Energy Comparison of Controllers Used for a Differential Drive Wheeled Mobile Robot

ALEXANDR STEFEK\textsuperscript{1}, THUAN VAN PHAM\textsuperscript{1}, VACLAV KRIVANEK\textsuperscript{2}, (Member, IEEE), AND KHAC LAM PHAM\textsuperscript{3}

\textsuperscript{1}Department of Air Defence, University of Defence, 66210 Brno, Czech Republic
\textsuperscript{2}Department of Military Robotics, University of Defence, 66210 Brno, Czech Republic
\textsuperscript{3}Department of Aircraft Technology, University of Defence, 66210 Brno, Czech Republic

Corresponding author: Thuan Van Pham (vanthuan.pham@unob.cz)

This work supported by the Czech Republic Ministry of Defence (University of Defence in Brno development program “Research of sensor and control systems to achieve battlefield information superiority”).

ABSTRACT In order to select the best controller for a Differential Drive Wheeled Mobile Robot (DDWMR), an energy consumption comparison relating to tracking accuracy is used as a very strict criterion. Therefore, this paper reviews some well-known controllers designed for the DDWMR. Furthermore, there are presented several experiments with the extensible open-source code programmed in Python. Such an extensible open-source code presentation could serve as a tool for simulating, comparing, and evaluating a set of different control algorithms. The kinematic and dynamic models of the DDWMR and control algorithms are implemented in this open-source code to determine a travel time, a distance between the robot’s position and a given path, a linear velocity, an angular velocity, a travel path length, and a total kinetic energy loss of the DDWMR. These simulation results are used to compare and evaluate the given control algorithms. Moreover, the simulation results also enable to answer the question of whether a significant increase in energy consumption is worth shortening the travel path by just a bit. Finally, this paper includes a direct link to the stored experiments which are runnable and could serve as a proof. Besides, users can also easily supplement with other controllers and different paths to evaluate robot tracking control algorithms.

INDEX TERMS Differential drive robots, energy model, fitness function, kinetic energy, robot’s energy, wheeled mobile robot control.

I. INTRODUCTION

Robotic Science has been developing rapidly due to its various useful applications in many aspects of common life, industry, medicine, the military, and especially its ability to operate in hazardous and toxic environments. Recently, the Differential Drive Wheeled Mobile Robot (DDWMR) has been increasingly noticed and widely applied within the scope of Robotic Science. It has a lot of advantages, such as flexible motion capabilities, a simple structure, lower production costs. Furthermore, it can operate independently for a long time without a direct human control.

The DDWMR has some different forms, such as a 2-wheel, a 3-wheel, or a 4-wheel type. We can find the 2-wheel type in [13], the 3-wheel type in [5], [7], [10], [11], [18]–[20], [23], and the 4-wheel type in [15], [17]. The 3-wheel type is the most popular form which comprises of two fixed powered wheels mounted on both left and right side of the robot platform and one passive castor wheel used for balance and stability.

The motion control algorithm of the DDWMR: In principle, the movement of the DDWMR is based on two separately driven wheels placed on either side of the robot body. Therefore, it changes its direction by changing the relative rotation speed of the two wheels. For the DDWMR, the problems of path planning and path tracking are the most important. However, the path tracking is more important because its accuracy directly affects the robot operability. Many authors have researched and published various control algorithms of path tracking for the DDWMR, such as the adaptive output feedback control [2], the input-output feedback linearization method [4], the two-step feedback linearization control [3], the backstepping-based control [5], the PID control [6], the Lyapunov function-based control [8], [9], the adaptive and sliding mode control [7], [10], [11], [18], the neural-network-based control [19], and the robust adaptive-based control [21]. All of the above studies refer only to continuous or smooth curves, whereas the energy consumption of the robot is not mentioned. In reality, robots often have to work in...
complex conditions (environments with many obstacles, slipping wheels, acceleration of robots varies when they move) over a long time. So in many cases, it is too difficult for the robot to track the given path smoothly with saved energy consumption.

A requirement for the DDWMR is that it has to move smoothly, with no or minimum vibration, and detects a minor tracking error while tracking the sharp turn path or the discontinuous path. In order to achieve that requirement, the robot needs to have a constant rotation of the two wheels or it has to move on a path in a circular arc format. However, it should be mentioned that the circular arc function does not fit all possible paths, which the robot might be required to take [12]. These problems have been studied and solved by several different control algorithms that were presented in some papers, such as the controllers proposed by Kanayama and Robins Mathew [1], a feedback-based control for circular path [12], the Lyapunov-based method [25], a clever trigonometry-based control [24], a fuzzy logic control [24], and a Dubins path-based control [16].

The comparison and evaluation of the given control algorithms can be based on many different criteria, such as the distance between the robot’s position and the given path (the cross-track error), the travel distance (the length of the path), the total energy loss, the travel time, etc. For the mobile robot motion control, accuracy and energy consumption are always strict criteria. So it is necessary to have a simulator to verify and compare these criteria when the different controllers are used. Therefore, this paper aims at simulating, and comparing the results of some well-known control algorithms for the 3-wheel DDWMR to track the sharp turn path or the discontinuous path in the same working conditions. Then the best controller will be designed to improve it so that the robot suits each working condition and specific working environment.

This paper is organized as follows: some different shapes, a kinematic model, a dynamic model, and an energy model of the DDWMR are introduced in Section II; a review of some well-known control algorithms used for the DDWMR is provided in Section III; simulation, result collection, comparison, and the used control algorithms evaluation in Section IV. A summary and a conclusion complete the paper in Section V.

II. DIFFERENTIAL DRIVE WHEELED MOBILE ROBOTS
Wheeled mobile robots are employed for inventory management, factory automation, military surveillance, etc. These robots are classified into different categories, such as car-like robots, omnidirectional robots, and differential drive robots; of which differential drive robots are a prominent class [1].

A. POPULAR SHAPES OF THE DIFFERENTIAL DRIVE WHEELED MOBILE ROBOT
Currently, the structure of the DDWMR has many different types, such as a 2-wheel, a 3-wheel, or a 4-wheel differential drive mobile robot, as shown in Fig. 1. Each one has different advantages but the 3-wheel type has more advantages than the others.

In this study, we mention only the 3-wheel DDWMR type because it is widely used in many applications, such as industrial automated guided vehicles and service robots. Furthermore, the robots of this type are controlled more easily than others and provide high maneuverability and rotation around the center of the robot [11]. Besides, the 3-wheel DDWMR has a simple structure and is suitable for many practical applications as well as being popular and prevailing in the market.

B. MODELS OF A 3-WHEEL DIFFERENTIAL DRIVE MOBILE ROBOT TYPE
1) KINEMATIC MODEL
The Differential Drive Wheeled Mobile Robots (DDWMRs) usually have two independently driven wheels and one or more unpowered wheels at the rear as a balance. An important issue of the differential driving of mobile robots which needs considering is that their motion controller design is mostly based on kinematic models. The main reason is that dynamic models are more complex than kinematic models and mobile robots usually use only the low speed of the motor to control the loop [1].

The geometry and kinematic parameters of this robot are shown in Fig. 2.

In Fig. 2, this robot is the DDWMR which has important parameters as below:

- $v$ is the linear velocity of the DDWMR (ms$^{-1}$),
- $\theta$ is the orientation of the DDWMR (rad),
- $\omega_r$ is the angular velocity of the right wheel (rads$^{-1}$),
- $\omega_l$ is the angular velocity of the left wheel (rads$^{-1}$),
- $v_r$ is the linear velocity of the right wheel (ms$^{-1}$),
- $v_l$ is the linear velocity of the left wheel (ms$^{-1}$),
- $r$ is the radius of the right and the left wheels (m),
- $b$ is the distance between the right and the left wheels (m),

![Figure 1](image-url)  
(a) a 2-wheel differential drive mobile robot, (b) a 3-wheel differential drive mobile robot, (c) a 4-wheel differential drive mobile robot.
We assume that [3], [13]:

\[ x_G = x_Q - a \cos \theta \]  (1)
\[ y_G = y_Q - a \sin \theta \]  (2)

We assume that [3], [13]:
- The wheels are rolling without slipping,
- The center of gravity G coincides with the point Q,
- The guidance axis is perpendicular to the robot plane.

Based on Fig. 2 we get:
\[ v_r = v + \frac{b}{2} \omega \]  (3)
\[ v_l = v - \frac{b}{2} \omega \]  (4)

Adding and subtracting (3) and (4) we get:
\[ v = \frac{1}{2} (v_r + v_l) \]  (5)
\[ \dot{\theta} = \frac{1}{b} (v_r - v_l) \]  (6)

Due to the non-slipping assumptions we have \( v_r = r \omega_r \) and \( v_l = r \omega_l \).

From Fig. 2 we get:
\[ \dot{x} = \dot{x}_Q = v \cos \theta \]  (7)
\[ \dot{y} = \dot{y}_Q = v \sin \theta \]  (8)
\[ \dot{\theta} = \omega \]  (9)

Equations (7), (8) and (9) are a kinematic model of the 3-wheel DDWMR.

2) MOTOR DYNAMICS

A dynamic model of motor behavior has to be modelled to consider its characteristic on the robot chassis [12].

The DC motors are usually used for the DDWMR. The motor dynamics is modelled as follows [30]:

The electrical circuit and the free body diagram of the motor rotor of a DC motor are depicted in Fig. 3.

\[ F(s) = \frac{\omega(s)}{V(s)} = \frac{K_t}{L_a + L_s(s) + R_a I_a} \]  (16)

Or:
\[ F(s) = \frac{b_1}{s^2 + a_0 s + a_1} \]

where \( b_1 = \frac{K}{L_a} \), \( a_0 = \frac{R_I + L_s}{L_a} \), \( a_1 = \frac{K_B + K_a}{L_a} \).
C. FITNESS FUNCTION
There are many kinds of fitness functions for a mobile robot, but this paper only presents behavioral fitness functions for the 3-wheel Differential Drive Wheeled Mobile Robots. In this case, we consider the behavior of the robots in relation to their energy consumption and their travel distance because the amount of energy consumption and the travel distance affects the uptime and ability to perform their tasks.

1) ENERGY MODEL OF DIFFERENTIAL DRIVE WHEELED MOBILE ROBOT
Overall Energy Model of the Differential Drive Wheeled Mobile Robot: After analyzing all lost components, the complete energy model equation is presented as [22]:

$$E_{\text{battery}} = E_{dc} + E_k + E_{\text{friction}} + E_{\text{elect}}$$  \hspace{1cm} (17)

where $E_{dc}$ is the energy loss for DC motor, $E_k$ is the kinetic energy losses, $E_{\text{friction}}$ is the energy losses due to friction, $E_{\text{elect}}$ is the energy losses in the electronics, $E_{\text{battery}}$ is the energy of battery used for the robot.

In this study, we only consider the kinetic energy losses of the robot and ignore the other components.

The kinetic energy of the robot at any time can be expressed as [29]:

$$E_k = \frac{1}{2} m (v(t))^2 + \frac{1}{2} I (\omega(t))^2$$  \hspace{1cm} (18)

From (18), the total kinetic energy loss from the initial time to the final time is

$$\sum E_k = \sum_{t_0}^{t_n} \left( \frac{1}{2} m (v(t))^2 + \frac{1}{2} I (\omega(t))^2 \right)$$

$$= \sum_{i=0}^{z} \left( \frac{1}{2} m (v(t_0 + i\Delta T))^2 + \frac{1}{2} I (\omega(t_0 + i\Delta T))^2 \right)$$  \hspace{1cm} (19)

where $t, t_0, t_n$ represent time, initial time, and final time respectively, $m$ and $I$ denote the mass and the moment of inertia of the robot, $i = \{0, 1, 2, 3, ..., z = (t_n - t_0)/\Delta T\}, \Delta T$ is step time.

2) TRAVEL DISTANCE
The robot travel distance is an important quantity used for evaluating the quality of moving along the track of the given path. The less the difference between the travel distance and the length of the given path is, the less the average of cross-track error is. In this study, we expect the average cross-track error to be as small as possible. So, the travel distance is as similar to the length of the given path as possible. The function of the travel distance is

$$S = \int_{t_0}^{t_n} |v(t)| dt$$  \hspace{1cm} (20)

where $S$ is the travel distance by the robot, $t_0$ and $t_n$ represent the initial time and final time, respectively.

In this paper, (19) and (20) are used as the fitness functions to calculate energy consumption and travel distance that are used to compare and evaluate the controllers based on the energy consumption in relation to the travel distance of the robot.

III. WELL-KNOWN MOTION CONTROL ALGORITHMS
Motion is an important activity of the Differential Drive Wheeled Mobile Robot. Controlling the DDWMR movement depends on the structure of the robot, on the task that it will perform, and on the environment in which it will move. Therefore, choosing suitable algorithms for robot motion control depends on each specific case.

Currently, there are a lot of different motion control methods for the DDWMR, such as problems of position (posture) tracking, trajectory (path) tracking, point to point tracking and leader following. Among them, the path tracking is of more concern. In the trajectory tracking problems, a reference point (the midpoint between the left and right wheel) on the robot must follow a desired trajectory in the Cartesian space starting from a given initial configuration. Some prevalent algorithms of the DDWMR path tracking are the adaptive output feedback control algorithm, the input-output feedback linearization method, the backstepping control method, the PID control, the Lyapunov-based control, the sliding mode-based control, the robust control method, the adaptive fuzzy control, the neural-network control, and the vision-based methods, etc.

Point to point motion control or waypoint tracking of a robot: the robot must move to a desired goal configuration from a given initial configuration. As mentioned in the introduction of this paper, the DDWMR needs to move smoothly and minimize error tracking even when it tracks the sharp turn path or the discontinuous path. Some well-known point to point motion controllers have been proposed, such as the feedback-based controller for circular path, the Dubins path-based controller, the Lyapunov-based controller, the clever trigonometry-based controller, the advance controller by Robins Mathew, and by the Kanayama’s controller, etc.

A. FEEDBACK-BASED CONTROLLER FOR CIRCULAR PATH
According to [12] and kinematic model (7), (8), (9) we get:

$$\dot{x} = \dot{x}_0 = v \cos \theta$$
$$\dot{y} = \dot{y}_0 = v \sin \theta$$
$$\dot{\theta} = \omega = \frac{v_r - v_l}{b} = \frac{2\Delta}{b}$$

where $v = \frac{v_r + v_l}{2}; \Delta = \frac{v_r - v_l}{2}, b$ is the distance between the right and the left wheel.

The DDWMR moves from point $A$ to point $B$. Based on feedback control, the [12] shows that:

$$\dot{\theta} = \omega = \frac{2\Delta}{b} = 2v \frac{(AB) \sin \theta - (AB) \cos \theta}{|AB|^2}$$  \hspace{1cm} (21)
In the (21)

\[ \overrightarrow{AB} = ((AB)_x, (AB)_y), |AB| = \sqrt{(AB)_x^2 + (AB)_y^2} \]

\((AB)_x\) is the projection of \(AB\) along the x-axis. \((AB)_y\) is the projection of \(AB\) along the y-axis.

The kinematic model with the angular velocity of the robot in (21) is a form of feedback-based controller for the DDWMR moving on a circular path segment.

The paper [12] introduces one advanced control law by adding a coefficient \(K_c\) to (21), so we get:

\[ \dot{\theta} = \omega = \frac{2\Delta}{b} K_c \frac{(AB)_x \sin \theta - (AB)_y \cos \theta}{|AB|^2} \]  
(22)

Coefficient \(K_c\) is greater than zero. (22) allows the running of multiple experiments and fine-tuning a path of a robot on a single line segment [12].

B. THE LYAPUNOV-BASED CONTROLLER

The DDWMR moves from point \(Q\) with linear velocity \(v\) and orientation \(\theta\) to point \(G\), depicted in Fig. 4.

\[ \begin{align*}
\vec{x} &= x + l \cos \theta \\
\vec{y} &= y + l \sin \theta
\end{align*} \]  
(25)

\[ \begin{align*}
\dot{\vec{x}} &= \vec{v} \cos(\theta_G - \theta) \\
\dot{\vec{y}} &= \vec{v} \sin(\theta_G - \theta)
\end{align*} \]  
(26)

where \(\vec{v}\) is the desired forward velocity, an expected input for steering the robot towards a goal, \(l\) is the distance from \(X\) to \(\vec{X}\). It is considered as a coefficient of the controller.

The \(l\) needs to tune. In general, a shorter value of \(l\) makes the robot turn quicker, but a longer \(l\) allows the robot to run more smoothly [23].

D. ADVANCED CONTROLLER BY KANAYAMA AND ROBINS MATHEW

In Fig. 6, the DDWMR follows a path connecting the waypoints from a source \(q_{w0} = (x_{w0}, y_{w0})\) to a destination \(q_{wn} = (x_{wn}, y_{wn})\) through a set of intermediate waypoints \(q_{wk} = (x_{wk}, y_{wk}), k = 1, 2, \ldots, n - 1\) [1]. In the paper [1], Robins Mathew has mentioned the controller proposed by Kanayama, which is one of the most popular controllers in case of trajectories with continuous curvature. This control equation is given in (29) and (30). Further, Robins Mathew proposes a new control method to make the DDWMR tightly track the line connecting the waypoints with a minor cross-track error.

The control law of Kanayama is given below:

\[ \begin{align*}
v_{ci} &= v_{ri} \cos \theta_{ie} + k_1 x_{ie} \\
\omega_{ci} &= \omega_{ri} + k_2 v_{ri} y_{ie} + k_3 v_{ri} \sin \theta_{ie}
\end{align*} \]  
(29)

The control law of Robins Mathew is given below:

\[ \begin{align*}
v_{ci} &= v_{ri} \cos \theta_{ie} \\
\omega_{ci} &= \omega_{ri} + k_1 v_{ri} T_{ie} + k_2 v_{ri} \sin \theta_{ie}
\end{align*} \]  
(31)
In (29) – (32), $v_{ci}, \omega_{ci}, v_{ri}, \omega_{ri}$, are the linear, the angular velocity, the linear reference, the angular reference velocity respectively, $(k_1, k_2, k_3) > 0$ are the control gains, $(x_i, y_i, \theta_i)$ is the current robot position, and $(x_{wk}, y_{wk}, \theta_{wk})$ is waypoint posture, $(x_{ie}, y_{ie}, \theta_{ie})$ is the robot posture error, where $x_{ie} = (x_{wk} - x_i), y_{ie} = (y_{wk} - y_i), \theta_{ie} = (\theta_{wk} - \theta_i), T_{ie} = (x_i - x_{wk})\sin\phi - (y_i - y_{wk})\cos\phi$ is the cross-track error, $\phi$ is defined as the angle made by the line connecting current waypoint with the previous waypoint.

E. DUBINS PATH – BASED CONTROLLER

Dubins curve consists of several circular segments and straight segments. The shortest Dubins curve consists of three circular segments and straight segments. The primary forms are: (RSL, LSL, RSR, LSR, RLR, LRL). Where L, R represent the circular turning to the left and right respectively, S represent the tangent connecting two turning circulars [16]. The primary forms (RSL, LSL, RSR, LSR, RLR, LRL) of the Dubins path form are presented in Fig. 7.

IV. RESULTS AND DISCUSSION

A. SIMULATING AND RESULTS COLLECTING

The input parameters of the robot used for simulation in this study are shown in Table 1.

The coefficients of the controllers are given in Table 2. Two different given paths are used to simulate the behavior of the robot. The first one is a square path with parameters given in Table 3 and the second one is a straightforward path that has a different angle at any waypoints of the path. The parameters of the straightforward path are given in Table 4.

The movement of a mobile robot will be directly affected by the speed that its actuator can provide. If the provided velocity is too large, it will cause some difficulty in tracking, a high tracking error, and a high energy loss. Conversely, if the provided velocity is too small, the robot will move too slowly so it may affect its performance. Therefore, in this study, the velocity will be limited to evaluate and compare the quality of the used controllers. Then the value of the provided velocity will be changed to investigate and evaluate the effect...
This simulation creates some experiments of these, three main experiments presented below:

The first experiment proves that using the input parameters given in Tables 1-4, the linear reference velocity $v_{ri} = 1.0$ ms$^{-1}$, the angular reference velocity $\omega_{ri} = 0$ rads$^{-1}$, the linear and the angular velocity of the controllers are limited ($v$ is limited in the range $[0.2$ ms$^{-1}$ to $1.0$ ms$^{-1}$], $\omega$ is also limited in the range $[-0.75$ rads$^{-1}$ to $0.75$ rads$^{-1}$]). Running the simulation, we will receive some important results shown in Figures 8 to 17 and in Tables 5 and 6 below:

The second experiment testifies that using the Lyapunov-based controller with the input parameters given in Tables 1-4, the linear reference velocity $v_{ri} = 1.0$ ms$^{-1}$, 1.1 ms$^{-1}$, 1.15 ms$^{-1}$, the angular reference velocity $\omega_{ri} = 0$ rads$^{-1}$, the linear and angular velocity of the controllers are limited too. When $v_{ri} = 1.0$ ms$^{-1}$, $v$ is limited in the range $[0.2$ ms$^{-1}$ to $1.15$ ms$^{-1}$]. When $v_{ri} = 1.1$ ms$^{-1}$, $v$ is limited in the range $[0.2$ ms$^{-1}$ to $1.1$ ms$^{-1}$], and when $v_{ri} = 1.15$ ms$^{-1}$, $v$ is limited in the range $[0.2$ ms$^{-1}$ to $1.15$ ms$^{-1}$]. $\omega$ is limited in the range $[-0.75$ rads$^{-1}$ to $0.75$ rads$^{-1}$]). Running the simulation, we will receive some important results, shown in Figures 18 and 19 and Tables 7 and 8 below:

Some important results of the third experiment are shown in Figure 20 and Table 9 below:
The third experiment above proves that using input parameters given in Tables 1, 3, 4, and adding some more waypoints in each segment of the straightforward path. The linear reference velocity \( v_{ri} = 1.0 \, \text{ms}^{-1} \), the angular reference velocity \( \omega_{ri} = 0 \, \text{rads}^{-1} \), linear and angular velocity of the controllers are limited (\( v \) is limited in range \([0.2 \, \text{ms}^{-1} \text{ to } 1.0 \, \text{ms}^{-1}]\), \( \omega \) is limited in range \([-0.75 \, \text{rads}^{-1} \text{ to } 0.75 \, \text{rads}^{-1}]\)).

We can also see the results in some extra experiments as follows:

The extra experiment 1: A diamond-shape path tracking of the robot. The results of the experiment are shown in Figure 21 and Table 10.

The extra experiment 2: A sharp turn path tracking of the robot. The results of the experiment are shown in Figure 22 and Table 11.

In order to compare the energy loss of other controllers, we add some more different controllers and paths to the
open-source code programmed in Python. The program will show us all of the results.

### B. DISCUSSION

#### 1) THE USED CONTROL METHODS COMPARING AND EVALUATING

From these simulation results, it can be seen that when the robot tracks the square path or the straightforward path, its travel distance and energy loss are different when the different control algorithms are used in each segment of the given path. Therefore, the total travel distance and total energy loss are different when the robot is controlled by the different control algorithms to move in a full path, as shown in Tables 5-8 and in Fig.8-10, Fig.13-15 above.

In more detail, Table 5 indicates that when the robot is controlled to track the square path based on the “Lyapunov-based controller”, and the straightforward path based on the “geometry-based control” algorithm, its travel distance is the smallest of all.

When it is controlled by the rest of the control algorithms, the travel distance is variable and depends on each kind of the given path.
Besides, from Table 5, Table 6, and Fig. 8, Fig. 13 it is obvious that the trajectory of the robot is always the furthest from all the segments of the given path when control is based on the “feedback-based controller for the circular path $G = 4$”. In other words, the average of the error-track is the biggest when the robot is controlled based on the “feedback-based controller for the circular path $G = 4$”. The cross-track average of the robot is the smallest when control is based on the “Lyapunov-based controller” to track the square path, and the cross-track average of the robot is the smallest when it is controlled based on the “geometry-based controller” to track the straightforward path. Thus, it is very intuitive to recognize and compare the travel distances by the robot when it is controlled by the respective controllers shown in Table 5, Table 6, and Fig.8, Fig.13. It is also evident that the cross-track error in each segment is dependent on not only the control algorithms used but also the angle created by two contiguous segments. The smaller that angle is, the bigger the cross-track error is.

Therefore, if we evaluate the robot control quality based on the criterion of the travel distance, the control algorithm of the “Lyapunov-based controller” is the best when the robot tracks the square path, and the “geometry-based controller” is the best when the robot tracks the straightforward path because the travel distance is the smallest. The control
algorithm “feedback-based controller for the circular path $G = 4$” is the least qualified, because the travel distance is the largest.

The [23] mentioned mobile robots rely on a battery as their power source for most of the time. However, batteries have a very limited energy capacity. This finite amount of energy leads to a very short operational time of the robot, which may not be enough for missions or tasks that require more time and energy to be completed. Therefore, despite the intelligence and capabilities, their use in such tasks is not feasible. Although the time of operation can be prolonged by increasing the number of batteries or by diverting the robot back to the charging station, both of these techniques come at the expense of the increased cost and size which can cause control problems. Another way to increase the time of operation in robots is by reducing the energy consumption in the robot system itself and thus increasing their energy efficiency [23]. So, using less energy is considered as a very important criterion to evaluate the quality of control algorithms for the robot. It means that the less energy the robot uses, the better control algorithm is.

According to the energy loss criterion and based on Table 5, 6, Fig.10, Fig. 15 it can be seen that the “feedback-based controller for circular path $G = 4$” algorithm is the best because the total energy loss is the smallest of all. The “Kanayama-based controller” is the worst when the robot tracks the square path, and the “geometry-based controller” is the worst when the robot tracks a straightforward path because the total energy loss is the most significant of all.

In this research, the Lyapunov, Robins Mathew, geometry, and Kanayama-based controllers are called the high energy controllers. The feedback-based controller for a circular path and the Dubins path-based controller are called the low energy controllers. It is also evident that when the robot is controlled by the high energy controllers, the energy consumption is significantly more excessive (more than 270%) than the energy consumption of the robot controlled by the low energy controllers. Meanwhile, when the robot is controlled by the high energy controllers the length of the travel path is only slightly shorter than when it is controlled by low energy controllers (less than 5.5207%). So it can be said that when the robot is controlled to track a given path, the high energy consumption has to increase (greater than 270%) to shorten only a little (less than 5.5207%), which is unnecessary in most robot missions. When it is necessary to reduce the length of the travel path, we need to improve the control algorithms to shorten the length of the travel path with no increase of energy loss or negligible increase of energy loss.

2) USING ONLY ONE CONTROLLER WITH DIFFERENT REFERENCE LINEAR VELOCITY

When the robot is controlled by only the “Lyapunov-based controller”, it is set on different values of the reference linear velocity. Based on Table 7, Table 8, and Fig.18, Fig.19 it can be found that the faster linear velocity increases, the bigger the energy loss is in both types of the given paths. It means the robot will move and reach the destination faster, but its total energy loss becomes greater.

In this case, it is clear that when the linear velocity is set from 1.0 ms-1 to 1.1 ms-1 and then goes up to 1.15 ms-1, the travel time is reduced significantly (about 5s) and the energy loss greatly increases (more than 20%) while the increase in the travel distance is negligible (less than 0.4154%). Thus, to shorten the travel time, the robot must increase energy consumption significantly. This result has raised a question if the use of high velocity is so important in case that higher energy must be provided. So, based on the specific tasks of the robot, users can decide to increase energy consumption significantly, or not, so that the robot can quickly move to its destination.

3) MORE WAYPOINTS ADDED INTO THE SEGMENTS OF THE GIVEN PATHS

When some more waypoints are added into the segments of the given paths, the robot will track that path tightly. So the cross-track error of the robot will be reduced significantly.

Figure 20 and Table 9 show that the travel distance will be increased, or reduced, while the cross-track error will be reduced and most of the total energy loss will increase a lot. This result also has raised a question, if the decrease of the cross-track error is so important that higher energy must be provided. The answer to that question depends on the specific tasks of the robot. If the robot has to track the given path tightly, it must increase energy consumption significantly. In this way, the robot can reduce its cross-track error. But in some tasks where the robot does not have to track the given path tightly, it is unnecessary to increase the energy consumption to reduce its cross-track error.

C. OPEN-SOURCE CODE TO IMPLEMENT EXPERIMENTS

This paper proposes an open-source code as a tool from https://colab.research.google.com/drive/1UP2ru4v51peFy9jO65ovh8tFQtiUJCMd which was programmed with Python language in Jupyter Notebook to implement and compare the different control algorithms introduced in section 3. Users must click on this link to open the simulation program. However, to edit it, users have to copy it to their google drive by clicking the “Copy to Drive” or “File > Save a copy in Drive”. After that, they must get access to https://drive.google.com/drive/u/0/my-drive to open and edit that copy of the simulation program by using Google Colaboratory.

This program consists of an inevitable program initialization section, a robot model, simulation tools, controllers, a simulation section, an interactive simulation and a simulation results section.

1) THE INEVITABLE PROGRAM INITIALIZATION SECTION

This section declares some libraries (NumPy, Math, SciPy, Integrate, matplotlib, pyplot, etc.) used for programming in the next sections.
2) THE ROBOT MODEL SECTION
The Robot model section introduces some different shapes of the DDWMR, robot parameters, basic equations used for the DDWMR, the kinematic model, kinematic model implementation, the dynamic model, and dynamic model implementation. Users can change the robot parameters in this section for their own robot parameters.

3) THE SIMULATION TOOLS SECTION
Simulation tools are used for numerical computation RK-45 function from SciPy library. They consist of computation, given path, robot model and controller connection, closure function for simulation, simulation runner, path plotter, value plotter, all in one plotter, and data extractor.

4) THE CONTROLLERS SECTION
The controllers section allows users to add some controllers or define their own control algorithm at “Define your controller here”. That function will return the velocity and omega value of the robot.

5) THE SIMULATION SECTION
The simulation section allows users to definite square path, straightforward path, or put their own path. In this section, the experiment is described and executed.
In the “Experiment Description”, users can change the value of velocity limitation and also can change the value of the coefficients in the controllers.

6) THE INTERACTIVE SIMULATION SECTION
In the Interactive Simulation section, you can choose different experiment sets on different paths by selecting different “experimTag” and “pathTag”.

7) THE SIMULATION RESULTS SECTION
In this section, users can get the simulation text results, change the simulation image output size, simulate all graphs in one figure, simulate the path time, simulate the travel distance over time, and simulate the robot kinetic energy loss over time.
Thus, it is clear that this open-source code allows users to use different types of paths for the robot by defining them in the section “Simulation” and subsection “Paths Definition”. At the same time, it also allows users to use other control algorithms by adding or modifying the codes in the section “Controllers” and subsection “Define your controller here”.

V. SUMMARY AND CONCLUSION
In this paper, the set of various well-known controllers for the differential driven robot has been discussed. As a main attribute of those controllers, the model of energy consumption has been taken into account. Visually, the ability to track the given path is important and thus the differences of the energy consumptions have been evaluated in relation to differences in the total lengths of travel paths. According to the carried experiments, the controllers from a group of the low energy controllers have been offering significantly less energy consumption and still considerable precision in the tracking of the given path. The experiments have shown that the shortage of the travel path is less than 5 percent while energy consumption is almost more than 200 percent. The comparison of low energy controllers and high energy controllers has raised a question if the advantages of high energy controllers are so crucial their real usage. As the comparison always matters, standard and open tools should be used. Such tools allow us to build a rich database of experiments and transparent comparison of any given controller with the old and already proven one. To support this idea the software has been developed with the use of the Google Colab® and Jupyter platforms and opened for all users.
In our future research, we will focus on improving a motion control algorithm for the DDWMR that tracks a given path with higher accuracy, less energy loss but smooth movement without or less vibration and a shorter travel time. The result will be tested by using this open-source code program.

REFERENCES
[1] R. Mathew and S. S. Hiremath, “Development of waypoint tracking controller for differential drive mobile robot,” in Proc. 6th Int. Conf. Control, Decis. Inf. Technol. (CoDIT), Paris, France, Apr. 2019, pp. 1121–1126.
[2] X. Zhou, B. Ma, and L. Yan, “Adaptive output feedback tracking controller for wheeled mobile robots with unmeasurable orientation,” in Proc. 37th Chin. Control Conf. (CCCC), Wuhan, China, Jul. 2018, pp. 412–417.
[3] M. Yallala and S. J. Mija, “Path tracking of differential drive mobile robot using two step feedback linearization based on backstepping,” in Proc. Int. Conf. Innov. Control, Commun. Inf. Syst. (ICICCI), Greater Noida, India, Aug. 2017, pp. 1–6.
[4] N. V. Thinh, N. T. Linh, P. T. Cat, P. M. Tuan, M. N. Anh, and N. P. T. Anh, “Modeling and feedback linearization control of a nonholonomic wheeled mobile robot with longitudinal, lateral slip,” in Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE), Fort Worth, TX, USA, Aug. 2016, pp. 996–1001.
[5] D. Diaz and R. Kelly, “On modeling and position tracking control of the generalized differential driven wheeled mobile robot,” in Proc. IEEE Int. Conf. Automatika (ICA-ACCA), Curico, Chile, Oct. 2016, pp. 1–6.
[6] J. Meng, A. Liu, Y. Yang, Z. Wu, and Q. Xu, “Two-wheeled robot platform based on PID control,” in Proc. 5th Int. Conf. Inf. Sci. Control Eng. (ICISCE), Zhengzhou, China, Jul. 2018, pp. 1011–1014.
[7] S. Peng and W. Shi, “Adaptive fuzzy output feedback control of a nonholonomic wheeled mobile robot,” IEEE Access, vol. 6, pp. 43414–43424, 2018.
[8] Y. Jinhua, Y. Suzhen, and J. Xiao, “Trajectory tracking control of WMR based on sliding mode disturbance observer with unknown skidding and slipping,” in Proc. 2nd Int. Conf. Cybern., Robot. Control (CRC), Chengdu, China, Jul. 2017, pp. 18–22.
[9] F.-G. Rojas-Conteras, A.-I. Castillo-Lopez, L. Fridman, and V.-J. Gonzalez-Villela, “Trajectory tracking using continuous sliding mode algorithms for differential drive robots,” in Proc. IEEE 56th Annu. Conf. Decis. Control (CDC), Melbourne, VIC, Australia, Dec. 2017, pp. 6027–6032.
[10] B. B. Mevo, M. R. Saad, and R. Fareh, “Adaptive sliding mode control of wheeled mobile robot with nonlinear model and uncertainties,” in Proc. IEEE Can. Conf. Elect. Comput. Eng. (CCECE), Quebec City, QC, Canada, May 2018, pp. 1–5.
[11] P. Petrov and V. Georgieva, “Adaptive velocity control for a differential drive mobile robot,” in Proc. 20th Int. Symp. Electr. Appl. Technol. (SIELA), Bourgas, Bulgaria, Jun. 2018, pp. 1–4.
[12] A. Štefek, V. Klívánek, Y. T. Bergeon, and J. Motsch, “Differential drive robot: Spline-based design of circular path,” Dynamical Systems: Theoretical and Experimental Analysis (Springer Proceedings in Mathematics & Statistics), vol. 182, J. Awrejcewicz, Ed. Cham, Switzerland: Springer, 2016, pp. 331–342.
