applied sciences

MDPI

Article

Design of a Saving-Energy Fuzzy Logic Controller for a Differential Drive Robot Based on an Optimization

Van Thuan Pham [1,*], Alexandr Stefek [2], Vaclav Krivanek [3] and Thi Son Nguyen [4]

[1] Department of Missile and Gunship, Vietnam Naval Academy, Tran Phu 30, Nha Trang 650000, Vietnam
[2] Department of Informatics and Cyber Operations, University of Defence, Kounicova 65, 662 10 Brno, Czech Republic
[3] Department of Military Robotics, University of Defence, Kounicova 65, 662 10 Brno, Czech Republic
[4] Department of Natural Sciences and Foreign Language, Vietnam Naval Academy, Tran Phu 30, Nha Trang 650000, Vietnam
* Correspondence: vanthuanhvhq@gmail.com

Abstract: In the problem of designing a Fuzzy Logic Controller (FLC) for robots in general and a Differential Drive Robot (DDR) in particular, the determination of the parameters of Membership Functions (MFs) and Fuzzy Rules (FRs) is a difficult, complicated, and time-consuming problem because this is mainly based on heuristics and the knowledge of experts. Therefore, this paper provides a new method to efficiently design an FLC for the DDR by using the Genetic Algorithm (GA). Here, the GA is used to generate and optimize both the parameters of MFs and the FRs according to the minimum kinetic energy loss criterion. For this purpose, a program is created in Google Colab® by using the Python language with the help of the "Pymoo" library to not only automatically generate all the suboptimal parameters of MFs and the suboptimal FRs but also the simulate and evaluate different used FLCs. This program is published as an open-source code so that all readers can browse, view, run, and modify the code themselves to design their FLC. The simulation results have shown that the designed FLC is much better than other used FLCs in terms of the minimum kinetic energy loss while other control performances are still good.

Keywords: differential drive robot; fuzzy logic control; robot motion control; energy saving control; genetic algorithm; optimal control; pymoo; wheeled mobile robot control

check for updates

Citation: Pham, V.T.; Stefek, A.; Krivanek, V.; Nguyen, T.S. Design of a Saving-Energy Fuzzy Logic Controller for a Differential Drive Robot Based on an Optimization. *Appl. Sci.* **2023**, *13*, 997. https://doi.org/10.3390/app13020997

Academic Editor: Alessandro Gasparetto

Received: 19 December 2022
Revised: 5 January 2023
Accepted: 9 January 2023
Published: 11 January 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the field of mobile robots, the design of a motion controller is a crucial, fundamental, and important problem because the control law directly affects the efficiency of the application. So far, there have been many different motion control algorithms that have been successfully applied to mobile robots such as the PID control [1,2]; the Lyapunov-based control [3]; the adaptive and sliding mode control [4–7]; the fuzzy logic-based control [8–10]; the neural network-based control [11]; etc. In the mentioned algorithms, the fuzzy logic-based control is widely applied because it has a number of advantages such as requiring no complicated mathematical modeling and understating when designing a Fuzzy Logic Controller (FLC); it is easy to design and implement an FLC; the Fuzzy Logic (FL) control can be combined easily and deliver improved efficiency with conventional control systems; it has an ability to deal with nonlinearities and uncertainties; the fuzzy logic control algorithms are often robust; and, most of all, it is very efficient and easy to implement and optimize the fuzzy logic controller thanks to the adjustment of its membership function's parameters and fuzzy rules.

Most FLCs are designed mainly based on the heuristics and the knowledge of experts as in [12,13], so the design is quite difficult, complicated, and time-consuming, while the performances of motion control are still poor. Thus, in order to reduce the complexity, as well as to shorten the design time and to cause the design to be less dependent on the

heuristics and the knowledge of experts, there have been many studies using different optimization algorithms to design and improve the FLC for Wheeled Mobile Robots (WMRs) in general and the Differential Drive Robot (DDR) in particular [14]. Some well-known optimization algorithms have been used to design the FLC such as the Bat Algorithm (BA) [15,16]; the Bee Colony Optimization (BCO) [17]; the Particle Swarm Optimization (PSO) [9,17]; the Genetic Algorithm (GA) [8–10]; etc. Among those algorithms, the GA is more suitable for the optimization of the FLC because it offers a number of different benefits such as the GA considering a population of points, not a single point; the GA can work directly with strings of characters representing the parameter set; the GA uses probabilistic rules to guide their search not deterministic rules; the GA has a world-wide usage and user support; and it is easy to implement the GA in different program languages with a number of different support libraries.

Until now, most of the studies using GA to design FLC mainly focus on studying and using GA to design and optimize the parameters of MFs, while the Fuzzy Rules (FRs) are designed based on heuristics or experts, as in [10,18–20]. The other studies use the GA to design and optimize the FRs while the parameters of the MFs are fixed, as in [21–23]. There are very few studies that use GA to optimize both the parameters of the MFs and the FRs, as in [8,24]. However, these studies just aim to solve the problem of designing the FLC to control the robot to reach the destination and have not solved the problem of energy optimization, path optimization, travel time optimization, etc. Furthermore, almost no research has been published with any program or any open-source code for readers to reproduce, use, and verify.

From the above-mentioned problems, the authors develop and extend the previously published paper [10] with the following main objectives: (1) to provide a new method to efficiently generate both the suboptimal parameters of MFs and the suboptimal FRs for the fuzzy logic motion controller in order to drive the DDR from the starting point to the destination through several given waypoints according to the minimum kinetic energy loss criterion while the other performances of motion control are still good; (2) to provide a program created in Google Colab® with the Python language with the help of the "Pymoo" library in the form of the open-source code used to automatically generate all the suboptimal parameters of MFs, the suboptimal FRs, and simulate and evaluate different previously used FLCs so that readers can browse, view, run, and modify the code themselves to design their FLC.

The remainder of this paper is organized as follows: the problem statement is outlined in Section 2; the DDR mathematical models, the fuzzy logic, and the genetic algorithm are introduced in Section 3; in Section 4, the procedure of designing an FLC using GA is presented in detail; In Section 5, the simulation result collection and a discussion are presented to highlight the performance of the proposed method. The conclusions are stated in Section 6.

2. Problem Statement

2.1. Problem Description

Without a loss of generality, let us assume that the DDR has to move from the starting point to the given destination (the goal) through all the given waypoints (W_1, W_2, W_3, and W_4) to fulfill its task as shown in Figure 1. Thus, a controller is needed for the DDR to navigate the robot from starting point to the given destination effectively.

In this study, an FLC is designed based on the genetic algorithm to help the robot solve its task effectively. Specifically, the FLC is designed for the DDR to drive the robot from the starting point to the given destination through all the given waypoints with the minimum kinetic energy loss criterion while the other performances of the motion control are still good.

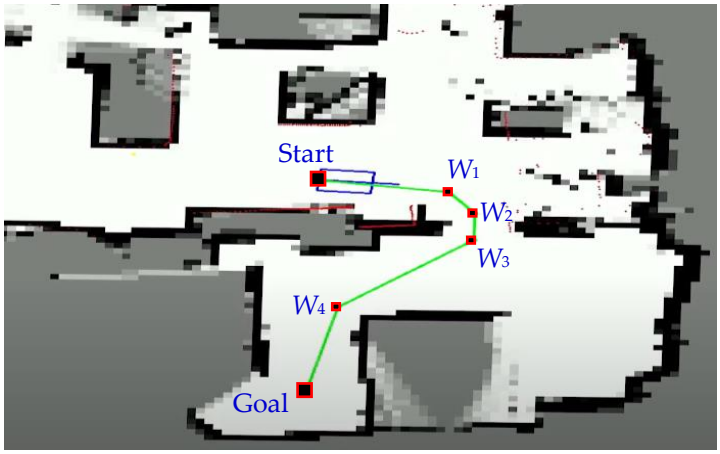


Figure 1. The moving task of the differential drive robot.

The relationship between the current robot position and the next waypoint is needed for the design process and it is indicated in Figure 2 [10].

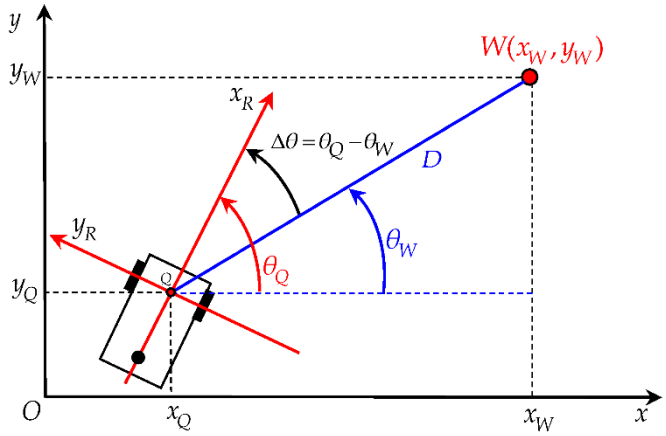


Figure 2. The parameters of the robot and the next waypoint.

In Figure 2, the distance (D) and the angle ($\Delta\theta$) are calculated as follows [10]:

$$D = \sqrt{(x_W - x_Q)^2 + (y_W - y_Q)^2} \tag{1}$$

$$\Delta\theta = \theta_Q - \theta_W \tag{2}$$

$$\tan\theta_W = \frac{y_W - y_Q}{x_W - x_Q} \tag{3}$$

2.2. The Limitations and Constraining Conditions

To solve the problem described in Section 2.1 above, some simplifications, limitations, and constraining conditions have been created and listed below.

- The robot moves on a horizontal plane.
- The friction of the robot's wheels with the floor is ignored.
- The robot moves without any slipping, skidding, or sliding.
- The center of mass of the robot coincides with the center of the axis between the right and the left wheels.

3. Preliminaries

In this section, the mathematic models of the DDR, the fuzzy logic control, and the genetic algorithm will be briefly reviewed.

3.1. Kinematic Model of the Differential Drive Robot

The kinematic model of the DDR describes how the robot's position changes over time from a geometric point of view. The geometry of the DDR is needed to understand the kinematic model of the DDR, which is shown in Figure 3 and Table 1 [10].

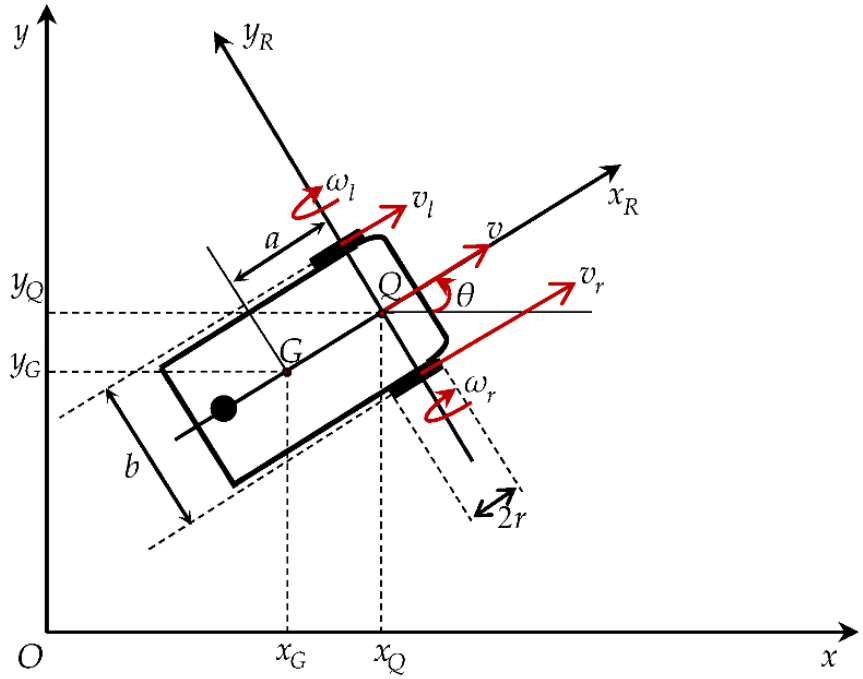


Figure 3. Geometry of a differential drive robot.

Table 1. The parameters of the differential drive robot.

No	Parameters (Notation)	Unit
01	The linear velocity of the DDR (v)	$m{\cdot}s^{-1}$
02	The orientation of the DDR (θ)	rad
03	The angular velocity of the right wheel (ω_r)	$rad{\cdot}s^{-1}$
04	The angular velocity of the left wheel (ω_l)	$rad{\cdot}s^{-1}$
05	The linear velocity of the right wheel (v_r)	$m{\cdot}s^{-1}$
06	The linear velocity of the left wheel (v_l)	$m{\cdot}s^{-1}$
07	The radius of the right and the left wheels (r)	m
08	The distance between the right and the left wheels (b)	m
09	The center of the axis between the right and the left wheels (Q)	-
10	The center of gravity of the DDR (G)	-
11	The distance between Q and G (a)	m

From Figure 3 and referring to [10,25,26], the kinematic model of the DDR is expressed by Equation (4) as follows:

$$\begin{cases} \dot{x} = x_Q = v\cos\theta \\ \dot{y} = y_Q = v\sin\theta \\ \dot{\theta} = \omega \end{cases} \tag{4}$$

3.2. The Actuator Dynamic Model

In this study, two brushed DC motors are used to drive the wheels of the DDR. The actuator model is needed to find the relationship between the control signal and the input of the mechanical system. The electrical circuit and the free body diagram of the rotor of a brushed DC motor are depicted in Figure 4 [25].

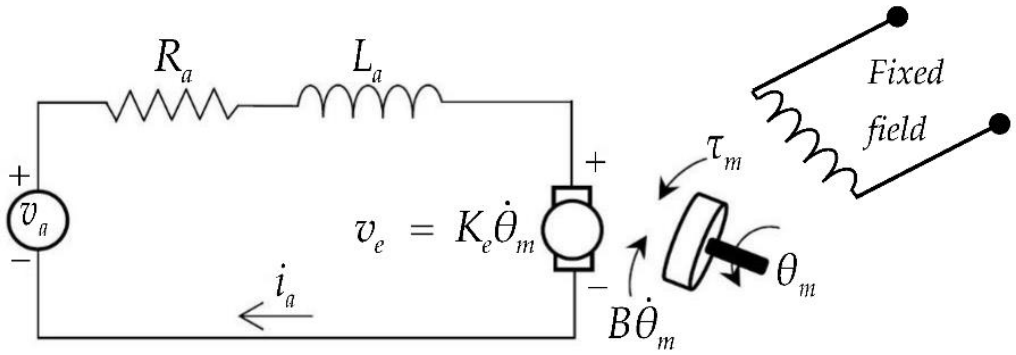


Figure 4. A basic electrical circuit and free body diagram of a DC motor.

An equation governing the mechanical dynamics of the DC motor is expressed in (5). The armature current that has its own dynamic equation is expressed in (6) as follows:

$$J\ddot{\theta}_m + B\dot{\theta}_m = K_t i_a \tag{5}$$

$$L_a \frac{\mathrm{d}i_a}{\mathrm{d}t} + R_a i_a = v_a - K_e\dot{\theta}_m \tag{6}$$

where J is the moment of inertia, θ_m is the angle of rotation of the output shaft, $\tau_m = K_t i_a$ is the mechanical torque developed in the motor rotor, K_t is the torque constant, i_a is the armature current, B is the damping coefficient, v_a is the input voltage, K_e is the back emf constant, R_a stands for the armature resistance, L_a is the armature inductance, and $v_e = K_e\dot{\theta}_m$ is the back emf [25].

Referring to [25], the relationship between the angular velocity ($\dot{\theta}_m$) and the input voltage (v_a) of the motor is expected in the form of a transfer function by Equation (7).

$$F(s) = \frac{\dot{\theta}_m(s)}{\dot{v}_a(s)} = \frac{K_t}{(R_a + sL_a)(Js + B) + K_t K_e} \tag{7}$$

Thus, the block diagram of the brushed DC motor can be expressed by Figure 5.

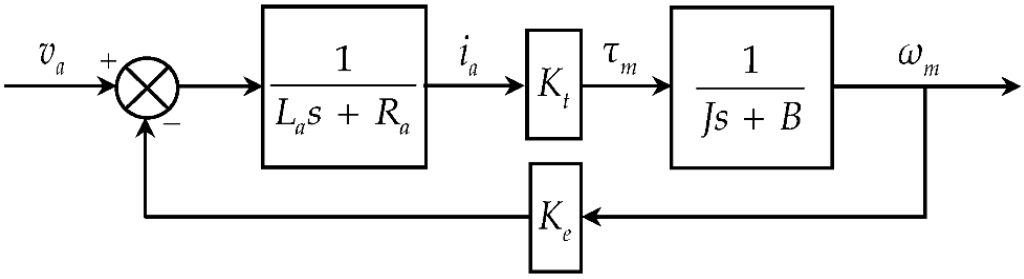


Figure 5. The block diagram of a brushed DC motor.

3.3. The Kinetic Energy of the Differential Drive Robot

The main energy losses of the DDR consist of the losses inside the motors, the kinetic energy loss, the losses due to the friction, the losses in the electronics, etc. [10,25,27]. However, in this study, for simplification, only the kinetic energy loss model is considered to design an energy-saving controller for the DDR.

When the DDR moves, it must accelerate, go with the constant velocity, and decelerate. Such actions of the robot will be related to its energy consumption. Specifically, part of the output power will be used to increase the kinetic energy and accelerate the robot. In theory, the kinetic energy will be transformed back during the deceleration phase, however, most of the transformed energy will be lost on heating [10,25]. The kinetic energy of the robot at any time is calculated by (8).

$$E_{\text{kinetic}} = \frac{1}{2}\left\{m[v(t)]^2 + I[\omega(t)]^2\right\} \tag{8}$$

where m and I are the mass and the moment of inertia of the robot and $v(t)$ and $\omega(t)$ are the linear velocity and angular velocity at time t of the robot, respectively.

3.4. The Fuzzy Logic Controller

The close loop schema of the DDR using the fuzzy logic controller can be depicted in Figure 6.

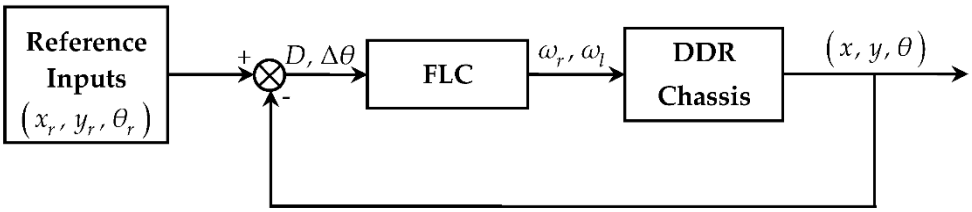


Figure 6. The close loop schema of the DDR using the fuzzy logic controller.

For the problem in this study, the Mamdani type is selected. The FLC used for the DDR in Figure 6 is structured as in Figure 7.

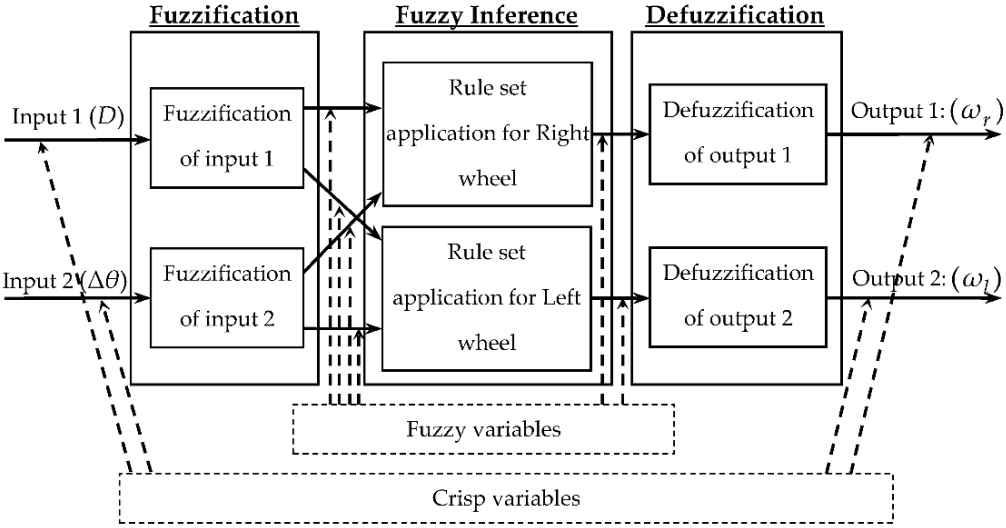


Figure 7. The fuzzy logic controller used for the differential drive robot.

The FLC has two inputs that are D and two outputs that are ω_r and ω_l. These inputs and outputs are indicated in Figure 2 and listed in Table 1. According to the value of the inputs, the FLC will decide the value of the outputs. Specifically, first, the crisp input values $(D, \Delta\theta)$ are converted into fuzzy values by the fuzzification. After that, a rule set in the form of IF-THEN is applied to obtain the fuzzy values of the outputs (ω_r, ω_l). Finally, the fuzzy values of the outputs are converted into crisp output values that are used to drive the DDR.

3.5. The Genetic Algorithm

As mentioned earlier, the GA has several outstanding benefits that are suitable for the design of an FLC. Therefore, in this study, the GA is selected to optimize both the parameters of the MFs and the FRs of the FLC for the DDR. The GA is based on the observations of the behavior of the nature [10,20]. Generally, a GA includes three operations, which are the selection, genetic operations, and replacement. For more details about the GA process, refer to [10,28]. In general, a GA cycle process is shown in Figure 8 [10].

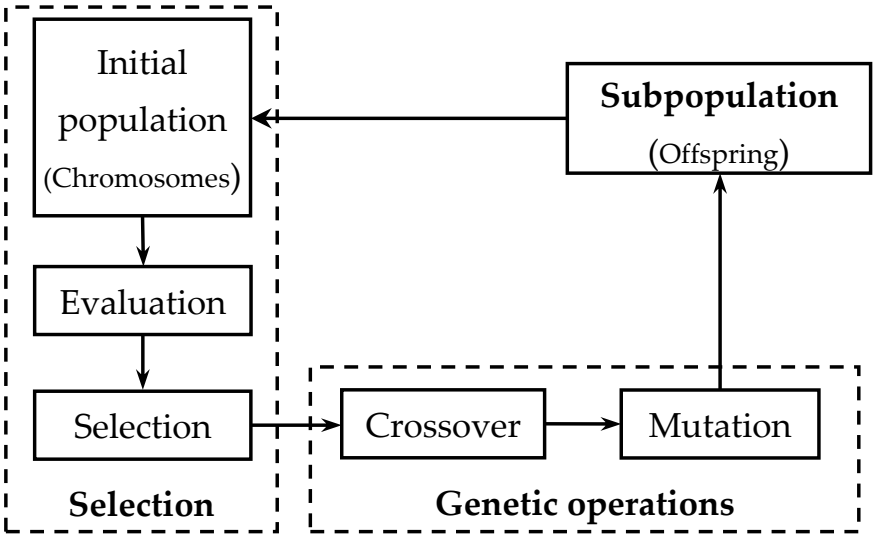


Figure 8. The schema of the genetic algorithm cycle.

4. The Design of the Fuzzy Logic Controller for the Differential Drive Robot

4.1. The Input and Output Variables for the Fuzzy Logic Controller

Several types of MFs can be used for FLC, such as the triangular MFs, the trapezoidal MFs, the Gaussian MFs, etc. However, in the WMR field, the triangular and trapezoidal MFs are applied more often than others [10]. In this study, to slightly simplify the problem, the triangular MFs are considered and each input/output are represented by only five MFs. The shape of each input and output MF is indicated in Figure 9. The name and notation of the input and output MFs are listed in Tables 2 and 3.

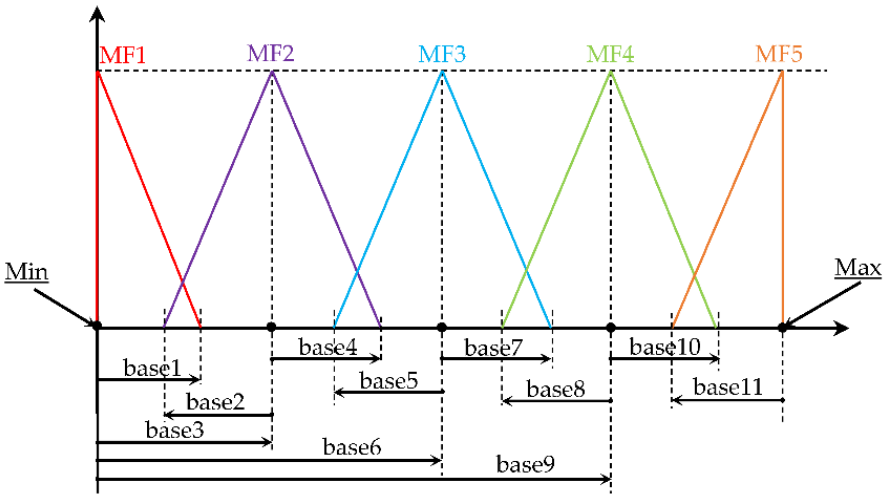


Figure 9. Representation of membership functions of each input and output.

Table 2. The input membership functions.

Input 1: (D)		Input 2: ($\Delta\theta$)	
Membership Functions	**Notation**	**Membership Functions**	**Notation**
Very Close	VC	Big Negative	BN
Close	C	Negative	N
Medium	M	Zero	Z
Far	F	Positive	P
Very Far	VF	Big Positive	BP

Table 3. The output membership functions.

Output 1: (ω_r)		Output 1: (ω_l)	
Membership Functions	**Notation**	**Membership Functions**	**Notation**
Very Small Right	VSR	Very Small Left	VSL
Small Right	SR	Small Left	SL
Medium Big Right	MBR	Medium Big Left	MBL
Big Right	BR	Big Left	BL
Very Big Right	VBR	Very Big Left	VBL

There are 44 parameters (from base1 to base44) for two inputs and two outputs of the FLC. The first 11 parameters (from base1 to base11) define five MFs of input1 (D). The next 11 parameters (from base12 to base22) define five MFs of input2 ($\Delta\theta$). Five MFs of output1 (ω_r) are represented by parameters (from base23 to base33). Additionally, five MFs of output2 (ω_l) are represented by parameters (from base34 to base44). The values of these parameters define the shape and size of all the MFs. The shape and size of the MFs directly affect the performance of the FLC, so it is important to find out a suboptimal set of all the input and output parameters (from base1 to base44).

4.2. The Fuzzy Rule Set for the Fuzzy Logic Controller

The FLC needs a rule set that allows it to derive the value of the fuzzy variable at the output from input variables. For the DDR, all the fuzzy linguistic variables of input1 (D) are combined with all the fuzzy linguistic variables of input2 ($\Delta\theta$) and there is one fuzzy linguistic representation of the output as a result [20]. Such operations can be represented by Tables 4 and 5.

Table 4. The rules of the right wheel angular velocity.

ΔD \ $\Delta\theta$	**BN**	**N**	**Z**	**P**	**BP**
VC	Out_R01	Out_R02	Out_R03	Out_R04	Out_R05
C	Out_R06	Out_R07	Out_R08	Out_R09	Out_R10
M	Out_R11	Out_R12	Out_R13	Out_R14	Out_R15
F	Out_R16	Out_R17	Out_R18	Out_R19	Out_R20
VF	Out_R21	Out_R22	Out_R23	Out_R24	Out_R25

Table 5. The rules of the left wheel angular velocity.

ΔD \ $\Delta\theta$	**BN**	**N**	**Z**	**P**	**BP**
VC	Out_L01	Out_L02	Out_L03	Out_L04	Out_L05
C	Out_L06	Out_L07	Out_L08	Out_L09	Out_L10
M	Out_L11	Out_L12	Out_L13	Out_L14	Out_L15
F	Out_L16	Out_L17	Out_L18	Out_L19	Out_L20
VF	Out_L21	Out_L22	Out_L23	Out_L24	Out_L25

The fuzzy linguistic representation of output1 (from Out_R1 to Out_R25) can be just one of VSR, SR, MBR, BR, or VBR and output2 (from Out_L1 to Out_L25) can be just one of VSL, SL, MBL, BL, or VBL. The selected one will directly affect the performance of the FLC, so the selection of the fuzzy output (VSR, SR, MBR, BR, or VBR) for output1 and (VSL, SL, MBL, BL, or VBL) for output2 are extremely important and need to be carried out.

4.3. The Optimization of the Fuzzy Logic Controller for the Differential Drive Robot

For the design of the FLC, determining the parameters of the MFs and the FRs is always a very difficult and extremely important problem because it directly affects the performance of the controller. Usually, these parameters and the FRs are determined based

on the heuristics and knowledge of the experts, which leads to great difficulty and is time consuming. Therefore, in this study, both the parameters of MFs and the FRs will be determined completely automatically by using the GA according to the minimum kinetic energy loss criterion.

4.3.1. Encoding Gene for Chromosome

In this study, both MFs and the rule set are optimized, so the 44 parameters (from base1 to base44), 25 fuzzy linguistic representation of output1 (from Out_R1 to Out_R25), and 25 fuzzy linguistic representation of output2 (from Out_L1 to Out_L25) need to be encoded into 94 genes of each chromosome used in the GA. This means that each chromosome has 94 genes that represent all the parameters of the input, output MFs, and all fuzzy linguistic representations of output1 and output2. The structure of each chromosome (Ch) is expressed in Figure 10. The best chromosome, which is the result of the GA process, will define the suboptimal FLC for DDR.

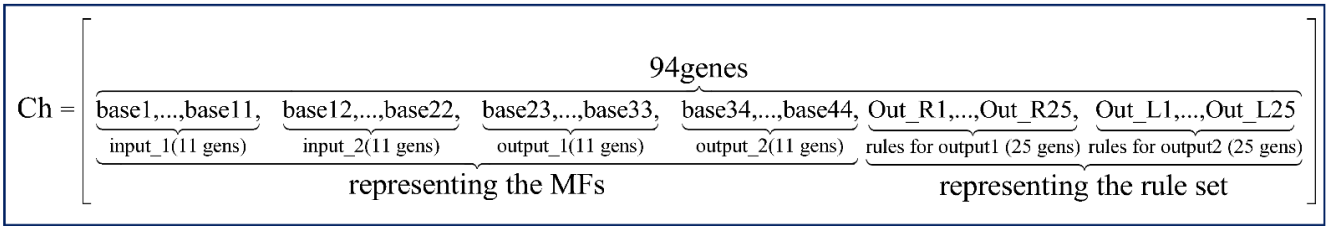


Figure 10. Representation of each chromosome.

4.3.2. Fitness Functions for the Genetic Algorithm

In the GA, an FF is defined to obtain the fitness value for each chromosome and to determine which one will be reproduced and brought into the next generation. Thus, with a specific chromosome, the FF returns a single numerical score, which is proportional to the ability of the chromosome (in this study, the score is a value of the robot's kinetic energy).

As mentioned in Section 3.3 before, the kinetic energy of the DDR at any time is calculated by (8). However, when implementing the movement of the DDR in the software, the discrete time is used. Thus, the kinetic energy of the robot at time (t_i) is calculated by Equation (9). The difference in kinetic energy between the two states is calculated by Equation (10). In addition, the kinetic energy loss is only for acceleration. Thus, the total kinetic energy loss at time (t_i) and, therefore, the FF at any time is calculated by Equation (11) as follows:

$$E_{\text{kinetic}}(i) = \frac{1}{2}m[v(t_i)]^2 + \frac{1}{2}I[\omega(t_i)]^2 \tag{9}$$

$$\Delta E_{\text{kinetic}}(i) = E_{\text{kinetic}}(i) - E_{\text{kinetic}}(i-1) \tag{10}$$

$$FF(i) = \sum E(i) = \sum E(i-1) + \Delta E_{\text{kinetic}}(i),\ (if\ \Delta E_{\text{kinetic}}(i) > 0) \tag{11}$$

where $E_{\text{kinetic}}(i)$ is the kinetic energy of the robot at time (t_i); $\Delta E_{\text{kinetic}}(i)$ is the difference in kinetic energy between the two states; $\sum E(i-1)$ is the total kinetic energy of the robot at time (t_{i-1}); and $FF(i) = \sum E(i)$ is the total kinetic energy of the robot at time (t_i).

4.3.3. The Optimization Process of the Fuzzy Logic Controller Using Genetic Algorithm

The optimization process of the FLC using the GA for the robot is shown in Figure 11.

As previously mentioned, all the parameters of the MFs and FRs are mapped to a chromosome, thus the FLC is created according to the chromosome. This chromosome represents all the parameters of the MFs and FRs, so its behavior can be tested by a simulation process. The results of the simulation are transformed into a fitness value. Such a process of mapping from the proposed solution to the FF result allows for the use of optimization methods. Therefore, it is expected that the same mapping applied to the

suboptimal solution will provide the wanted FLC. The tuning process using the GA in Figure 11 is executed by the flow chart of the GA, which is shown in Figure 12 [10].

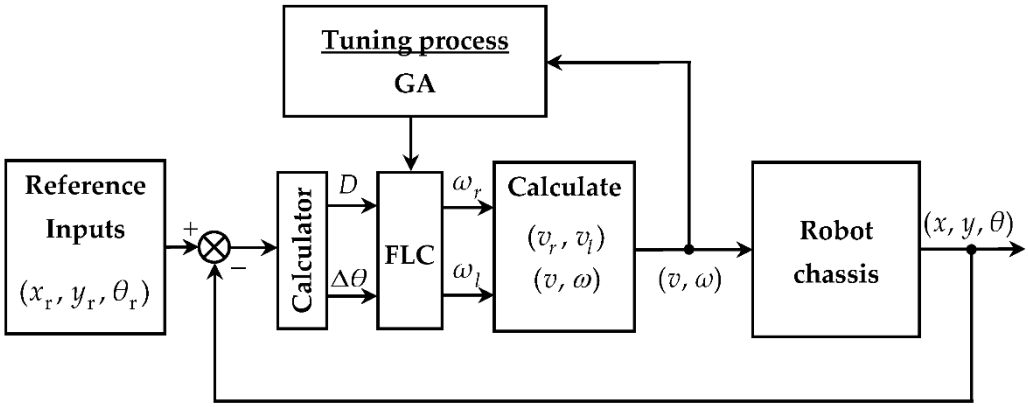


Figure 11. The optimization process of the fuzzy logic controller.

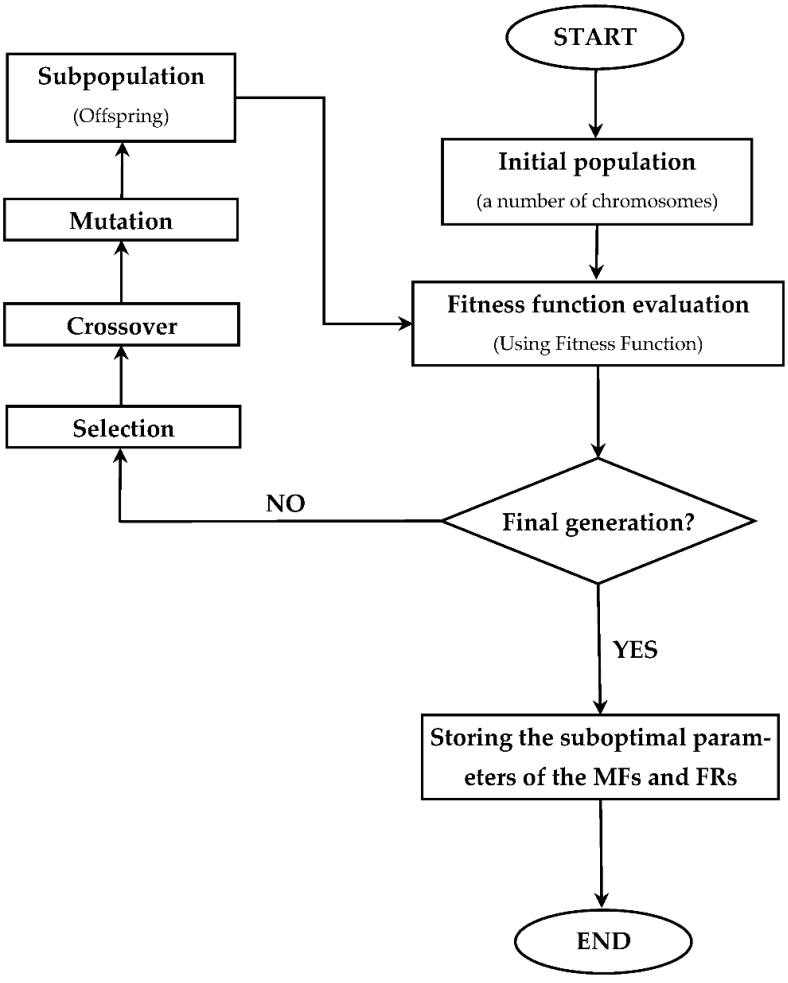


Figure 12. The flow chart of the genetic algorithm process.

The full optimization process of the FLC using the GA is implemented and programmed in Google Colab® using the Python language with the help of the Pymoo library to obtain the best chromosome. This means that the suboptimal parameters of the MFs and FRs are found and, thus, the suboptimal FLC is designed.

Pymoo is a multi-objective optimization framework in Python. It serves to solve the problems of practical needs, such as the parallelization of function evaluations, the

methods to visualize low- and high-dimensional spaces, and, especially, having tools for multi-criteria decision-making. For more details about Pymoo, refer to [29]; and readers should visit: https://pymoo.org (accessed on 14 November 2022) to learn how to optimize the code, what the GA settings are, and what integration with Pymoo looks like.

5. The Simulation Results and Discussion

In this section, the optimization process is implemented using a simulation program to find the suboptimal parameters of the MFs and FRs for the FLC in order to control the DDR to move to the goal through all the given optimal waypoints sequentially with a minimum of the kinetic energy loss.

5.1. Simulation Setup

5.1.1. The Main Parameters of the Robot

The robot's parameters used for this simulation are from the parameters of a real DDR. Some of them are provided by the manufacturer and the other parameters are identified and estimated by the tool in MATLAB-Simulink®. The main parameters of the DDR used to simulate in this study are listed in Table 6.

Table 6. The main parameters of the DDR used for simulation.

Parameters (Notation)	Value
The mass of the robot (m)	1.733 kg
The moment of inertia of the DDR (I)	0.08 kg·m^2
The torque constant (K_t)	0.001 Nm/A
The back emf constant (K_e)	0.03 V/m/s
The electrical resistance (R_a)	0.7 Ω
The electrical inductance (L_a)	0.01 H
The damping coefficient (B)	0.004 N·m·s
The moment of inertia of the motor (J)	0.037 kg·m^2·s^{-2}
The gear ratio of the DC motor (N)	30
The distance between two drive wheels ($2b$)	0.2 m
The drive wheel's radius (r)	0.0425 m

5.1.2. The Parameters of the Waypoints

First, a zigzag path is used as a standard complex path for the optimization process of the FLC. Then, several popular defined paths will be used to test the robot motion control by using suboptimal FLC that have been found in the optimization process. These paths are defined by the several given waypoints in a unit of a meter as listed in Table 7.

Table 7. Parameters of the paths used for simulation.

Paths	Waypoints (x, y)				
	Starting Point	1st Point	2nd Point	3rd Point	Goal
Zigzag path	(0, 0)	(2, 0)	(2, 2)	(4, 2)	(4, 4)
Square path	(0, 0)	(2, 0)	(2, 2)	(0, 2)	(0, 0)
Sharp turn path	(0, 0)	(2, 0)	(1, 2)	-	-
Straightforward path	(0, 0)	(2, 1.5)	(4, 3)	-	-
Diamond-shape path	(0, 0)	(2, –1)	(4, 0)	(2, 1)	(0, 0)

5.1.3. Intervals of the Optimization Problem Variables

All the variables of inputs and outputs in this study need to be bounded according to the real robot's parameters for the optimization process. This optimization problem works with the distance (D), angle ($\Delta\theta$), and the wheel's angular velocities (ω_r, ω_l). The interval of each variable is selected based on the robot parameters, the path, the environment, etc. So, based on the parameters of the DDR and the paths used in the experiments of this study,

the intervals of the input and output variables used for optimization problem are selected and listed in Table 8.

Table 8. The limit of the optimization problem variables.

Optimization Variables	Low Bound	High Bound
input1 (D)	0 m	10 m
input2 ($\Delta\theta$)	$-\pi$ rad	π rad
output1 (ω_r)	0 rad·s^{-1}	30 rad·s^{-1}
output2 (ω_l)	0 rad·s^{-1}	30 rad·s^{-1}

Therefore, the optimization parameters of input1 MFs (from base1 to base11) will be in the range [from 0 to 10] m. The optimization parameters of input2 MFs (from base12 to base22) will be in the range [from $-\pi$ to π] rad. The optimization parameters of output1 MFs and output2 MFs (from base23 to base44) will be in the range [from 0 to 30] rad·s^{-1}. In addition, for the optimization the FRs, the values of the genes represent the outputs (from Out_R1 to Out_R25) and (from Out_L1 to Out_L25) will be selected in the range [from 0.0 to 1.0]. The value of these genes including (from Out_R1 to Out_R25) and (from Out_L1 to Out_L25) will define the notation of each output (VSR/VSL, SR/SL, MBR/MBL, BR/BL, and VBR/VBL) as in Table 9.

Table 9. The mapping of genes into the notations of the output membership functions.

Genes Represent Outputs	Value	Notation of Output
From Out_R1/Out_L1 to Out_R25/Out_L25	[0.0–0.2)	VSR/VSL
From Out_R1/Out_L1 to Out_R25/Out_L25	[0.2–0.4)	SR/SL
From Out_R1/Out_L1 to Out_R25/Out_L25	[0.4–0.6)	MBR/MBL
From Out_R1/Out_L1 to Out_R25/Out_L25	[0.6–0.8)	BR/BL
From Out_R1/Out_L1 to Out_R25/Out_L25	[0.8–1.0]	VBR/VBL

5.1.4. Genetic Algorithm Setup

In this study, the Python language has been chosen to implement the optimization process of the FLC by using the GA. For the Python language, there is the Pymoo library that has already implemented a number of optimization algorithms, thus the GA will be implemented in Google Colab® with the help of the Pymoo library so that the suboptimal FLC can be found. For the optimization of the FLC, the GA was implemented in the Pymoo library with the size of the population set to 50 chromosomes and the generation count was set to 200. The size of the population and the number of the generation have been found as a good trade-off between the fast optimization process and the behavior of the resulting FLC.

5.2. Optimization Process Results

After running the optimization process program, the convergence of the fitness value over generations, the suboptimal MFs and suboptimal FRs when optimizing both MFs and FRs are found as listed in Tables 10–13 and shown in Figures 13 and 14.

Table 10. The founded suboptimal parameters of the input membership functions.

The Distance Error ΔD (m)/The Angle Error $\Delta\theta$ (rad)		
Name	**Notation**	**Range of Value**
Very Close/Big Negative	VC/BN	[0.0, 0.0, 3.95]/[$-\pi$, $-\pi$, -1.72]
Close/Negative	C/N	[2.93, 3.97, 4.55]/[-3.59, -1.57, 1.95]
Medium/Zero	M/Z	[4.13, 4.77, 6.54]/[-0.16, 0.41, 2.40]
Far/Positive	F/P	[7.30, 7.47, 15.79]/[-0.09, 1.56, 3.32]
Very Far/Big Positive	VF/BP	[7.24, 10, 10]/[0.004, π, π]

Table 11. The founded suboptimal parameters of the output membership functions.

The ω_r (rad/s)/The ω_l (rad/s)		
Name	**Notation**	**Range of Value**
Very Small Right/Left	VSR/VSL	[0, 0, 8.94]/[0, 0, 9.32]
Small Right/Left	SR/SL	[−9.15, 1.59, 14.57]/[6.40, 7.02, 7.80]
Medium Big Right/Left	MBR/MBL	[13.97, 16.41, 18.07]/[15.40, 16.73, 17.53]
Big Right/Left	BR/BL	[17.03, 17.10, 19.00]/[20.84, 27.36, 18.32]
Very Big Right/Left	VBR/VBL	[16.55, 30, 30]/[19.59, 30, 30]

Table 12. The suboptimal rules for the right wheel angular velocity.

$\Delta\theta$ / ΔD	**BN**	**N**	**Z**	**P**	**BP**
VC	MBR	VSR	SR	BR	VSR
C	VSR	VSR	VBR	VBR	VSR
M	VSR	SR	SR	VSR	VSR
F	MBR	MBR	BR	VBR	VSR
VF	BR	VBR	VBR	VSR	VSR

Table 13. The suboptimal rules for the left wheel angular velocity.

$\Delta\theta$ / ΔD	**BN**	**N**	**Z**	**P**	**BP**
VC	MBL	SL	SL	VSL	VSL
C	SL	VSL	MBL	SL	VSL
M	SL	MBL	SL	VBL	BL
F	BL	BL	BL	VBL	BL
VF	VBL	SL	VBL	BL	BL

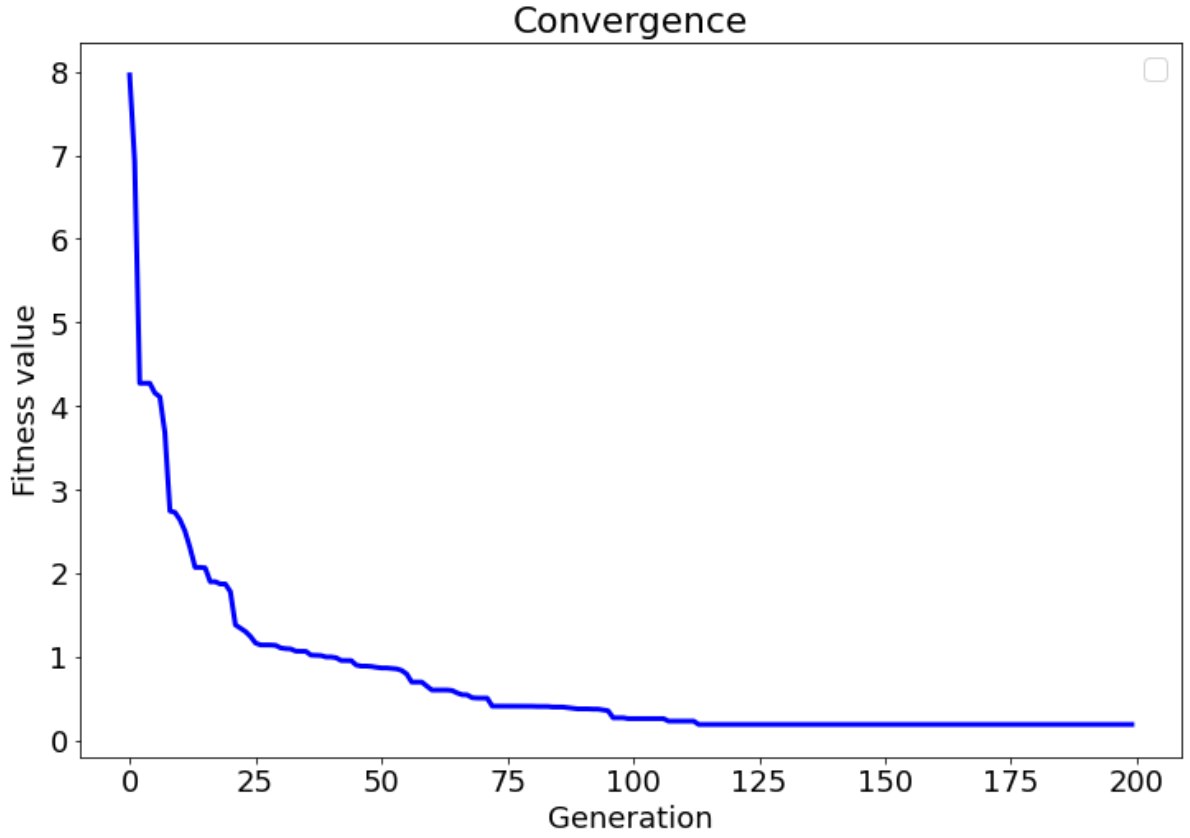


Figure 13. The convergence of the fitness value over generations.

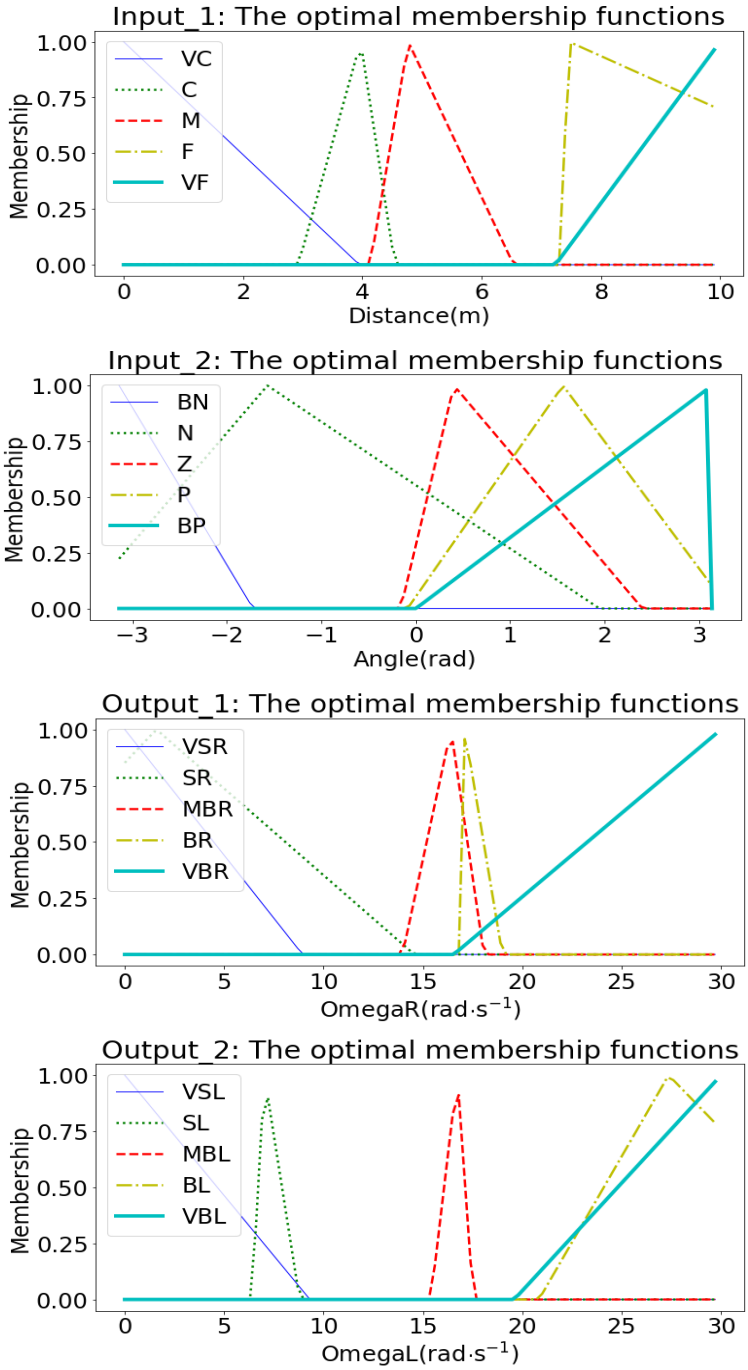


Figure 14. The suboptimal membership functions.

5.3. Simulation Results

In order to evaluate the effectiveness of the "Optimal_MFs_FRs_FLC", in which both the MFs and the FRs are optimized by the optimization process, two motion FLCs are selected and implemented to compare them with the "Optimal_MFs_FRs_FLC". Two motion FLCs are the "Thoa controller" and the "GA_FLC controller" in [10]. In this study, the "Thoa controller" is called "Expert_FLC" and the "GA_FLC controller" is called "Optimal_MFs_FLC". The "Expert_FLC" is designed by an expert. The MFs and the FRs of the "Expert_FLC" are designed based on the heuristics and knowledge of an expert. For the "Optimal_MFs_FLC", only its MFs are optimized, while its FRs are designed by an expert. For more details refer to [10,13]. The main simulation results are shown in Figures 15–19 and listed in Tables 14–18.

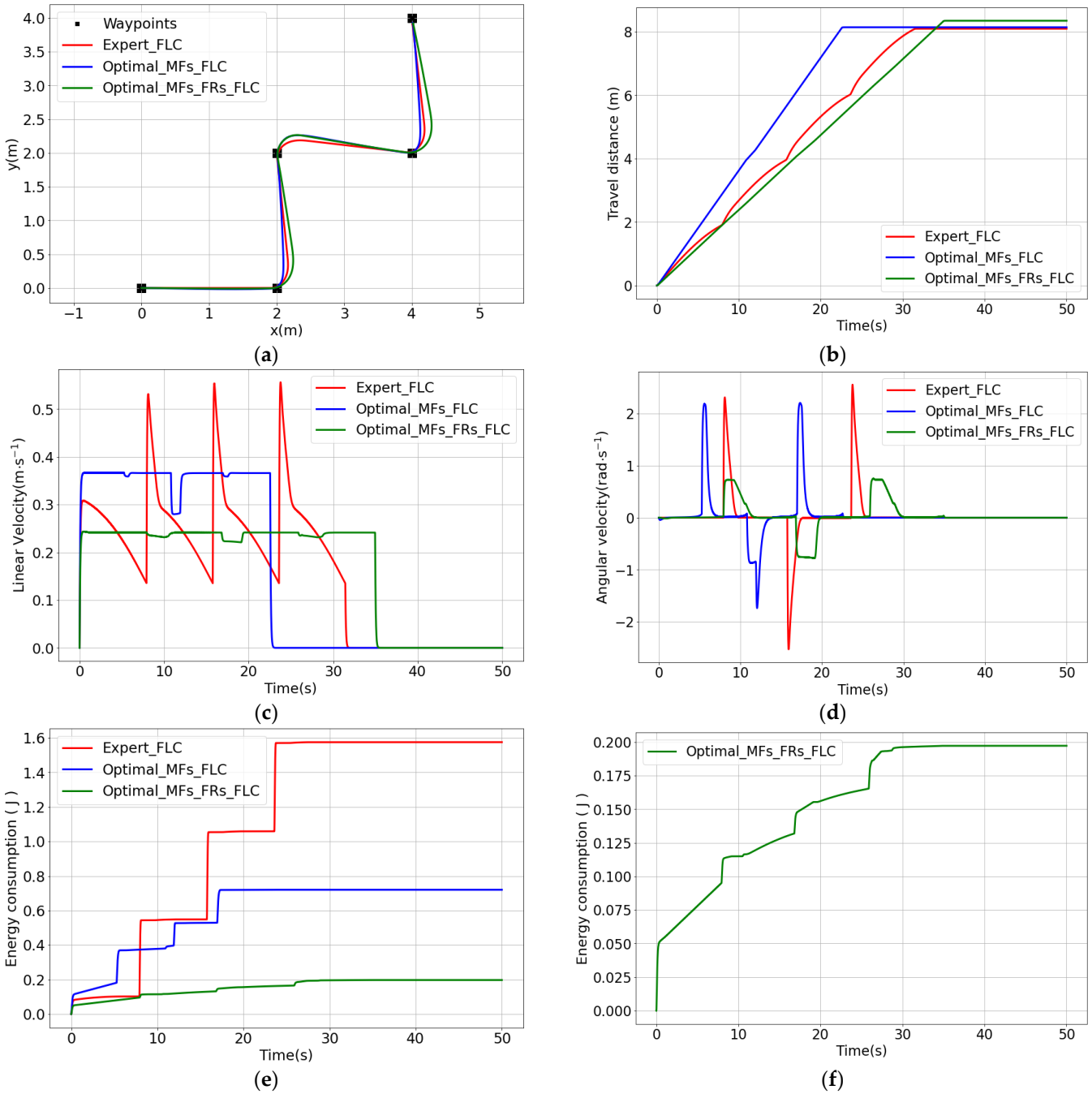


Figure 15. The simulation results in the case of using the zigzag path. (**a**) The trajectory of the robot; (**b**) The robot's travel distance over time; (**c**) The robot's linear velocity over time; (**d**) The robot's angular velocity over time; (**e**) The robot's kinetic energy loss over time; and (**f**) The zoom of the robot's kinetic energy loss over time.

Table 14. The kinetic energy loss and the travel distance of the robot when the robot tracks the zigzag path by the different controllers.

Controllers	Energy Consumption		Travel Distance		Travel Time
	E (J)	E (%)	s (m)	s (%)	Time(s)
Expert_FLC	1.575194	798.56	8.096951	100.00	28.33
Optimal_MFs_FLC	0.720139	365.08	8.142723	100.57	22.53
Optimal_MFs_FRs_FLC	0.197255	100.00	8.348827	103.11	34.95

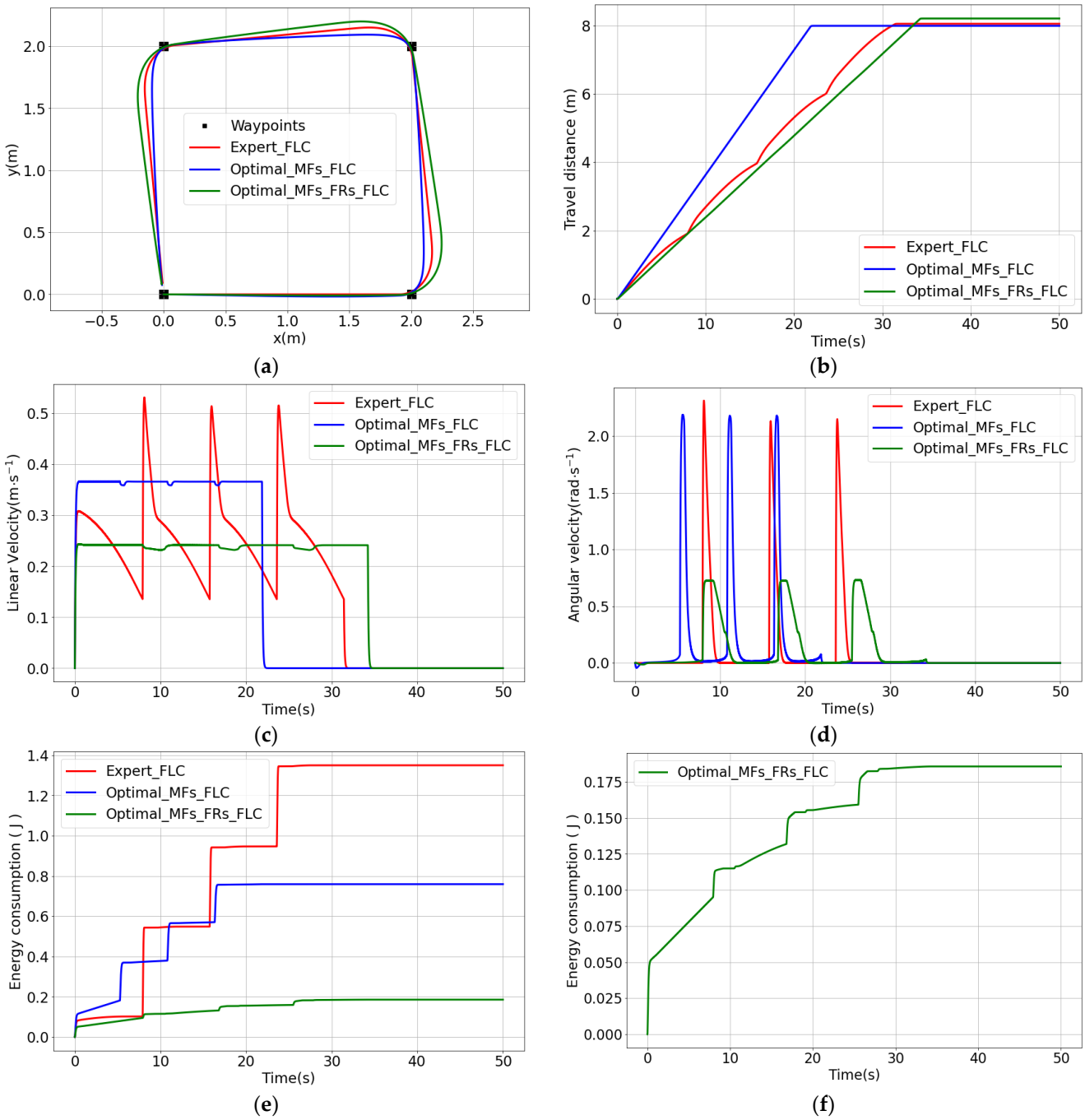


Figure 16. The simulation results in the case of using the square path. (**a**) The trajectory of the robot; (**b**) The robot's travel distance over time; (**c**) The robot's linear velocity over time; (**d**) The robot's angular velocity over time; (**e**) The robot's kinetic energy loss over time; and (**f**) The zoom of the robot's kinetic energy loss over time.

Table 15. The kinetic energy loss and the travel distance of the robot when the robot tracks the square path by the different controllers.

Controllers	Energy Consumption		Travel Distance	
	E (J)	E (%)	s (m)	s (%)
Expert_FLC	1.350298	727.19	8.048057	100.74
Optimal_MFs_FLC	0.759424	408.98	7.989019	100.00
Optimal_MFs_FRs_FLC	0.185687	100.00	8.202735	102.68

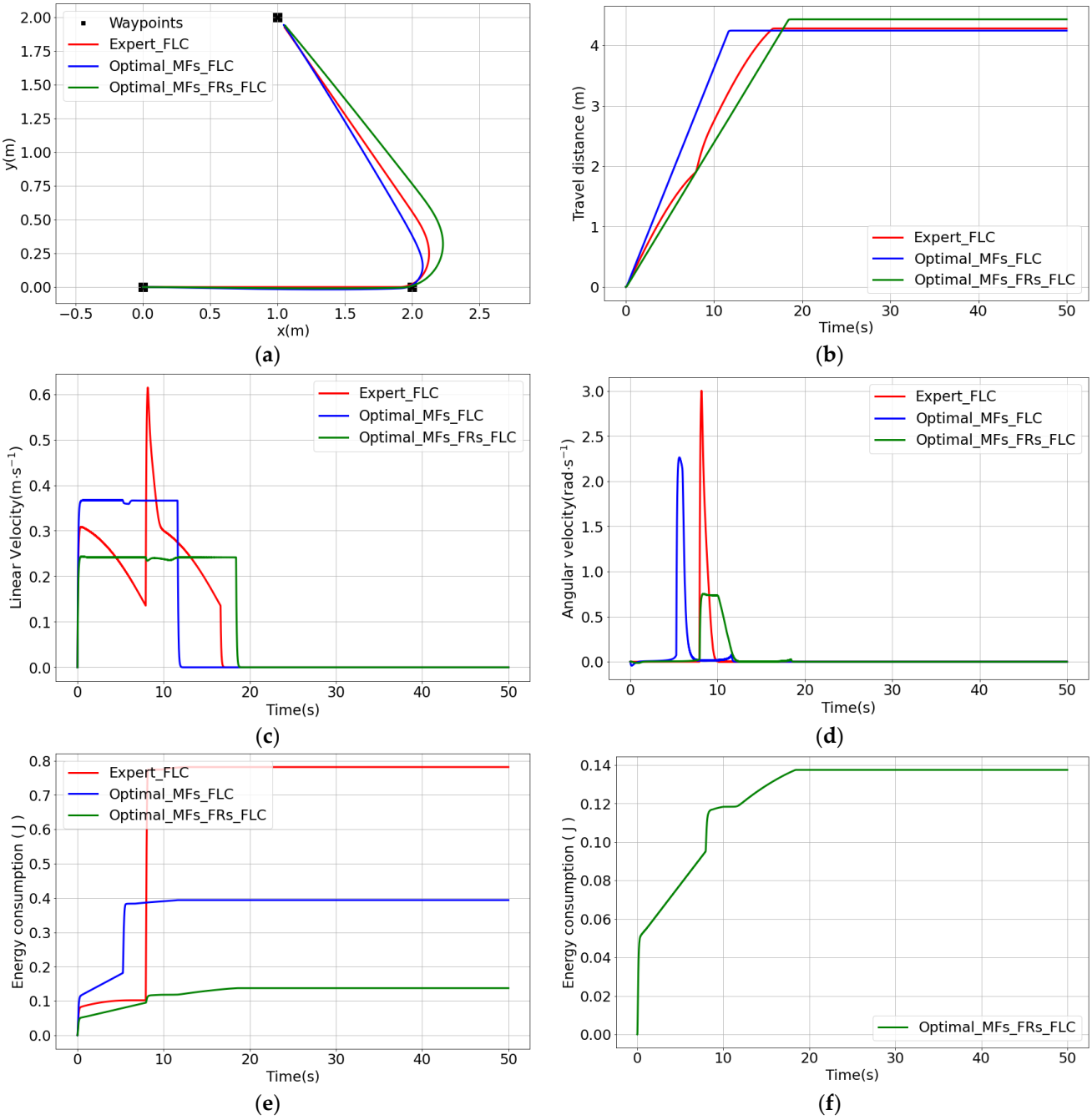


Figure 17. The simulation results in the case of using the sharp turn path. (**a**) The trajectory of the robot; (**b**) The robot's travel distance over time; (**c**) The robot's linear velocity over time; (**d**) The robot's angular velocity over time; (**e**) The robot's kinetic energy loss over time; and (**f**) The zoom of the robot's kinetic energy loss over time.

Table 16. The kinetic energy loss and the travel distance of the robot when the robot tracks the sharp turn path by the different controllers.

Controllers	Energy Consumption		Travel Distance	
	E (J)	E (%)	s (m)	s (%)
Expert_FLC	0.781463	568.37	4.279199	100.83
Optimal_MFs_FLC	0.393902	286.49	4.243791	100.00
Optimal_MFs_FRs_FLC	0.137491	100.00	4.430193	104.39

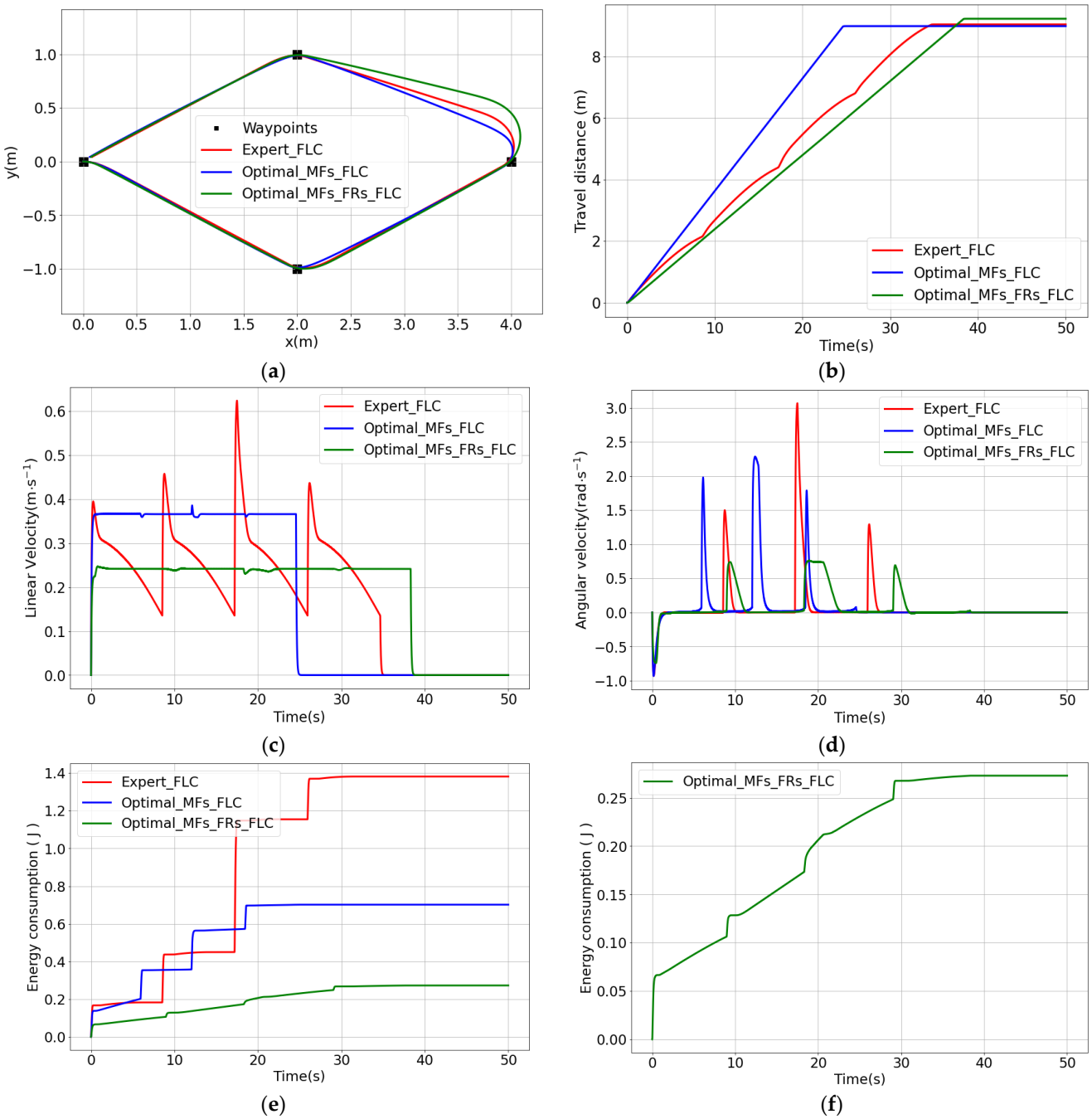


Figure 18. The simulation results in the case of using the diamond-shaped path. (**a**) The trajectory of the robot; (**b**) The robot's travel distance over time; (**c**) The robot's linear velocity over time; (**d**) The robot's angular velocity over time; (**e**) The robot's kinetic energy loss over time; and (**f**) The zoom of the robot's kinetic energy loss over time.

Table 17. The kinetic energy loss and the travel distance of the robot when the robot tracks the diamond-shaped path by the different controllers.

Controllers	Energy Consumption		Travel Distance	
	E (J)	E (%)	s (m)	s (%)
Expert_FLC	1.381601	506.32	9.037577	100.60
Optimal_MFs_FLC	0.701540	257.09	8.983455	100.00
Optimal_MFs_FRs_FLC	0.272872	100.00	9.222562	102.66

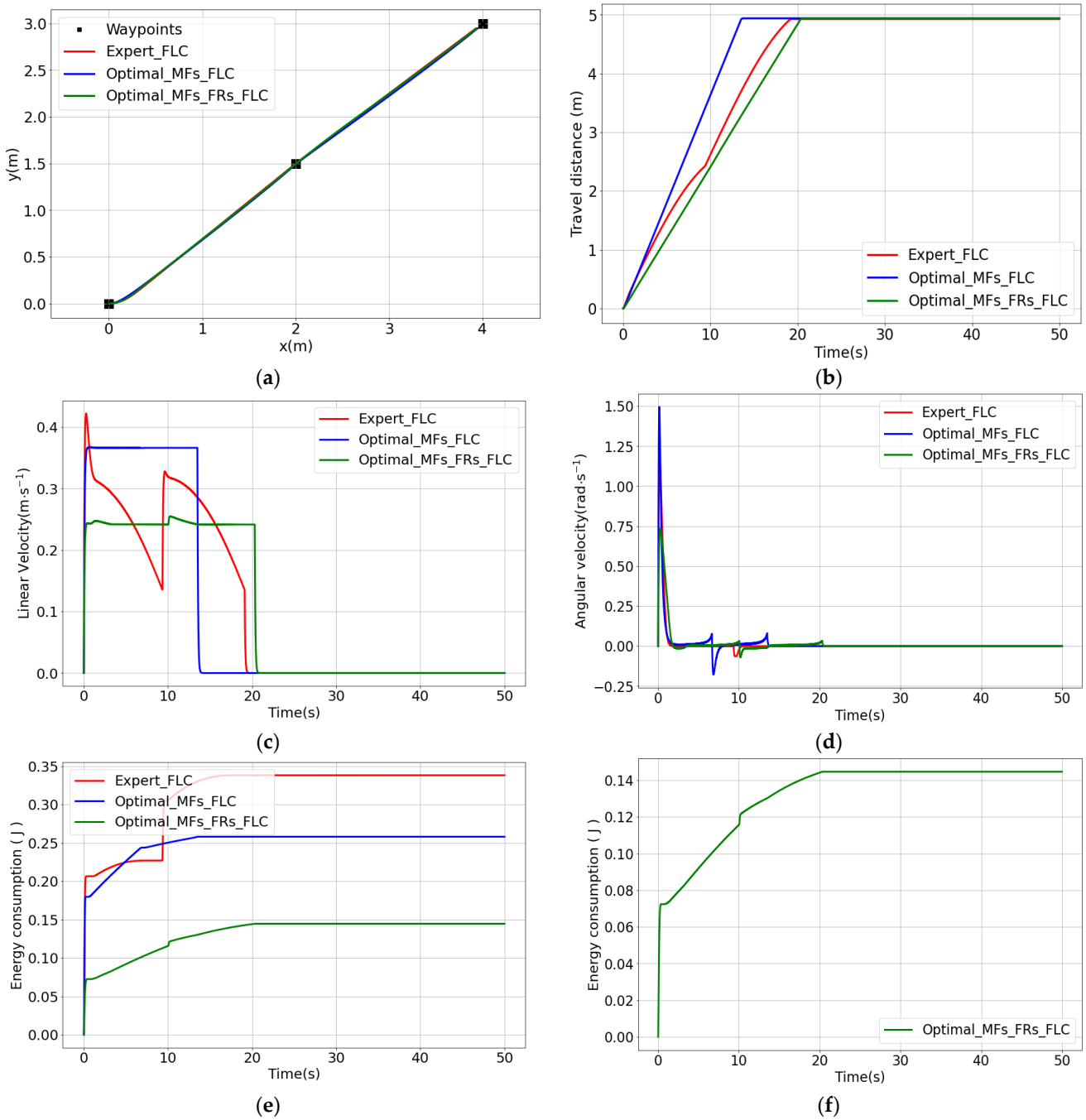


Figure 19. The simulation results in the case of using the straightforward path. (**a**) The trajectory of the robot; (**b**) The robot's travel distance over time; (**c**) The robot's linear velocity over time; (**d**) The robot's angular velocity over time; (**e**) The robot's kinetic energy loss over time; and (**f**) The zoom of the robot's kinetic energy loss over time.

Table 18. The kinetic energy loss and the travel distance of the robot when the robot tracks the straightforward path by the different controllers.

Controllers	Energy Consumption		Travel Distance	
	E (J)	E (%)	s (m)	s (%)
Expert_FLC	0.338241	233.85	4.926989	100.00
Optimal_MFs_FLC	0.258087	178.44	4.938676	100.24
Optimal_MFs_FRs_FLC	0.144638	100.00	4.934045	100.14

5.4. Discussion

5.4.1. The Robot Tracks the Zigzag Path

The zigzag path is a popular and complex path that is selected for the optimization process in order to find both the suboptimal MFs and the suboptimal FRs for the "Optimal_MFs_FRs_FLC" according to the minimum kinetic energy loss criterion. Figure 13 shows the value of the FF (kinetic energy loss). In this case, the value of the FF is rapidly reduced during the optimization process (from the 1st generation to the 50th generation). Then, the value of this FF is slightly decreased from the 51st generation to the 115th generation. From the 116th generation onwards, the value of the FF converges to E = 0.197255 J and remains at this value. This optimization process is based on random behavior and has been executed repeatedly, so it can be concluded that the FF in the optimization process has converged and reached a stable suboptimal value.

The suboptimal MFs and the suboptimal FRs of the "Optimal_MFs_FRs_FLC" are shown in Tables 10–13 and Figure 14. These are the main results of the optimization process. Using the "Optimal_MFs_FRs_FLC", the "Expert_FLC", and the "Optimal_MFs_FLC" to drive the DDR, all the simulation results are shown in Figures 15–19 and in Tables 14–18. Figure 15 and Table 14 have served as a base for a comparison between the "Optimal_MFs_FRs_FLC" and the other used controllers. Specifically, when the DDR is controlled by the "Optimal_MFs_FRs_FLC" to track the zigzag path, the kinetic energy loss of the robot (E = 0.197255 J) is much lower than the kinetic energy loss of the robot controlled by the other used controllers (for the "Expert_FLC", E = 1.575194 J; and for the "Optimal_MFs_FLC", E = 0.720139 J). In this case, the "Optimal_MFs_FRs_FLC" has saved 698.56% kinetic energy loss while the travel distance in this case is just a bit bigger (3.11%) in comparison with the case of using the "Expert_FLC" and the "Optimal_MFs_FRs_FLC" has saved 265.08% kinetic energy loss while the travel distance in this case is just a bit bigger (2.54%) in comparison with the case of using the "Optimal_MFs_FLC".

The velocity of the robot when it is controlled by the "Optimal_MFs_FRs_FLC" is lower than the velocities of the robot when it is controlled by the other controllers. Therefore, in this case, the travel time of the robot is bigger than the travel times of the robot when it is controlled by the other controllers (see Figure 15b,c and Table 14). Specifically, the travel times are 28.33 s, 22.53 s, and 34.95 s when the robot is navigated by the "Expert_FLC", the "Optimal_MFs_FLC", and the "Optimal_MFs_FRs_FLC", respectively. For the main purpose of minimizing kinetic energy loss, such a difference in speed and travel time is acceptable. Fortunately, this leads to the advantage of causing the robot to move more smoothly.

In addition, Figure 15c,d show that the linear velocity and the angular velocity of the DDR controlled by the "Expert_FLC" oscillated over time, which has led to the vibrations and jerks of the robot. However, these problems of the vibrations and jerks have been fixed quite well by the "Optimal_MFs_FLC" and the "Optimal_MFs_FRs_FLC". When the "Optimal_MFs_FLC" and the "Optimal_MFs_FRs_FLC" are used, the linear velocity is almost unchanged, its value just slightly changes at each corner of the path. Moreover, Figure 15d clearly shows that the angular velocity of the robot controlled by the "Optimal_MFs_FRs_FLC" is always the smallest, especially at the time when the robot turns at each corner of the path. Therefore, the issues about the robot vibration and the robot jerk have been solved quite well by this controller.

5.4.2. The Robot Tracks the Other Typical Path

In order to investigate the operating ability of the "Optimal_MFs_FRs_FLC", this controller, the "Expert_FLC", and the "Optimal_MFs_FLC" are used to control the robot to move in several typical different paths including square path, sharp turn path, diamond-shaped path, and straightforward path (see Figures 16–19 and Tables 15–18).

Figures 16–19 and Tables 15–18 clearly indicate that the performances of the motion control are still very good when the robot is controlled by the "Optimal_MFs_FRs_FLC" compared to the "Expert_FLC" and the "Optimal_MFs_FLC" in all the cases of using

the given paths. Specifically, a 408.98%, 286.49%, 257.09%, and 178.44% reduction in the kinetic energy loss was achieved using the proposed controller compared to the best of the used controllers when the DDR tracks the square path, the sharp turn path, the diamond-shaped path, and the straightforward path, respectively. Especially, the limitations of the other controllers such as the vibration and the jerk have been well solved by the "Optimal_MFs_FRs_FLC". So, it can be also concluded that the "Optimal_MFs_FRs_FLC" is the best controller for the DDR on the different given paths.

As mentioned earlier, the main goal of this study is to design an FLC to drive the DDR from the starting point to the destination through all the given waypoints by the optimization process with the help of the GA based on minimizing the kinetic energy loss criterion. Thus, from all the analyses above, it could be concluded that the "Optimal_MFs_FRs_FLC" is the most suitable one in the used FLCs. In other words, the "Optimal_MFs_FRs_FLC" is much better than the others in terms of energy-saving while the other performances of motion control are still good.

6. Conclusions

The paper presents the DDR motion control. The content of the paper focuses on the problem of designing the fuzzy logic energy-saving motion controller for the DDR so that the robot can move to the given destination through all the given waypoints with the minimum kinetic energy loss and still ensure the other performances of the motion control. This study solves the difficulties in the problem of determining the parameters of MFs and the FRs when designing the FLC for DDR. Specifically, first, the zigzag path is used for the optimization process by using the GA based on the minimum energy consumption criterion to find out both the suboptimal MFs and the suboptimal FRs. Then, this suboptimal FLC is used to evaluate its performances of motion control when the DDR tracks the other popular paths. Finally, the "Optimal_MFs_FRs_FLC", the "Optimal_MFs_FLC", and the "Expert_FLC" are successfully implemented in Google Colab® by using the Python language to compare them with each other. The simulation results have shown that the "Optimal_MFs_FRs_FLC" is significantly better than the used controllers in terms of the minimum kinetic energy loss while the other performances of the motion control are still good.

The main contributions of this study are: (1) providing a new method to efficiently design an FLC by using the GA for the DDR. This method allows any user even a naïve user, to design their own effective FLC; (2) both the MFs and the FRs are automatically generated by the optimization process; (3) providing a program in Google Colab® by using the Python language, which serves as a tool to execute the optimization process, simulate and evaluate the several robot motion control algorithms. This program is an open-source code for so that the readers can browse, view, run, and modify the code themselves. The open-source codes are stored in the following link: https://colab.research.google.com/drive/1gIcl7ZUu4wAbekBebDa-RBonOQ3nsvh1?usp=sharing (accessed on 14 November 2022). In future work, the "Optimal_MFs_FRs_FLC", the "Expert_FLC", and the "Optimal_MFs_FLC" will be implemented on a real DDR to test and evaluate the performances of those controllers.

Author Contributions: Conceptualization, V.T.P.; methodology, V.T.P., A.S. and V.K.; software, V.T.P. and A.S.; validation, V.T.P., A.S., V.K. and T.S.N.; formal analysis, V.T.P., A.S., V.K. and T.S.N.; investigation, V.T.P. and A.S.; resources, A.S. and V.T.P.; data curation, V.T.P.; writing—original draft preparation, V.T.P. and T.S.N.; writing—review and editing, V.T.P. and T.S.N.; visualization, V.T.P., A.S. and T.S.N.; supervision, V.T.P.; project administration, V.T.P.; funding acquisition, A.S. and V.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by internal fund "DZRO Cyber forces and resources" of University of Defence, the Czech Republic.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The simulation as open-source code is linked to https://colab.research.google.com/drive/1gIcl7ZUu4wAbekBebDa-RBonOQ3nsvh1?usp=sharing (accessed on 14 November 2022).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

List of Abbreviations

DDR	Differential Drive Robot
FF	Fitness Function
FL	Fuzzy Logic
FLC	Fuzzy Logic Controller
FRs	Fuzzy Rules
GA	Genetic Algorithm
WMR	Wheeled Mobile Robot

List of Variables

v	The linear velocity of the DDR	$m \cdot s^{-1}$
v_r	The linear velocity of the right wheel	$m \cdot s^{-1}$
v_l	The linear velocity of the left wheel	$m \cdot s^{-1}$
θ	The orientation of the DDR	rad
ω	The angular velocity of the DDR	$rad \cdot s^{-1}$
ω_r	The angular velocity of the right wheel	$rad \cdot s^{-1}$
ω_l	The angular velocity of the left wheel	$rad \cdot s^{-1}$
r	The radius of the right and the left wheels	m
b	The distance between the right and the left wheels	m
m	The total mass of the robot	kg
I	The total equivalent moment of inertia of the DDR	$kg \cdot m^2$
K_t	The torque constant of the DC motor	$N \cdot m \cdot A^{-1}$
K_e	The back emf constant of the DC motor	$V \cdot m^{-1} \cdot s^{-1}$
R_a	The electrical resistance of the DC motor	Ω
L_a	The electrical inductance of the DC motor	H
$\Delta\theta$	The angle error between the current robot's position and the next waypoint	rad
ΔD	The distance error between the current robot's position and the waypoint's position	m

References

1. Armah, S.; Yi, S.; Abu-Lebdeh, T. Implementation of autonomous navigation algorithms on two-wheeled ground mobile robot. *Am. J. Eng. Appl. Sci.* **2014**, *7*, 149–164. [CrossRef]
2. Kumar, U.D.; Nisha, M.; Mathivanan, N. Tracking of a PID driven differential drive mobile robot. *Int. J. Mechatron. Electr. Comput. Technol.* **2018**, *8*, 3690–3704.
3. Wang, Y.; Miao, Z.; Zhong, H.; Pan, Q. Simultaneous stabilization and tracking of nonholonomic mobile robots: A lyapunov-based approach. *IEEE Trans. Control Syst. Technol.* **2015**, *4*, 1440–1450. [CrossRef]
4. Shuying, P.; Wuxi, S. Adaptive fuzzy output feedback control of a nonholonomic wheeled mobile robot. *IEEE Access* **2018**, *6*, 43414–43424.
5. Cui, M. Observer-based adaptive tracking control of wheeled mobile robots with unknown slipping parameters. *IEEE Access* **2019**, *7*, 169646–169655. [CrossRef]
6. Azza, E.; Sayed, B.I. Wheeled mobile robot trajectory tracking using sliding mode control. *J. Comput. Sci.* **2016**, *12*, 48–55.
7. Yasmine, K.; Mohamed, B.; Tarak, D. Adaptive sliding-mode dynamic control for path tracking of nonholonomic wheeled mobile robot. *J. Autom. Syst. Eng.* **2015**, *2*, 119–131.
8. Ran, Z.; Dong, H.L.; Hong, K.L. Mobile robot navigation using optimized fuzzy controller by genetic algorithm. *Int. J. Fuzzy Log. Intell. Syst.* **2015**, *15*, 12–19.
9. Mohammed, F.; Mohammed, A.; Bencherif, M.A.; Habib, D.; Mohamad, M.A.R. Human expertise in mobile robot navigation. *IEEE Access. J.* **2017**, *6*, 1694–1705.
10. Štefek, A.; Pham, V.T.; Krivánek, V.; Pham, K.L. Optimization of fuzzy logic controller used for a differential drive wheeled mobile robot. *Appl. Sci.* **2021**, *11*, 6023. [CrossRef]
11. Yue, M.; Wang, L.; Ma, T. Neural network based terminal sliding mode control for WMRs affected by an augmented ground friction with slippage effect. *IEEE/CAA J. Autom. Sin.* **2017**, *3*, 498–506. [CrossRef]
12. Ritesh, B.M. Motion control of wheeled mobile robots using fuzzy logic. *Int. J. Recent Technol. Eng.* **2013**, *2*, 89–93.

13. Thoa, T.M.; Cosmin, C.; Keyser, D.R.; Trung, D.T.; Thich, V. MIMO fuzzy control for autonomous mobile robot. *J. Autom. Control Eng.* **2016**, *4*, 65–70.
14. Mehdi, Z.; Ebrahim, A.; Mehdi, F. A Review on optimization of fuzzy controller parameters in robotic applications. *IETE J. Res.* **2020**, *68*, 4150–4159.
15. Perez, J.; Valdez, F.; Castillo, O.; Melin, P.; Gonzalez, C.; Martinez, G. Interval type-2 fuzzy logic for dynamic parameter adaptation in the bat algorithm. *Soft Comput.* **2017**, *3*, 67–85. [CrossRef]
16. Pérez, J.; Valdez, F.; Castillo, O. A New bat algorithm augmentation using fuzzy logic for dynamical parameter adaptation. In *Mexican International Conference on Artificial Intelligence*; Springer: Cham, Switzerland, 30 December 2015; pp. 33–43.
17. Frumen, O.; Leticia, A.A.; Jonathan, P.; Camilo, C.; Fevrier, V.; Oscar, C. Comparative study of type-2 fuzzy particle swarm, bee colony and bat algorithms in optimization of fuzzy controllers. *Algorithms. J.* **2017**, *10*, 101.
18. Ahmet, A.; Mehmet, K. Determination of fuzzy logic membership functions using genetic algorithms. *Fuzzy Sets Syst. J.* **2001**, *118*, 297–306.
19. Mohammadian, M.; Stonier, R.J. Tuning and optimisation of membership functions of fuzzy logic controllers by genetic algorithms. In Proceedings of the 1994 3rd IEEE International Workshop on Robot and Human Communication, Nagoya, Japan, 18–20 July 1994; pp. 356–361.
20. Stefek, A.; Frantis, P. Optimization of fuzzy logic based virtual pilot for wargaming. *Mathematics* **2021**, *9*, 3169. [CrossRef]
21. Mucientes, M.; Moreno, D.L.; Bugarin, A.; Barro, S. Design of a fuzzy logic in mobile robotics using genetic algorithms. *Appl. Soft Comput.* **2007**, *7*, 540–546. [CrossRef]
22. Cintra, M.E.; Camargo, H. Fuzzy rules generation using genetic algorithms with self-adaptive selection. In Proceedings of the 2007 IEEE International Conference on Information Reuse and Integration, Las Vegas, NV, USA, 13–15 August 2007; pp. 261–266.
23. Kumarasamy, K.S. Single robot motion planning using fuzzy-genetic technique. *Control Intell. Syst. J.* **2010**, *1*, 1–8.
24. Dilip, K.P.; Kalyanmoy, D.; Amitabha, G. A genetic-fuzzy approach for mobile robot navigation among moving obstacles. *international J. Approx. Reason.* **1999**, *20*, 145–172.
25. Štefek, A.; Pham, V.T.; Krivánek, V.; Pham, K.L. Energy comparison of controllers used for a differential drive wheeled mobile robot. *IEEE Access J.* **2020**, *8*, 170915–170927. [CrossRef]
26. Spyros, G.T. *Introduction to Mobile Robot Control*, 1st ed.; Elsevier: Amsterdam, The Netherlands, 2014; pp. 137–317.
27. Mudasser, W.; Fernando, R.; Adel, E.S. Energy modeling of differential drive robots. In Proceedings of the IEEE SoutheastCon 2015, Fort Lauderdale, FL, USA, 9–12 April 2015; pp. 1–6.
28. Zilouchian, A.; Jamshidi, M. *Intelligent Control Systems Using Soft Computing Methodologies*, 1st ed.; Ali, Z., Mo, J., Eds.; CRC Press: Boca Raton, FL, USA, 27 March 2001; pp. 211–234, ISBN 9780849318757.
29. Blank, J.; Deb, K. Pymoo: Multi-Objective Optimization in Python. *IEEE Access* **2020**, *8*, 89497–89509. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.