Trajectory optimization using learning from demonstration with meta-heuristic grey wolf algorithm

Adam Pawlowski[1], Slawomir Romaniuk[1], Zbigniew Kulesza[1], Milica Petrovic[2]
[1]Faculty of Electrical Engineering, Bialystok University of Technology, Bialystok, Poland
[2]Faculty of Mechanical Engineering, University of Belgrade, Belgrade, Serbia

Article Info

Article history:

Received Jun 9, 2022
Revised Sep 11, 2022
Accepted Sep 21, 2022

Keywords:

Differential drive
Grey wolf optimizer
Learning from demonstration
Trajectory optimization
Ultra-wideband system

ABSTRACT

Nowadays, most robotic systems perform their tasks in an environment that is generally known. Thus, robot's trajectory can be planned in advance depending on a given task. However, as a part of modern manufacturing systems which are faced with the requirements to produce high product variety, mobile robots should be flexible to adapt to changing and diverse environments and needs. In such scenarios, a modification of the task or a change in the environment, forces the operator to modify robot's trajectory. Such modification is usually expensive and time-consuming, as experienced engineers must be involved to program robot's movements. The current paper presents a solution to this problem by simplifying the process of teaching the robot a new trajectory. The proposed method generates a trajectory based on an initial raw demonstration of its shape. The new trajectory is generated in such a way that the errors between the actual and target end positions and orientations of the robot are minimized. To minimize those errors, the grey wolf optimization (GWO) algorithm is applied. The proposed approach is demonstrated for a two-wheeled mobile robot. Simulation and experimental results confirm high accuracy of generated trajectories.

Corresponding Author:

Zbigniew Kulesza
Faculty of Electrical Engineering, Bialystok University of Technology
Wiejska 45D, 15-351 Bialystok, Poland
Email: z.kulesza@pb.edu.pl

1. INTRODUCTION

Nowadays, usage of mobile autonomous platforms in the field of manufacturing applications is becoming more and more popular. Engagement of the mobile platforms in such applications builds an intensive demand for solving several crucial problems, which one amongst them is the proper, optimal path planning algorithm. An interesting example of the production process is when the workpiece is placed inside a strictly defined workspace in a completely random way, so that its position and orientation are variable, and the path has to be continuously adapted. Such cases can occur, for example, in processes of welding [1], [2], painting [3], [4], cleaning [5], transporting [6].

Several techniques can be used for robot optimal path planning, when moving during manufacturing processes. In [7] the authors present an original modification of the grey wolf optimization (GWO) algorithm to find an optimal path for a multi robot case in order to maintain proper distance from other robots and obstacles. GWO is a meta-heuristic algorithm inspired by grey wolves. Benchmarks presented in [8] show that the GWO algorithm acquires better performance in minimizing test functions in relation to other algorithms such as particle swarm optimization (PSO), gravitational search algorithm (GSA), differential evolution (DE), fast evolutionary programming (FEP) techniques. A solution presented in [9] utilizes the fish swarm algorithm to find a path avoiding obstacles existing in the surrounding environment of the robot. At the final step the authors apply Bessel curve theory to achieve smoothed trajectory. Usage of the evolutionary algorithms for the purpose of path optimization is also frequently spotted in the form of many different modifications. As an example, the mutated cuckoo optimization algorithm can be mentioned [10], used as the global path planner together with a genetic algorithm. Bioinspired algorithms are now widely implemented in both 2D and 3D trajectory planning. In the paper [11], the authors used GWO and golden eagle optimizer (GEO) and their modifications to plan the unmanned aerial vehicle (UAV) path carrying out inspection works. When cooperation of several robots is required to process or manufacture a given element, artificial potential fields may be used to maintain proper structure and distances between the platforms [12], [13], [14].

If the process of path planning must be sped up, particularly in the case of variable pose of the manufactured element, the learning from demonstration (LfD) approach [15], [16] can be used instead of building the entire path from the beginning. At the first step of the LfD, the operator remotely moves the robot around the processed object roughly driving through the work spots and focusing on proper avoidance of the obstacles [17]. Next, LfD utilizes the demonstrated trajectory in the optimization algorithm to achieve satisfactory trajectory indicators. There are different implementations of robot learning algorithms. The most popular of them are neural networks, optimization algorithms and dynamic movement primitives (DMP) [18]. Metaheuristic optimization methods, such as genetic algorithms, are gaining more and more popularity due to their speed as well as efficiency to find optimal solution. On the other hand, neural networks show greater adaptation to changing conditions, but at the same time consume large computing resources. Another method used in LfD is DMP. This method can be applied to model robot's motion based on demonstration [19]. However, it requires the implementation of complex mathematical operations.

Navigational systems are used to increase precision and create a closed feedback loop for the robot position controller. In order to verify performance of different navigational algorithms and to evaluate accuracy of the generated trajectory, it is required to measure position of the mobile platform. In indoor applications global navigational satellite systems (GNSS) may not be effectively used. Therefore, other navigational systems are discussed shortly.

Lateration based positioning systems, apart of anchors positions, utilize the distance between the tag and the anchors. Those systems can be further classified by the way the distance is measured or by the utilized measurement technology [20]. Most accurate way to determine the distance is the time of flight (TOF) or time-difference of arrival (TDOA). Beside those two, there are also phase of arrival (POA) methods and methods that use a received signal strength indicator (RSSI). Lateration based positioning systems utilizing the following measurement technologies were built and successfully evaluated: magnetic field, ultrasonic, infrared, radio frequency (Wi-Fi, Bluetooth), and vision, which can also be used for obstacle avoidance applications [21], [22]. Popular ultrasonic based systems are characterized by high accuracy (up to 1 cm), yet still suffer of low range and significant interference from wind, and other sound sources [23]. A promising radio frequency (RF) technology for indoor positioning purposes is the ultra-wide band (UWB) technique [24]. Accuracy of the UWB based positioning systems can be established in the range from 10 to 30 cm, yet UWB signals can propagate through thin obstacles, and are resistant to most of electromagnetic interferences.

In the current paper, the problem of mobile robot path planning, in which processed elements are sequentially randomly placed in the predefined area, is considered. The task is realized by a mobile platform, having the form of a two-wheeled robot, equipped with a proper work tool, and operating in the planar surface. The result of the proposed algorithm is not the path in the form of successive robot positions, but the values controlling the operation of robot's drives. The objective of the path optimization algorithm is to achieve the predefined work spots with the best possible accuracy in robot position and orientation.

Efficiency of the proposed path planning algorithm is demonstrated with a practical application in which the mobile two-wheeled platform travels to certain base points in order to perform the tasks that are specific to the manufacturing process. In the considered case, the main problem are various positions and orientations of processed elements. By using the LfD technique, together with the GWO algorithm the controller can generate control signals for the drives, which move the platform to required base points along the optimized trajectory learned. The proposed path planning algorithm can be used in such manufacturing processes as: welding, painting, or cleaning of the elements.

A novel contribution of the proposed approach is that the shape of the required mobile robot trajectory is generated and stored in robot's controller memory only once, during the LfD phase. Then, that learned trajectory can be multiply used by robot's controller, by recreating its shape for different positions and orientations of the processed element. There is no need to again teach the controller this trajectory for those new positions and orientations.

2. EXAMPLE MANUFACTURING PROCESS

In the considered manufacturing process a processed element is placed inside a specified area Figure 1. There might be two or more work spots on the element that have to be processed by a mobile platform equipped with a specialized equipment or a manipulator. For the purpose of the exemplification, manufacturing process of the painting system in considered, in which only two work spots are included. Accuracy of element placement is limited, as its position and orientation may vary due to inaccuracy of transporting machines. Therefore, it is impossible to define one universal trajectory of the mobile platform.

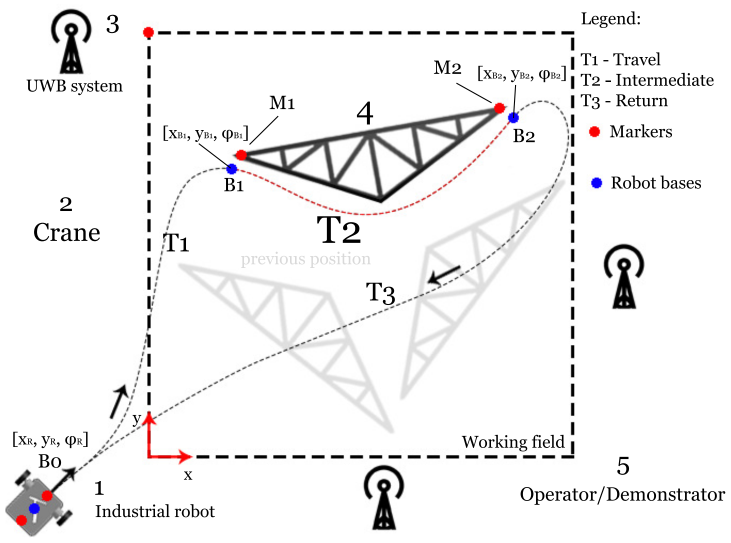


Figure 1. The hall of manufacturing operations

The processed element (4) is transported to the assembly hall in one part Figure 1. The object is placed at the working area by the crane (2). Due to large dimensions and weight of the object, the crane does not operate precisely, and thus the actual position and orientation of the object relative to the local coordinate system xy can vary. Therefore, the position x, y and orientation φ of the object is detected by the positioning system (3) utilizing two markers placed at points M1, M2. The locations of base points B1 and B2 are determined by shifting the position of markers M1, M2 by a constant offset value. The entire trajectory of the mobile platform (1) consists of three parts: travel trajectory T1, intermediate trajectory T2, and return trajectory T3. The task of the mobile platform is to adjust its motion trajectory to the changing position and orientation of the processed element. Aforementioned feature can be seen in Figure 2, where trajectories T1p and T3p differ from those of T1, T3 in Figure 1.

The position and orientation of the intermediate trajectory T2 depends on positions of base points B1 and B2, which are strictly related to the pose of the processed element. However, the shape of the trajectory remains the same regardless of the position and orientation of the painted element. It is required that trajectory T2 ends exactly at the base point B2. In order to fulfill such requirement, the two-stage LfD algorithm (section 3.3.) is utilized to generate the trajectory avoiding painted elements and ending exactly at point B2. In general, trajectory T2 may cover many base points, depending on the number of work spots on the processed element.

2.1. Travel/return trajectory

Travel trajectory T1 is generated between base points B0, B1 assuming that the travel path is free of obstacles Figure 1. Trajectory T1 is obtained as a result of the operation of position and orientation controller presented in section 3.1. The controller uses the UWB system and the digital compass to measure the position and orientation of the moving platform and generates control values for drive units.

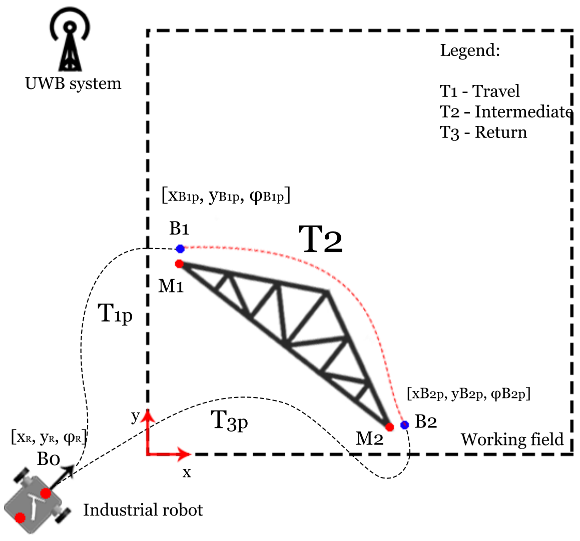


Figure 2. The hall of manufacturing operations: other position of the processed element

At the end of travel trajectory T1, the robot reaches base point B1, defined with $(x_{B1}, y_{B1}, \varphi_{B1})$. Due to friction and wheel slips this point is reached with some position and orientation errors. Then, the robot is ready to recreate the learned, intermediate trajectory T2. After completing the task at base points B1, B2 (i.e. after traveling the intermediate trajectory T2), the platform is returned to the starting point B0. This time the platform travels along trajectory T3, which is obtained similarly as trajectory T1.

2.2. Intermediate trajectory

Intermediate trajectory T2 is the most important part of the entire robot path. Its shape is determined based on the poses of base points located around the processed element. In general the intermediate trajectory may contain any number of base points. The trajectory is obtained as a result of the proposed LfD algorithm presented in section 3.3.

3. METHOD

The following section summarizes the methods used to control the robot. The main focus is put on the control algorithm for travel/return trajectory T1, T3, and the learning from demonstration algorithm for the intermediate trajectory T2. The proposed methods can be easily implemented in the mobile platform controller.

3.1. Control algorithm

The simplest controller, which may be used to control the robot over a desired trajectory, is the proportional controller Figure 3. Such controller is used to drive the mobile platform over travel/return trajectories T1, T3. The control loop consists of two separate parts. The first one is the orientation controller, which can be described by the following (1):

$$\omega(t) = K_\varphi e_\varphi(t) \tag{1}$$

where $\omega(t)$ is the desired angular velocity of the robot, K_φ is the controller gain, $e_\varphi(t) = (\varphi_{\text{ref}} - \varphi(t))$ is the actual orientation error at time t, φ_{ref} is the desired orientation, $\varphi(t)$ is the actual orientation at time t. The second part of the control loop is the position controller

$$v(t) = K_v e_v(t) \tag{2}$$

where $e_v(t) = \sqrt{(x_{\text{ref}} - x(t))^2 + (y_{\text{ref}} - y(t))^2}$ is position error, $v(t)$ is desired linear velocity, K_v is controller gain, x_{ref}, y_{ref} are desired positions, $x(t)$, $y(t)$ are actual positions of the robot at time t.

In real applications, both linear and angular velocities are limited to an acceptable level by the limiter. This level is determined based on the limitations resulting from mechanical/electrical parameters of the robot drive unit. Those limitations are included in the robot control system, as presented in Figure 3.

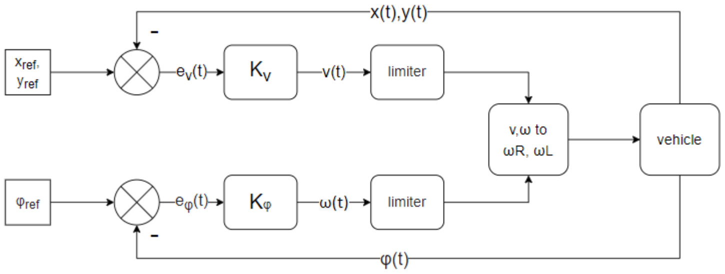


Figure 3. Block diagram of the position and orientation control system of the wheeled mobile robot

Different algorithms can be used to generate travel/return trajectories T1, T3. One method is to control the robot to a destination point by using an intermediate direction [25]. The intermediate direction is obtained basing on a right-angled triangle that is presented in Figure 4. One leg of that triangle connects robot's current position with the destination point B1. Orientation of this leg is φ_r and its length D. The other leg is a segment of constant length r. The resulting hypotenuse determines the intermediate direction φ_t. The intermediate direction navigational algorithm consists of two stages. In the first stage, the robot goes to the intermediate direction with the orientation $\varphi_t = \varphi_r + \beta(t)$, where $\beta(t)$ is the angle between the cathetus and the hypotenuse. When the value of angle β exceeds the value of angle α, the second stage takes place. During that stage, the controller tries to achieve the proper orientation $\varphi_{\text{ref}} = \varphi_r + \alpha(t)$, where $\alpha(t)$ is an angle achieved as a result of the subtraction of cathetus orientation and reference orientation. The use of intermediate direction smoothies the trajectory of robot's movement. The error $e_\varphi(t)$ in robot's orientation depends on the relationship between angles α and β, i.e.

$$e_\varphi(t) = \varphi_r(t) - \varphi(t) + \begin{cases} \alpha(t) & \text{for} \quad |\alpha(t)| < |\beta(t)| \\ \beta(t) & \text{otherwise} \end{cases} \tag{3}$$

where

$$\alpha(t) = \varphi_r(t) - \varphi_{\text{ref}} \tag{4}$$

$$\beta(t) = \begin{cases} \operatorname{atan}\left(\frac{r}{D}\right) & \text{for} \quad \alpha(t) > 0 \\ -\operatorname{atan}\left(\frac{r}{D}\right) & \text{otherwise} \end{cases} \tag{5}$$

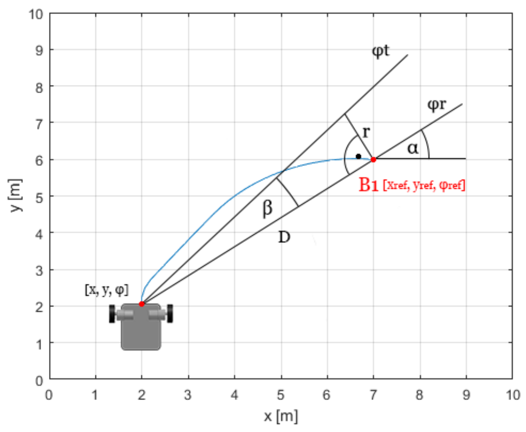


Figure 4. Generating travel trajectory by using an intermediate direction

3.2. Model of the differential drive

An algorithm for trajectory optimization in section 3.3.2 evaluates performance of trajectories based on the estimated final position of the robot. Hence, it is required to simulate robot's trajectory based on the angular velocity of its wheels. In the following section relation between control outputs $\omega_R(t)$, $\omega_L(t)$ and the position of the robot is presented. The main geometry parameters that are important for description of vehicle kinematics in the form of differential drive's model are axle length L and radius of the wheels r. The pose of the robot at the 2D plane is described by position coordinates x, y and orientation angle φ. Considering robot's motion in the local coordinate system, robot's linear speed v is referenced to point K in the center of the driving axle Figure 5. Assuming the relation between linear and angular speeds of the wheels is $v_R = r\omega_R$, $v_L = r\omega_L$, linear speed $v(t)$ of the robot may be expressed as:

$$v(t) = \frac{r(\omega_R(t) + \omega_L(t))}{2} \tag{6}$$

rotational speed of the platform $\omega(t)$ can be presented as:

$$\omega(t) = \frac{r(\omega_R(t) - \omega_L(t))}{L} \tag{7}$$

when the robot turns round, it moves around a circle, whose center point is referenced as the ICR (instant center of rotation). Radius $R(t)$ of that circle can be calculated as:

$$R(t) = \frac{L}{2}\frac{\omega_R(t) + \omega_L(t)}{\omega_R(t) - \omega_L(t)} \tag{8}$$

forward kinematics of the robot with a differential drive in the local coordinate system is expressed as:

$$\begin{bmatrix}\dot{x}(t)\\ \dot{y}(t)\\ \dot{\varphi}(t)\end{bmatrix} = \begin{bmatrix}\cos(\varphi(t)) & 0\\ \sin(\varphi(t)) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v(t)\\ \omega(t)\end{bmatrix} \tag{9}$$

using trapezoidal approximation, (9) can be presented as:

$$x(k+1) = x(k) + v(k)T\cos(\varphi(k) + \frac{\omega(k)T}{2}) \tag{10}$$

$$y(k+1) = x(k) + v(k)T\sin(\varphi(k) + \frac{\omega(k)T}{2}) \tag{11}$$

$$\varphi(k+1) = \varphi(k) + \omega(k)T \tag{12}$$

where T is the time step.

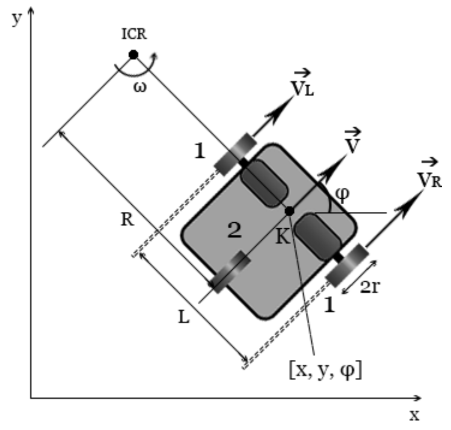


Figure 5. Model of a wheeled robot with a differential drive

In (10)-(12) are used to simulate robot's trajectory during the optimization process of control variables $\omega_{R\text{opt}}, \omega_{L\text{opt}}$. The presented simplified model of robot's forward and inverse kinematics assumes ideal ambient conditions. Its formulation does not include model unaccuracies due to friction, wheels slip, and loading forces.

3.3. Learning from demonstration

Learning from demonstration technique can be divided into two stages. During the first stage, the trajectory between the base points is demonstrated by the operator. Data obtained from the demonstration run is processed with the use of an optimization algorithm in the second stage of the LfD. The second stage is used to increase the precision of the final position and orientation of the robot in the base points. The optimization algorithm used during the second stage is the grey wolf optimization algorithm.

3.3.1. Demonstration stage

At the beginning of T2 trajectory, it is assumed that the platform is at the base point B1 $(x_{B1}, y_{B1}, \varphi_{B1})$, which has been reached using the previously mentioned travel trajectory T1. The platform learns how to generate control signals for the intermediate trajectory T2 during the demonstration process. During the process, the operator takes control of the platform, switching it to the manual mode. Then, using the wireless control unit, the operator drives the robot around the manufactured object from the base point B1 to the base point B2. The operator takes into account obstacles in robot's path as well as the shape and size of the processed element. While controlling the robot, the operator tries to move the platform accurately to the close vicinity of point B2. During this process, the values of control signals delivered to either left $\omega_{L\text{dem}}$ or right $\omega_{R\text{dem}}$ wheel motors are stored into the system memory. The operator makes several demonstration trips Figure 6 in order to collect data that will be further used during the optimization process.

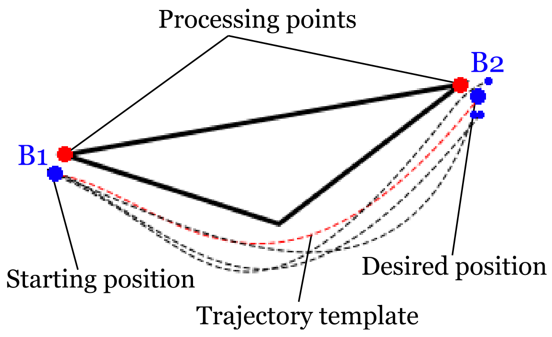


Figure 6. Demonstration trajectories performed by the operator

3.3.2. Optimization stage - GWO algorithm

In the next step the demonstrated trajectory is optimized through the GWO algorithm. Grey wolf optimizer (GWO) is meta-heuristic algorithm inspired by grey wolves. The GWO algorithm imitates the hunting mechanism of grey wolves in nature. Four types of grey wolves such as alpha, beta, delta, and omega are employed for simulating the leadership hierarchy. The algorithm includes four most important steps: hunting, searching for prey, encircling prey, and attacking prey [8].

In this step the best trajectory from the demonstration set is selected as the template for further optimization. The criterion for the best demonstration trajectory is the final accuracy of achieving point B2 by the mobile platform. For the selected trajectory template, the vector of control outputs is considered during the optimization. In each optimization step, the new value of the wheel control signals for left and right wheels is accordingly calculated as

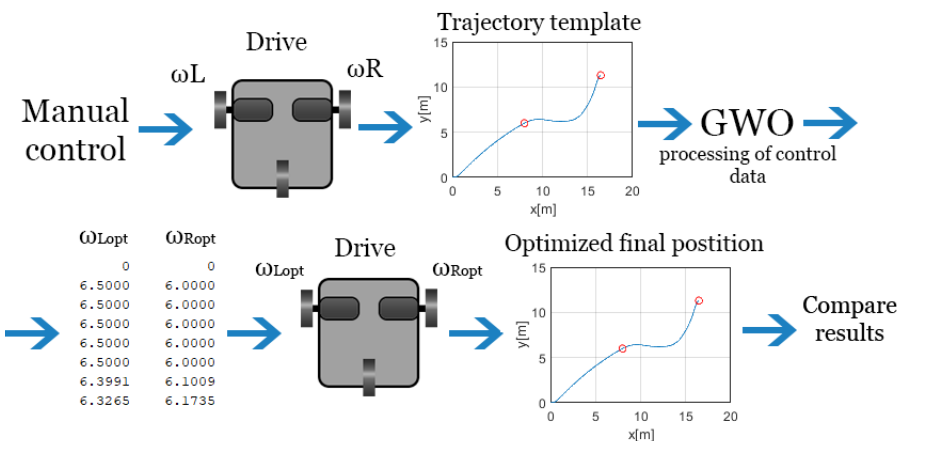
$$\begin{aligned}\omega_{L\text{opt}}(Nk) &= \omega_{L\text{dem}}(Nk) + \Delta\omega_L(k)\\ \omega_{R\text{opt}}(Nk) &= \omega_{R\text{dem}}(Nk) + \Delta\omega_R(k)\end{aligned} \tag{13}$$

where N is the sampling step of the demonstration run, $\omega_{L\text{opt}}$ is the optimized left wheel control value, $\omega_{L\text{dem}}$ is the left wheel control value from the demonstration, $\Delta\omega_L$ is the k-th modifier of the control value returned by the GWO algorithm.

In the next step the control values of the left and right wheels are converted to the linear and angular speed of the robot using (6)-(7). Then, using odometry (10)-(12), the linear and angular velocity is converted to the position of the robot. Values of robot's end position and orientation angle are used to calculate the goal function:

$$F(x, y, \phi) = \sqrt{(x_{T2} - x_{\text{des}})^2 + (y_{T2} - y_{\text{des}})^2 + (\varphi_{T2} - \varphi_{\text{des}})^2} \tag{14}$$

where x_{T2}, y_{T2} are the final planar position coordinates of the robot, φ_{T2} is the final orientation of the robot, x_{des}, y_{des}, φ_{des} are the desired values of the final position coordinates and orientation of the robot. Next, the optimized control values $\omega_{R\text{opt}}$, $\omega_{L\text{opt}}$ are experimentally verified. For that purpose, the robot uses the optimized control values to perform the T2 trajectory starting from the base point B1. When the robot finishes its drive, the final position and orientation are used to evaluate the trajectory generated during the optimization stage. Apart from the evaluation of the resulting position and orientation accuracy, operator also supervises robot's maneuvers to avoid any physical collision of the robot with the processed object. Supposing the operator is satisfied with the achieved results, the optimized trajectory is approved for further usage. The complete teaching/control routine is shown in Figure 7.



```
ωLopt     ωRopt
     0         0
6.5000    6.0000
6.5000    6.0000
6.5000    6.0000
6.5000    6.0000
6.5000    6.0000
6.3991    6.1009
6.3265    6.1735
```

Figure 7. Usage of the GWO algorithm for the optimization process of the robot trajectory

4. RESULTS AND DISCUSSION

4.1. Simulation results

Evaluation of the presented trajectory optimization algorithm started with simulations. The differential drive model presented in section 3.2 was utilized throughout simulations to calculate subsequent positions of the robot according to control inputs ω_R, ω_L. Simulation process can be divided into two separate stages.

During the first stage of the simulation, the assumed demonstration trajectories are utilized in the GWO algorithm in order to achieve the optimized intermediate trajectory T2. The presented research examines two different cases of workplace conditions, in which the autonomous robot is utilized to perform painting tasks. The first case deals with a single element that has to be avoided. The second case deals with two painted elements. For each of this conditions a demonstration trajectory that avoids the specified painted element is established. Thus, cosine-based (15) and sine-based (16) functions are assumed as if they are to avoid one or two obstacles accordingly.

$$y_1(x) = 1.5 \cos\left(2\pi \frac{x}{10}\right) \tag{15}$$

$$y_2(x) = 1.5 \sin\left(2\pi \frac{x}{10}\right) \tag{16}$$

where y_1, y_2, x are coordinates in the 2D plane. The cosine demonstration path, (15), is shown in Figure 8, while the sine path (16), is shown in Figure 9.

For the purpose of trajectory optimization, the GWO algorithm is used. For each optimization trial the GWO algorithm is run for 100 iterations, where each generation consists of 30 individuals. The optimization is performed directly on control signals ω_R, ω_L, and focuses on minimizing the goal function specified in (14). On the basis of a single demonstration trajectory, optimization is performed several times in order to achieve a set of different solution candidates. Several solutions acquired after the optimization stage for the demonstration trajectories y_1 and y_2 are shown in Figures 8 and 9. Optimized trajectories are slightly different, but all of them converge to a desired point B2. In Table 1 the values of the objective function for individual solutions are shown.

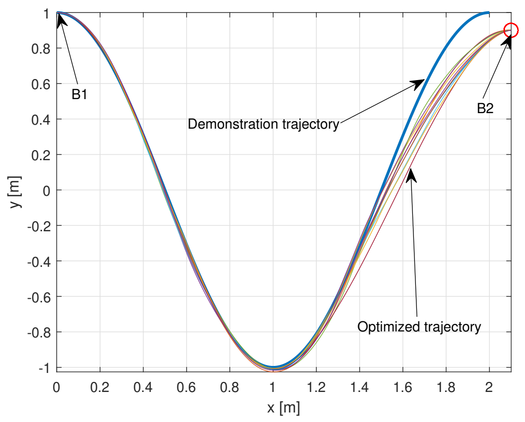


Figure 8. Five different trajectories generated using GWO for the cosine-based function

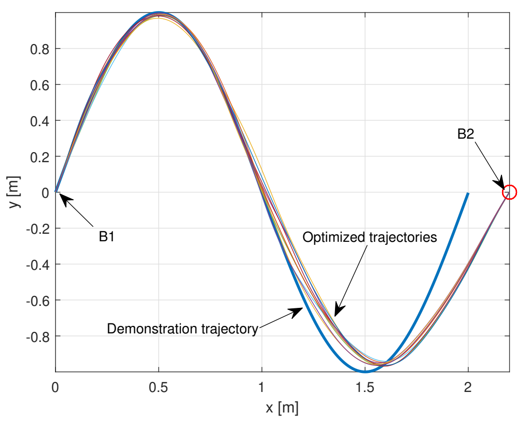


Figure 9. Five different trajectories generated using GWO for the sine-based function

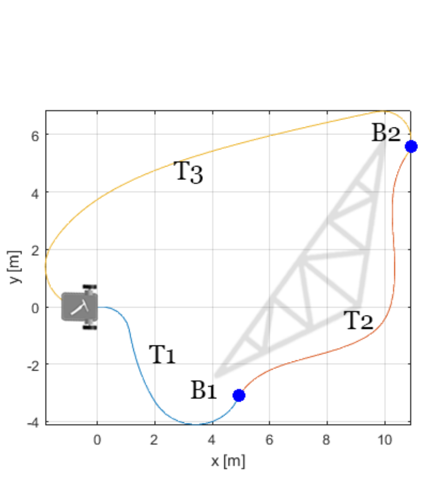
Table 1. Parameters of different trajectories obtained during subsequent optimization processes

No	Accuracy - goal func., (14)		Final position $[\mathbf{x}[\mathbf{m}], \mathbf{y}[\mathbf{m}], \varphi[\mathbf{rad}]]$	
	$1.5\cos\left(2\pi\frac{x}{10}\right)$	$1.5\sin\left(2\pi\frac{x}{10}\right)$	$1.5\cos\left(2\pi\frac{x}{10}\right)$	$1.5\sin\left(2\pi\frac{x}{10}\right)$
1	0.00038	0.00108	[10.001, 1.248, 0.000]	[9.503, −0.104, 0.781]
2	0.00099	0.00033	[10.000, 1.254, 0.003]	[9.501, −0.120, 0.782]
3	0.00057	0.00112	[10.001, 1.241, −0.001]	[9.504, −0.118, 0.776]
4	0.00149	0.00069	[10.001, 1.257, 0.005]	[9.496, −0.103, 0.790]
5	0.00140	0.00095	[9.996, 1.262, 0.002]	[9.500, −0.106, 0.785]

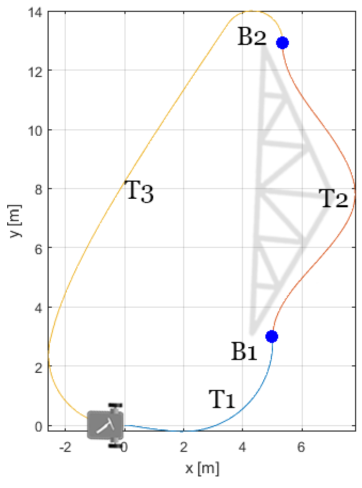
In the second stage of the simulation the entire painting problem including travel T1, intermediate T2, and return T3 trajectories is considered. Since intermediate trajectory is optimized during the first stage of simulation, it is necessary to determine shape of trajectories T1 and T3. During the travel trajectory T1, the robot begins to move from the starting position B0, defined with the coordinates vector $q = \begin{bmatrix}0 & 0 & 0\end{bmatrix}$ and stops at point B1. Using (1)-(5), the control algorithm described in section 3.1. calculates control signals to drive the robot from the starting position B0 to the first base point B1.

Then, the actual travel around the painted object is started along the optimized trajectory T2 determined in the first part of the simulation. When the robot finishes it is movement along the optimized trajectory T2, it is located at the base point B2. The robot stays at points B1 and B2 for the time required to finish painting tasks.

Finally, the platform returns to the starting position B0 using the return trajectory T3 generated with the same equations as for the travel trajectory T1. Simulation results of trajectory generation for the painting problem described through the demonstration trajectory y_1 are shown in Figure 10. Two different alignments of the painted object are analyzed, where Figure 10(a) presents trajectory generated for position B1 in $[4.0, -2.2]$ and B2 in $[10.0, 6.0]$, and Figure 10(b) presents trajectory generated for B1 in $[4.2, 3.0]$ and B2 in $[4.4, 12.8]$. Summary of the control signals generated through the control algorithm for the entire trajectory for both placements of the painted object is presented in Figures 11 and 12.



(a)



(b)

Figure 10. Platform trajectories generated in MATLAB for two different locations of the processed object: (a) B1 in $[4.0, -2.2]$, B2 in $[10.0, 6.0]$ and (b) B1 in $[4.2, 3.0]$, B2 in $[4.4, 12.8]$

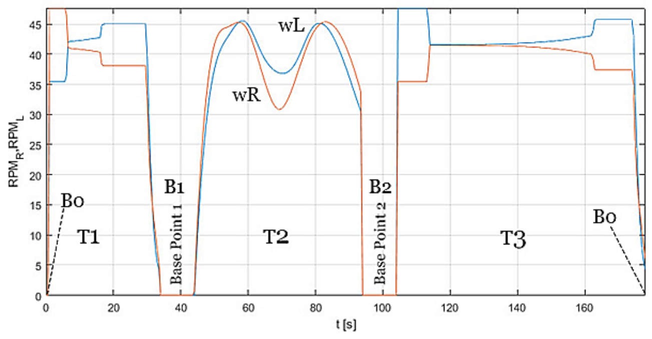


Figure 11. Robot control variables for trajectories from Figure 10(a)

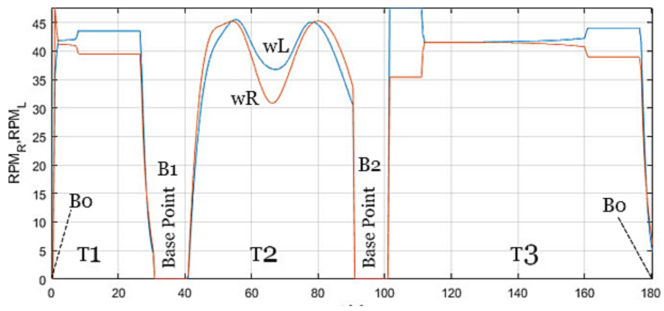


Figure 12. Robot control variables for trajectories from Figure 10(b)

4.2. Experimental results

The experiment focused only on the optimization of trajectory T2 between base points B1 and B2 along the processed object. In the experimental stage both demonstration functions described by (15) and (16) were evaluated. The robot moved inside the tested area of approximately 4×5 m. The experiment was conducted on a board that was specially designed to include supportive localization lines Figure 13.

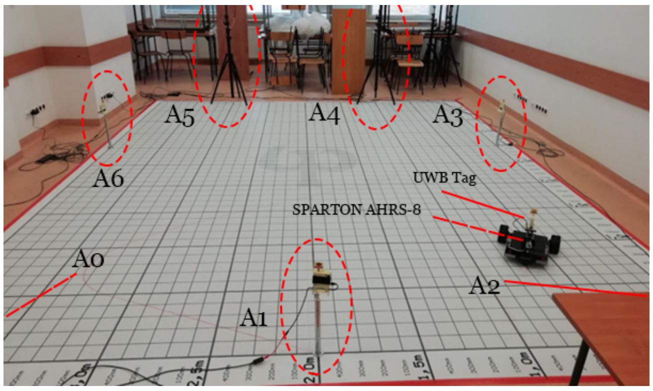


Figure 13. Measurement environment with a tracking system UWB

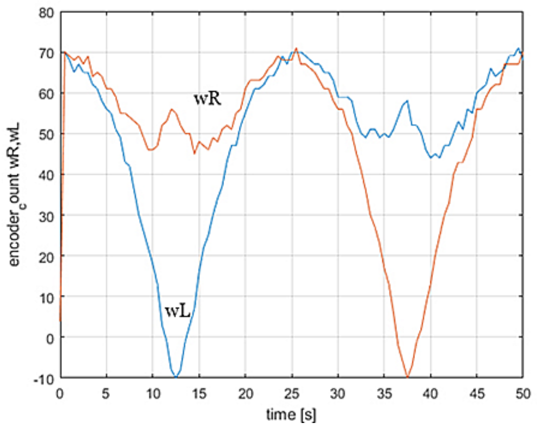


Figure 14. Cosine shape demonstration trajectory - right and left wheel control values from encoders

The experiment was conducted with the platform utilizing the differential, two-wheel drive. Platform's motors were driven by a PID unit coupled with optical encoders located on the motor shaft to control rotational speed of the wheels. The UWB positioning system was used to record robot's position. The UWB system utilized during the test consisted of seven anchors positioned at the edges of the tested area, and one tag placed on the robot. An average accuracy error, the UWB navigational system achieved in the aforementioned setup was 0.017 m in x axis, and 0.034 m in y axis. The components of the UWB system are described with labels A0-A6 and are marked with red circles in Figure 13. Orientation of the platform was measured with the SPARTON AHRS-8 digital compass.

The experiment was started with the demonstration run. Similarly to the simulation, sine and cosine trajectories were recreated. For each of them, subsequent values of rotational speeds of the right ω_R and left ω_L wheel were registered Figures 14 and 15). Then the obtained data was optimized using the GWO algorithm in order to achieve desired target pose B2. The control values optimized in that way were loaded into the robot and the drive was performed for each of the desired poses B2. The trajectories recorded for different target positions B2 are shown in Figure 16, where Figure 16a refers to B2 in $[1.5, 0, \pi/3]$, Figure 16b refers to B2 in $[2.5, 0.5, 0]$, Figure 16c – to B2 in $[1.75, 0, \pi/3]$, and Figure 16d – to B2 in $[2.5, 0, 0]$. During the optimization, the goal function given by (14) was used. Mean value of the objective function before optimization was equal 1.5. After applying the optimization algorithm, the value of the objective function dropped below 0.05. This means that the target point was reached with the required accuracy while maintaining the curvature of the path.

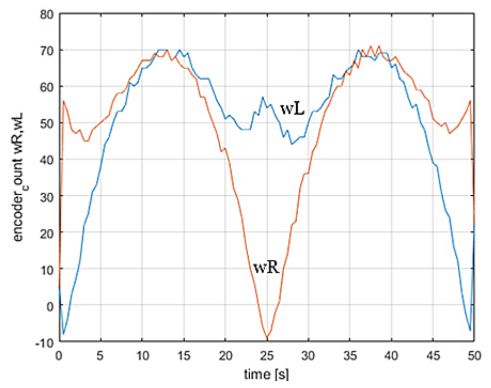


Figure 15. Sine shape demonstration trajectory – right and left wheel control values from encoders

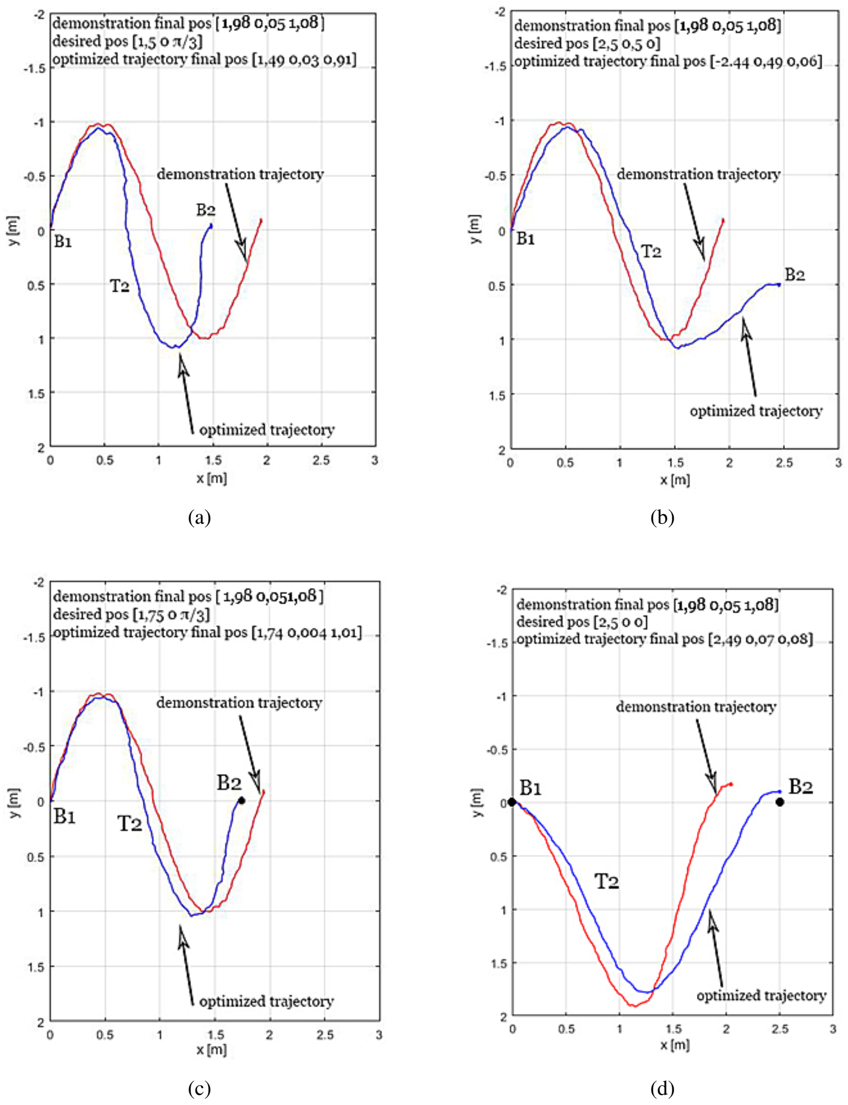


Figure 16. Experimental results of the trajectory optimizer obtained for different desired positions B2 of the mobile platform: (a) B2 in $[1.5, 0, \pi/3]$, (b) B2 in $[2.5, 0.5, 0]$, (c) B2 in $[1.75, 0, \pi/3]$ and (d) B2 in $[2.5, 0, 0]$

5. CONCLUSION

The article presents a new approach in the generation of the trajectory for an autonomous platform in modern manufacturing tasks. The main advantage of the discussed solution is the possibility of optimizing the trajectory demonstrated by the operator in order to improve accuracy of achieving a desired pose. Involvement of the operator for demonstration of the intermediate trajectory ensures that all of the obstacles are avoided in the preliminary stage. Since the presented solution includes a control algorithm to navigate through the travel and return trajectories, it is independent from the position and orientation of the elements that have to be avoided. It is achieved by combining the positioning system with the position controller in a closed loop. Hence the presented solution does not require qualified employees to operate and can be used by a person without experience and knowledge in the field of robotics. The teaching runs do not have to be precise; it is enough for the robot to avoid obstacles and reach the base position roughly. The algorithm is capable of generating reproducible control values. Simulations and experimental research showed that the obtained results are satisfactory and confirm the usefulness of the GWO based trajectory optimization solution in the practical applications. The time required to achieve solution through the GWO algorithm mainly depends on the length of the intermediate trajectory and settings adopted during the optimization. The GWO algorithm used in the presented research needed usually around 30 s to calculate the optimized trajectory on the Intel i3 based laptop. The system is not yet fully adapted to practical use, the obstacles that may appear on robot's optimized trajectory are not taken into account. In simulation studies, the obtained accuracy of the final position was 10 times higher than in the experiments. Such accuracy deterioration was caused by the use of a simple differential model in a real experiment and limitations of the measuring devices. In further work, implementation of a more complex robot model that is closer to reality may give more precise results. Moreover it is significant to improve the reliability of the generated trajectories, by taking into account positions and shapes of the obstacles, which have to be avoided during the GWO optimization.

ACKNOWLEDGEMENT

This work was supported by Bialystok University of Technology, grant number WZ/WE-IA/4/2020.

REFERENCES

[1] N. Ku *et al.*, "Development of a mobile welding robot for double-hull structures in shipbuilding," Journal of Marine Science and Technology, vol. 15, no. 4, pp. 374–385, Dec. 2010, doi: 10.1007/s00773-010-0099-5.

[2] G. Pan, E. Guan, F. Yang, A. Ren, and P. Gao, "Optimal motion planning for mobile welding robot," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10463, 2017, pp. 124–134.

[3] X. Liu, G. Wang, and K. Chen, "Option-based multi-agent reinforcement learning for painting with multiple large-sized robots," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2022, doi: 10.1109/TITS.2022.3145375.

[4] R. Zhang, J. Wu, and Y. Wang, "Stability analysis of a novel mobile spray-painting robot for touch-up painting in vehicle repair plant," *Journal of Mechanical Science and Technology*, vol. 36, no. 5, pp. 2571–2584, May 2022, doi: 10.1007/s12206-022-0438-6.

[5] P. Nattharith and T. Kosum, "Development of mobile robot system for monitoring and cleaning of solar panels," *GMSARN International Journal*, vol. 16, no. 3, pp. 302–306, 2022.

[6] Y. Zhang, Y.-L. Hsueh, W.-C. Lee, and Y.-H. Jhang, "Efficient cache-supported path planning on roads," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 4, pp. 951–964, Apr. 2016, doi: 10.1109/TKDE.2015.2507581.

[7] R. Kumar, L. Singh, and R. Tiwari, "Path planning for the autonomous robots using modified grey wolf optimization approach," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 5, pp. 9453–9470, 2021, doi: 10.3233/JIFS-201926.

[8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[9] F.-F. Li, Y. Du, and K.-J. Jia, "Path planning and smoothing of mobile robot based on improved artificial fish swarm algorithm," *Scientific Reports*, vol. 12, no. 1, p. 659, Dec. 2022, doi: 10.1038/s41598-021-04506-y.

[10] M. Alireza, D. Vincent, and W. Tony, "Experimental study of path planning problem using EMCOA for a holonomic mobile robot," *Journal of Systems Engineering and Electronics*, vol. 32, no. 6, pp. 1450–1462, Dec. 2021, doi: 10.23919/JSEE.2021.000123.

[11] J.-X. Lv *et al.*, "A new hybrid algorithm based on golden eagle optimizer and grey wolf optimizer for 3D path planning of multiple UAVs in power inspection," *Neural Computing and Applications*, vol. 34, no. 14, pp. 11911–11936, Jul. 2022, doi: 10.1007/s00521-022-07080-0.

[12] C. Kownacki and L. Ambroziak, "Asymmetrical artificial potential field as framework of nonlinear PID loop to control position tracking by nonholonomic UAVs," *Sensors*, vol. 22, no. 15, Jul. 2022, doi: 10.3390/s22155474.

[13] C. Kownacki, "Self-adaptive asymmetrical artificial potential field approach dedicated to the problem of position tracking by nonholonomic UAVs in windy enivroments," *Acta Mechanica et Automatica*, vol. 15, no. 1, pp. 37–46, Mar. 2021, doi: 10.2478/ama-2021-0006.

[14] C. Kownacki and L. Ambroziak, "Adaptation mechanism of asymmetrical potential field improving precision of position tracking in the case of nonholonomic UAVs," *Robotica*, vol. 37, no. 10, pp. 1823–1834, 2019, doi: 10.1017/S0263574719000286.

[15] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, May 2009, doi: 10.1016/j.robot.2008.10.024.

[16] M. Mitić, N. Vuković, M. Petrović, and Z. Miljković, "Chaotic metaheuristic algorithms for learning and reproduction of robot motion trajectories," *Neural Computing and Applications*, vol. 30, no. 4, pp. 1065–1083, 2018, doi: 10.1007/s00521-016-2717-6.

[17] A. Vakanski, F. Janabi-Sharifi, I. Mantegh, and A. Irish, "Trajectory learning based on conditional random fields for robot programming by demonstration," in *Proceedings of the IASTED International Conference on Robotics and Applications*, 2010, pp. 401–408, doi: 10.2316/P.2010.706-061.

[18] M. Ginesi, D. Meli, A. Roberti, N. Sansonetto, and P. Fiorini, "Dynamic movement primitives: volumetric obstacle avoidance using dynamic potential functions," *Journal of Intelligent and Robotic Systems*, vol. 101, no. 4, Apr. 2021, doi: 10.1007/s10846-021-01344-y.

[19] C. Yang, C. Chen, W. He, R. Cui, and Z. Li, "Robot learning system based on adaptive neural control and dynamic movement primitives," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 777–787, Mar. 2019, doi: 10.1109/TNNLS.2018.2852711.

[20] M. Ciezkowski, "A prototype of static IR beacon-receiver positioning system based on triangulation method," *Measurement*, vol. 128, pp. 149–159, Nov. 2018, doi: 10.1016/j.measurement.2018.06.039.

[21] Z. Song, G. Jiang, and C. Huang, "A survey on indoor positioning technologies," in *Communications in Computer and Information Science*, vol. 164, 2011, pp. 198–206.

[22] J. Cieśluk, Z. Gosiewski, L. Ambroziak, and S. Romaniuk, "Computationaly simple obstacle avoidance control law for small unmanned aerial vehicles," *Acta Mechanica et Automatica*, vol. 9, no. 1, pp. 50–56, Mar. 2015, doi: 10.1515/ama-2015-0010.

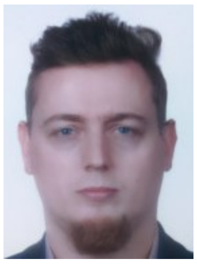
[23] C. Medina, J. C. Segura, and Á. de la Torre, "A synchronous TDMA ultrasonic TOF measurement system for low-power wireless sensor networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 3, pp. 599–611, Mar. 2013, doi: 10.1109/TIM.2012.2218056.

[24] A. Alarifi *et al.*, "Ultra wideband indoor positioning technologies: analysis and recent advances," *Sensors*, vol. 16, no. 5, May 2016, doi: 10.3390/s16050707.

[25] M. Bowling and M. Veloso, "Motion control in dynamic multi-robot environments," in *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 168–173, doi: 10.1109/CIRA.1999.810044.

BIOGRAPHIES OF AUTHORS

Adam Pawlowski graduated with M.Sc. in Electronics and Telecommunications from Bialystok University of Technology, Bialystok, Poland in 2020. Currently he is a Ph.D. student at the Faculty of Electrical Engineering, Bialystok University of Technology. His research interests include physics, electronics, and robotics. He can be contacted at a.pawlowski100@gmail.com.



Slawomir Romaniuk works as the lecturer at the Bialystok University of Technology, Faculty of Electrical Engineering, Department of Automatic Control and Robotics. Sławomir Romaniuk received his PhD degree in Automation and Robotics at AGH University of Science and Technology - Faculty of Mechanical Engineering and Robotics, Krakow, Poland. His research focus on Navigation Systems, Positioning Accuracy Improvement, Calibration algorithms, Optimization methods, Unmanned Aerial Vehicles, Robotics, Machine Learning and Artificial Intelligence. He can be contacted at s.romaniuk@pb.edu.pl.

Zbigniew Kulesza graduated with M.Sc. in Automatic Control and Robotics from Bialystok University of Technology, Bialystok, Poland in 1995. In 2003, he obtained his Ph.D. in Machine Building and Maintenance from Warsaw University of Technology, Warsaw, Poland. In 2013 he was awarded with D.Sc. in Automatic Control and Robotics from AGH University of Science and Technology, Krakow, Poland. Currently he is an Associate Professor at Bialystok University of Technology, Faculty of Electrical Engineering. He has research interests in dynamics and control of manipulators and mobile robots, analysis and design of control systems, dynamics of rotating machines, modeling fluid power systems. He can be contacted at z.kulesza@pb.edu.pl.

Milica Petrovic with M.Sc. in Mechanical Engineering from University of Belgrade, Belgrade, Serbia in 2010. She received her Ph.D. in Mechanical Engineering from University of Belgrade in 2016. Currently she is an Assistant Professor at the Faculty of Mechanical Engineering, University of Belgrade. Her research interests include intelligent manufacturing systems and processes, process planning and scheduling, optimization algorithms, computational intelligence, machine learning and artificial intelligence, neural networks, robotics, multi-agent systems. She can be contacted at mmpetrovic@mas.bg.ac.rs.