer (Figure 13). Figure 10 shows the waveforms describing the position of the mobile robot and the target with respect to the axis of the stationary xy coordinate system. It can be seen from Figs. 10a and 10b that the position of the mobile robot changes over time, following the position of the moving target. The fact that the coordinates of the mobile robot do not reach the position of the quadruped robot is due to the need to maintain a safe distance between the robots. The apparent distortion of the target position waveform occurring around 40 [s] is due to

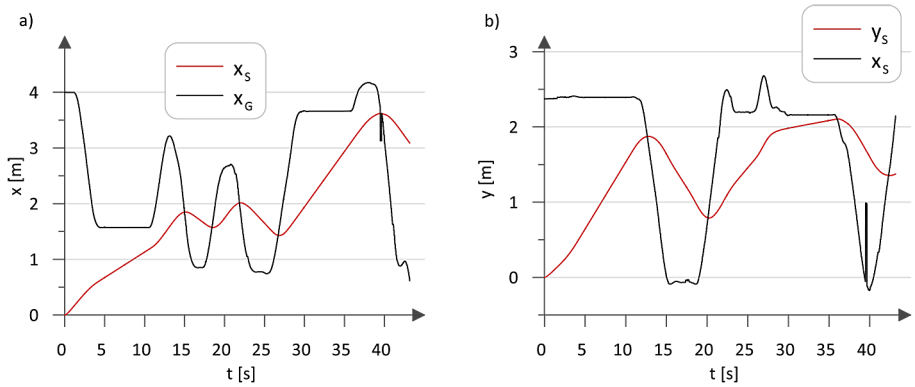

Figure 10. Test results: position of the target (x_G) and mobile robot (x_S) relative to the x-axis (a), position of the target (y_G) and mobile robot (y_S) relative to the y-axis (b)

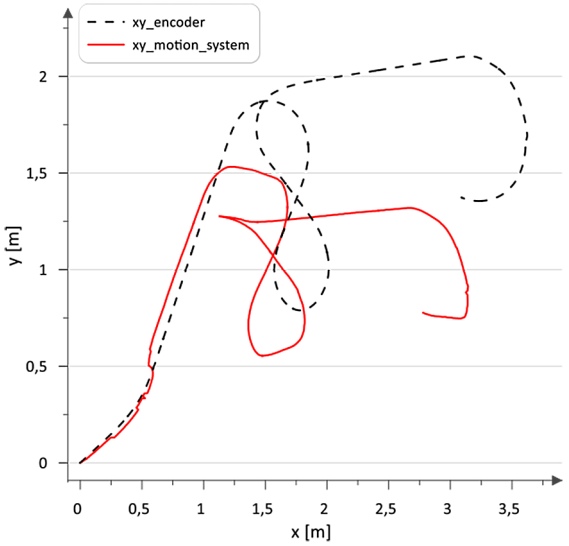

Figure 11. The motion path of the characteristic point S belonging to the mobile robot: read from encoders (xy_encoder), read from the motion capture system (xy_motion_system)

a temporary misreading of the position generated from the motion capture system, which may have been caused by the obscuration of the markers placed on the quadruped robot.

Figure 11 shows the motion track read from the encoders and the motion track read from the motion capture system. The position of the robot from the motion capture system was used to verify the odometry measurement of the robot. Position determination by the Vicon system was performed by the RoboDataLink.py application, as part of the described communication between the CPS systems. The obtained waveforms are analogous in shape, but there is a relative shift of the graphs by approximately 0.75 m with respect to the y-axis. The reason for this situation is most likely the temporary loss of contact between the Mecanum wheels and the ground that was observed during the experimental tests.

Figure 12a shows the distances between the position of the target and the position of the mobile robot. Based on the distance measurement, a robot speed control signal is generated (Fig. 12b). A motion trajectory is then generated, from which the obtained Mecanum wheel rotation angles (Fig. 12c) and angular velocities (Fig. 12d) are sent to the internal control layer. In Fig. 12a, the apparent distance to the target, excluding the initial phase of movement, oscillates between -2 and 2 [m], indicating that the robot is effectively trying to follow the changing position of the target. In Figs. 12c and 12d, the generated waveforms of the angular motion parameters are the same for the wheel pair 1, 4 and 2, 3. This is due to the assumption made regarding the constant orientation angle of the robot frame ($\beta = 0$). The angular velocities of the Mecanum wheels assume a non-zero value in the final phase of the experiment. This is due to the fact that the position of the target is continuously variable, so that the mobile robot does not perform a braking process before the end of experiment. The maximum accepted duration of the experiment was approximately 45 s, after which time the robot stopped.

Figure 13a shows the waveforms of the control signals generated by the PD controller in the inner layer of the control system. The waveforms of the Mecanum wheel angle tracking error (Fig. 13b) and angular velocity (Fig. 13c) are relatively small and limited, which indicates the stability of the control system. The sudden increase in angular velocity error observed at about 40 s is due to the occurrence of the previously discussed error in reading the target position affecting the sudden change in the set angular parameters of the robot's movement.

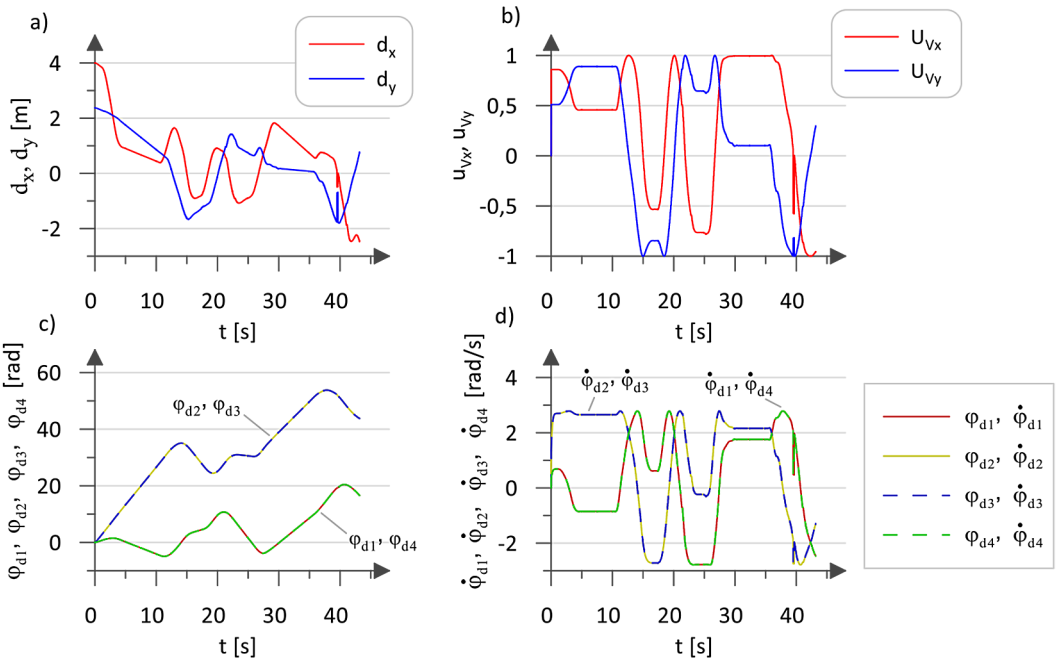

Figure 12. Test results: distances between the position of the target and the position of the mobile robot relative to the x-axis and y-axis (a), waveform of the robot speed control signal (b), waveforms of the generated angles of rotation of the Mecanum wheels (c), waveforms of the generated angular velocities of the Mecanum wheels (d)

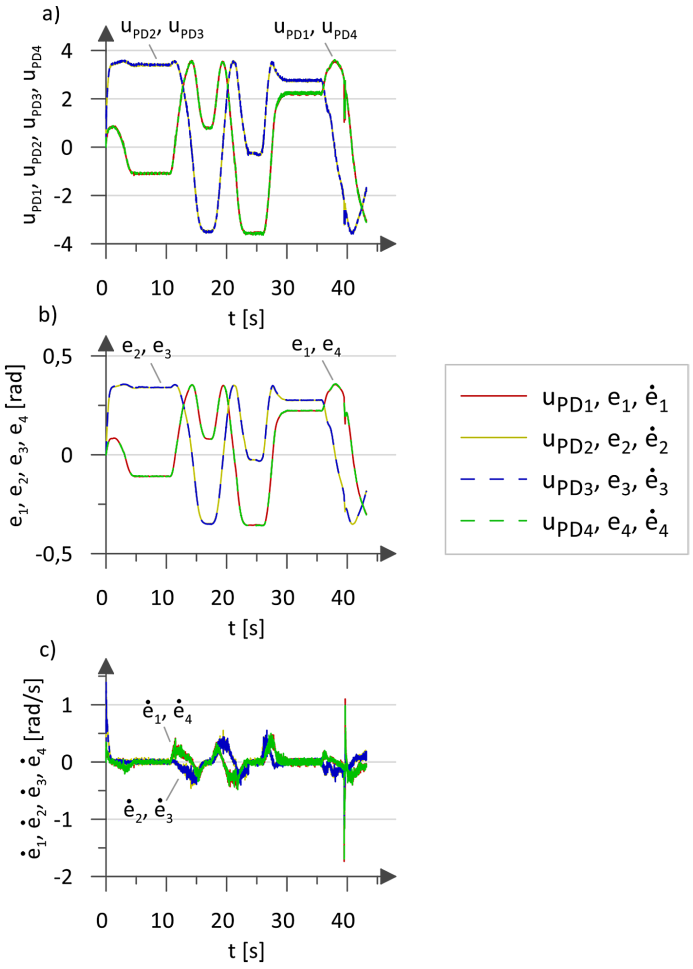

Figure 13. Test results: waveforms of control signals generated in the inner layer (a), waveform of Mecanum wheel angle tracking errors (b), waveform of Mecanum wheel angular velocity tracking errors (c)

The presented case study demonstrates the integration of various fields (computer science, robotics, and control systems) that coexist to practically implement the concepts of Industry 4.0 through CPS. This example thus confirms the challenges associated with Industry 4.0 as outlined in the works of [6, 13]. By addressing the communication problem identified in the studies of [14, 17, 35], a specific solution has been proposed to ensure hardware-software compatibility.

The presented experimental results confirm the effectiveness of the presented control algorithm in performing the task of following a moving target. Furthermore, the experiment confirmed the feasibility of using RoboDataLink.py as an agent for communication between CPS systems.

CONCLUSIONS

This paper application that solves the problem of hardware-software compatibility in communication between CPS (cyber-physical system) components or between CPSs. The program RoboDataLink.py works in an input-output architecture and, through a modular thread--based design, allows easy implementation of new input-output communication standards. In consequence, thanks to the modular structure this solution can be treated as is an IoT architecture. The multidisciplinary nature of proposed solution was verified in an experiment consisting of: two robots (a mobile robot with Mecanum wheels and a quadruped robot) and a motion capture system. The experiment carried out the task: following a moving target, based on a hierarchical, behavioral control system. The RoboDataLink.py application used in the test provided communication between the two CPSs. An agent in the form of RoboDataLink.py was used to ensure hardware-software compatibility between the motion capture system and the Dspace signal card. While this paper has demonstrated the usefulness of the author's RoboDataLink.py application in the context of an Industry 4.0 implementation, the authors see room for development of the solution presented. This is the implementation of a graphical interface with an editor for configuration files, testing in virtual environments or direct connection to the Matlab/Simulink engineering environment. In addition, in future research work it is planned to extend the functionality of the mobile robot control algorithm with an obstacle avoidance function realized through the use of Lidar. In order to achieve software and hardware compatibility with, for example, the Dspace card, the RoboDataLink.py application will be used.

Acknowledgments

The research leading to these results has received funding from the commissioned task entitled "VIA CARPATIA Universities of Technology Network named after the President of the Republic of Poland Lech Kaczyński", under the special purpose grant from the Minister of Science, contract no. MEiN/2022/DPI/2578 action entitled "In the neighborhood – inter-university research internships and study visits".

REFERENCES

1. Gola A. Design and management of manufacturing systems. Applied Sciences 2021;11:2216. https://doi.org/10.3390/app11052216
2. Nahavandi S. Industry 5.0—A Human-centric solution. Sustainability 2019;11:4371. https://doi.org/10.3390/su11164371
3. Pizoń J, Witczak M, Gola A, Świć A. Challenges of human-centered manufacturing in the aspect of industry 5.0 assumptions. IFAC-PapersOnLine 2023;56:156–61. https://doi.org/10.1016/j.ifacol.2023.10.1562
4. Shukur M H, Askar S, R M. Zeebaree S. The utilization of 6G in Industry 4.0. Appl Comput Sci 2024;20:75–89. https://doi.org/10.35784/acs-2024-17
5. Culot G, Nassimbeni G, Orzes G, Sartor M. Behind the definition of Industry 4.0: Analysis and open questions. International Journal of Production Economics 2020;226:107617. https://doi.org/10.1016/j.ijpe.2020.107617
6. Alcácer V, Cruz-Machado V. Scanning the Industry 4.0: A literature review on technologies for manufacturing systems. Engineering Science and Technology, an International Journal 2019;22:899–919. https://doi.org/10.1016/j.jestch.2019.01.006
7. Motyl B, Baronio G, Uberti S, Speranza D, Filippi S. How will change the future engineers' skills in the industry 4.0 framework? A questionnaire survey. Procedia Manufacturing 2017;11:1501–9. https://doi.org/10.1016/j.promfg.2017.07.282
8. Gilchrist A. Introducing industry 4.0. Industry 4.0, Berkeley, CA: Apress; 2016; 195–215. https://doi.org/10.1007/978-1-4842-2047-4_13
9. Saucedo-Martínez J A, Pérez-Lara M, Marmolejo-Saucedo J A, Salais-Fierro T E, Vasant P. Industry 4.0 framework for management and operations: a review. J Ambient Intell Human Comput 2018;9:789–801.

https://doi.org/10.1007/s12652-017-0533-1

10. Krot K, Iskierka G, Poskart B, Gola A. Predictive monitoring system for autonomous mobile robots battery management using the industrial internet of things technology. Materials 2022; 15(19):6561. https://doi.org/10.3390/ma15196561
11. Asaad H, Askar S, Kakamin A, Faiq N. Exploring the impact of artificial intelligence on human-robot cooperation in the context of Industry 4.0. Appl Comput Sci 2024;20:138–56. https://doi.org/10.35784/acs-2024-21
12. Padhan S, Turuk A K. A technique to detect and mitigate false data injection attacks in Cyber–Physical Systems. Computers & Security 2025;150:104253. https://doi.org/10.1016/j.cose.2024.104253
13. Liu Y, Peng Y, Wang B, Yao S, Liu Z. Review on cyber-physical systems. IEEE/CAA J Autom Sinica 2017;4:27–40. https://doi.org/10.1109/JAS.2017.7510349
14. Lee E. The past, present and future of cyber-physical systems: a focus on models. Sensors 2015;15:4837–69. https://doi.org/10.3390/s150304837
15. Majerník M, Daneshjo N, Malega P, Drábik P, Barilová B. Sustainable development of the intelligent industry from industry 4.0 to industry 5.0. Adv Sci Technol Res J 2022;16:12–8. https://doi.org/10.12913/22998624/146420
16. Aceto G, Persico V, Pescape A. A survey on information and communication technologies for industry 4.0: state-of-the-art, taxonomies, perspectives, and challenges. IEEE Commun Surv Tutorials 2019;21:3467–501. https://doi.org/10.1109/COMST.2019.2938259
17. Tlach V, Kuric I, Zajačko I, Kumičáková D, Rengevič A. The design of method intended for implementation of collaborative assembly tasks. Adv Sci Technol Res J 2018;12:244–50. https://doi.org/10.12913/22998624/86476
18. Kuraś P, Strzalka D, Kowal B, Mazurek J. REDUCE – A python module for reducing inconsistency in PairwiseComparison matrices. Adv Sci Technol Res J 2023;17:227–34. https://doi.org/10.12913/22998624/170187
19. Manual P n.d. https://husarion.com/manuals/panther/ (accessed February 14, 2025).
20. dSPACE GmbH. DS1103 Hardware Installation and Configuration 2010.
21. Unitree. Go1 User Manual 2021.
22. Go1 Software Manual 2022. https://unitree-docs.readthedocs.io/en/latest/get_started/Go1_Edu.html# (accessed February 14, 2025).
23. Camera Vicon Vero. Technical information. n.d. https://www.vicon.com/hardware/cameras/vero/#technical-information
24. Boubertakh H, Tadjine M, Glorennec P-Y. A Simple Goal Seeking Navigation Method for a Mobile Robot Using Human Sense, Fuzzy Logic and Reinforcement Learning. In: Lovrek I, Howlett RJ, Jain LC, editors. Knowledge-Based Intelligent Information and Engineering Systems, Berlin, Heidelberg: Springer Berlin Heidelberg; 2008; 666–73.
25. Adriansyah A, Gunardi Y, Badaruddin B, Ihsanto E. Goal-seeking behavior-based mobile robot using particle swarm fuzzy controller. Telkomnika 2015;13:528. https://doi.org/10.12928/telkomnika.v13i2.1111
26. Motlagh O, Nakhaeinia D, Tang SH, Karasfi B, Khaksar W. Automatic navigation of mobile robots in unknown environments. Neural Comput & Applic 2014;24:1569–81. https://doi.org/10.1007/s00521-013-1393-z
27. Hendzel Z, Szuster M. Neural sensor-based navigation of wheeled mobile robot in unknown environment. Pomiary Automatyka Robotyka 2013;17:114–20.
28. Hendzel Z. Fuzzy reactive control of wheeled mobile robot. Journal of Theoretical and Applied Mechanics 2004;42:503–17.
29. Goeller M, Steinhardt F, Kerscher T, Zöllner JM, Dillmann R. Proactive Avoidance of Moving Obstacles for a Service Robot Utilizing a Behavior-Based Control n.d.
30. Kerr EP, Vance P, Kerr D, Coleman SA, Das GP, McGinnity TM, et al. Biological Goal Seeking. 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India: IEEE; 2018; 1602–7. https://doi.org/10.1109/SSCI.2018.8628696
31. Moreno-Caireta I, Celaya E, Ros L. Model predictive control for a Mecanum-wheeled robot navigating among obstacles. IFAC-PapersOnLine 2021;54:119–25. https://doi.org/10.1016/j.ifacol.2021.08.533
32. Benbouabdallah K, Zhu Q-D. A Behavior-Based Controller for a Mobile Robot Tracking a Moving Target in Multi-obstacles Environment. 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China: IEEE; 2013; 418–23. https://doi.org/10.1109/IHMSC.2013.247
33. Burghardt A, Szybicki D, Kurc K, Muszyńska M. Mechatronic designing and prototyping of a mobile wheeled robot driven by a microcontroller. Journal of Theoretical and Applied Mechanics 2020;58:127–42. https://doi.org/10.15632/jtam-pl/115332
34. Szeremeta M, Szuster M. Neural tracking control of a four-wheeled mobile robot with mecanum wheels. Applied Sciences 2022;12:5322. https://doi.org/10.3390/app12115322
35. Szybicki D, Obal P, Penar P, Kurc K, Muszyńska M, Burghardt A. Development of a dedicated application for robots to communicate with a laser tracker. Electronics 2022;11:3405. https://doi.org/10.3390/electronics11203405