sciendo

DOI 10.2478/ama-2021-0027

acta mechanica et automatica, vol.15 no.4 (2021)

GENERALISED REGRESSION NEURAL NETWORK (GRNN) ARCHITECTURE-BASED MOTION PLANNING AND CONTROL OF AN E-PUCK ROBOT IN V-REP SOFTWARE PLATFORM

Vikas Singh PANWAR*, Anish PANDEY*, Muhammad Ehtesham HASAN*

*School of Mechanical Engineering, KIIT Deemed to be University, An Institute of Eminence, Patia, Bhubaneswar, PIN - 751024, Odisha, India

vikas09panwar@gmail.com, anish06353@gmail.com, hasan.adab@gmail.com

received 1 June 2020, revised 21 August 2021, accepted 31 August 2021

Abstract: This article focuses on the motion planning and control of an automated differential-driven two-wheeled E-puck robot using Generalized Regression Neural Network (GRNN) architecture in the Virtual Robot Experimentation Platform (V-REP) software platform among scattered obstacles. The main advantage of this GRNN over the feedforward neural network is that it provides accurate results in a short period with minimal error. First, the designed GRNN architecture receives real-time obstacle information from the Infra-Red (IR) sensors of an E-puck robot. According to IR sensor data interpretation, this architecture sends the left and right wheel velocities command to the E-puck robot in the V-REP software platform. In the present study, the GRNN architecture includes the MIMO system, i.e., multiple inputs (IR sensors data) and multiple outputs (left and right wheel velocities). The three-dimensional (3D) motion and orientation results of the GRNN architecture-controlled E-puck robot are carried out in the V-REP software platform among scattered and wall-type obstacles. Further on, compared with the feedforward neural network, the proposed GRNN architecture obtains better navigation path length with minimum error results.

Key words: e-puck robot, generalised regression neural network architecture, virtual robot experimentation platform software, scattered obstacle, Infra-Red sensor

1. INTRODUCTION

Research on wheeled robotics can be categorised into two types: navigation control and stability control. For navigation control, we mainly focus on the kinematic part and control the displacement, velocity and acceleration of a wheeled robot. Similarly, we consider the dynamic part of a wheeled robot and apply force to generate torque to achieve stability control over a robot. The researchers use various soft-computing methods like Fuzzy, Neural Network, Neuro-fuzzy, nature-inspired algorithms, and evolutionary algorithms for navigation and stability control. The meaning of navigation control of a wheeled robot is to search a collision-free optimal path from the starting location to the target among obstacles in the workspace. Obstacle avoidance and wall following problems come under navigation control, and the trajectory tracking problem is solved by using stability control torque equations. The authors Elmi and Efe (2020) implemented a grasshopper algorithm to control the motion of a wheeled robot between static and dynamic hurdles. However, the authors present only computer simulations, and real-time experiments on an experimental wheeled robot have not been found. Long et al. (2020) designed the hybridisation of A* algorithm with a bacterial foraging optimisation algorithm, and present the global path planning for an unmanned surface vehicle in a grid map environment.

Ben Jabeur and Seddik (2020) developed an optimised PID neural network with a hybrid fuzzy controller for trajectory tracking and motion control of a two-wheeled mobile robot in unknown and complex environments. Although the 3D experiments of a two-wheeled mobile robot were not carried out in that work, further research in this direction continues. Another work relevant for mention here would be that by Protik et al. (2019), where the authors apply chemical reaction optimisation algorithms with Euclidean-based fitness functions to determine an optimal motion and orientation control for a wheeled robot among uneven obstacles. In Zhao et al. (2020), the authors designed a genetic algorithm optimised type-2 fuzzy controller and implemented this developed controller to minimise the locomotion of a wheeled robot from a start point to the target between scattered obstacles. Pandey et al. (2019) applied an adaptive neuro-fuzzy inference system to perform an autonomous motion between static and dynamic obstacles for a wheeled robot. Multilayer perceptron feedforward neural network (Singh and Thongam, 2018; Pandey and Parhi, 2016) had been implemented for wheeled robot navigation in the computer simulation environment between static and moving objects. Hadi and Younus (2020) designed the path tracking control architecture for a nonholonomic wheeled robot. Moreover, the authors briefly explain the kinematic and dynamic models of a wheeled robot in MATLAB graphical user interface environments. An extensive review article on the navigation of swarm robotics and their various control methods was reported in the reference (Osaba et al., 2019; Nedjah and Junior, 2019).

Grid-map-based navigation and collision avoidance scheme for a wheeled robot were presented by Tripathy et al. (2021). In their work, different grid size environments are considered to show the motion results among obstacles. However, they did not show any real-time 3D navigation results in that work. Narasimhan and Bettyjane (2020) implemented the fuzzy rule-based architecture to design a local path planner for two co-operating wheeled

robots in an unstructured environment. Wang (2021) used Genetic Algorithm (GA) to minimise the navigation path length of a wheeled robot in MATLAB software-based grid-map environments. As we know, the convergence rate of the GA is high, and that is why it takes more time to reach the goal from the source point. Almeida et al. (2021) study deep neural network-controlled visual landmarks-based motion planning for a differential-driven automated guided vehicle. The authors of Quan et al. (2021) applied harmony search algorithm and Morphin algorithm for a global path planning of an autonomous mobile robot between moving obstacle conditions. However, only two-dimensional (2D) simulation results were presented in that work. Teli and Wani (2021) designed the hybrid controller by taking the fuzzy method with an artificial potential field method and implemented this controller to steer a wheeled robot autonomously between C and H types of complicated obstacle conditions. Proportional–Integral–Derivative (PID) controller has been used in a study (Khan et al., 2021) for the speed control of both DC motors of a wheeled robot during its navigation and obstacle avoidance behaviours.

The application of different soft-computing methods like grasshopper algorithm (Elmi and Efe, 2020), bacterial foraging optimisation algorithm (Long et al., 2020), chemical reaction optimisation algorithm (Protik et al., 2019), genetic algorithm (Zhao et al., 2020), adaptive neuro-fuzzy inference system (Pandey et al., 2019) and feedforward neural network (Ben Jabeur and Seddik, 2020; Singh and Thongam, 2018; Pandey et al., 2019) in a wheeled robot motion planning problem motivates the authors to undertake the present research. After summarising the above-cited elements in the literature, we have observed that most of the researchers (Ben Jabeur and Seddik, 2020; Singh and Thongam, 2018; Pandey and Parhi, 2016; Almeida et al., 2021) have used feedforward or backpropagation neural network to control the motion and orientation of a wheeled robot among obstacles in various scenarios. Moreover, we have found that the GRNN provides accurate results in a short period with minimal error compared to other neural networks like feedforward or backpropagation neural networks (Specht, 1991). This strength of the GRNN attracts the authors to select this method in the current work. Therefore, in this study, the authors choose GRNN architecture to control the motion of an E-puck robot among scattered and wall-type obstacles. Further on, compared with the feedforward neural network (Singh and Thongam, 2018), the proposed GRNN architecture obtains better navigation path length with minimum path error. The contributions of this work are presented as follows: - Section 2 presents the kinematic study of an automated differential-driven two-wheeled E-puck robot. Section 3 provides the design of a GRNN architecture for motion and orientation planning and control of an E-puck robot among scattered and wall-type obstacles in the V-REP software environment. The outcomes of the experiments and comparative analysis are organised and elaborated in Section 4. Finally, the conclusion and future research work of the present work are summarised in Section 5.

2. KINEMATIC STUDY OF AN AUTOMATED DIFFERENTIAL-DRIVEN TWO-WHEELED E-PUCK ROBOT

This section derives the kinematic equations for an automated differential-driven two-wheeled E-puck robot, which control the motion and orientation of an E-puck robot during navigation among scattered obstacle in the V-REP software platform. Fig. 1 illustrates the schematic representation of kinematic parts of an automated differential-driven two-wheeled E-puck robot in the two-dimensional rigid plane. The E-puck robot is a differential-driven two-wheeled mobile robot developed by École Polytechnique Fédérale de Lausanne. The diameter of the E-puck robot is 7 cm, 5 cm height with the wheel diameter of 4 cm; and the total weight of the DDER is 0.16 kg. The E-puck robot can move in the forward and backward directions; and it can turn with a top speed of 0.15 m/s. It includes eight IR sensors with eight LEDs. IR sensors of the E-puck robot can read the obstacles between 0.01 m to 0.06 m range approximately. Fig. 2 shows the top view of the E-puck robot with the distribution of the eight IR sensors from S_0 to S_8. The following IR sensors readings: S_0, S_1, S_6, and S_7 are used during motion and orientation controls in the present study. The front senor obstacle reading (D_f) takes the minimum data of S_0 and S_7. Similarly, the left senor obstacle reading (D_l) is the least value of S_6 and S_7. Next, we use the minimum data of S_0 and S_1 as a right senor obstacle reading (D_r). The data of D_f, D_l, and D_r, along with left and right wheel velocities, are used in the GRNN architecture to control the motion and orientation of an E-puck robot.

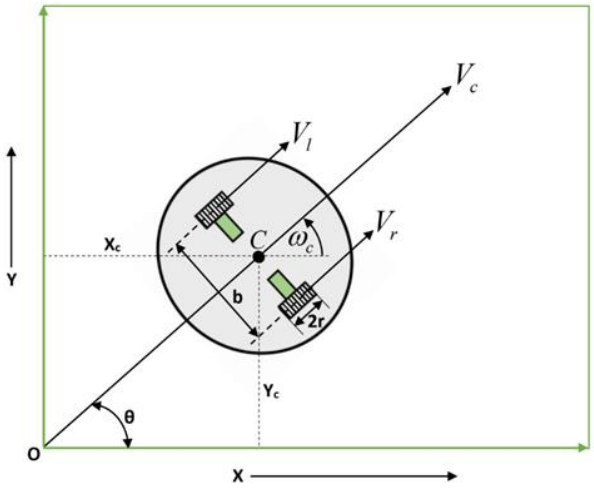

Fig. 1. Schematic representation of kinematic parts of an automated differential-driven two-wheeled E-puck robot in the two-dimensional rigid plane

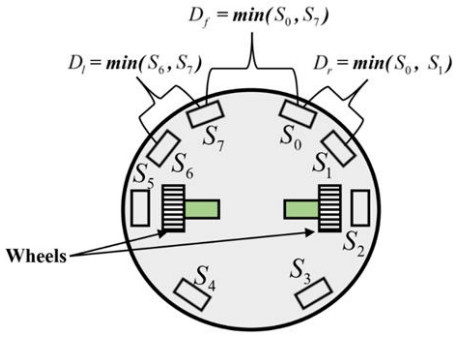

Fig. 2. Top view of an E-puck robot with the distribution of eight IR sensors

Further on, the E-puck robot consists of two independent driving wheels that carry the mechanical chassis of a robot. The two driving wheels attached with the two separate stepper motors drive the E-puck robot. The V_L denotes the left wheel velocity, and

V_R indicates the right wheel velocity in the kinematic equations. The E-puck robot moves in the solid plane, and a rigid chassis makes it. The axes (x_c, y_c) are the current posture of the E-puck robot from the origin O point in the global frame $\{O, X, Y\}$. In Fig. 1, θ denotes the orientation angle of the E-puck robot with respect to an axis (O, X), b is the track width between the left and right wheel drive systems, r is the radius of the driving wheels and C is the center of the mass of E-puck robot system. The following kinematic equations control the velocities and steering angle of the E-puck robot: -

$$V_c = \frac{r}{2} \cdot (\omega_R + \omega_L) = \frac{(V_R + V_L)}{2} \tag{1}$$

$$\omega_c = \dot{\theta} = \frac{r}{2} \cdot (\omega_R - \omega_L) = \frac{(V_R - V_L)}{b} \tag{2}$$

where V_c and ω_c represent the center (mean) linear velocity and center angular velocity of the E-puck robot, respectively, these V_c and ω_c control the motion and orientation of the E-puck robot in the V-REP software platform, respectively. Next, ω_R and ω_L denote the angular velocities of the right and left wheel driving systems, respectively.

Next, the following equations express the velocity (linear and angular) with respect to time (t): -

$$\frac{dx}{dt} = \dot{x}(t) = V_c \cdot \cos\theta = \frac{r}{2} \cdot (\omega_R + \omega_L) \cdot \cos\theta \tag{3}$$

$$\frac{dy}{dt} = \dot{y}(t) = V_c \cdot \sin\theta = \frac{r}{2} \cdot (\omega_R + \omega_L) \cdot \sin\theta \tag{4}$$

$$\frac{d\theta}{dt} = \dot{\theta}(t) = \omega_c = \frac{r}{b} \cdot (\omega_R - \omega_L) \tag{5}$$

3. DESIGN OF A GRNN ARCHITECTURE FOR MOTION AND ORIENTATION PLANNING AND CONTROL OF AN E-PUCK ROBOT AMONG SCATTERED OBSTACLES IN THE V-REP SOFTWARE ENVIRONMENT

This section reveals a brief description of a GRNN architecture that controls the motion and orientation of an E-puck robot among scattered obstacles in the V-REP software platform. GRNN belongs to the family of the Statistical Neural Network (SNN) group, and the rest of this GRNN, the Radial Basis Function Neural Network and Probabilistic Neural Network are also the members of this SNN group. GRNN works based on the regression method, which was developed by Specht in 1991. The main advantage of this GRNN over the feedforward neural network is that it does not need an iterative-based training procedure (Specht, 1991). GRNN makes a linear or non-linear regression function between dependent variables (outputs) and independent variables (inputs) to provide the expected outcome. In the present work, the GRNN architecture takes the IR sensors data information as three inputs (D_f, D_l, and D_r) and wheel velocities (V_L and V_R) as two outputs of an E-puck robot. Tab. 1 reveals the inputs and outputs data set, which are fed into the GRNN architecture to control the motion and orientation of an E-puck robot in the V-REP software platform among scattered obstacles. The ranges of inputs (D_f, D_l, and D_r) have taken from IR sensor range of an E-puck robot, and the ranges of outputs (V_L and V_R) are fixed between 0.063 m/sec to 0.150 m/sec. Fig. 3 shows the basic structure of the MIMO system GRNN architecture, which combines different layers. The GRNN architecture consists of four layers: an input layer, pattern layer, summation layer, and output layer. As shown in Fig. 3, the input layers connect the second pattern layer through the weight of the pattern layer (w_p). Similarly, each pattern layer is connected to the summation layer through the weight of the summation layer (w_s). The summation layer can be divided into the D-summation and S-summation neurons. The different steps of the GRNN architecture can be represented as follows:

$$G\left|\frac{v}{u}\right| = \frac{\int_{-\infty}^{\infty} v \cdot f(u,v)dy}{\int_{-\infty}^{\infty} f(u,v)dy} \tag{6}$$

Eq. (6) expresses the regression equation of the dependent variable (v) on the independent variable (u). In Eq. (6), $u = [u_1, u_1, \ldots, u_n]^T$ denotes the number of inputs, and $v = [v_1, v_1, \ldots, v_j]^T$ represents the number of outputs. Next, the function $f(u, v)$ calculates the probability density for u and v.

Tab. 1. Inputs and outputs data set for GRNN architecture

D_f in meter	D_l in meter	D_r in meter	V_R in meter/second	V_L in meter/second
0.01	0.024	0.01	0.075125	0.062125
0.024	0.01	0.01	0.075125	0.0775
0.01	0.01	0.024	0.0639	0.0775
0.016	0.032	0.012	0.097875	0.071625
0.032	0.016	0.012	0.097875	0.096375
0.012	0.016	0.032	0.067875	0.099
0.032	0.012	0.016	0.092625	0.101625
0.032	0.032	0.032	0.098125	0.11025
0.048	0.024	0.036	0.100375	0.124875
0.024	0.048	0.036	0.100375	0.110375
0.036	0.024	0.048	0.08225	0.12875
0.014	0.022	0.03	0.0735	0.09625
0.03	0.014	0.022	0.085375	0.102875
0.022	0.03	0.014	0.098625	0.07775

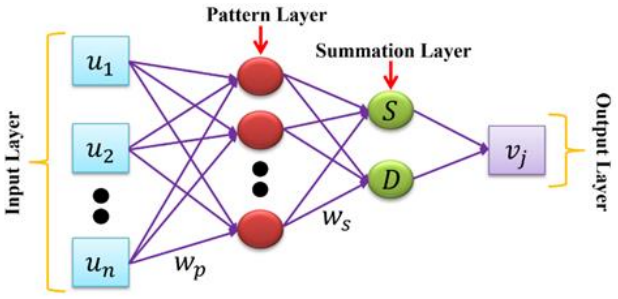

Fig. 3. The basic structure of the MIMO system GRNN architecture

The input layer collects the information from inputs (u) and stores this information. After getting inputs, the number of neurons is allotted to each input. Next, the input neurons of the input layers are sent to the pattern layer. The pattern layer uses a Gaussian function (φ_i) as follows: -

$$\varphi_i = exp\left\{-\frac{(u-U_i)^T(u-U_i)}{2\sigma^2}\right\} \tag{7}$$

where i = 1, 2,...,n; and σ represents the width or spread notation, which adjusts the value of final network outputs, the

present study takes σ = 1. U_i is the training vector of the i^{th} neuron in the pattern layer. In the fourth layer (final layer), the following function calculates the final outputs (v_i) of GRNN architecture:

$$v_i = \frac{\sum_{i=1}^{n} w_i \cdot \varphi_i}{\sum_{i=1}^{n} \varphi_i} \quad (8)$$

where w_i makes the weight connection between the i^{th} neuron of the pattern layer and summation layer node.

4. EXPERIMENTAL RESULTS OF AN E-PUCK ROBOT AMONG THREE-DIMENSIONAL SCATTERED OBSTACLES AND COMPARISON WITH FEEDFORWARD NEURAL NETWORK (SINGH AND THONGAM, 2018) IN THE TWO-DIMENSIONAL PLATFORM

This section reveals the motion and orientation results of GRNN architecture-controlled E-puck robot in the three-dimensional (3D) V-REP software platform among scattered obstacles. Also, the feedforward neural network (Singh and Thongam, 2018) is selected for comparison in the 2D platform. The 3D simulation environments with an E-puck robot and scattered obstacles are built in the V-REP software. The GRNN architecture with inputs and outputs data set and the kinematic equation are scripted in the MATLAB programming language. The remote Application Program Interface (API) function makes an interface between the MATLAB and V-REP software. After interfacing, we run the MATLAB script, and simultaneously we start the simulation in V-REP software. The script sends the motion control command to the E-puck robot in the 3D V-REP software platform, and the GRNN architecture gives the right, and left wheel velocities command to the E-puck robot according to the front, left, and right IR sensor readings. Fig. 4 illustrates the snapshot of an automated differential-driven two-wheeled E-puck robot that performs the experiments in the 3D V-REP software platform among scattered obstacles.

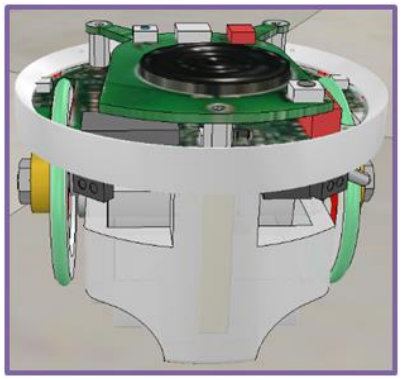

Fig. 4. Automated differential-driven two-wheeled E-puck robot

Further on, Fig. 5 reveals the 3D motion control results of an E-puck robot among scattered obstacles in the V-REP software platform using a GRNN architecture. The axis size is taken 230 × 230 cm^2 in Fig. 5. Next, the E-puck robot begins the motion from (210 cm, 10 cm) and reaches the target location, placed at the left corner (10 cm, 210 cm). Five green color cuboids and four yellow color cylindrical obstacles are randomly placed in the software platform to test the performance of the GRNN architecture in an E-puck robot. At first, the E-puck robot goes to reach the target, and after moving some distance, the E-puck robot finds obstacles within the specified sensory range. Then GRNN architecture is activated and sends the left and right wheel velocity control command to E-puck robot to avoid the obstacles. Fig. 8 shows the real-time recorded angular velocities (degree/seconds) of a right wheel (magenta color) and left wheel (cyan color) of an E-puck robot during motion control in the V-REP software platform of Fig. 5.

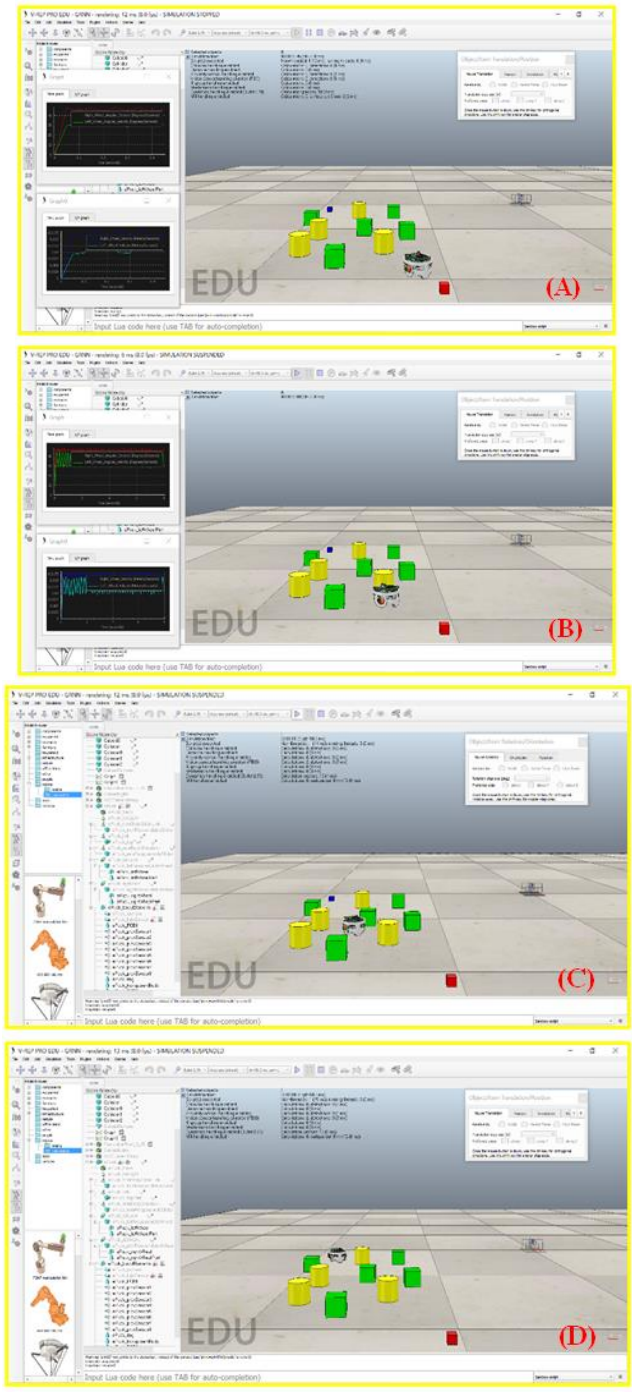

Fig. 5. 3D motion control results of an E-puck robot among scattered obstacles in the V-REP software platform using a GRNN architecture

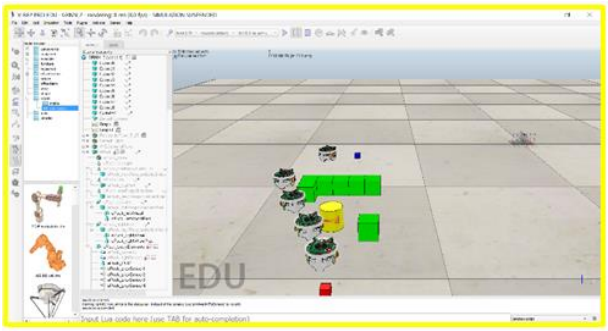

Fig. 6. 3D motion control results of an E-puck robot among wall-typeobstacles in the V-REP software platform using a GRNN architecture

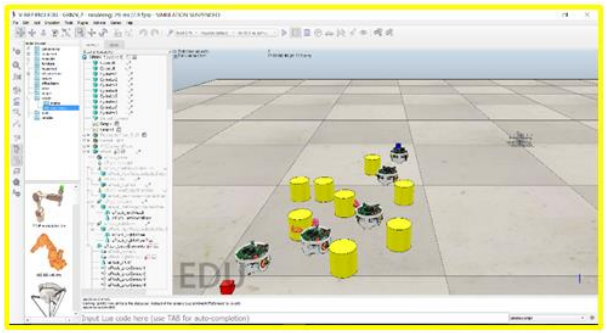

Fig. 7. 3D motion control results of an E-puck robot among many cylindrical shape obstacles in the V-REP software platform using a GRNN architecture

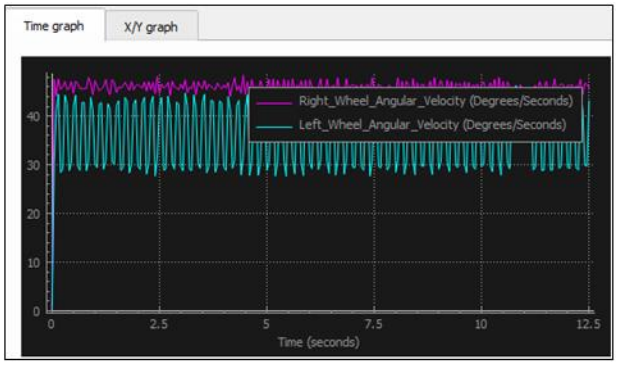

Fig. 8. Real-time recorded angular velocities (degree/seconds) of a right wheel (magenta color) and left wheel (cyan color) of an E-puck robot during motion control in the V-REP software platform of Fig. 5

Similarly, Fig. 9 displays the real-time recorded linear velocities (m/s) of a right wheel (yellow color) and left wheel (blue color) of an E-puck robot during motion control in the V-REP software platform of Fig. 5. As we can see in Fig. 5, the target (blue color small cuboid) is placed at the left corner, and that is why, most of the time, the E-puck robot takes a left turn to reach the goal. It means that the GRNN architecture increases the angular and linear velocity of the right wheel compared to the left wheel, as shown in Figs. 8 and 9, respectively. The E-puck robot covers 130 cm distance to reach the target from the starting location between scattered obstacles and takes 35 s. Figs. 6 and 7 show the motion control results of an E-puck robot among wall-type and many cylindrical shape obstacles in the V-REP software platform using a GRNN architecture, respectively. Further on, these figures present the different positions of an E-puck robot during navigation and obstacle avoidance in a single-single snapshot. The axis size is taken 230 × 230 cm^2 in both Figs. 6 and 7. In Fig. 6, the robot starts moving from (100 cm, 25 cm) and reaches the target location, placed at the coordinate (120 cm, 200 cm). Similarly, in Fig. 7, the robot begins motion from (20 cm, 20 cm) and reaches the target, located at the coordinate (210 cm, 210 cm). In both these figures, it can be seen that after moving some distance, the sensors of the robot detect the obstacles. Then, the GRNN architecture is activated and provides the left and right wheel velocity control command to the E-puck robot to avoid wall-type and many cylindrically shaped obstacles and reach the target successfully without any collision.

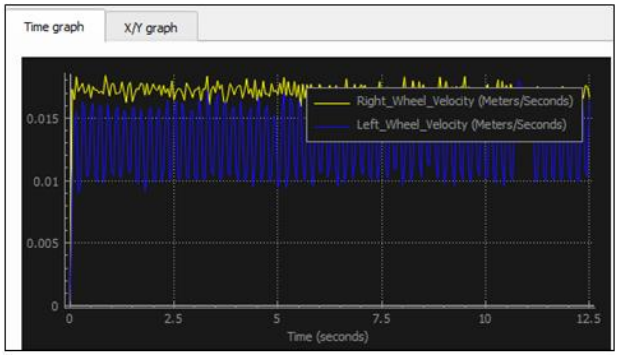

Fig. 9. Real-time recorded linear velocities (meter/seconds) of a right wheel (yellow color) and left wheel (blue color) of an E-puck robot during motion control in the V-REP software platform of Fig. 5

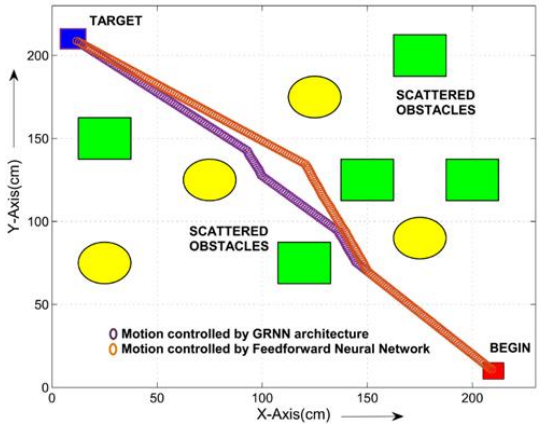

Fig. 10. 2D motion planning and control comparison result between proposed GRNN architecture and feedforward neural network (Singh and Thongam, 2018) among scattered obstacles in the same platform

After obtaining the 3D motion-control results of an E-puck robot, we perform the comparative study between the proposed GRNN architecture and previously developed feedforward neural network (Singh and Thongam, 2018) in the same 2D platform. Fig. 10 reveals the 2D motion planning and control comparison result between the proposed GRNN architecture and feedforward neural network (Singh and Thongam, 2018) among scattered obstacles. The purple color trajectory presents the GRNN architecture controlled motion result of an E-puck robot in Figure Fig. 10. Similarly, the red color trajectory shows the navigation result of a feedforward neural network (Singh and Thongam, 2018) driven E-puck robot. As shown in Figure Fig. 8, the GRNN architecture controlled E-puck robot utilises a shorter distance to reach the target compared to the feedforward neural network (Singh and

Thongam, 2018) driven E-puck robot because GRNN provides accurate network output with minimal error (Specht, 1991). Further on, Tab. 2 compares the GRNN architecture and feedforward neural network (Singh and Thongam, 2018) in terms of trajectory path length and time.

Tab. 2. Result comparison between the GRNN architecture feedforward neural network (Singh and Thongam, 2018) in terms of trajectory path length and time

Name of applied method	Figure number	Begin	Target	Trajectory path length (cm)	Trajectory time (seconds)	Trajectory path length error (cm)
GRNN architecture	Fig. 10	(210 cm, 10 cm)	(10 cm, 210 cm)	130 cm	35 s	1.34 cm
Feedforward neural network (Singh and Thongam, 2018)	Fig. 10	(210 cm, 10 cm)	(10 cm, 210 cm)	138 cm	38 s	2.91 cm

5. CONCLUSION AND FUTURE WORK

This article has presented the motion planning and control technique for an E-puck robot by applying GRNN architecture in the V-REP software platform among scattered obstacles. The GRNN architecture receives real-time obstacle information as inputs from the IR sensors of an E-puck robot. According to IR sensor data interpretation, this architecture sends the left and right wheel velocities command as outputs to the E-puck robot during navigation among obstacles. The programming of GRNN architecture and kinematic equations has been written in the MATLAB script. This script controls the motion and orientation of an E-puck robot in the V-REP software platform through the remote-API function. The different 2D and 3D simulation results demonstrate that the GRNN architecture successfully autonomously controls the motion and orientation of an E-puck robot.

Moreover, as compared to the feedforward neural network (Singh and Thongam, 2018), the proposed GRNN architecture has provided better results in a short period with minimal error. Future work can include dynamic scattered obstacles instead of static obstacles. Also, this presented method can be used for motion and orientation control of multiple E-puck robots.

REFERENCES

1. **Almeida T., Santos V., Mozos O. M., Lourenço B.** (2021), Comparative Analysis of Deep Neural Networks for the Detection and Decoding of Data Matrix Landmarks in Cluttered Indoor Environments. *Journal of Intelligent & Robotic Systems*, 103(1), 1-14.
2. **Ben Jabeur C., Seddik H.** (2020), Design of a PID optimized neural networks and PD fuzzy logic controllers for a two-wheeled mobile robot, *Asian Journal of Control*, 22(1), 1–19.
3. **Elmi Z., Efe M. Ö.** (2020), Online path planning of mobile robot using grasshopper algorithm in a dynamic and unknown environment, *Journal of Experimental & Theoretical Artificial Intelligence*, 32, 1–19.
4. **Hadi N. H., Younus K. K.** (2020), Path tracking and backstepping control for a wheeled mobile robot (WMR) in a slipping environment, *IOP Conference Series: Materials Science and Engineering*, 671, 1–17.
5. **Khan H., Khatoon S., Gaur P.** (2021), Comparison of various controller design for the speed control of DC motors used in two wheeled mobile robots. *International Journal of Information Technology*, 13(2), 713-720.
6. **Long Y., Zuo Z., Su Y., Li J., Zhang H.** (2020), An A*-based Bacterial Foraging Optimisation Algorithm for Global Path Planning of Unmanned Surface Vehicles, *The Journal of Navigation*, 73(3), 1–16.
7. **Narasimhan G. E., Bettyjane J.** (2020), Implementation and study of a novel approach to control adaptive cooperative robot using fuzzy rules. *International Jopurnal of Information Technology*, 1–8. https://doi.org/10.1007/s41870-020-00459-z
8. **Nedjah N., Junior L. S.** (2019), Review of methodologies and tasks in swarm robotics towards standardization, *Swarm and Evolutionary Computation*, 50, 1–26.
9. **Osaba E., Del Ser J., Iglesias A., Yang X. S.** (2019), Soft Computing for Swarm Robotics: New Trends and Applications, *Journal of Computational Science*, 39, 1–4.
10. **Pandey A., Kashyap A. K., Parhi D. R., Patle, B. K.** (2019), Autonomous mobile robot navigation between static and dynamic obstacles using multiple ANFIS architecture, *World Journal of Engineering*, 16(2), 275–286.
11. **Pandey A., Parhi D. R.** (2016), New algorithm for behaviour-based mobile robot navigation in cluttered environment using neural network architecture, *World Journal of Engineering*, 13(2), 129–141.
12. **Pandey K. K., Parhi D. R.** (2019), Trajectory Planning and the Target Search by the Mobile Robot in an Environment Using a Behavior-Based Neural Network Approach, *Robotica*, 37(1), 1–15.
13. **Protik P., Das S., Islam M. R.** (2019, October). Chemical Reaction Optimization for Mobile Robot Path Planning. *International Joint Conference on Computational Intelligence, Springer, Singapore*, 191–203.
14. **Quan Y., Ouyang H., Zhang C., Li S., Gao L.** (2021), Mobile Robot Dynamic Path Planning Based on Self-adaptive Harmony Search Algorithm and Morphin Algorithm. *IEEE Access*, 10.1109/ACCESS.2021.3098706
15. **Singh N.H, Thongam K.** (2018), Neural network-based approaches for mobile robot navigation in static and moving obstacles environments, Intelligent Service *Robotics*, 12(1), 55–67.
16. **Specht D. F.** (1991), A general regression neural network. *IEEE transactions on neural networks*, 2(6), 568–576.
17. **Teli T. A., Wani M. A.** (2021), A fuzzy based local minima avoidance path planning in autonomous robots. *International Journal of Information Technology*, 13(1), 33-40.
18. **Tripathy H. K., Mishra S., Thakkar H. K., Rai D.** (2021), CARE: A Collision-Aware Mobile Robot Navigation in Grid Environment using Improved Breadth First Search. *Computers & Electrical Engineering*, 94, 107327.
19. **Wang M.** (2021), Real-time path optimization of mobile robots based on improved genetic algorithm. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 235(5), 646-651.
20. **Zhao T., Xiang Y., Dian S., Guo R., Li S.** (2020), Hierarchical interval type-2 fuzzy path planning based on genetic optimization, *Journal of Intelligent & Fuzzy Systems*, 32, 1-12.

Vikas Singh Panwar: https://orcid.org/0000-0001-5874-2945

Anish Pandey: https://orcid.org/0000-0001-9089-3727