Received: June 13, 2022. Revised: July 1, 2022.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Intelligent Path Planning Algorithm and Control Strategy Design for Multi-Mobile Robots based on a Modified Elman Recurrent Neural Network

Zainab E. Kanoon[1]* Ahmed Sabah Al-Araji[1] Mohammed Najm Abdullah[1]

[1]*Computer Engineering Department, University of Technology –Iraq, Baghdad, Iraq*
* Corresponding author's Email: 120105@uotechnology.edu.iq

Abstract: The goal of the navigation process is to find the optimal path for the mobile robot and control its motion on that path without any oscillation. This work aims to find the optimal paths for multi-mobile robots working in the same static environment. To achieve this goal, the proposed quarter orbits particle swarm optimization (QOPSO) algorithm, which is an enhancement of the cell decomposition algorithm, will be used. The main advantage of using the QOPSO algorithm is to generate the shortest path and avoid collision with static obstacles. Moreover, to direct the motion of the three mobile robots on the desired predefined paths, a proposed inverse differential kinematic neural network trajectory tracking (IDKNNTT) controller based on a modified Elman recurrent neural network (MERNN) will be used. This proposed controller is used to control the nonlinear kinematics mobile robots' system to smoothly and quickly generate the left and right wheels' velocities of the multi mobile robots, which are used to control the orientation and position of each mobile robot. Furthermore, using the proposed controller minimizes the tracking error in the X-axis and the Y-axis positions, approximately zeroes the orientation error, and provides no oscillation in the responses. In particular, the controller guarantees that all the mobile robots will follow their desired paths quickly and correctly. Finally, we validate the numerical simulation results of the proposed control strategy by comparing them to those of other types of controllers in terms of the maximum error enhancement in the X-position and the Y-position. Particularly, when the proposed controller was compared to the convolutional neural network trajectory tracking (CNNTT) controller, the comparison results show that the proposed controller improves the tracking error rate on the X-axis by 75.5 % and enhances the tracking error rate on the Y-axis by 21.2 %. In addition, the proposed controller was compared to the MIMO-PID-MENN controller, and the comparison results show that the proposed controller improves the tracking error rate on the X-axis by 33.3 % and on the Y-axis by 40.6 %.

Keywords: Quarter orbits particle swarm optimization algorithm, Path planning, Inverse differential kinematic controller, Modified Elman recurrent neural network, Mobile robot model, Trajectory tracking.

1. Introduction

There are numerous applications for mobile robots in various areas of our lives, including robotics, medicine, virtual reality, bioinformatics, and search and rescue operations [1]. The design and building of mobile robot systems that work autonomously in complex, dynamic, and uncertain situations have remained a difficulty throughout the previous decade. Such systems must be able to accomplish many jobs, which necessitates the integration of a range of knowledge-intensive information processes at various levels of abstraction to provide real-time execution, robustness, adaptability, and scalability [2]. Mobile robots are now being developed to be used in multi-robot systems, in which numerous mobile robots are linked to one another. The systems may provide increased efficiency, dependability, and flexibility in various applications [3]. Multi-robot teams can be used in a variety of situations, including salvaging in hazardous situations and emergencies, product transit, environmental monitoring, exploration of an unknown environment and so on [4]. However, this research focuses on its use in exploration. When compared to a single robot, multiple robot systems are widely known for their synchronization process and improved geographical distribution capability. This coordination tackles the

issue of how teams of autonomous mobile robots can share the same workspace while avoiding interference, colliding with static obstacles, and/or reaching group motion goals [5]. A mobile robot is a robot that can travel from one location to another by employing several wheels. The robot requires a control system to alter the movements of each wheel so that it can reach the required position. While numerous trajectory tracking control methods can be used to monitor the mobile robot, the key goal is to operate the system inexpensively and effectively without losing the controller's robustness and reliability. In this context, many approaches have been presented to handle path-tracking issues and ensure that mobile robots follow the predetermined path without slipping. Particularly, the majority of articles used a nonlinear back-stepping technique to solve the problem of mobile robot motion control and stability of a non-holonomic dynamic model of a mobile robot [6]. However, the limitation is that the control parameters of the back-stepping controller are fixed and not updated by an on-line optimization method. In 1998, Fierro and Lewis [7] used a multi-layer feedforward neural network to construct a neural network-based model that combined the torque controller with the back-stepping tracking technique, allowing the neural network to learn the dynamics of the mobile robot online. Nonetheless, the drawback of the back-stepping controller is that the control gains are selected by a try-and-error method and the utilized learning algorithms are extremely complicated and computationally expensive [7]. In addition, although the authors in [8] presented a back-stepping control methodology, the design procedure and the resulting controller structure are extremely complicated and used off-line tuning control parameters. Furthermore, a wheeled mobile robot was equipped with a feedforward neural controller and a feedback kinematics controller to track various types of desired trajectories in [9] but the problem of the error propagation mechanism is still considerable and was not eliminated because the parameters of the feedback controller are fixed. On the other hand, the researchers in [10] reviewed the path tracking control method based on model predictive control (MPC) and found that the existing MPC-based path tracking control methods can be divided into four types: linear model predictive control (LMPC), linear error model predictive control (LEMPC), nonlinear model predictive control (NMPC), and nonlinear error model predictive control (NEMPC). By comparing the four types, it was found that the real-time performance of LMPC and LEMPC is good, but the problem of (LMPC) is that it is less robust to reference paths and there are small positioning errors. In this regard, the NMPC performs well when the reference velocity is high and the radius of the reference path is small. It is also robust to positioning errors. However, the real-time performance of the NMPC is slightly worse. Moreover, in [11] explained a cognition path planning algorithm based on the particle swarm optimization (PSO) algorithm that generated an optimal path with free navigation and designed traditional PID controller based on modified Elman neural network MIMO-PID-MENN with PSO control gain algorithm for a motion the non-holonomic wheeled mobile robot. The limitation of the MIMO-PID-MENN was the minimum size of the neural network with the minimum learning set of the desired path, which led to generating a tracking error for the mobile robot during following the reference path.

Specifically, this work's problem definition is separated into two parts: the initial part of our task will be to develop an optimal or near-optimal desired path with the achievement of the two requirements; the shortest path with collision avoidance for three mobile robots working in the same environment. The second part of our work is to create a motion controller for each mobile robot trajectory tracking to guarantee that the mobile robots move on the predefined path without any slipping and with minimum position and orientation tracking errors. The goal of this study is to solve the problem statement by i) using the proposed hybrid method called quarter orbits particle swarm optimization (QOPSO) to generate an optimal or near-optimal smooth desired path for each mobile robot with the shortest path and collision avoidance with the static obstacle or with other mobile robots working in the same environment, ii) generating a smooth and best values of two-wheel velocities' control actions performed by numerical simulation using the suggested inverse differential kinematic neural network trajectory tracking (DIKNNTT) controller based on the modified Elman recurrent neural network control technique.

This paper is organized as follows: Section 2 presents the non-holonomic wheeled mobile robot system. Section 3 discusses the proposed methodology, and section 4 describes the simulation result, while section 5 shows the conclusion.

2. Non-holonomic wheeled mobile robot model

Drive systems can be classified based on how the robot moves into the holonomic and non-holonomic drives [12]. Holonomic drive and non-holonomic

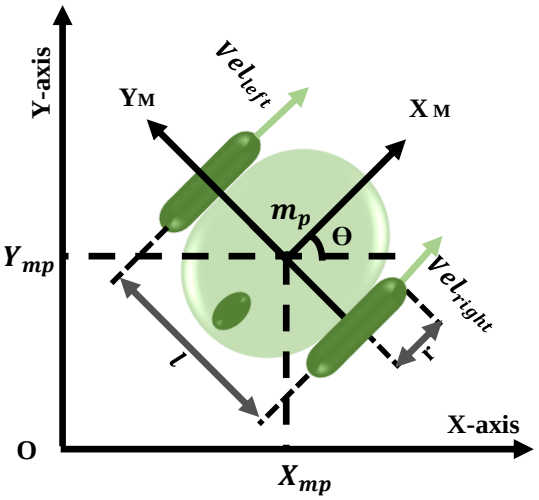


Figure. 1 A non-holonomic wheeled mobile robot

Table.1 The equations' parameters

Parameter	Definition
Vel_{left}	The left wheel velocity
Vel_{right}	The right wheel velocity
l	The distance between the two wheels
T	The sampling time of the mathematical calculation

drive are terms used to describe the relationship between a robot's controlled degree of freedom and its overall degrees of freedom [13]. In particular, the degree of freedom that can be controlled is specified. A robot with a holonomic drive has controllable degrees of freedom equal to its total degrees of freedom. On the other hand, a non-holonomic drive occurs when the robot's controlled degrees of freedom are less than its total degrees of freedom. Fig. 1 demonstrates a non-holonomic wheeled mobile robot system. It contains two driving wheels positioned within the same axis and one caster wheel at the front or at the back of the platform and this wheel is used for the stability of the mobile robot [14]. For mobility and platform steering, the wheeled robot's right and left wheels are controlled by two independent analog direct current (DC) motors. The center mass of the mobile robot is located at the point (m_p), and the two drive wheels are linked to the axis center [10].

The mobile robot model is a multi-input multi-output system. It has two input states (left and right wheels' velocities), and three output states based on its position in the global coordinate frame [O, X-axis, and Y-axis]. The pose surface is X_{mp} and Ym_p representing the coordinates of the point m_p. The kinematics equation of the mobile robot platform has a highly nonlinear state time-variant output, and it also has an under-actuated model. As a result, these three generalized coordinates can be used to define the mobile robot's configuration. More specifically, the computer simulation equations are as follows [15]:

$$X_{mp}(j) = \left[\frac{1}{2}\left(V_{left} + V_{right}\right) \times cos\left(\theta(j)\right) \times T\right] + X_{mp}(j-1) \quad (1)$$

$$Y_{mp}(j) = \left[\frac{1}{2}\left(V_{left} + V_{right}\right) \times sin\left(\theta(j)\right) \times T\right] + Y_{mp}(j-1) \quad (2)$$

$$\theta(j) = \left[\frac{1}{l}\left(Vel_{left} + Vel_{right}\right)T\right] + \theta(j-1) \quad (3)$$

3. Methodology of path planning and control strategy design

The proposed methodology of this work consists of two steps. The first step is the path planning of the multi-mobile robot system based on a hybrid method called the quarter orbits particle swarm optimization algorithm (QOPSO) to find the best desired path from the starting point to the target point, while the second step is the motion control for the same multi-mobile robot system based on the inverse differential kinematic trajectory tracking controller in order to follow the desired path.

3.1 Path planning methodology

The path planning algorithm is required to move the mobile robot from the starting point to the target point taking many objectives into consideration. This work covers two aspects of the path planning problem: the first one is avoiding collisions with obstacles or with other mobile robots, and the second one is determining the shortest path for a mobile robot to achieve its destination in a complex environment. To solve these two problems, a hybrid method called the quarter orbits particle swarm optimization (QOPSO) algorithm [16] will be used, and this algorithm was developed by combining the quarter orbits algorithm with the particle swarm optimization method.

3.1.1. Quarter orbits algorithm

The quarter orbits techniques are derived from the cell decomposition algorithm, with certain modifications to the path search procedure and the cell form. The quarter orbits method divides the robot's static environment into quarter orbits cells, and the navigation process is represented in Fig. 2. The cell decomposition method is improved by the

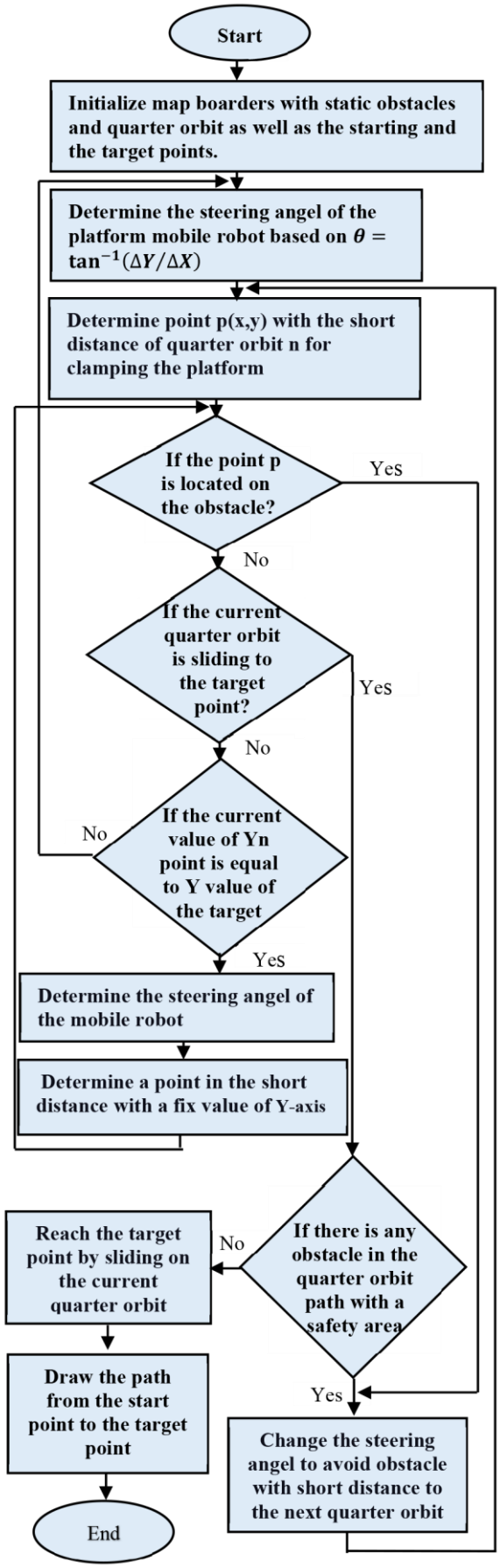


Figure. 2 The quarter orbits algorithm flowchart [16]

quarter orbits algorithm. By driving the robot toward the goal using quarter orbits cells instead of grid cells and vertical lines cells, it decreases the path length required to reach the objective. When a collision-free path is available, this procedure creates the path. Using this strategy, however, does not guarantee that the shortest distance path will be found [16].

3.1.2. Particle swarm optimization algorithm

Kennedy and Eberhart introduced the PSO in 1995 as an evolutionary computation technique based on the behaviour of swarming animals such as birds and fish [17]. Specifically, the PSO tries to mimic the behaviour of a social animal, but it does not require a group leader to accomplish its mission. When a flock of birds is on the hunt for food, they don't need leaders; instead, they just follow one of the individuals which is the closest to the meal. As a result of effective communication with the other particles, the flock of birds achieves its targeted goal. The pseudocode of the PSO algorithm is represented in Fig. 3, and the update functions of the velocity and position vectors at the N^{th} iteration [18] can be represented in Eqs. (4) and (5) whose parameters are represented in Table 2.

Table 2. The parameters' definition of PSO

Parameter	Definition
$V_j(N)$	j^{th} particle's velocity at iteration N
$X_j(N)$	j^{th} position vectors at iteration N
$P_{jbest}(N)$	The best fitness values for the j^{th} particle
$G_{best}(N)$	The best global fitness value for the whole swarm
w	Inertia weight of the velocity (0.751)
c_1 and c_2	The acceleration coefficients (1.75, 1.75)
r_1 and r_2	Random numbers with a uniform distribution of the range [0, 1].

Step 1: Determine the maximum number of iterations.
Step 2: Initialize each particle.
Step 3: For each particle**,** check the fitness value, if it is greater than the best fitness value (P_{jbest}), then set the current value as (new P_{jbest})
Step 4: For each particle:

- Find the particle in the particle neighbourhood with the best global fitness (G_{best}).
- Calculate the particle's velocity $V_j(N)$, according to Equation (4).
- Apply the new value of the velocity.
- Calculate the new particle position $X_j(N)$, according to Equation (5).
- Apply the new value of the position.

Figure. 3 The pseudocode of the PSO method

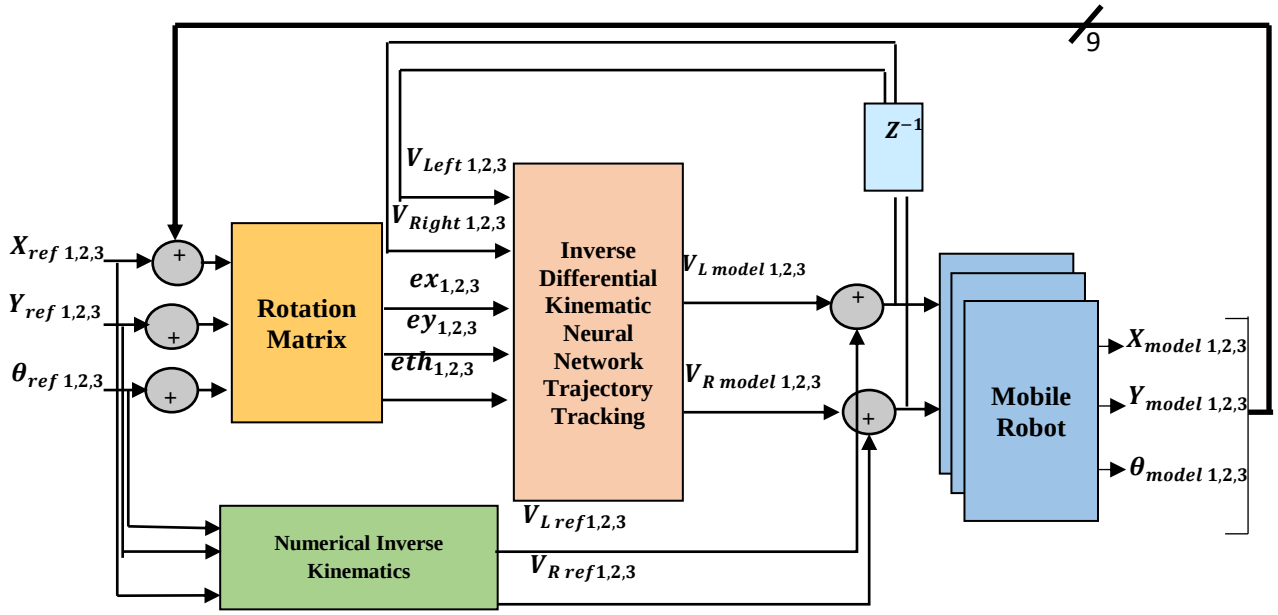


Figure. 4 The main structure of the multi-mobile robots control strategy

$$V_j(N+1) = wV_j(N) + c_1 r_1 \left(P_{jbest}(N) - X_j(N)\right) + c_2 r_2 \left(G_{best}(N) - X_j(N)\right) \quad (4)$$

$$X_j(N+1) = X_j(N) + V_j(N+1) \quad (5)$$

While the PSO algorithm can solve path planning problems effectively and produce a smooth path, it can quickly fall into local optima in many optimization problems. Furthermore, in a complex environment, this algorithm cannot guarantee to provide the best solution.

3.1.3. Hybrid quarter orbits particle swarm optimization (QOPSO) algorithm

To generate the shortest path with collision avoidance, this hybrid algorithm combines the advantages of both the quarter orbits and the PSO algorithms [16]. By driving the mobile robot toward the target and moving it from orbit to orbit with obstacle avoidance until it reaches the destination, the quarter orbits method ensures that a path from the start point to the target point is found. The PSO method will next be used to produce the smoothest and shortest path using the boundaries of the generated path [16].

3.2 Control strategy design

Because the mobile robot platform has time-variant output states, a highly nonlinear kinematics model, and an under-actuated system, the proposed control strategy in this work is to solve the problem of designing a motion controller of trajectory tracking for a multi-mobile robots system. As a result, the suggested controller can generate the optimal left and right wheels' velocities precisely and fast to track the desired routes equations with a minimal tracking position error and without oscillation.

The main structure of the trajectory tracking control strategy for multi-mobile robots is shown in Fig. 4.

The proposed trajectory-tracking controller for a mobile robot is learned using a neural network based on a modified Elman recurrent neural network (MERNN) structure. The MERNN is a partial neural network architecture first developed for speech processing [19].

In this regard, Elman neural networks (ENNs) are a special class of neural networks (NNs) that are made up of a large number of neuron cell models that follow specific rules. Specifically, the NN is a mathematical model that can process data concurrently, with an associative memory function, high level of fault tolerance, and ability to adapt [20]. The capabilities of the modified Elman neural network structure in the proposed controller give the new controller structure many powerful features such as high adaptation performance, fast learning, high order control performance (due to the context units, which remember the previous activations of the hidden units), no output oscillation, good dynamic characteristic, and strong robustness performance (due to the self-connection in the context units, which increase the order of the hidden units). The structure of the MERNN consists of four layers:

- The input layer: It passes the data without transformation, so it acts as a buffer.
- The hidden layer: It is an active layer that has non-linear activation functions.
- The context layer: It has the same behaviour as that of the input layer without any activation function and it is used as memory for increasing the speed of learning. The calculated value of the context node is the same value of the hidden output node simply shifted via recurrent connections with a unit delay. The number of the context layer's nodes is equal to the number of the hidden layer's nodes.
- The output layer: Each node of the output layer is described by a linear activation function.

Fig. 5 shows the block diagram of the proposed controller for three mobile robots working in the same environment. The inputs to this controller are the feedback of the mobile robot's left and right velocities$VR_{MR}(k)$ and $VL_{MR}(k)$ of the three mobile robots with configuration errors $ex_{MR}(k)$ and $ey_{MR}(k)$.

The $e\theta_{MR}(k)$ of the three mobile robots can be calculated using Eq. 6, as follows [2]:

$$\begin{bmatrix} ex_{MR}(k) \\ ey_{MR}(k) \\ e\theta_{MR}(k) \end{bmatrix} = \begin{bmatrix} \cos\theta_{MR} & \sin\theta_{MR} & 0 \\ -\sin\theta_{MR} & \cos\theta_{MR} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{ref} - X_{model} \\ Y_{ref} - Y_{model} \\ \theta_{ref} - \theta_{model} \end{bmatrix} \quad (6)$$

Where X_{ref} , $Y_{ref,}$ and θ_{ref} are the reference position and orientation of the mobile robot.

The first step is to calculate the weighted sum net_j of the inputs as given in Eq. (7).

$$net_j = \sum_{i=1}^{I} UH_{ij} \times D_i + \sum_{z=1}^{Z} UC_{jz} \times h_z^o \quad (7)$$

Where UH_{ij} denotes the weight matrix of the hidden layer, D_i denotes the inputs of the controller, UC_{jz} denotes the weight matrix of the context layer, h_z^o represents the output of the context layer, and *I*, *Z*, and *J* represent the number of nodes in the input layer, the context layer, and the hidden layer, respectively.

The output of the context layer is given by the following Eq. (8):

$$h_z^o(k) = \alpha h_z^o(k-1) + \beta h_j(k-1) \quad (8)$$

Where α represents the self-connection feedback gain, which is selected randomly between (0-1) and β represents the connection weight from the hidden unit to the context unit (0-1).

In addition, Eq. (9) represents a sigmoid activation function that is used in the hidden nodes. Therefore, the output of the neuron h_j of the hidden layer can be expressed as given in Eq. (10).

$$H(net_j) = \frac{2}{1+e^{-net_j}} - 1 \quad (9)$$

$$h_j = H(net_j) \quad (10)$$

In the output layer, the fifteen linear neurons are used to calculate the weighted sum $neto_n$ of their inputs, as expressed in Eq. (11).

$$neto_n = \sum_{j=1}^{J} VO_{jn} \times h_j \quad (11)$$

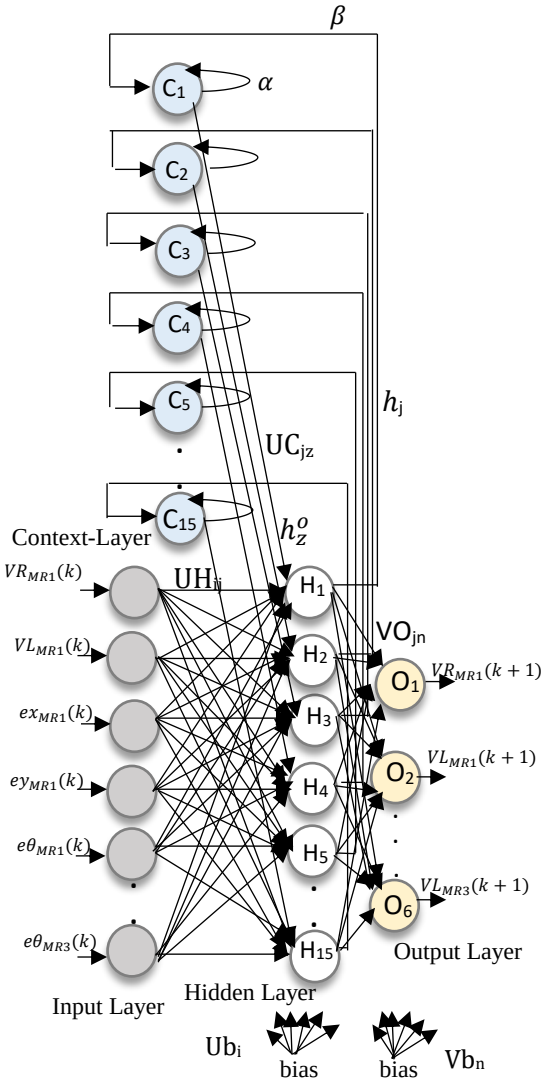


Figure. 5 The block diagram of the proposed neural network controller

Where VO_{jn} denotes the weight matrix of the output layer.

The six linear function neurons in the output layer are used to find the output of the proposed neural network controller representing the six velocities of six wheels of three mobile robots.

To perform the off-line learning of the proposed neural network controller using the back propagation algorithm for adjusting the weights of the modified Elman recurrent neural network, the first stage is to divide the data set into two parts. The first part is called the learning set, which represents half of the data set, while the other half is the testing set. The cost function (mean square error) is chosen as the criterion for estimating the control model performance for each mobile robot, as shown in Eq. (12), which determines whether the neural network outputs (left and right wheels' velocities) are modelled equal to the reference velocity or not.

$$MSE=\frac{1}{M}\sum_{1}^{M}\sqrt{(V_{L\,ref}-V_{L\,model})^2+(V_{R\,ref}-V_{R\,model})^2} \quad (12)$$

Where M is the total number of patterns in the learning set.

Then the total cost function is proposed as in Eq. (13) for three mobile robots. This cost function updates the proposed neural network controller weights as well as expands the number of epochs.

$$MSE_{Total} = MSE_{MR1} + MSE_{MR2} + MSE_{MR3} \quad (13)$$

After training the proposed controller, it will be able to generate left and right wheels' velocities for each mobile robot. The output of the proposed controller is the control action modelled velocities (right and left velocities) of the three mobile robots. To ensure that all the three mobile robots follow their desired paths correctly, a testing set will be used to show the difference between the reference velocities that can be calculated from the numerical inverse kinematic and the left and right wheels' velocities for the mobile robot in order to prove that the neural network is excellently learned.

4. Simulation result

For an effective navigation process of the three mobile robots working on [700×700] cm workspace filled with static obstacles, as shown in Fig. (6), the first step is to generate the desired path that has the shortest distance and collision avoidance with the obstacles for each mobile robot. The second step is to generate the appropriate control action for each of the kinematic mobile robot models, which is an under-actuated system with two inputs (left and right wheels' velocities) and three states' outputs (position (x and y), and orientation) with time-variant and highly

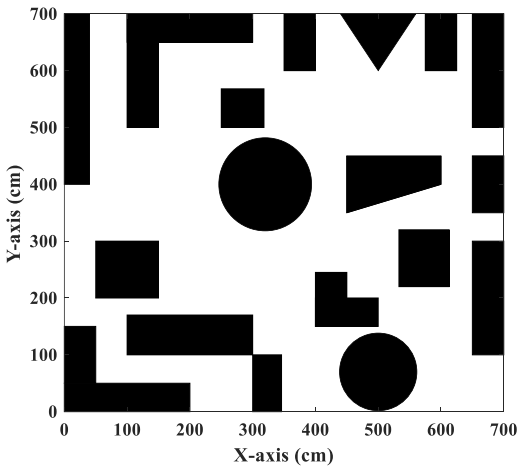
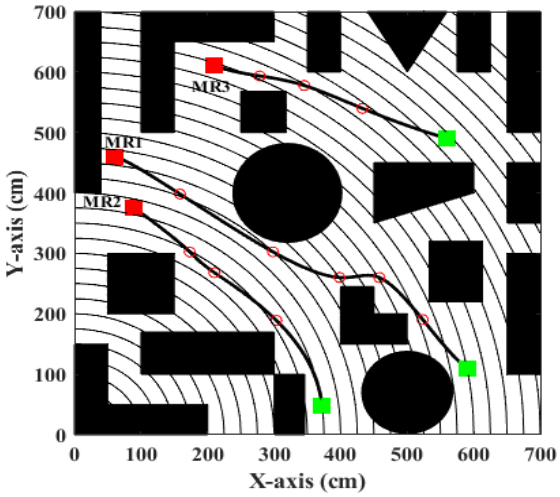
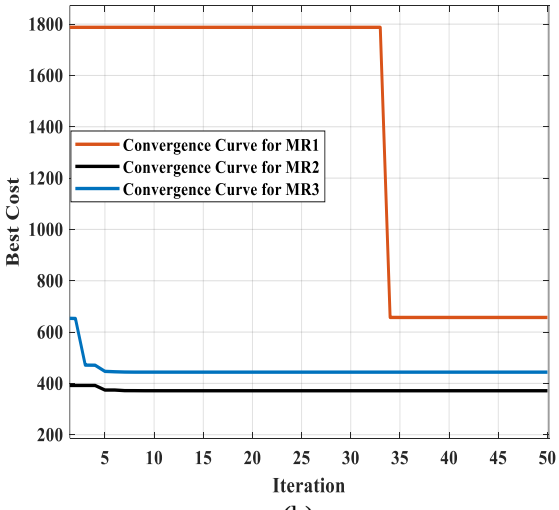


Figure. 6 The suggested workspace



(a)



(b)

Figure. 7 The simulation result using the QOPS algorithm (a) path planning and (b) the best cost function

nonlinear behaviour.

The MATLAB 2021a package with computer hardware specifications of Intel Core i5-1035G7 with 8.00 GB of RAM and CPU of 1.20GHz were used. To satisfy the requirement of the path planning, a hybrid QOPSO algorithm was used [15]. The first mobile robot (MR1) was transferred from a source (50, 460) cm to the destination (590, 110) cm, the second mobile robot (MR2) was transferred from a source (90, 375) cm to the destination (372, 473) cm, and the third mobile robot (MR3) was transferred from a source (210, 610) cm to the destination (560, 490) cm. The mobile robot platform used in this article has a wheels' radius of 0.075 m, and the distance between the wheels is 0.2m, with a 0.1 sampling time.

The QOPSO algorithm was used to find the shortest distance path for each mobile robot in the suggested environment using a maximum of 50 iterations, as shown in Fig. 7.

The shortest path for each mobile robot as well as the number of iterations needed to find this solution are presented in Table 3.

The optimal reference paths for the three mobile robots after applying the QOPSO algorithm are represented in the reference paths of Eqs. (14), (15), and (16) in order to generate the learning and testing sets for the input $D_i$to the proposed neural network controller, as shown in Fig. 5.

$$y_1(x_1) = 7.371e-13x_1^6 - 1.448e-9x_1^5 + 1.087e-6x_1^4 - 0.0003921x_1^3 + 0.07047x_1^2 - 6.52x_1 + 676 \quad (14)$$

$$y_2(x_2) = -1.373e-9x_2^5 + 1.283e-6x_2^4 - 0.0004499x_2^3 + 0.07397x_2^2 - 6.569x_2 + 621 \quad (15)$$

$$y_3(x_3) = 2.637e-6x_3^3 - 0.00352x_3^2 + 1.127x_3 + 497.6 \quad (16)$$

In order to train the proposed neural network controller shown in Fig. 5 and to behave as an inverse differential kinematic trajectory tracking controller for each mobile robot, Fig. (8) illustrates the response of the controller's off-line learning performance index for the three mobile robots that have six wheels' velocities depending on the learning data set "optimal reference paths for the three mobile robots", as in Eqs. (14), (15), and (16) as well as the numerical inverse kinematic" during 1000 epochs with a maximum size of the dataset (learning and testing sets) equals 530 samples for (MR1), 282 samples for (MR2), and 350 samples for (MR3). The mean square error reaches a small value of less than 0.0104 based on Eq. (13) without any learning problems, such as over-learning or overfitting.

Table 3. Three mobile robots' path lengths and the iterations required

Mobile robot	Path length	No. of iterations
MR1	656.99 cm	34
MR2	443.86 cm	19
MR3	371.36 cm	16

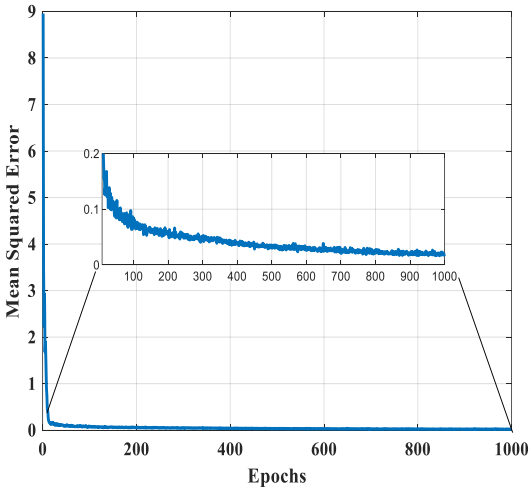


Figure. 8 Performance index for the proposed controller in the learning process

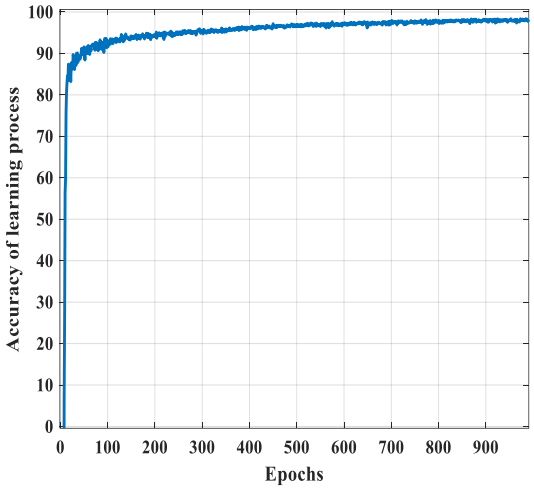
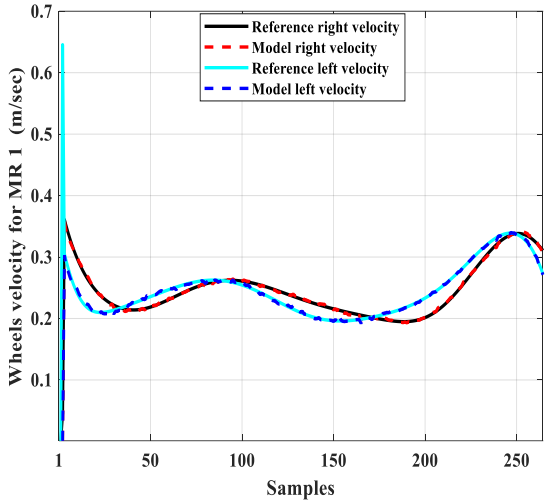


Figure. 9 The accuracy of the proposed controller in the learning process

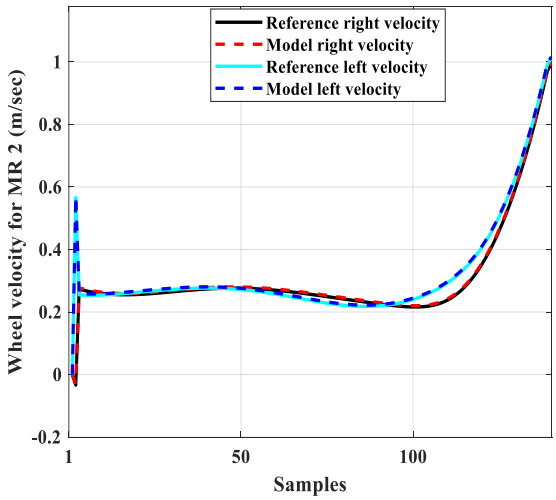
Fig. 9 shows the learning accuracy of the proposed neural network controller, which reaches over 98.975 percent at 1000 epochs.

After 1000 epochs of the off-line back propagation algorithm, the proposed controller was able to generate left and right wheels' velocities with the same reference velocity for the three mobile robots, as shown in Fig. (10) a, b, and c.

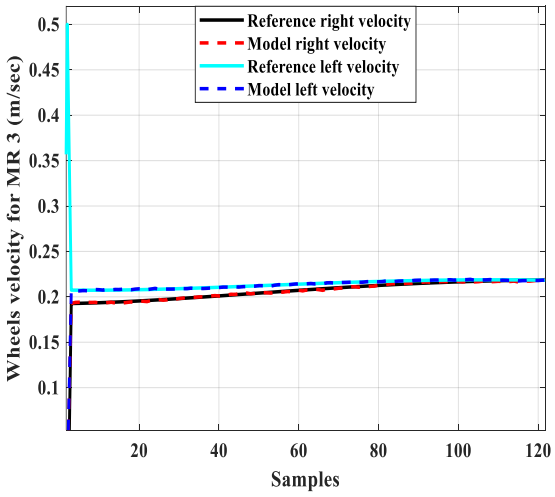
Then, to verify that the proposed neural network controller does not encounter the over-learning



(a)

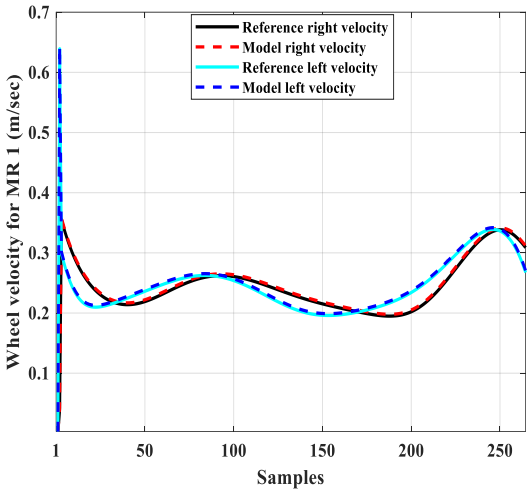


(b)

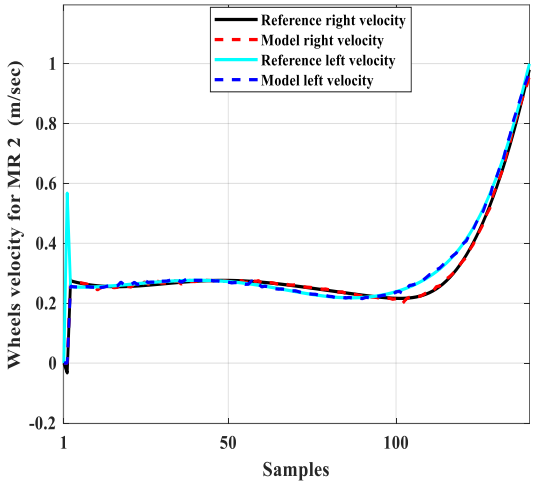


(c)

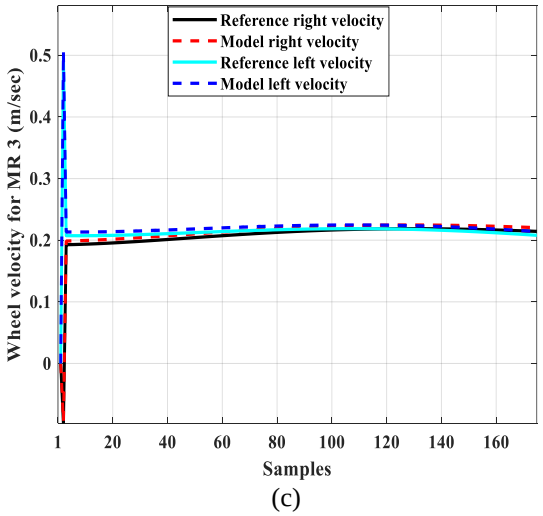
Figure. 10 The result of the learning process (a) wheels' velocities for MR1, (b) wheels' velocities for MR2, and (c) wheels' velocities for MR3



(a)



(b)



(c)

Figure. 11 The result of the testing process (a) wheels' velocities for MR1, (b) wheels' velocities for MR2, and (c) wheels' velocities for MR3

problem, the testing set is applied to the proposed controller model to generate the left and right wheels' velocities for the three mobile robots without the

over-learning problem, as shown in Fig. (11) a, b, and c. This result indicates that the proposed neural network controller has the capability to learn all the paths of the multi-mobile robot.

These responses of the proposed controller outputs' for the left and right wheels' velocities have very small errors compared to the reference velocities, keeping in mind that the learning set has a rich input signal to excite all regions of the proposed controller model without encountering the over-learning problem. Hence, the proposed controller is prepared to track various types of desired paths for three mobile robots. To show the effectiveness of the proposed inverse differential neural network controller for trajectory tracking for each mobile robot, Fig. 12 demonstrates a 2D simulation of the desired paths based on Eqs. 14, 15, and 16 and the output of the three pose kinematics mobile robots model. The actual output of the three mobile robots is

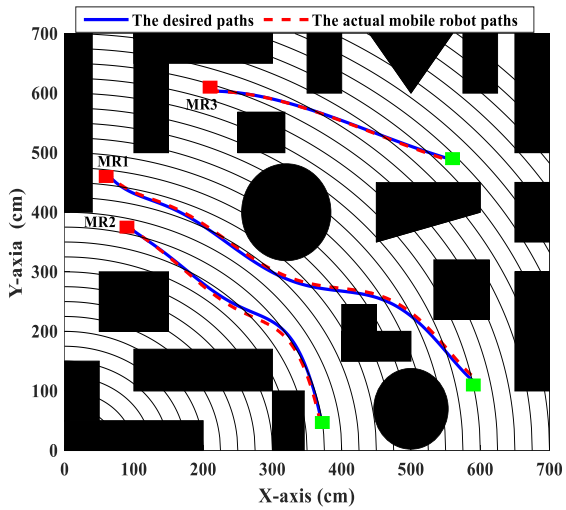


Figure. 12 The 2D simulation result of the actual paths and the desired paths for the three mobile robots

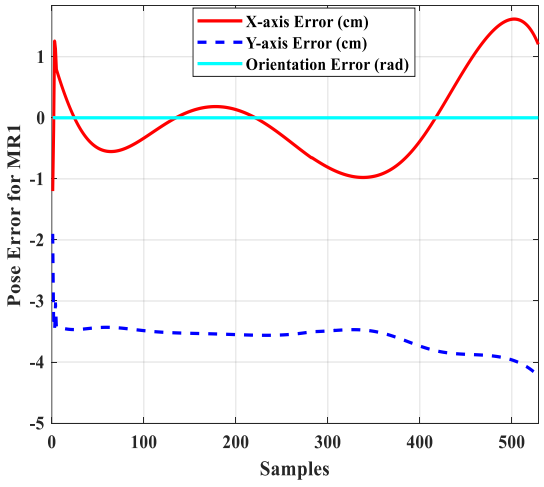


Figure. 13 The tracking error for MR1

fast and without any oscillation during 530 samples (MR1 uses all the numbers of the samples, MR2 uses only 282 samples, and MR3 uses 350 samples) for tracking the desired paths with minimum trajectory-tracking errors in the X-position and the Y-position.

In addition, the orientation errors of the three mobile robots are shown in Figs. 13, 14, and 15 that have very small error in each mobile robot.

The small values of the tracking error in the X-axis and the Y-axis with the orientation error for the three mobile robots are presented in Table 4.

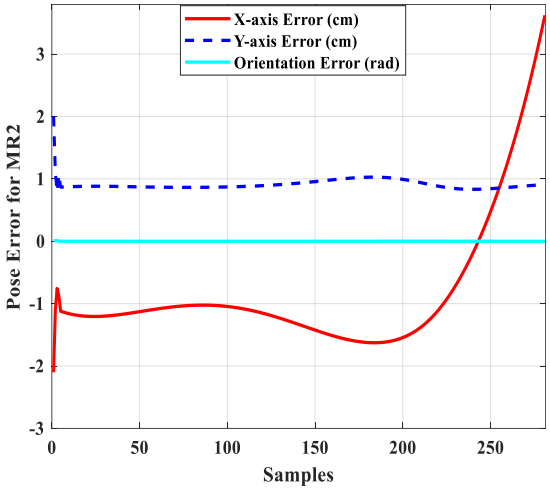


Figure. 14 The tracking error for MR2

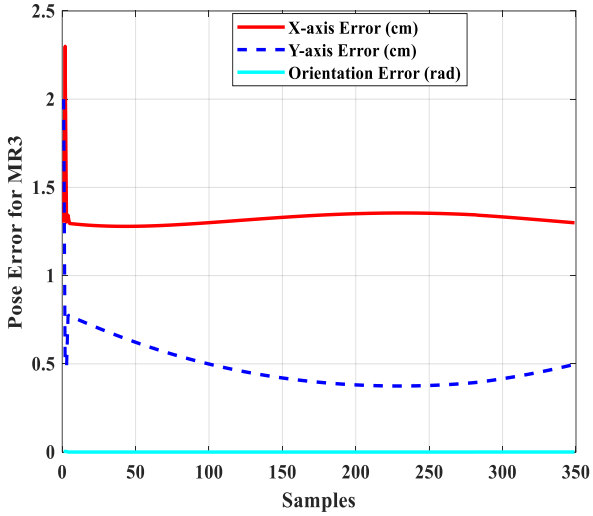
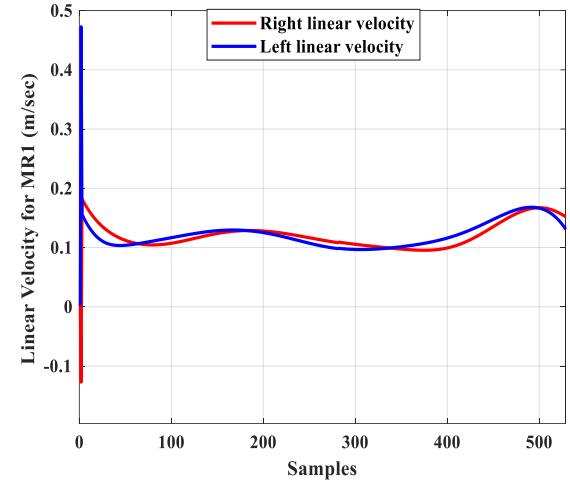


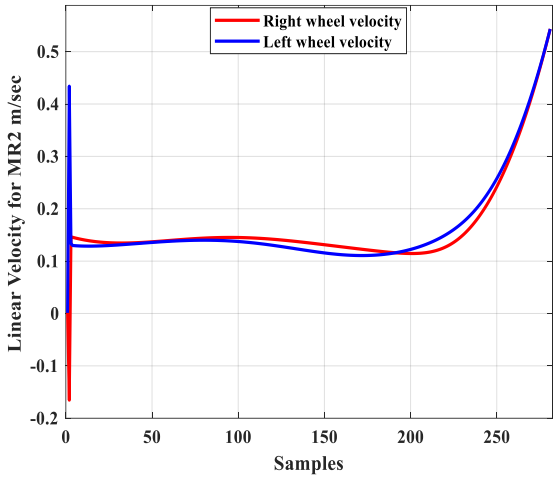
Figure. 15 The tracking error for MR3

Table 4. The tracking error for the three mobile robots

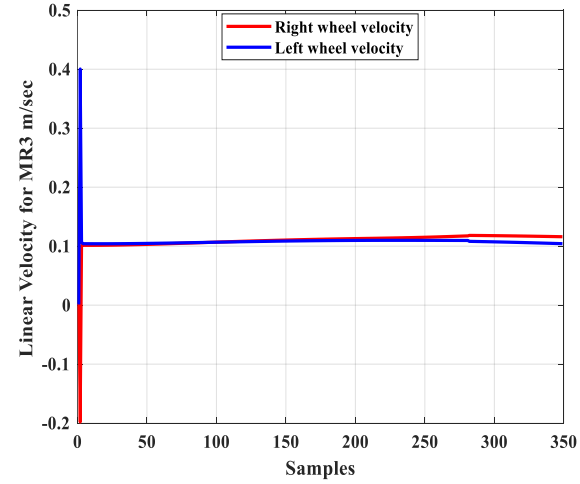
Mobile Robot	Pose Error		
	X-axis	Y-axis	Orientation
MR1	1.72 cm	4.201 cm	0.028 rad
MR2	3.6 cm	2 cm	0.093 rad
MR3	2.3 cm	2 cm	0.034 rad



(a)



(b)



(c)

Figure. 16 Linear velocity for the three mobile robots (a) Linear velocity for MR1, (b) Linear velocity for MR2, (c) Linear velocity for MR3

The proposed controller's output response is shown in Fig. 16, which shows the smooth and fast control actions' responses of the three mobile robots' left and right wheels' linear velocities. These actions do not exceed 1 m/sec and do not have a saturation state. Therefore, they successfully led to tracking the desired paths for each mobile robot.

To confirm the effectiveness of this proposed work in terms of trajectory tracking for the mobile robot, we compared the numerical simulation results of the proposed controller with those of other types of controllers based on the maximum enhancement in the tracking error of the mobile robot in the X-axis position and the Y-axis position. Firstly, the proposed methodology was compared with the research work in [22], which uses a convolutional neural network trajectory tracking (CNNTT) controller to control the nonlinear kinematics mobile robot system. This research produces a maximum error in the X-axis position equals 4 cm and it also produces a maximum error in the Y-axis position equals 2.5 cm. The researchers in [22] used a static environment with a [500 ×500] cm workspace. The optimal reference path for the mobile robot after applying the QOPSO algorithm is shown in Fig. 17 and is represented in the reference path of Eq. (17).

$$y(x) = 0.001608 \times x^2 - 0.8315 \times x + 280.9 \quad (17)$$

Then we applied the generated reference path equation on the suggested controller represented in Fig. 4 by stopping the two other robots.

The difference between the desired path and the actual path generated from the output of the kinematics mobile robot model is represented in Fig. 18. The actual output of the mobile robot is fast and without any oscillation during 350 samples.

Fig.19 represents the maximum improvement in error in the x-position, which is 0.98 cm and the maximum enhancement in error in the y-position, which is 1.97 cm.

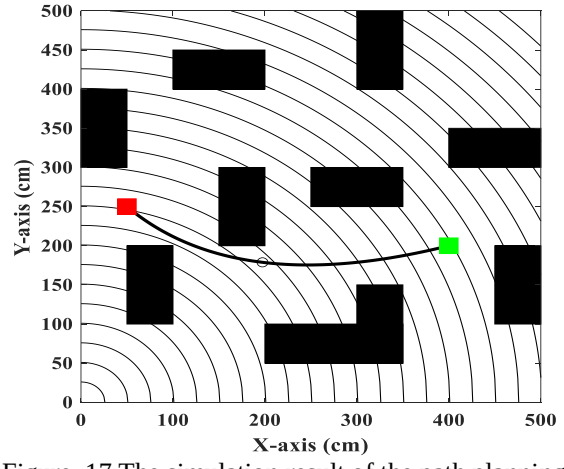


Figure. 17 The simulation result of the path planning based on the QOPSO algorithm

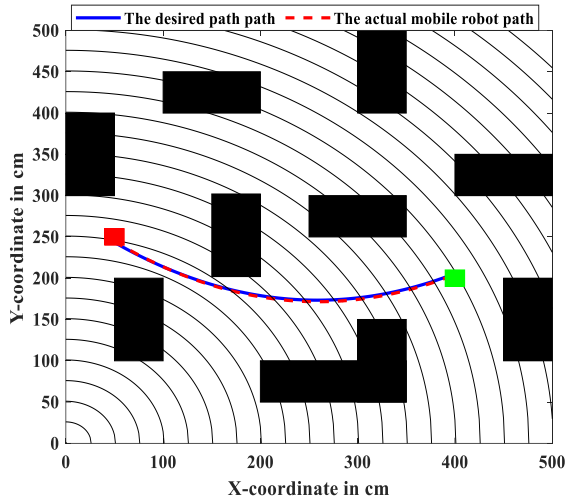
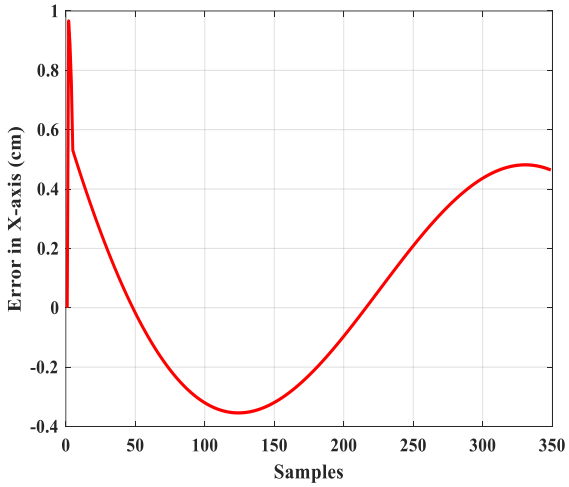
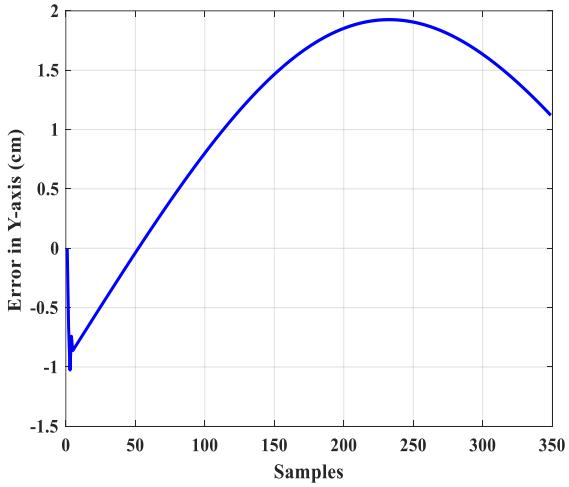


Figure. 18 The 2D simulation result of the actual path and the desired path



(a)



(b)

Figure. 19 The tracking error for the mobile robot (a) Error in the X-axis, (b) Error in the Y-axis

The simulation results in this case indicate that the proposed controller produces a smaller tracking error compared to the convolutional neural network trajectory tracking (CNNTT) controller [22], as shown in Table 5.

Table 5. Comparison of simulation results

Tracking error in axes	**Convolutional Neural Network Trajectory Tracking CNNTT [22]**	**The proposed controller**	**The tracking error enhancement ×(100%)**
The error in the X-position	4cm	0.98cm	$\frac{4-0.98}{4}$ =75.5%
The error in the Y-position	2.5cm	1.97 cm	$\frac{2.5-1.97}{2.5}$ 21.2%

This superiority is achieved because the proposed inverse differential neural network controller consists of [15:15:15:6] nodes, including fifteen nodes for the input layer, fifteen nodes for the hidden layer with a nonlinear activation function, fifteen nodes for the context layer, and six nodes for the output layer with a linear activation function. Moreover, there are three different learning sets of the reference paths. On the other hand, the CNNTT controller consists of [5:11:2] nodes, including five nodes for the input layer, eleven nodes for the hidden layer, and two nodes for the output layer. In addition, the number of the learning sets in [22] equals 80 samples for one desired path. In particular, these datasets were used in the fully connected layer as only one vector (5×16, 1). The goal was to reduce the search space in [22]. Moreover, the authors used a backpropagation algorithm to train the neural network with a maximum number of epochs of 1000. These reasons may lead to learning impairment in the trajectory-tracking controller due to increasing the tracking error in the x-axis and the y-axis of the mobile robot. On the other hand, in our work, we used three different learning sets, where the first path has 530 samples, the second path has 282 samples, and the third path has 350 samples. This means that the goal is to increase the search space in this work in order to excite all regions of the proposed controller, which did not fall into the overlearning problem.

The difference between the proposed controller and CNNTT [22] is in the number of data sets “search space” and the number of nodes in the neural network, offering the proposed neural network controller excellent learning to generate the best control action to track the desired path and reach the target point without oscillation and with a minimum tracking error rate on the X-axis and the Y-axis compared to the CNNTT controller.

Secondly, we compared the proposed

methodology with the research work in [11] that uses the nonlinear MIMO-PID-MENN controller design for the wheeled mobile robot based on the modified Elman recurrent neural network. This research produces an error in the X-axis position equals 6 cm and an error equals 3.1 cm in the Y-axis position. This research uses a static environment with a [400×400] cm workspace. The optimal reference path for the mobile robot after applying the QOPSO algorithm is shown in Fig. 20 and is represented in the reference path of Eq. (18).

$$y(x) = 2.153e-10 \times x^5 - 2.067e-7 \times x^4 + 7.205e-5 \times x^3 - 0.01232 \times x^2 + 2.098 \times x + 5.381 \quad (18)$$

Then we applied the generated reference path equation on the suggested controller represented in Fig. 4 by stopping the two other robots. The

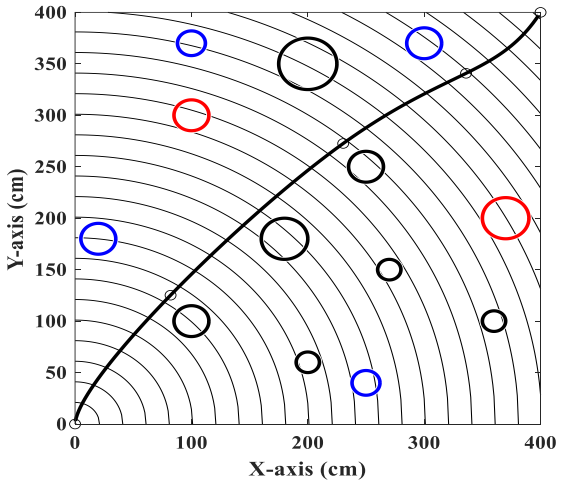


Figure. 20 The simulation result of the path planning based on the QOPSO algorithm

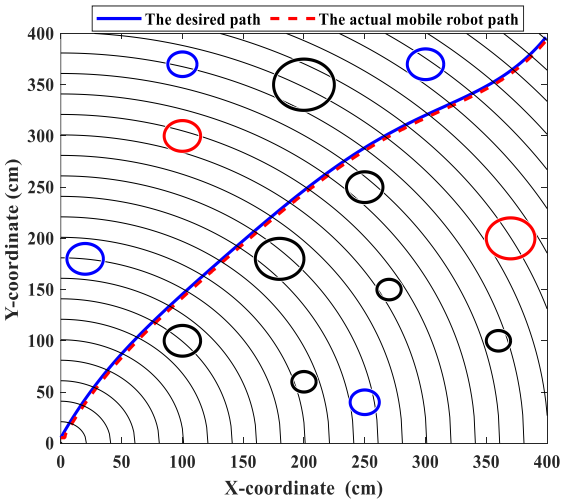
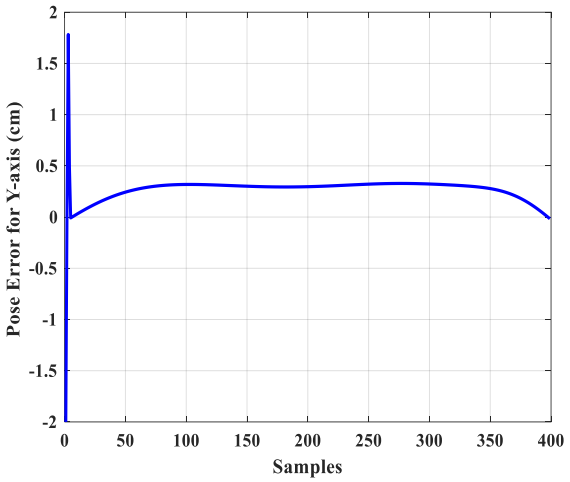
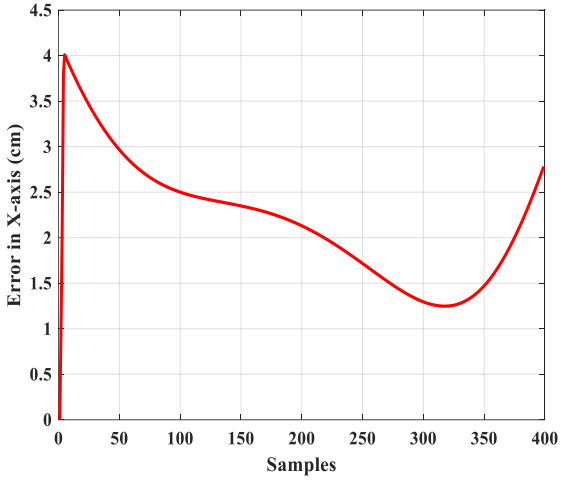


Figure. 21 The 2D simulation result of the actual path and the desired path



(a)



(b)

Figure. 22 The tracking error for the mobile robot. (a) Error in the X-axis, (b) Error in the Y-axis

Table 6. Comparison of simulation results

Tracking error in axes	Nonlinear MIMO-PID-MENN controller[11]	The proposed controller	The tracking error enhancement (100%)
The error in the X-position	6cm	4cm	$\frac{6-4}{6}$ =33.3%
The error in the Y-position	3.1cm	1.84 cm	$\frac{3.1-1.84}{3.1}$ 40.6%

difference between the desired path and the actual path generated from the output of the kinematics mobile robot model is represented in Fig. 21. The actual output of the mobile robot is fast and without any oscillation during 400 samples.

Fig. 22 shows the maximum enhancement in tracking error in the x-position, which is 4 cm, and the maximum improvement in tracking error in the y-position, which is 1.84 cm for the mobile robot.

The simulation results in this case indicate that the proposed controller produces a smaller tracking error compared to the nonlinear MIMO-PID-MENN controller, which is based on the modified Elman recurrent neural network, as shown in Table 6.

Again, this superiority is due to the fact that the proposed neural network controller consists of [15:15:15:6] nodes, while the nonlinear MIMO-PID-MENN controller [11] consists of [3:3:1:2] nodes, three nodes for the input layer, three nodes for the hidden layer, one node for the context layer, and two nodes for the output layer. In addition, the number of the learning sets in [11] equals 400 samples for one desired path.

The difference between the numbers of neurons in the neural network and the size of the data set for learning and testing gives the proposed controller the ability to generate the best velocity control action for the mobile robot to track the desired path and reach the target point without oscillation and with a minimum error rate on the X-axis and the Y-axis compared to the nonlinear MIMO-PID-MENN controller [11].

Finally, all the simulation results indicate that the suggested inverse differential kinematic neural network controller has the ability to generate the best velocity control action that makes the mobile robot track the desired path with minimum tracking errors in the X-axis position and the Y-axis position and successfully reach the target point without oscillation.

5. Conclusions

This paper proposed the design of a trajectory tracking neural network controller based on the modified Elman recurrent neural network for tracking the desired path equations of three mobile robots. These paths were generated using the hybrid quarter orbits particle swarm optimization (QOPSO) as a path-planning algorithm that provides the shortest path with collision avoidance for each mobile robot.

In particular, this research focused on solving two problems of the navigation process. The first part was to create an optimal or near-optimal desired path that meets two requirements: the shortest path with collision avoidance for three mobile robots working in a static environment. This problem has been solved using the proposed QOPSO algorithm. The second problem of our work involves developing a motion controller for each mobile robot trajectory tracking to ensure that each mobile robots move along the predefined path with minimal slipping and minimal position and orientation errors for each mobile robot and this has been done using the proposed neural network controller, which minimizes the tracking errors in the X-axis position and the Y-axis position. In addition, the error is almost zero in the orientation. The proposed controller provides the ideal and smooth values of the left and right wheels' velocities that were learned on each kinematics mobile robot platform model control action. Therefore, each mobile robot was excellent at tracking the desired path and successfully reaches the target point without oscillation. To ensure the effectiveness of the proposed controller, a comparison study has been conducted in terms of the maximum tracking errors in the X-position and the Y-position considering other researchers that use different types of controllers. Specifically, the proposed controller was compared with the convolutional neural network trajectory tracking controller [22]. The comparison results showed that the proposed controller improves the tracking error rate on the X-axis by 75.5 % and 21.2 % on the Y-axis compared to the CNNTT controller. Moreover, the proposed controller was compared with the MIMO-PID-MENN controller and the comparison results showed that the proposed controller enhances the tracking error rate on the X-axis by 33.3 % and by 40.6 % on the Y-axis compared to the MIMO-PID-MENN controller [11].

For future work, we suggest adjusting the proposed controller to work with multiple robots whose paths intersect with each other. In addition, we suggest that the experimental work of the path planning algorithm and the proposed control strategy for the mobile robot system be practically implemented.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Zainab E. Kanoon, Ahmed Sabah Al-Araji and Mohammed Najm Abdullah enhanced and developed a trajectory tracking controller for controlling three mobile robot models. Zainab E. Kanoon described the proposed Modified Elman Recurrent Neural Network controller. Ahmed Sabah Al-Araji explained the kinematics mobile model. In addition, Zainab E. Kanoon and Mohammed Najm Abdullah discussed the simulation results of this work.

References

[1] A. A. A. Rasheed, A. S. A. Araji, and M. N. Abdullah, "Static and Dynamic Path Planning Algorithms Design for a Wheeled Mobile Robot Based on a Hybrid Technique", *International*

Journal of Intelligent Engineering and Systems, Vol. 15, No. 4, pp. 167-181, 2022, doi: 10.22266/ijies2022.0831.16.

[2] N. A. K. Zghair and A. S. A. Araji, "A One Decade Survey of Autonomous Mobile Robot Systems", *International Journal of Electrical and Computer Engineering*, Vol. 11, No. 6, pp. 4891–4906, 2021.

[3] A. P. Widanis, T. Agustinah, and A. Santoso, "Trajectory Tracking of Multi-Robot System Using a Singularity Approach", In: *Proc. of International Seminar on Intelligent Technology and Its Applications*, Surabaya, Indonesia, pp. 266-272, 2020.

[4] Y. Han, D. Li, J. Chen, X. Yang, Y. Hu, and G. Zhang, "A Multi-Robots Task Allocation Algorithm based on Relevance and Ability with Group Collaboration", *International Journal of Intelligent Engineering and Systems*, Vol. 3, No. 2, pp. 33-41, 2010.

[5] A. Ayari and S. Bouamama, "A New Multiple Robot Path Planning Algorithm: Dynamic Distributed Particle Swarm Optimization", *Robotics and Biomimetics*, Vol. 4, No. 1, pp. 1-15, 2017.

[6] H. G. Tanner and K. J. Kyriakopoulos, "Backstepping for Nonsmooth Systems", *Automatica*, Vol. 39, No. 7, pp. 1259-1265, 2003.

[7] R. Fierro and F. L. Lewis, "Control of a Nonholonomic Mobile Robot Using Neural Networks", *IEEE Transactions on Neural Networks*, Vol. 9, No. 4, pp. 589-60, 1998.

[8] Z. P. JIANGdagger and H. Nijmeijer, "Tracking Control of Mobile Robots: A Case Study in Backstepping", *Automatica*, Vol. 33, No. 7, pp. 1393-1399, 1997.

[9] M. Asif, M. J. Khan, M. Safwan, and M. Rehan, "Feedforward and Feedback Kinematic Controllers for Wheeled Mobile Robot Trajectory Tracking", *Journal of Automation and Control Engineering*, Vol. 3, No. 3, pp. 178-182, 2015.

[10] G. Bai, Y. Meng, L. Liu, W. Luo, Q. Gu, and L. Liu, "Review and Comparison of Path Tracking based on Model Predictive Control", *Electronics*, Vol. 8, No. 10, pp. 10-32, 2019.

[11] A. S. A. Araji, A. K. Ahmed, and K. E. Dagher, "A Cognition Path Planning with a Nonlinear Controller Design for Wheeled Mobile Robot based on an Intelligent Algorithm", *Journal of Engineering*, Vol. 25, No. 1, pp. 64-83, 2019.

[12] S. Prongnuch and S. Sitjongsataporn, "Differential Drive Analysis of Spherical Magnetic Robot Using Multi-Single Board Computer", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 264-275, 2021, doi: 10.22266/ijies2021.0831.24.

[13] M. Fuad, T. Agustinah, and D. Purwanto, "Collision Avoidance of Multi-Modal Moving Objects for Mobile Robot Using Hybrid Velocity Obstacles", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 3, pp. 407-421, 2020, doi: 10.22266/ijies2020.0630.37.

[14] A. S. Araji, A. K. Ahmed, and M. K. Hamzah, "Development of a Path Planning Algorithms and Controller Design for Mobile Robot", In: *Proc. of Scientific Conference of Electrical Engineering, Baghdad*, Iraq, pp. 72-77, 2018.

[15] A. S. A. Araji, K. E. Dagher, and B. A. Ibraheem, "An Intelligent Cognitive System Design for Mobile Robot based on Optimization Algorithm", In: *Proc. of The Scientific Conference of Electrical Engineering*, Baghdad, Iraq, pp. 84-89, 2018.

[16] Z. E. Kanoon, A. S. A. Araji, and M. N. Abdullah, "Enhancement of Cell Decomposition Path-Planning Algorithm for Autonomous Mobile Robot Based on an Intelligent Hybrid Optimization Method", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 3, pp. 161-175, 2022, doi: 10.22266/ijies2022.0630.14.

[17] K. Menghour and L. S. Meslati, "Hybrid ACO-PSO based Approaches for Feature Selection", *International Journal of Intelligent Engineering and Systems*, Vol. 9, No. 3, pp. 65-79, 2016.

[18] R. H. A. Rubayi, M. K. Abd, and F. M. F. Flaih, "A New Enhancement on PSO Algorithm for Combined Economic-Emission Load Dispatch Issues", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 1, pp. 77-85, 2020, doi: 10.22266/ijies2020.0229.08.

[19] L. J. Rothkrantz and D. Nollen, "Speech Recognition Using Elman Neural Networks", In: *Proc. of International Workshop on Text, Speech and Dialogue*, Springer, Berlin, Heidelberg, pp. 146-151, 1999.

[20] G. Ren, Y. Cao, S. Wen, T. Huang, and Z. Zeng, "A modified Elman Neural Network with a New Learning Rate Scheme", *Neurocomputing*, Vol. 286, pp. 11-18, 2018.

[21] A. S. A. Araji, M. F. Abbod, and H. S. A. Raweshidy, "Design of an Adaptive Neural Predictive Nonlinear Controller for Nonholonomic Mobile Robot System Based on Posture Identifier in the Presence of Disturbance", *International Journal of Simulation–Systems, Science & Technology*, Vol. 12, No. 3, pp. 17–28, 2012.

[22] O. A. R. A. Wahhab and A. S. A. Araji, "Path Planning and Control Strategy Design for Mobile Robot Based on Hybrid Swarm Optimization Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 565-579, 2021, doi: 10.22266/ijies2021.0630.48.