АҚПАРАТТЫҚ-КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР
ИНФОРМАЦИОННО-КОММУНИКАЦИОННЫЕ ТЕХНОЛОГИИ
INFORMATION AND COMMUNICATION TECHNOLOGIES

АВТОМАТТАНДЫРУ ЖӘНЕ БАСҚАРУ
АВТОМАТИЗАЦИЯ И УПРАВЛЕНИЕ
AUTOMATION AND CONTROL

DOI 10.51885/1561-4212_2025_2_144
IRSTI 28.17.31

A.L. Krasavin[1], G.M. Nazenova[2], D.L. Alontseva[3], K. Askaduly[4]
D. Serikbayev East Kazakhstan Technical University, Ust-Kamenogorsk, Kazakhstan
[1]*E-mail: akrassavin@ektu.kz*
[2]*E-mail: nazenova.g@edu.ektu.kz**
[3]*E-mail: dalontseva@ektu.kz*
[4]*E-mail: kaskaduly@edu.ektu.kz*

TRAJECTORY TRACKING SIMULATION OF A WHEELED MOBILE ROBOT

ДӨҢГЕЛЕКТІ МОБИЛДІ РОБОТТЫҢ ҚОЗҒАЛЫС ТРАЕКТОРИЯСЫН МОДЕЛЬДЕУ

МОДЕЛИРОВАНИЕ ТРАЕКТОРИИ ДВИЖЕНИЯ КОЛЕСНОГО МОБИЛЬНОГО РОБОТА

Abstract. *This paper presents an event-driven structure for modeling trajectory tracking of a wheeled mobile robot. The robot's controller is modeled by a virtual microcontroller. Control algorithms are implemented in event handlers, which can be viewed as analogs of interrupt handlers of a real microcontroller. The goal of the research was to create a simulator that allows testing trajectory control methods for wheeled robots, implemented as programs for microcontrollers operating as part of control systems. In this case, for one dynamic model of the robot, different control algorithms are implemented by different program code of the event handlers of the virtual controller. The feature of the developed simulator of the microcontroller control system for a wheeled mobile robot is that it allows testing the program code implementing the software implementation of the mobile robot control controller. This feature distinguishes this new type of simulator from simulators created using traditional methods and opens up good prospects for using such simulators in education for organizing laboratory work in disciplines related to the design of microcontroller control systems and allows using virtual equivalents of control system components that are not available or poorly suited for conducting full-scale experiments.*

Keywords: *Trajectory tracking simulation, wheeled mobile robot, event-driven programming, dynamic modeling of a mobile robot.*

Аңдатпа. *Бұл мақалада доңғалақты мобильді роботтың траекториясын бақылауды модельдеуге арналған оқиғаға негізделген құрылым ұсынылған. Робот контроллері виртуалды микроконтроллер арқылы модельденеді. Басқару алгоритмдері оқиға өңдеушілерінде жүзеге асырылады, олардыі нақты микроконтроллердің үзу өңдеулерінің аналогтары деп санауға болады. Зерттеудің мақсаты басқару жүйелерінің бөлігі ретінде жұмыс істейтін микроконтроллерлерге арналған бағдарламалар ретінде іске асырылатын доңғалақты роботтардың траекториясын басқару әдістерін сынауға мүмкіндік беретін симуляторды құру болды. Сонымен қатар, бір динамикалық робот үлгісі үшін әртүрлі басқару алгоритмдері виртуалды контроллер оқиғаларын өңдеушілердің әртүрлі бағдарламалық кодтары арқылы жүзеге асырылады. Доңғалақты мобильді роботқа арналған микроконтроллерді басқару жүйесінің әзірленген симуляторының ерекшелігі, ол мобильді роботты басқару контроллерінің бағдарламалық іске асыруын жүзеге асыратын бағдарламалық кодты сынауға мүмкіндік береді. Бұл мүмкіндік тренажерлардың жаңа түрін дәстүрлі әдістермен жасалған тренажерлерден ерекшелендіреді, және микроконтроллерлерді басқару жүйелерін жобалаумен байланысты пәндер бойынша зертханалық жұмыстарды ұйымдастыру үшін білім беруде осы тектес тренажерларды пайдаланудың жақсы перспективаларын ашады, және толық ауқымды эксперименттер жүргізу үшін қол жетімсіз немесе жарамсыз басқару жүйесінің құрамдас бөліктерінің*

виртуалды эквиваленттерін пайдалануға мүмкіндік береді.

Түйін сөздер: *Траекторияны қадағалауды модельдеу, доңғалақты мобильді робот, оқиғаға негізделген бағдарламалау, мобильді роботты динамикалық модельдеу.*

Аннотация. *В данной статье представлена событийно-управляемая структура для моделирования отслеживания траектории колесного мобильного робота. Контроллер робота моделируется виртуальным микроконтроллером. Алгоритмы управления реализуются в обработчиках событий, которые можно рассматривать как аналоги обработчиков прерываний реального микроконтроллера. Целью исследования было создание симулятора, позволяющего тестировать методы траекторного управления колесными роботами, реализуемыми как программы для микроконтроллеров, работающих в составе систем управления. При этом для одной динамической модели робота различные алгоритмы управления реализуются разным программным кодом обработчиков событий виртуального контроллера. Особенность разработанного симулятора микроконтроллерной системы управления колесным мобильным роботом в том, что он позволяет тестировать программный код, реализующий программную реализацию контроллера управления мобильным роботом. Эта черта отличает новый тип симуляторов от симуляторов, создаваемых традиционными методами, и открывает хорошие перспективы применения симуляторов такого рода в образовании для организации лабораторных работ по дисциплинам, связанным с проектированием микроконтроллерных систем управления и позволяет использовать виртуальные эквиваленты компонентов системы управления, недоступные или малопригодные для проведения натурных экспериментов.*

Ключевые слова: *Моделирование отслеживания траектории, колесный мобильный робот, событийно-управляемое программирование, динамическое моделирование мобильного робота.*

Introduction. The two-wheeled differential drive robot is a widely popular design among mobile robots (Manoharan et al., 2024; Hassan & Saleem, 2022; Mata-Machuca et al., 2021). This design features two independently powered wheels and a third, freely rotating castor wheel for stability. By controlling the power supplied to the motors, the robot can move forward, rotate in place, or navigate along any arbitrary curve within a plane (Zhao et al., 2023; He et al., 2024). Due to the simplicity of the design and availability of the necessary components, Differential Drive Whiled Robot (DDWR) robots are popular in amateur robotics and are widely used in education (Fahmizal et al., 2024). MATLAB/Simulink environment is often used for computer simulation of DDWR robots (Channareth et al., 2022; Fernando & GanLim, 2021). This approach to computer simulation of wheeled mobile robot control systems is very popular and well-documented in the technical literature. In particular, articles (Nair et al., 2016; Mirzaeinejad et al., 2018; Gao et al., 2021) provide a detailed description of the application of this approach to the task of computer simulation of the trajectory control process of a DDWR robot that we are interested in. Typically, when using the Simulink/Matlab combination, the control laws are defined using the same graphical design methods as the physical-mathematical model of the robot. This is a very convenient approach for modeling control systems that operate on continuous signals, as the modeling environment offers numerous ready-to-use tools for signal operations described by continuous functions of time, such as ordinary differential equation (ODE) solvers, differentiation and integration blocks, and so on. However, in the vast majority of cases, mobile robots (particularly DDWRs) operate under the control of a microcontroller or a single-board computer. The data read by the microcontroller unit (MCU) from sensors (feedback signals of the microcontroller control system) are discrete, not continuous signals. The mobile robot control controller is implemented in software and represents a discrete control system. As is known, even in cases where a microcontroller control system simulates the operation of a continuous control system (as an example, one can cite numerous software implementations of PID controllers), the specifics of the control system implementation often make its simulation inadequate by means intended for simulating continuous systems. On the other hand, debugging microcontroller programs controlling mobile robots (especially autonomous mobile robots) is often extremely difficult. It would be highly desirable to have a simulation environment for microcontroller control systems for mobile robots that allows simulating the operation of the microcontroller controlling the system. It is particularly desirable that such a simulation environment allows

simulating the execution of specific software code loaded into the microcontroller, similar to the Proteus electronic circuit simulation environment. This paper describes a prototype software simulator for a DDWR robot's microcontroller control system, which possesses these desirable features. The main distinguishing feature of our proposed scheme for simulating the operation of a wheeled mobile robot's microcontroller control system is the concept of *a virtual microcontroller* that interacts with *the hardware environment simulator*. The software virtual microcontroller is a set of user-defined functions simulating interrupt handlers of the real microcontroller controlling the DDWR robot. During the simulation session, the hardware environment simulator code calls these functions, simulating hardware interrupts from timers setting the control system timing, as well as interrupts generated by the microcontroller's built-in capture blocks. The simulation system provides the user with an API that allows reading data from virtual sensors generated by the hardware environment simulator during the execution of virtual interrupt handler code, as well as setting so-called control codes, simulating robot motor control. The set of virtual interrupt handlers corresponds to the event-driven MCU control application framework, practically applied for creating embedded applications. This framework is described further in Section 4 of this paper. The hardware environment of the simulated microcontroller control system for the mobile robot is described in Section 3. Obviously, in order to create a simulator, it is necessary to have both a kinematic and a dynamic model of the DDWR. Descriptions of the kinematic and dynamic models of the DDWR robot used are given in Sections 1 and 2 respectively.

1. Kinematic and dynamic models of a DDWR robot.

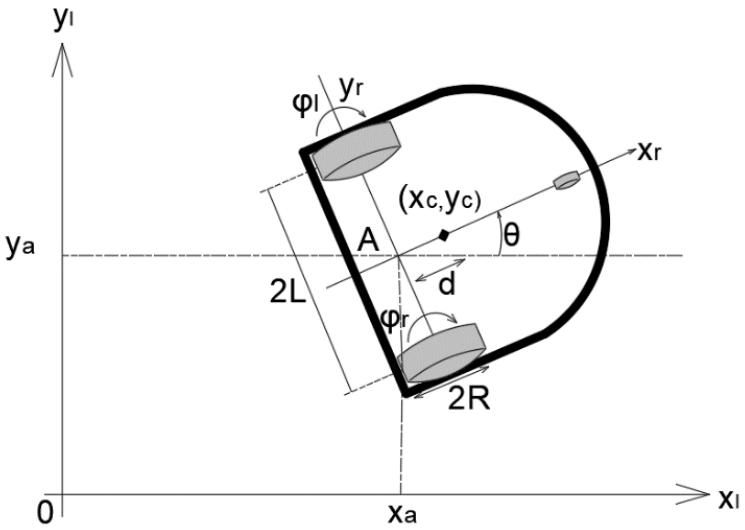

Figure 1. A diagram of a DDWR robot

Note – compiled by the authors based on (Dhaouadi & Hatab, 2013)

Let $\{X_I, Y_I\}$ denote the fixed Cartesian inertial frame and $\{X_r, Y_r\}$ denote the moving Cartesian frame of the robot. The origin of the robot frame is at point A, located at the midpoint of the axis between the wheels. The center of mass of the robot (point C) is assumed to be located on the axis of symmetry, at a distance d from point A.

The motion of the differential drive mobile robot is characterized by two nonholonomic constraint equations, which are obtained using two main assumptions:

No lateral slip motion: This constraint simply means that the robot can move only in a curved motion (forward and backward) but not sideward. In the robot frame, this condition means that the velocity of the center-point A is zero along the lateral axis:

$$\dot{y}_\alpha^r = 0 \tag{1}$$

Pure rolling constraint: The pure rolling constraint represents the fact that each wheel maintains a one contact point P with the ground. There is no slipping of the wheel in its longitudinal axis and no skidding in its orthogonal axis. The velocities of the contact points in the robot frame are related to the wheel velocities by:

$$\begin{cases} v_{pR} = R\dot{\varphi}_R \\ v_{pL} = R\dot{\varphi}_L \end{cases} \tag{2}$$

the three constraint equations can be written in the following matrix form:

$$\Lambda(q)\dot{q} = 0, \tag{3}$$

where

$$\Lambda(q) = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 & 0 \\ \cos\theta & \sin\theta & L & -R & 0 \\ \cos\theta & \sin\theta & -L & 0 & -R \end{bmatrix} \tag{4}$$

and

$$\dot{q} = \left[\dot{x}_\alpha\ \dot{y}_\alpha\ \dot{\theta}\ \dot{\varphi}_R\ \dot{\varphi}_L\right]^T \tag{5}$$

The kinematic model of the differential drive mobile robot is actually quite simple and is described by two well-known equations (6) and (7) for the linear velocity of the robot's center of mass v and the angular velocity $\omega = \dot{\theta}$ of the robot.

$$v = \frac{v_R + v_L}{2} = R\frac{(\varphi_R + \varphi_L)}{2} \tag{6}$$

$$\omega = \frac{v_R - v_L}{2L} = \frac{(\dot{\varphi}_R - \dot{\varphi}_L)}{2} \tag{7}$$

The Lagrange-Euler equations for the DDWR have the form (8):

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) + \frac{\partial L}{\partial q_i} = F - \Lambda^T(q)\lambda, \tag{8}$$

where L=T-V is the Lagrangian function, T, is the kinetic energy of the system, V is the potential energy of the system, qi are the generalized coordinates, F is the generalized force vector, Λ is the constraints matrix, and λ is the vector of Lagrange multipliers associated with the constraints. As shown in the paper (Dhaouadi & Hatab, 2013), the equations in Lagrange form (8) are equivalent to the equation of motion of the mobile robot (9).

$$\begin{cases} \left(m + \dfrac{2I_w}{R^2}\right)\dot{v} - m_c d\omega^2 = \dfrac{1}{R}(\tau_r + \tau_L) \\ \left(I + \dfrac{2L^2}{R^2}I_w\right)\dot{\omega} + m_c d\omega v = \dfrac{L}{R}(\tau_r - \tau_L), \end{cases} \tag{9}$$

where τ_R and τ_L – moments of friction forces applied to the right and left wheels of the robot respectively (since we assume that the wheels move without slipping, τ_R and τ_L are equal to the

torque of the corresponding electric motors), d – distance from the center of mass of the robot to the axis passing through the centers of the robot's wheels, m_c – mass of the mobile robot without wheels and motors attached to the wheels, $m = m_c + 2 \cdot m_w$ – the total mass of the robot (m_w – mass of the wheel with a gearmotor attached to it), $I = I_c + m_c d^2 + 2m_w L^2 + 2I_m$ the moment of inertia of the mobile robot about the vertical axis passing through point A (the center of the robot's wheel axle) (I_c – the moment of inertia of the robot (without wheels and motors) about the vertical axis passing through the center of mass, I_m - the moment of inertia of the wheel about the vertical axis passing through the center of the wheel), I_w – the moment of inertia of the robot wheel about the horizontal axis passing through the center of the wheel. Next, taking into account the kinematic equalities (6) and (7), we obtain from (9) the equations of motion in the form (10).

$$\begin{cases} \ddot{\varphi}_R = f_1(\dot{\varphi}_R, \dot{\varphi}_L . \tau_R, \tau_L) \\ \ddot{\varphi}_L = f_2(\dot{\varphi}_R, \dot{\varphi}_L . \tau_R, \tau_L) \end{cases} \quad (10)$$

We do not give the analytical expressions for f_1 and f_2 due to their complexity, but we note that obtaining these expressions is a completely trivial task. The DC commutator motor, installed on the robot are described by equations (11).

$$\begin{cases} v_a = R_\alpha + L_\alpha \dfrac{di_a}{dt} + e_\alpha \\ e_\alpha = K_b \omega_{m\alpha} \\ e_\alpha = K_i i_\alpha \\ \tau = N\tau_m \end{cases}, \quad (11)$$

where, i_a is the armature current,(R_α, L_α) (where the index α takes the values $\{R, L\}$) is the resistance and inductance of the armature winding respectively, e_α is the back emf, $\omega_{m\alpha}$ is the rotor angular speed, τ_m is the motor torque, (K_i , K_b) are the torque constant and back emf constant respectively, N is the gear ratio, and is τ the output torque applied to the wheel. Since in the DDMR the motors are mechanically coupled to the robot wheels through the gears, the mechanical equations of motion of the motors are linked directly with the mechanical dynamics of the DDMR. Therefore, for each dc motor, the expression (12) holds.

$$\omega_{m\alpha} = N\dot{\varphi}_\alpha \quad (12)$$

From system (11) taking into account (12), we can obtain the equation (13):

$$\dot{\tau}_a = \left(\frac{K_t}{L_a}\right) v_a - \left(\frac{R_a K_t}{L_a}\right) \tau_\alpha - \left(\frac{K_t K_b N}{L_a}\right) \dot{\varphi}_\alpha \quad (13)$$

In the four-dimensional phase space $(\dot{\varphi}_R, \dot{\varphi}_L, \tau_R, \tau_L)$ equations (10) and (13) define the equation of motion of the system in canonical form (14):

$$\dot{x} = f(x, u), \quad (14)$$

where $x(t) = \left(\dot{\varphi}_R(t), \dot{\varphi}_L(t), \tau_R(t), \tau_L(t)\right)$ – the state vector of the system, and $u(t) = \left(u_L(t), u_R(t)\right)$ – the control signal vector.

2. Structure of the Microcontroller Control System for a Two-Wheeled Differential Drive Mobile Robot. We assume the mobile robot is controlled by a microcontroller or a single-board computer. Consequently, we consider a discrete-time control system, whose block diagram is depicted in Figure 2. We assume that the mobile robot is equipped with DC commutator gearmotors with integrated rotary encoders. The rotary encoder outputs a predefined number of pulses per revolution of the gearbox shaft. Processing data from the integrated encoders allows achieving two goals. First, based on the encoder data, it is possible to reconstruct discrete signals

representing the dependence of the mobile robot's wheel rotation angular velocity on time. Second, access to the encoder data allows building an odometric system for determining the position and orientation of the mobile robot (obviously, in the absence of slippage of the robot's wheels).

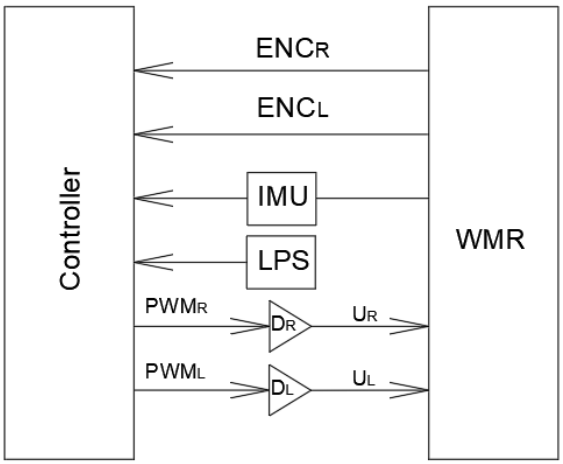

Figure 2. Block Diagram of the Microcontroller Control System for a Two-Wheeled Mobile Robot
Note – compiled by the author

The controller is a microprocessor board or a single-board computer. D_R and D_L denote the DC commutator motor drivers of the mobile robot. PWM_R and PWM_L are low-voltage pulse-width modulated signals controlling the motor drivers of the right and left wheel, respectively. ENC_R and ENC_L are the signals from the encoders of the robot's right and left motors, IMU is a MEMS inertial sensor unit, LPS (Local Positioning System) is a unit of the local positioning system.

3. Structure of a microcontroller program controlling a mobile robot. Timing organization. Let T denote the duration of the working cycle of the microcontroller-based control system. At time instants $t_k = k \cdot T$, where k is an integer index, the control signal $u(t)$ is updated. During the working cycle (over the time interval $[t_k, t_{k+1}]$) the controller is engaged in collecting data from sensors. The timing is set by one of the microcontroller's built-in timers. Timing is set by one of the microcontroller's built-in timers. Timer interrupts occur at a fixed time interval τ. During timer interrupt processing, sensors are polled (such as MEMS inertial sensors and local positioning systems) and the acquired readings of discrete signals are stored in corresponding data structures (e.g., circular buffers) in the microcontroller's RAM memory. Therefore, the interval τ can be considered as the discretization step for feedback signals. The duration of the working cycle T is chosen so that it contains a whole number M of time intervals τ, i.e. $T = M \cdot T$. The time instants t_k are determined by the overflow of a software-implemented counter modulo M (the counter is incremented at each timer interrupt). Encoder pulses are processed by the microcontroller's built-in capture units. The use of hardware capture units allows measuring the time intervals between encoder pulse edges in the clock cycles of the timer source associated with the respective capture unit. To organize encoder data reading, interrupt processing is performed, the source of which are the engaged capture units. Interrupts are generated upon the arrival of encoder pulse edges (it is obvious that interrupts generated by capture units are not synchronized with timer interrupts). After the working cycle is completed, code implementing a software control unit for managing the mobile robot is executed, i.e., control codes c_k^1 и c_k^2 are calculated

based on the readings of discrete sensor signals (feedback signals). Subsequently, after control code calculation, depending on the hardware implementation of the robot's motor drive, the control codes are either loaded into the duty cycle control registers of the pulse-width modulated (PWM) signal (when controlling motors using H-bridge drivers) or loaded into digital-to-analog converters (DACs) controlling the motor drive with analog control input, or transmitted over one of the standard serial communication interfaces (such as UART, SPI, I2C) to digitally controlled drives. Let us $u_1^*(t)$ and $u_2^*(t)$ as step functions, defined by the equations (15).

$$\forall t \in [t_k, t_{k+1}] \;\; u_j^*(t) = u_j^k \;\; u_j^k = u_{max} \cdot \left[\frac{c_k}{c_{max}}\right], \tag{15}$$

where $j = \{1,2\}$, and $c_{max} = 2^N - 1$. In the last equation, the integer N p is the resolution of the PWM or DAC, used for controlling the mobile robot's motors. Obviously, it is assumed that the values of the control codes c_k^1 и c_k^2 are limited to the value c_{max}, т.е. $\forall k \; c_k^1, c_k^2 \in \{0,1, \dots, c_{max}\}$. Assuredly, the actual control signals should be described by a continuous function $u_j(t)$. We adopt a control signal model of the form (16), where the parameter value $\lambda > 0$ is chosen such that $\frac{1}{\lambda} \ll T$.

$$\forall t \in [t_k, t_{k+1}] \;\; u_j(t) = u_j^{k-1} + \left(1 - e^{-\lambda(t-t_k)}\right)\left(u_j^k - u_j^{k-1}\right), \tag{16}$$

4. Structure and principles of operation of a simulator for an autonomous wheeled mobile robot with a microcontroller-based control system. The simulator is structurally divided into two modules: a hardware environment simulator module and a virtual control controller module. Functionally, the virtual controller represents four user-defined functions – three virtual interrupt handlers with predefined names and signatures, and also a user-defined $Init()$ function, as well as the simulator's API – a set of functions implemented in the hardware environment simulator code, available to the user when writing the code for virtual interrupt handlers. All user-defined functions of the virtual controller are called only from the hardware environment simulator code. The $Init()$ function is called by the hardware environment simulator at the beginning of the simulation process. It is assumed that user-defined variables are initialized and dynamic data structures are created in the code of the $Init()$ function. Furthermore, it is assumed that the call to the $Init()$ function coincides with the beginning of the first working cycle of the simulation process and the user has the ability to set the initial values of the control codes c_0^1 and c_0^2 in the code of the $Init()$ function by calling the simulator API function $SetControlCodes(int\ cc_1, int\ cc_2)$. The actual simulation process is based on sequentially calculating the dependence of the system state vector $x(t) = \left(\dot{\varphi}_R(t), \dot{\varphi}_L(t), \tau_R(t), \tau_L(t)\right)$ on time at time intervals $[t_k, t_{k+1}]$ working cycles of the virtual mobile robot controller). At the beginning of each simulation step, the hardware environment simulator calls a user-defined virtual interrupt handler function from timer $timer\_1\_interrupt\_handler()$. It is assumed that timer 1 interrupts correspond to the end of the current working cycle of the virtual microcontroller. In the interrupt handler, new values of the control codes c_k^1 and c_k^2 are set for the next k – the working cycle ($[t_k, t_{k+1}]$ time interval). Since the values of the control codes c_k^1 and c_k^2 Since the values of the control codes $u(t)$ on the interval $[t_k, t_{k+1}]$ according to equations (16), solving the system dynamics equation ($\dot{x} = f(x, u)$) on the interval $[t_k, t_{k+1}]$ reduces to an initial value problem for an ODE system of the form (17):

$$x(t_k) = x_{t_k} \;\; \dot{x} = g(x, t), \tag{17}$$

where the initial value x_{t_k} is the value of $x_{pr}(t_k)$ ($x_{pr}(t)$ the dependence of the system state vector $x(t)$ on time on the interval $[t_{k-1}, t_k]$, calculated on the previous simulation step, and the function $g(x, t)$ is defined by equation (18).

$$g(x,t) = f(x,u(t)) \quad (18)$$

For numerical solution of the system of ordinary differential equations (4), we used the solve_ivp ODE solver from the SciPy library. Among the input parameters of the solve_ivp there is the *method* parameter, which allows us to choose one of several numerical methods implemented in the library for solving the initial value problem for ODE systems.

We used the Explicit Runge-Kutta method of order 5 (17) (Dormand & Prince, 1980). The solution of the ODE system (18) gives discretely defined functions $\dot{\varphi}_R(t)$ and $\dot{\varphi}_L(t)$ on the interval $[t_k, t_{k+1}]$ (dependencies on time of the angular velocities of rotation of the right and left wheels of the mobile robot). First, based on these data, kinematic characteristics such as the dependence of the magnitude of the linear velocity of the mobile robot on time and the dependence of the angular velocity of the robot on time are restored. Combining these data with the data obtained in previous steps of the simulation process, numerical integration methods are used to calculate the robot's trajectory on the plane. The next step of the simulation process is to perform a sequence of calls to the user-defined virtual interrupt handler function from timer $timer\_2\_interrupt\_handler()$. It is assumed that timer 2 interrupts correspond to the cycles of data collection by the virtual microcontroller program. In the $timer\_2\_interrupt\_handler()$ interrupt handler, data on the current position and orientation of the virtual robot is typically requested. After the procedures described above are completed, the simulation proceeds to the next step for the next working cycle (time interval $[t_{k+1}, t_{k+2}]$).

While articles (Zangina et al., 2020) and (Mikova, 2023) provide valuable insights into kinematic and dynamic modeling, their simulation approaches are less suited for capturing the complexities of microcontroller-based control systems. Our proposed method distinguishes itself with a more realistic and practical simulation approach, making it a more powerful tool for testing and evaluating control algorithms for wheeled mobile robots, particularly in educational settings.

Conclusions. The wheeled mobile robot microcontroller system simulator presented in this paper enables testing of program code that implements the software for the robot's control system. This feature distinguishes the presented new type of simulator from simulators created using traditional methods in a radical way, opening up promising opportunities for using such simulators in education.

Primarily, simulators of this type are well-suited for organizing laboratory work in disciplines related to designing microcontroller control systems. The simple API of the simulation system, available when writing virtual controller code, allows for a focus on control problems without being distracted by numerous minor issues characteristic of low-level embedded programming. Additionally, the simulator allows for the use of virtual equivalents of control system components that are unavailable or poorly suited for conducting full-scale experiments.

Conflict of interest. The authors declare no conflict of interest.

Acknowledgements. This research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP19679327).

References

Manoharan, S. K., Raghavan, D., Megalingam, R. K., Parakat, P., & Sudheesh, S. K. (2024). Fine-tuning mobility: PID driven velocity control optimization for two-wheel differential drive robot. Proceedings of the 2024 Parul International Conference on Engineering and Technology (PICET). https://doi.org/10.1109/PICET60765.2024.10716137

Hassan, N., & Saleem, A. (2022). Neural network-based adaptive controller for trajectory tracking of wheeled mobile robots. IEEE Access, 10(4), 13582–13597. https://doi.org/10.1109/ACCESS.2022.3146970

Mata-Machuca, J. L., Zarazua, L. F., & Aguilar-Lopez, R. (2021). Experimental verification of the leader-follower formation control of two wheeled mobile robots with obstacle avoidance. IEEE Latin America Transactions, 19(8), 1417–1424. https://doi.org/10.1109/TLA.2021.9475873

Zhao, T., Wang, Y., Li, G., & Hou, G. (2023). Lyapunov-based global trajectory tracking control of wheeled mobile robot. Journal of Physics: Conference Series, 2478(10), 102018. https://doi.org/10.1088/1742-6596/2478/10/102018

He, X., Han, X., Wei, T., & Li, X. (2024). Tracking control of wheeled mobile robots via intermittent control. Mathematical Biosciences & Engineering, 21, 3774-3783. https://doi.org/10.3934/mbe.2024167

Fahmizal, F., Pratikno, M. S., Isnianto, H. N., Mayub, A., Maghfiroh, H., & Anugrah, P. (2024). Control and navigation of differential drive mobile robot with PID and Hector SLAM: Simulation and implementation. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI), 10(3), 594-607. https://doi.org/10.26555/jiteki.v10i3.29428

Channareth, S., Vannsith, L., & Virbora, N. (2022). Experimental in head tracking control of a four-OMNI wheeled mobile robot system. Indonesian Journal of Engineering and Science, 3(3), 15-26. https://doi.org/10.51630/ijes.v3i3.67

Fernando, A., & GanLim, L. (2021). Velocity analysis of a six-wheel modular mobile robot using MATLAB-Simulink. IOP Conference Series: Materials Science and Engineering, 1109(1), 012037. https://doi.org/10.1088/1757-899X/1109/1/012037

Nair, L., & Saju, K. K. (2016). Modelling and trajectory tracking of wheeled mobile robots. Procedia Technology, 24, 538-545. https://doi.org/10.1016/j.protcy.2016.05.094

Mirzaeinejad, H., & Shafei, A. M. (2018). Modeling and trajectory tracking control of a two-wheeled mobile robot: Gibbs–Appell and prediction-based approaches. Robotica, 36(10), 1551-1570. https://doi.org/10.1017/S0263574718000565

Gao, X., Yan, L., & Gerada, C. (2021). Modeling and analysis in trajectory tracking control for wheeled mobile robots with wheel skidding and slipping: Disturbance rejection perspective. Actuators, 10(9), 1-15. https://doi.org/10.3390/act10090222

Dhaouadi, R., & Hatab, A. (2013). Dynamic modelling of differential-drive mobile robots using Lagrange and Newton-Euler methodologies: A unified framework. Advances in Robotics and Automation, 2(2). https://doi.org/10.4172/2168-9695.1000107

Dormand, J. R., & Prince, P. J. (1980). A family of embedded Runge-Kutta formulae. Journal of Computational and Applied Mathematics, 6(1), 19-26. https://doi.org/10.1016/0771-050X(80)90013-3

Zangina, U., Buyamin, S., Abidin, M., Mahmud, M., & Hasan, H. S. (2020). Non-linear PID controller for trajectory tracking of a differential drive mobile robot. Journal of Mechanical Engineering Research and Developments, 43(7), 255-270.

Mikova, L. (2023). Design of a functional model of a three-wheeled mobile robot. Technical Sciences and Technologies, 33(3), 53-58. https://doi.org/10.25140/2411-5363-2023-3(33)-53-58

Information about authors

Krasavin Alexander – a candidate of physical and mathematical sciences, D. Serikbaev East Kazakhstan technical university, Ust-Kamenogorsk, Kazakhstan, akrassavin@ektu.kz

Nazenova Gaukhar – a doctoral student, D. Serikbaev East Kazakhstan technical university, Ust-Kamenogorsk, Kazakhstan, nazenova.g@edu.ektu.kz, ORCID: 0000-0002-5415-094X, +7 707 903 12 09

Alontseva Darya – a professor, D. Serikbaev East Kazakhstan technical university, Ust-Kamenogorsk, Kazakhstan, dalontseva@ektu.kz

Askaduly Kanat – a doctoral student, D. Serikbaev East Kazakhstan technical university, Ust-Kamenogorsk, Kazakhstan, kaskaduly@edu.ektu.kz