DEVELOPING A MOBILE ROBOT WITH VOICE COMMANDS

PAVOL BOZEK[1], ELENA PIVARCIOVA[2]

[1]Slovak University of Technology in Bratislava, Faculty of Materials Science and Technology, Institute of Production Technologies, J. Bottu 25, 917 24 Trnava, Slovakia

[2]Technical University in Zvolen, Faculty of Technology, Department of Manufacturing and Automation Technology, Studentska 26, 960 01 Zvolen, Slovakia

DOI: 10.17973/MMSJ.2023_06_2023013

e-mail to corresponding author: pavol.bozek@stuba.sk

The paper deals with the development of a transport robot control system with voice commands and simulation of their movement in Microsoft Robotics Developer Studio (MRDS) software product. The advantages of MRDS over other programmes are discussed. The programming language used was VPL, a visual programming environment for creating and debugging applications. It was designed multi-level control of the robot. At the top level of control, human voice commands are recognised, setting the direction of movement of the robot and its motion functions. At the lower level, the wheel speed is controlled by a microcontroller, which generates pulse width modulation output signals.

KEYWORDS

Mobile robot, programming, simulation, voice

1 INTRODUCTION

We live in an age where the most complex or repetitive tasks are automated. Smart robots have the potential to revolutionize how we perform all kinds of tasks with high accuracy and efficiency [Staple 2021]. Handling machines are increasingly being used in all sectors of the industry. Knowledge of the theory of transport and handling machines are basic prerequisites for their further technical development [Blatnicky 2020, Krenicky 2022].

Examples of intelligent control systems for transport robots are given in the papers [Nikitin 2017 and 2022, Bozek, 2021]. With programming voice-controlled IoT applications deals [Allwork 2023]. Study of problems involved in assembling complete, modern electric drive systems involving mechanical, electrical, and electronic elements [Crowder 2020] covers the different topics from mechatronics and robotics, including mechatronics basics, robotics arms and manipulators, sensors and actuators in mechatronics [Gacovski 2021]. Artificial intelligence, autonomous robot navigation, intelligent robot system design, intelligent sensing and control, and machine vision are described in literature [Kim 2018].

The use of transport robots nowadays makes it possible free humans from the task of carrying heavy loads indoors. A transport robot can be used as a platform for wheelchairs. Transport robots are planned to be widely used in industry as part of the Industry 4.0 industrial revolution. The transport robot is most conveniently controlled by voice, reducing worker fatigue and increasing speed and flexibility of command transmissions.

A comparative characteristic of Java Robocode, Pascal Robot, Snake Battle, and Microsoft Robotics Developer Studio (MRDS) is presented in the summary Table 1. The «+» sign in the table indicates the presence of the characteristic and the «–» sign indicates the absence of the characteristic.

Table 1. Comparative characteristics of existing equivalents

Program	Robocode	Pascal Robot	Snake Battle	MRDS
Easy to write code	+	–	–	+
Display actions in 3D	–	–	–	+
Ability to run tournaments	+	+	+	–
Create tournament reports	–	–	–	–
Built-in editor	+	+	–	+
Laws of Physics	–	–	–	+
Portability to real robots	–	–	–	+
Faster tournament time	+	+	–	–
Ability to evaluate designed robots	+	+	+	–
Use of a large number of sensors	–	–	–	+

A study of the subject area and a search for similar projects revealed that there are several systems for computer game strategy competitions. Each system has both, advantages, and disadvantages.

The comparison table shows that systems such as Pascal Robot and Snake Battle are long outdated and have very poor functionality. The process of demonstrating computer strategies is low-informative and not trivial due to the image quality. The user interface is virtually non-existent.

Java Robotics is a more appropriate system for computer games strategy competitions compared to systems like Pascal Robot and Snake Battle. It has better 2D graphics, an intuitive interface, and visibility of the action. But the system is not without its drawbacks. The robots have a minor set of devices that are used to program their behavior. In Java Robotics there is no possibility of making reports on the tournaments held, there is no possibility of displaying what is happening in 3D, the laws of physics do not operate on robots, and certainly the program written for the robot in Java Robotics environment cannot be applied to the real-life robot.

The summary table clearly shows that the MRDS platform is of great interest for creating software for computer game strategy programming competitions. This environment is great for programming robots, both real and virtual. The program written for the virtual robot can be applied to the real robot equipped with the necessary number of sensors [Murcinkova 2013]. User only needs to change the manifest and specify new ports for services. The ability to show what happens to the robot in 3D makes MRDS much more attractive in comparison to the above-mentioned analogues. Another important feature of the MRDS environment is a 3D physic-based engine that provides realistic simulation, all objects of the simulation world obey the laws of physics.

MRDS has a number of advantages over the considered systems of computer game strategy competitions, but this environment lacks the main infrastructure for tournaments between robots, i.e. there is no possibility of entering code developed by a participant into a single game environment, there is no possibility to evaluate the developed robots according to the specified criteria, there is no possibility of making convenient reports on tournaments. The software developed as part of this thesis solves these problems, making the MRDS platform the best system for conducting competitions, in relation to the reviewed counterparts. MRDS, a Windows based environment for robot control, is used for simulation. This product is designed for academic or commercial development and supports a wide variety of robot hardware.

RDS is based on CCR (Concurrency and Coordination Runtime) library, a .NET implementation library for handling concurrent and asynchronous data flows using messaging and DSS (Decentralized Software Services), a lightweight tool for creating distributed service-based applications that provide management of multiple services to adjust overall behavior.

Features of Microsoft's Visual Programming Language for creating and debugging software applications for robots include web- and windows-oriented interfaces, 3D modeling, simplified access to robot sensors and actuators, support for multiple languages including C#, Visual Basic NET, JScript and IronPython. Microsoft Robotics Developer Studio supports modularity to add new services to the suite. Currently available: Soccer Simulation, Microsoft's Sumo Competition, and the community-developed Maze Simulator, a program for creating worlds and walls that can be explored by a virtual robot.

The advantage of Microsoft Robotics Developer Studio (MRDS) is that it's free and suitable for schools and universities to learn robotics. MRDS is aimed at programmers of all levels and includes support for a vast range of hardware. The package is compatible with the following robotics platforms: Aldebaran Robotics, iRobot Create, Mindstorms NXT, CoroWare CoroBot, KUKA Robotics, Boe-Bot, Parallax Scribbler, Kondo KHR-1 and several others.

The environment can be thought of as having four interrelated components:

Visual Programming Language created by Microsoft specifically for MRDS. Programs in VPL look like diagrams where all elements are connected with each other, and each has its own functionality. The language is intended for beginner programmers and requires basic knowledge of variables and algorithmic principles.

Concurrency and Coordination Runtime is a library for working with asynchronous and parallel data flows, based on .NET Framework. It simplifies code creation for scaling and parallel execution on latest-generation multicore processors, allowing, for example, robots to correctly respond to data from multiple sensors simultaneously. The library has been widely used in third-party projects, greatly simplifying the development of multi-threaded applications. An example is mySpace.com, which uses CCR in its server side.

Decentralized Software Services is a lightweight CCR-based environment for the development of distributed applications, providing management of various services that correct the behavior of robots in general.

Visual Simulation Environment is a visualization environment that simulates the behaviour of robots in a 3D virtual world. It allows for experimenting with different models, testing and debugging algorithms in cases where no real robot is available. It uses NVIDIA PhysX technology to create realism.

Pluses of the simulator include the following:

- it is easy enough to get started,
- It's a great tool for learning and research,
- the scenic approach allows to simulate the world around by creating objects and placing them in the right places,
- advanced graphics (thanks to the capabilities of 3D accelerators) allow the visual component of the environment to be accurately modelled,
- physics simulation is fully supported.

The simulator has a number of drawbacks:

- The world in the simulator is idealised, i.e. there is no data distortion as in the real world. For example, a surface the robot rides on may not create friction, whereas in the real world no such surface exists. However, when writing services for specific 'virtual' sensors, it is possible to add noise similar to the experiment.
- It is impossible to fully describe a robot as it is in the real world. When describing a robot in a simulator, we can try to make it as similar as possible to the real one, but the model will always only be an approximation. Thus, in a simulator, user has to deal with incomplete or inaccurate models.
- The more accurate the model we want to create, the clearer the setup has to be, and this is very time consuming.
- The physics model in Microsoft Robotics Studio is quite simplified, so the simulation is not suitable where ultra-precise calculations are needed.
- There is no consideration and support for the real environment in which the robot operates (its movement surface type, weather conditions, etc.).
- When controlling a real robot, its simulation is used, which may not be exactly the same as the real prototype.
- The more accurate the model, the more adjustments it requires.
- Although PhysX is used, the physics in MRDS is greatly simplified.

2 VOICE CONTROL METHODS FOR THE MOBILE ROBOT

The aim of this work is to develop systems to control mobile robots in nondeterministic environments by voice commands.

To achieve this goal, the following tasks are solved: a review of analogues and prototypes of control systems with voice control, the development of a control system for transport robots based on voice control, simulation of the movement of transport robots in the software product MRDS.

The sentence structure, phrase values and speech morphology are to be programmed, in particular in the form of "rules" for the robot, and then transmitted via Bluetooth.

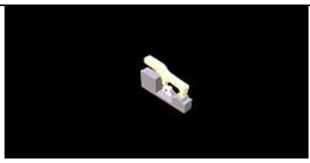
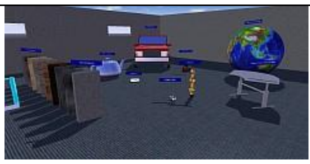
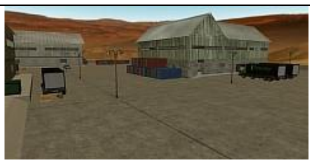
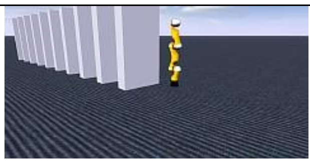
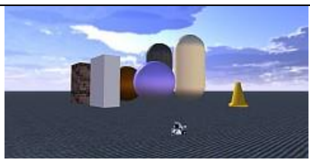
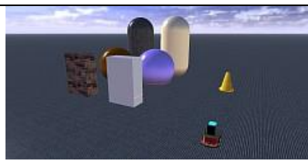
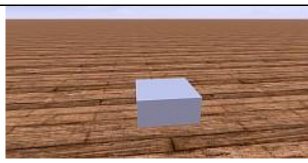
There are a large number of apps available for the Android operating system that can be used to control the robot using Bluetooth. The app allows to control the robot with your voice, buttons or the phone's accelerometer. The slider allows to control the speed of the robot. There are also two buttons to activate the robot's front and rear lights. Indications include a flashing light when the phone is connected to the robot and arrows that show the direction the robot is moving.

Motion simulations of the voice-controlled transport robot were performed in the Microsoft Robotics Developer Studio software environment using the Visual Programming Language (VPL). VPL is a visual programming environment for creating and debugging applications. Microsoft Robotics Developer Studio is a visual programming language for building and debugging software applications for robots that creates a 3D motion simulation of the robot, allows simulation of robot behavior in the virtual world using NVIDIA PhysX technology, which allows the use of modern physical model, as well as this language has simplified access to sensors and actuators the robot, and supports multiple languages, including C #, Visual

Basic .NET, JScript and IronPython. This VPL language also has a disadvantage: there is no consideration and support for the real-world operating environment, and only a simulation is used to control the transport robot, which may not fully match the real prototype. The more accurate the model, the more adjustments it requires.

MRDS includes a set of objects that allow users to quickly assemble fairly complex robotic platforms in a variety of simulated environments. Sixteen different environments are included in the simulation environment. Table 2 gives examples of settings in the simulation environment.

Table 2. Examples of settings in the simulation environment

Name	Image of the setting
Apartment Environment	
Collada Scene	
Entities	
Factory	
House Floor plan	
iRobot Create Simulation	
KUKA LBR3 Arm	
Lego NXT Tribot Simulation	
ModernHouse	
Multiple Simulated Robots	
Outdoor Environment	
Pioneer 3DX Simulation	
Simple Simulated Robot	
Simulated Air Resistance	
Simulated Sumo	
Urban Environment	

3 EXPERIMENTAL PART

Consider a mobile robot with two independent active wheels whose axes lie in a straight line. The wheels are driven by electric motors. The axes of the wheels are rigidly attached to the body, an absolutely solid body, which can move in a plane parallel to each other. The robot moves on a perfectly rough plane, the wheels do not slip at the points of contact with the plane.

The model of the robot is shown in Figure 1. The figure shows: 1 – left wheel, 2 – right wheel, 3 – supporting wheel, 4 – left motor, 5 – right motor.

Usually, when a robot moves with two actuators, its position is described by four parameters (I_1, I_2, ω_1, ω_2). In this case, it is difficult to mathematically describe the relationship between the motion parameters and the analytically given trajectory of motion. Let's add the rotation of the platform ωMR and synthesize a five-axis model of the robot's motion on the plane. The parameters then are the electric currents and angular velocities of the two actuators, and the angular velocity of the robot's platform rotation: I_1, I_2, ω_1, ω_2, ω_{MR}. The centre of mass of the robot body is located in the middle of the wheel axis. We consider the influence of the passive wheel on the motion of the system to be insignificant. We assume that during the motion of the robot, the coordinates, together with the angular velocities of the wheels, are continuous.

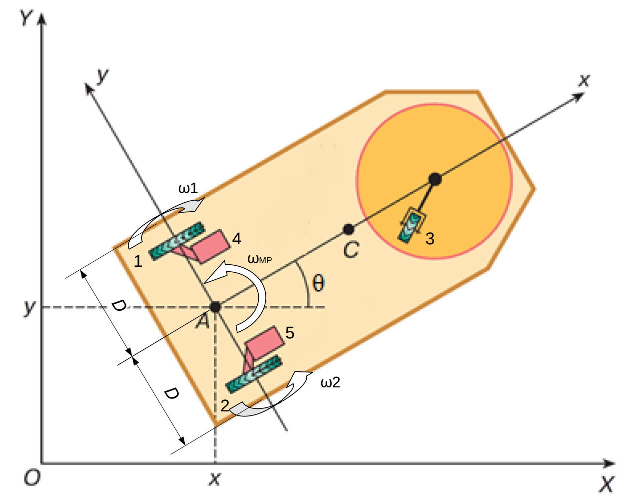


Figure 1. Kinematic diagram of the mobile robot

Consider the problem of transition of the robot from some *k*-th state on the motion path to (*k* + 1) state. In this case the robot has to move to (*k* + 1)-point with a priori known velocity (V_{MR} = const) and velocity vector directed tangentially to the trajectory at this point. Let the basic motions of the robot be linear velocity motion (acceleration or deceleration when moving to a point with a given velocity) and circular arc motion from a trajectory point with one curvature to a trajectory point with another curvature.

Mathematical model of a mobile robot

The equation of motion dynamics of a mobile robot in the horizontal plane, including the dynamics of the actuator motors, is as follows:

$$J\frac{d\omega}{dt}=\frac{D}{r}(M_2-M_1) \qquad (1)$$

where J is the moment of inertia of the robot, including the moments of inertia of the trolley, motors and wheels, M_1, M_2 are torques of the motors, *D* is distance from the centre of the axle to the wheel, *r* is radius of the wheel.

By expressing the torques of the motors through electric currents, we obtain

$$J_{MR}\frac{d\omega}{dt}=\frac{DC_M}{r}(I_2-I_1) \qquad (2)$$

where I_i is the current in the armature winding of the *i*-th motor, C_M is constant of DC motor.

Moment of inertia applied to each wheel as it moves

$$J=\frac{mr^2}{2}+\frac{J_{MR}}{2} \qquad (3)$$

where *m* is total mass of motor and wheel, *r* is wheel radius, J_{MR} is moment of inertia of the mobile robot.

The sensor located on the wheel shaft is a disc with twenty holes (an infrared diode on one side and a phototransistor on the other). The diameter of the wheel determines the distance that corresponds to one pulse of the sensor.

If one wheel is not turning, knowing the movement of the second wheel, it is possible to determine the angle of rotation and the speed of the wheel. By calculating the speed of the wheels, it is possible to determine the speed at which the robot moves.

A code has been developed that allows a person to control the robot using speech. The speech recognition was set up with the help of the Windows control panel. The robot executes the following voice commands:

- "Drive Forwards" – set the DrivePower variable in both wheels to 0.1,
- "End Drive" – set the DrivePower variable in both wheels to 0.0,
- "Turn Left" – Turn the robot to the left by 45 degrees (using the RotateDegrees call),
- "Turn Right" – Turns the robot to the right by 45 degrees,
- "Turn Around" – turns the robot 180 degrees.

Three additional commands were used:

- "Begin Learning" – In the learning state, the robot writes commands,
- End Learning – instructs the robot to stop recording commands,
- "Perform Actions" – if the robot has received a task (and training has ended), it must respond to this command by performing the actions described in the task.

A speech recognition service has been added to receive a verbal command. A speech recognizer from Servicestoolbox has been selected. The speech recognizer service uses grammars that define words and phrases. The speech recognizer Guiservice was used to set up the grammar.

As a result, after training the robot with voice commands, the code of the transport robot motion control program is obtained, which is shown in Figure 2.

After training the robot, it executes voice commands in a virtual environment in the VPL.

The software environment used is Microsoft Robotics Developer Studio.

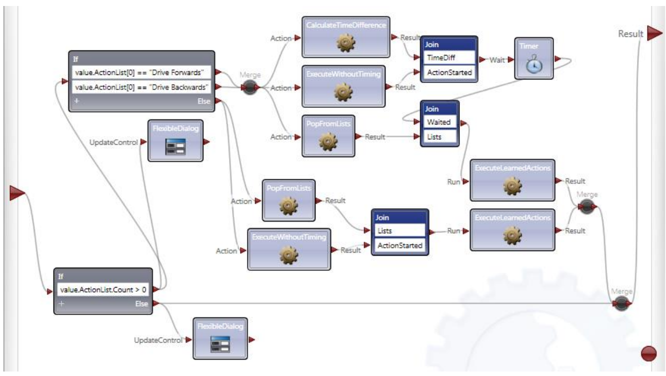


Figure 2. Example code for a mobile robot motion control programme in MATLAB

4 COMPUTATIONAL EXPERIMENT

The computation experiment is shown in Figure 3.

The folder contains 12 wav files of the training sample, i.e. four voice commands, each recorded three times. The names of the voice commands are:

Command "forward": 1.wav, 5.wav, 9.wav,

Command "back": 2.wav, 6.wav, 10.wav,

Command "left": 3.wav, 7.wav, 11.wav,

Command "right": 4.wav, 8.wav, 12.wav.

The "ya_sdam" folder contains the voice commands of the test sample. The names of the commands are:

Command "forward": 1.wav, 5.wav,

The "back" command: 2.wav, 6.wav,

The "left" command: 3.wav,

Command "right": 4.wav.

Start the project by specifying the name of the audio file to be recognized from the test folder. Figure 3 shows a program section for selecting an audio file for recognition.

```
nagib.m*
clear;
nTest=1;     % number of file
dlina=1000;  % number of neurons
m=4;         % number of commands
t=3;         % number of different sounding commands
a=0.1;       % learning rate
N=10;        % number of network training
```

Figure 3. Selecting an audio file for recognition (nTest =1)

In this case, the audio file 1.wav is the voice command "forward". As a result, the software recognises the voice command "forward" and outputs it on the screen.

Result of voice command recognition is recognised command: forward.

At the lower level, the wheel speed is controlled by a microcontroller, which generates pulse width modulation output signals. Figure 4 shows the PWM signal conditioning diagram using a speed feedback sensor of mobile robot.

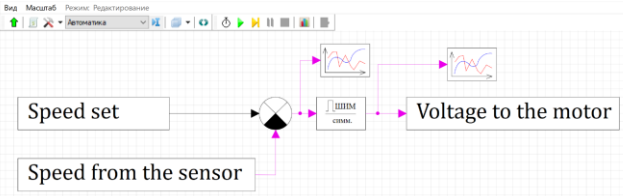


Figure 4. PWM signal conditioning diagram using a speed feedback sensor of mobile robot

Figure 5 shows a graph of the variation in velocity deviation with time. Figure 6 shows the PWM signal for the motor.

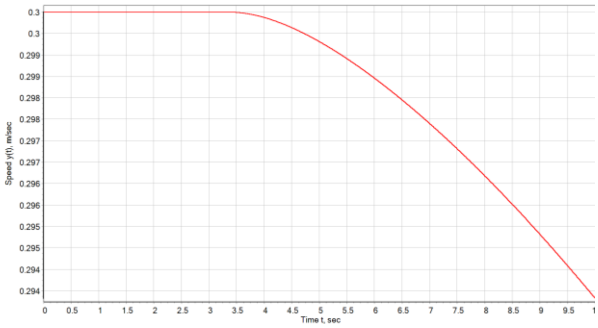


Figure 5. Graph of the variation in velocity deviation of mobile robot with time

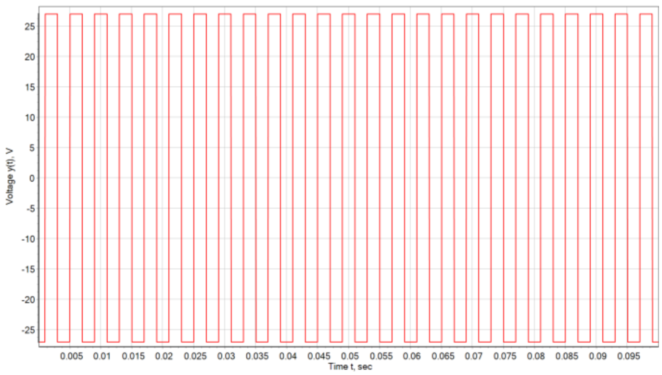


Figure 6. The PWM signal for the motor of mobile robot

5 CONCLUSIONS

In the article mathematical and simulation models of mobile robot have been developed. These models are needed to develop the power electronics and control system for the mobile robot. A program was developed in Matlab Simulink software to recognize the voice commands of the mobile robot control. Voice commands of training samples were prepared. Voice commands in wav file format were used for this purpose. The results of the computational experiment showed that the recognition of voice commands was successful.

The developed control system allows the mobile robot to follow from one target point to another using voice control. At the lower level, the wheel speed is controlled by a microcontroller, which generates pulse width modulation output signals.

Advantages of the MRDS simulator include: advanced graphics allowing accurate simulation of the visual environment, 3D robot position and action simulation based on a scene-based approach, full physics simulation supported, environment for computational experiments and research, simple, visual, interface.

Recording algorithms in the form of flowcharts is studied by students at the initial stage of studying the section "programming". Parallel study of VPL language will contribute to students' understanding of practical importance of using flowcharts and will introduce them to the actual, dynamically developing nowadays technology of visual programming.

The research results obtained can be used to build diagnostic systems for robot drives [Nikitin 2020], for industrial robot [Tlach 2017 and 2020], simulation of humanoid robot [Virgala 2014], the issue of collaborative robots and possibilities of human-robot collaboration in a collaborative workspace [Tlach 2019].

In future research, it is recommended to focus on the possibilities of using an inertial navigation system for the inertial navigation of a mobile robot [Qazizada 2016], using the object-oriented programming platform LabVIEW [Krenicky 2018], using of artificial intelligence model [Nikitin 2020] or genetic algorithms [Cubonova 2019] and an optimisation method [Hartansky 2017].

ACKNOWLEDGMENTS

This contribution was prepared within the framework of the project KEGA 006STU-4/2021: "Progressive form of interdisciplinary education and support for the development of the study of vocational subjects in the university environment".

REFERENCES

[Allwork 2023] Allwork, J. Programming Voice-controlled IoT Applications: with Alexa and Raspberry Pi. Elektor, 2023. 272 p. ISBN 978-3-89576-531-5.

[Blatnicky 2020] Blatnicky, M., Dizo, J., Saga, M., Gerlici, J., Kuba, E. Design of a Mechanical Part of an Automated Platform for Oblique Manipulation. Applied Sciences, 2020, Vol. 10, Issue 23, Article No. 8467, DOI: 10.3390/app10238467.

[Bozek 2021] Bozek, P. and Nikitin, Y. The Development of an Optimally-Tuned PID Control for the Actuator of a Transport Robot. Actuators, 2021, Vol. 10, Issue 8, 195. DOI: 10.3390/act10080195.

[Crowder 2020] Crowder, R. Electric Drives and Electromechanical Systems: Applications and

Control. 2nd Edition. Butterworth-Heinemann, 2020, 309 p. ISBN 978-0-08-102884-1.

[Cubonova 2019] Cubonova, N., Dodok, T. and Sagova, Z. Optimisation of the Machining Process Using Genetic Algorithm. Scientific Journal Of Silesian University Of Technology - Transport, 2019, Vol. 104, pp. 15-25. DOI: 10.20858/sjsutst.2019.104.2.

[Gacovski 2021] Gacovski, Z. Mechatronics and Robotics. Arcler Press, 2021, 437 p. ISBN 978-1-77407-775-7.

[Hartansky 2017] Hartansky, R., Smiesko, V. and Rafaj, M. Modifying and accelerating the method of moments calculation. Computing and Informatics, 2017, Vol. 36, No. 3, pp. 664–682.

[Kim 2018] Kim, JH., Myung, H., Kim, J., Xu, W., Matson, ET. and Jung, JW. Robot Intelligence Technology and Applications 5. Springer, 2018, 579 p. ISBN 978-3319784519

[Krenicky 2018] Krenicky, T. and Ruzbarsky, J. Alternative Concept of the Virtual Car Display Design Reflecting Onset of the Industry 4.0 into Automotive. 2018 IEEE 22nd international conference on intelligent engineering systems, pp. 407–410.

[Krenicky 2022] Krenicky, T., Hrebenyk, L., Chernobrovchenko, V. Application of Concepts of the Analytic Hierarchy Process in Decision-Making. Management Systems in Production Engineering, 2022, Vol. 30, No. 4, pp. 304-310. https://doi.org/10.2478/mspe-2022-0039

[Kuric 2020] Kuric, I., Tlach, V., Cisar, M., Sagova, Z. and Zajacko, I. Examination of industrial robot performance parameters utilizing machine tool diagnostic methods. International journal of advanced robotic systems 2020, Vol. 17, Issue 1, DOI: 10.1177/1729881420905723.

[Murcinkova 2013] Murcinkova, Z., Krenicky, T. Implementation of virtual instrumentation for multiparametric technical system monitoring. In: SGEM 2013: 13th Int. Multidisciplinary Sci. Geoconf., Vol. 1; 16-22 June, 2013, Albena, Bulgaria. Sofia: STEF92 Technology, 2013, pp. 139-144. ISBN 978-954-91818-9-0.

[Nikitin 2017] Nikitin, Y.R. and Teplyakova, M.Y. Transport robot control by voice commands. Intelligent Systems in Manufacturing (Upravlenie transportnym robotom golosovymi komandami). 2017, Vol. 15, No. 3, pp. 112–117. DOI: 10.22213/2410-9304-2017-3-112-117. (in Russian).

[Nikitin 2020] Nikitin, Y.R. BLDC motor identification based on optimal control model in the state space. IOP Conf. Series: Materials Science and Eng., 2020, Vol. 971, Modeling of technical systems. CAD/CAM/CAE technologies.

[Nikitin 2020] Nikitin, Y., Bozek, P. and Peterka, J. Logical-Linguistic Model of Diagnostics of Electric Drives with Sensors Support. Sensors 2020, Vol. 20, Issue 16, DOI 10.3390/s20164429

[Nikitin 2022] Nikitin Y., Turygin A. and Stollmann V. Multilevel control of a transport robot. MM Science Journal, 2022, Vol. June, pp. 5662-5669. DOI: 10.17973/MMSJ.2022_06_2022089.

[Qazizada 2016] Qazizada, M.E. and Pivarciova, E. Mobile robot controlling possibilities of inertial navigation system. In: Int. Conf. on Manuf. Eng. and Materials ICMEM 2016, Slovakia, pp. 404–413.

[Staple 2021] Staple, D. Learn Robotics Programming: Build and control AI-enabled autonomous robots using the Raspberry Pi and Python. 2nd edition. Packt Publishing, 2021. 602 p. ISBN 9781839218804.

[Tlach 2017] Tlach, V., Cisar, M., Kuric, I. and Zajacko, I. Determination of the Industrial Robot Positioning Performance. 13th International Conference on Modern Technologies in Manufacturing, 2017, Vol. 137, DOI: 10.1051/matecconf/201713701004.

[Tlach 2019] Tlach V., Kuric I., Sagova Z. and Zajacko I.: Collaborative assembly task realization using selected type of a human-robot interaction, 13th International Scientific Conference On Sustainable, Modern And Safe Transport 2019, Vol. 40, pp. 541–547, DOI: 10.1016/j.trpro.2019.07.078-

[Virgala 2014] Virgala, I., Kelemen, M., Varga, M. and Kurylo, P. Analyzing, Modeling and Simulation of Humanoid Robot Hand Motion. Modelling of mechanical and mechatronic systems 2014, Vol. 96, pp. 489–499.

CONTACTS:

Dr.h.c. Prof. Pavol Bozek, PhD.
Slovak University of Technology in Bratislava
Faculty of Materials Science and Technology
Institute of Production Technologies
J. Bottu 25, 917 24 Trnava, Slovakia
Tel.: 00421 903 240 686,
E-mail: pavol.bozek@stuba.sk