International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

NEAT based Artificial Neural Network with Genetic Algorithms for BLDC Motor Control

Corneliu-Inocențiu Colpacci, MSc
Department of Automatic Control and Electronics
University of Craiova
A.I. Cuza 13, 200585, Craiova, Romania

Abstract— **This paper wants to explore the application of genetic algorithms in system control. This paper proposes the implementation of a neural network with genetic algorithms based on NEAT (Neuroevolutionary of Augmenting Topologies) technique. The purpose is to see the effect of implementing neuronal network algorithms in controlling complex systems such an BLDC (Brushless DC) motors. Traditional systems such as PID controllers often encounter difficulties in solving nonlinearities. In contrast the proposed implementation gives the advantage that the algorithm creates a custom neuronal network to solve the task given. The main feature of the given system is that the topology of the network is not fixed but evolve over time, this means that the algorithm does not merely search for the best weights for a given architecture but automatically discovers the minimal and sufficient structure to solve the problem.**

I. INTRODUCTION

In this paper, the development of an intelligent controller for a brushless DC (BLDC) motor is proposed, using the NEAT (Neuroevolutionary of Augmenting Topologies) technique. The objective is to obtain a control algorithm capable of creating a neural network capable of maintaining speed, ensure stability and energy-efficiency requirements, creating the topology of the network from a simple one and evolving it during training. The choice of the BLDC as the target system is justified by its fast and nonlinear dynamics and its wide usage in robotics, modern system and industries.

The implementation of a controller based on artificial intelligence algorithms can bring significant improvements over conventional and classic solutions. By learning from data during training, the controller can reduce tracking error, shorten settling time, and optimize energy consumption under variable conditions. In addition, an evolved model can integrate anomaly detection and predictive diagnostic mechanisms, that result in reduced maintenance costs. Practically, these advantages translate into better performances and potential cost savings.

NEAT starts with an advantage from other algorithms, because the network topology is not fixed but evolves over time. This means the algorithm does not merely search for the best weights for a given architecture but automatically discovers the minimal and sufficient structure to solve the problem.

Key contributions. The primary contribution of the study is:

- Development and integration of a genetic algorithm for creating neuronal networks in controlling complex systems.
- The effect of controlling a system with complex dynamic using an AI based controller.
- Finding a rapid solution for controlling a system using genetic algorithm.

II. RELATED WORK

With the emergence of reliable and robust algorithms for artificial intelligence we can see it use spread in different domains. Its use of controlling systems also adds significant improvement such as cost reduction; efficiency increase and robustness.

Different studies implement different solutions to integrate the newly found capabilities of AI. They can serve as a standalone controller [1], [6], where by designing a capable neuronal network, they can achieve better performances than the classical solution such as PID. In this process the network is crafted by the user and trained in order to evaluate its performance.

Other implementations combine them with the classical solution to enhance them [2], [7], [10] hence bringing the best these two solutions can offer or in other cases they can detect early fault [7] and improve efficiency and reduce cost [9].

In contrast, NEAT algorithm represents a different approach on how we view the design of neural networks. Instead of choosing a fixed architecture from the start, we let an evolutionary process gradually build the structure and parameters needed to solve the given task [4], [5].

This idea becomes particularly valuable in control problems where the system dynamics are complex, present nonlinear or where manually designing an efficient network is difficult or costly. By starting from simple solutions and increasing complexity only when it discovers improvements, NEAT offers a new approach in designing system controllers.

Applying NEAT to control also generates new problems: evaluating the population can be computationally expensive, and the resulting networks may be difficult to analyze from the standpoint of stability guarantees. Moreover, without

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

designing the correct implementation of the fitness function, evolution can produce unstable or excessively complex solutions. For this reason, integrating NEAT into critical systems requires mechanisms to penalize complexity, rigorous validation in simulation, and deterministic fallback strategies.

III. PROPOSED METHODOLOGY

A. BLDC Motor Modeling

Understanding the dynamic behavior of the BLDC motor is crucial in designing a stable controller capable of completing the given task. The motor's dynamics are represented by a set of differential equations that describe both the electrical and mechanical aspects.

$$V = RI + L\frac{dI}{dt} + K_e\omega \quad (1)$$

where: V = voltage applied to the motor phase; R = Stator winding resistance; I = Current through the winding; L = Phase inductance; di/dt = rate of change of current over time; E =back electromotive force (Back-EMF)

The rotor position "θ", angular velocity "ω", and motor torque "T" constitute the essential state variables, and based on the relation between then, can be used to control and analyze the system behavior.

$$J\frac{d\omega}{dt} = K_t I - T_L - B\omega \quad (2)$$

where: J = rotor moment of inertia; ω = angular velocity; T = motor torque; TL = load torque; B = friction coefficient

B. NEAT algorithm and implementation

The AI-based controller uses the NEAT algorithm, an evolutionary method that optimizes the topology of the neuronal network and the weight and biases. Unlike a classical PID controller, this type of controller does not rely on an explicit mathematical model of the system, but its structures evolve overtime during training, generating or removing connection in order to discover the best solution for the given task.

The algorithm starts by codifying each network with a genome, where each connection has a weight and an innovation number. On each generation, the network is group based on genetic differences so that new architectures can be created after they are evaluated. In the crossover period the number of innovations permit correction of genes between parents, and different topologies and unique characteristics are kept or removed based on the fitness of the parents. Mutation can be in weight or structural modification, by adding new connection or nodes.

As observed in Fig. 1, not only the weight is modified to find the optimal solution but the topology of the network as well. By introducing node 5 for Parent 2 and the new connection between node 2 and node 4. NEAT starts from simple neural networks and gradually develops them by adding new neurons and connections, using principles inspired by natural evolution such as selection, mutation, and crossover.

In the Simulink simulation, the AI controller receives the system error and reference value as inputs and outputs the motor speed.

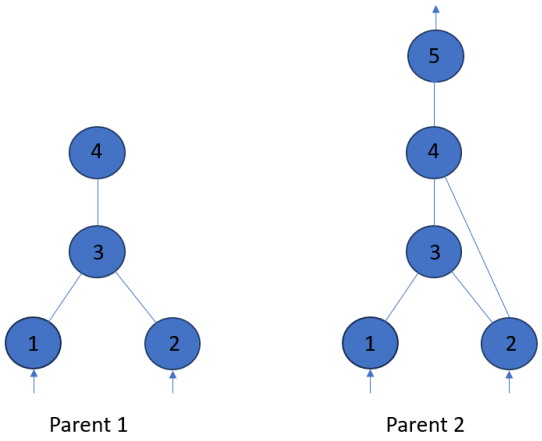

Fig. 1. Evolution of the network after each generation

The performance of each network is evaluated according to the fitness function.

$$fitness = -\sum|speed_{ref} - speed| \quad (3)$$

This approach enables an adaptive controller capable of handling nonlinear or complex systems, resulting in better solutions than classical methods.

IV. SIMULATION ENVIRONMENT

The purpose of the system is to maintain the motor at a stable and desired speed. To achieve this performance, the system is designed in a structure in which several interdependent subsystems are integrated. These include different electromechanical components that interact with the motor mechanisms, sensors dedicated in measuring use for controlling operating parameters, as well as software components that implement the control algorithms and coordinate the operation.

The NEAT algorithm (Neuroevolutionary of Augmenting Topologies), developed by Kenneth O. Stanley [4], is one method for implementing evolving neural networks and is based on principles of genetic algorithms. It starts from a simple network with different connections and weights. Each network is evaluated in a simulation of the controlled system, and performance is measured by a fitness function that takes into account speed error and different parameters for improving the network.

Networks with better results are selected for reproduction, and through genetic operations such as crossover and mutation, a new generation is produced. Unlike other methods, NEAT does not optimize only the connection weights and biases but also the network topology, allowing new neurons and connections so that complexity is increased over time. After a given number of generations specified by the designer, the algorithm can produce a network capable of controlling the motor at a high level of performance.

Fig. 2. Block Diagram for training the model.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

To train the model, a specialized simulation is created, integrated with Python and MATLAB/Simulink environment, which will reproduce the motor dynamics, the behavior of sensors, and the load conditions necessary for evaluation. The simulation will contain different parameters such as speed and error signals that can be analyzed and compared with classical methods.

We note that in this simulation created, see "Figure 2. Block Diagram for training the model" for training the network the control loop has been removed, since it is implemented in Python environment and run in Simulink, the system will run continuously in closed-loop mode.

V. SIMULATION RESULT AND ANALYSIS

By integrating the AI-based control module that uses a genetic algorithm, the Simulink program undergoes the following modification: the PI control block is replaced with a library custom made that incorporate the new created neuronal network, see Fig. 3.

We observe the system's evolution over time, where a very short settling time of approximately 0.17 seconds, but this performance produces an overshoot phenomenon, with the peak value reaching as high as 99.3 rpm, as observed in Fig. 5, whereas the classical system with a PI controller stabilizes after approximately 1 second, observed in Fig. 4.

Considering the relatively short training time during which the model was trained for 20 generations with a population of 150 individuals per generation, the results obtained show the algorithm can achieve the desired outcome of stabilizing the motor speed at the reference value given, even at a relatively low training time.

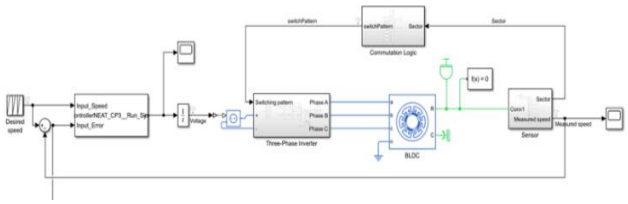

Fig. 3. Simulation Block

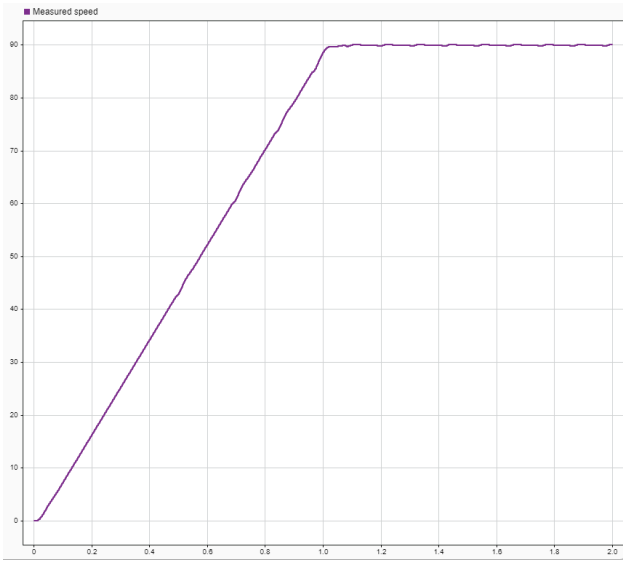

Fig. 4. Measured speed with PI.

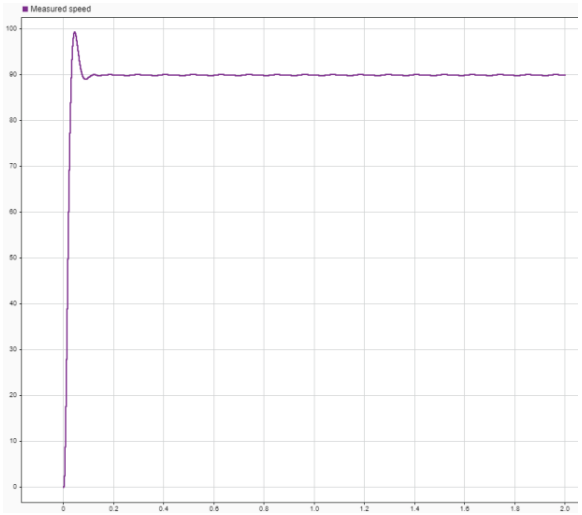

Fig. 5. Measured speed with AI Controller.

Another advantage of integrating the genetic algorithm-based control module is the system's predictable and consistent behavior; this is evidenced by the motor's power consumption, which remains unchanged throughout the simulation, achieving the result desired much faster at a cost expected by the designer.

This behavior can be observed in Fig. 6 and Fig. 7. With the PI controller we can see that voltage is increased over time until it reaches the necessary value to maintain the desired speed. Wherever with the AI controller the necessary voltage is applied directly from the start.

To better observe the system's performance, metrics based on integrals of the error such as IAE, ISE, ITAE, and ITSE were also performed; these allow a more detailed evaluation of the dynamic behavior, highlighting both the magnitude and the time distribution of the errors, thereby providing a clear picture of the efficiency and quality of the implemented control.

Although performance metrics, see Table 1, indicate that error persists over time, it can be observed that the designed controller is robust. Even in the presence of variations in error values, the system's overall behavior is not significantly affected.

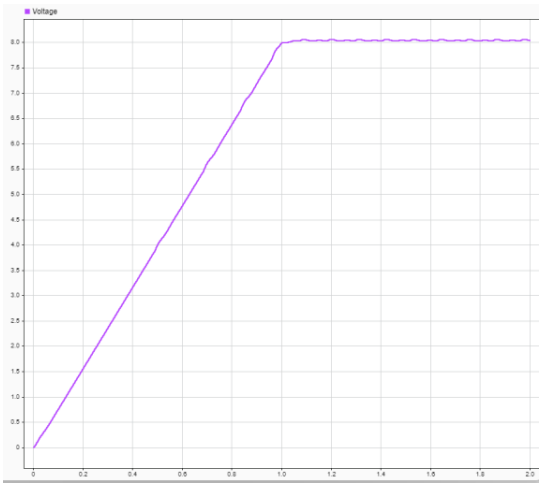

Fig. 6. Voltage consumption with PI.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

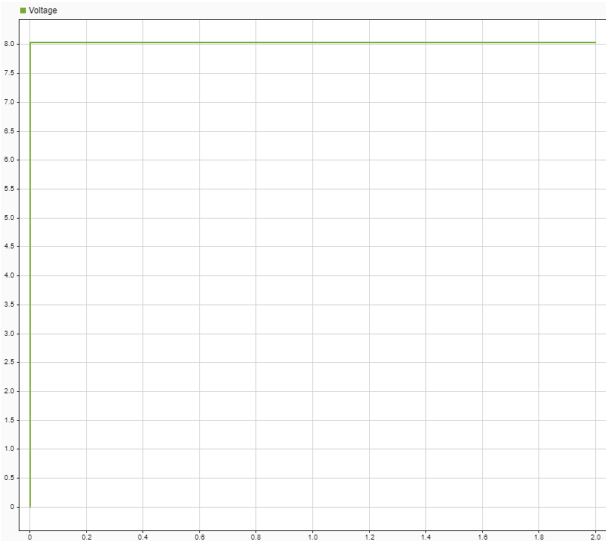

Fig. 7. Voltage consumption with AI controller.

This is evidenced by the rapid stabilization of the response and the maintenance of constant consumption, which confirms the controller's ability to ensure stable performance under varying operating conditions.

TABLE 1. PERFORMANCE METRICS

Metric	T0: 0s	T1: 0,5s	T2: 1,0s	T3: 1,5s	T4: 2,0s
IAE	0	32,79	43,75	43.80	43.84
ISE	0	2244,34	2575,20	2575,21	2575,22
ITAE	0	7,52	14,95	15,01	15,07
ITSE	0	476,13	672,53	672,54	672,55

After applying a load consisting of a rotational damper and a rotational spring, we can observe the evolution of the system without training the controller to take into consideration the newly added data, as we can see in Fig. 8.

After adding a rotational damper and a rotational spring, we observed that the AI controller still operated within the desired parameters without retraining it with the new load data, as the graphs Fig. 9 and Fig. 10.

Very short settling time: the AI controller's response stabilizes in approximately 0.17 s, much faster than the roughly 1 s for the PI regulator. Rapid algorithm convergence: training for 20 generations with 150 individuals per generation led to a satisfying solution, demonstrating NEAT's ability to quickly find effective topologies and weights.

Persistent error metrics: IAE, ISE, ITAE, and ITSE indicate the presence of residual error, but this does not compromise the overall robustness of the system.

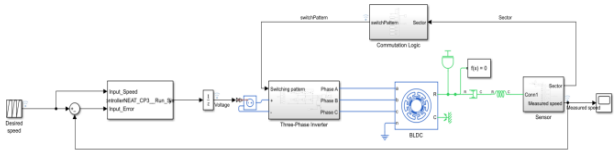

Fig. 8. Simulation Block with load added.

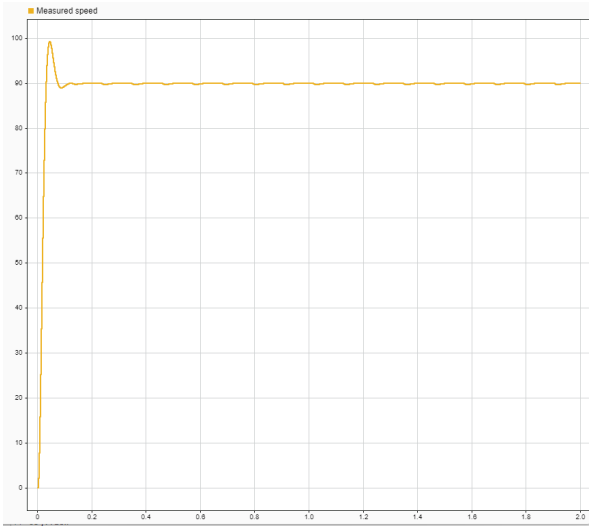

Fig. 9. Measured speed with AI Controller after load.

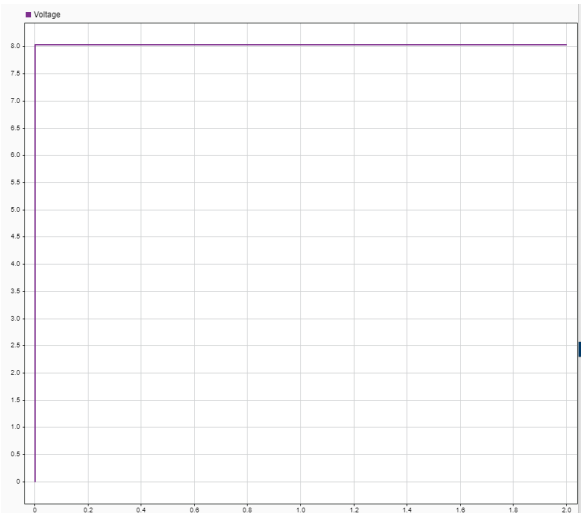

Fig. 10. Voltage consumption with AI controller after load.

Nevertheless, NEAT offers the following advantages: evolving neuronal network, where topology optimization enables finding architectures suitable for nonlinear or hard-to-model dynamics; improved dynamic performance, since much shorter response times can be achieved without relying on an explicit mathematical model; robustness, with rapid stabilization and constant power consumption indicating resilience to variations in operating conditions; and training efficiency, with good results obtained using a relatively small number of generations and a moderate population size.

VI. CONCLUSION

The developed system shows that integrating an evolutionary controller based on genetic algorithm can provide fast, adaptive, and energy-efficient control of the motor speed, achieving better results in certain dynamic metrics compared to a classical PI regulator.

However, despite these advantages, certain limitations were observed: overshoot and potential transient instability, since the reduced settling time came at the cost of significant overshoot, which may be unacceptable in sensitive applications; residual error, as integral metrics show

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

persistent error that can affect long-term precision; and dependence on the training scenario, because final performance reflects the fidelity of the simulation and the load conditions used during training, so generalization to real conditions may need additional modification to create a suitable environment.

VII. FUTURE SCOPE

The implementation of an evolutionary controller based on NEAT opens multiple development directions aimed at both improving simulation performance and ensuring a safe, robust control solution to real-world applications.

Experimentally, it is essential to extend training by increasing the number of generations and diversifying the population, which will improve the networks' capability to reduce errors and increase stability. Training across various scenarios such as extreme loads and sensor noise, should also be integrated into the training process to test the robustness of solutions under conditions close to real operation and unexpected scenarios that could impact the user.

To achieve an optimal trade-off between speed, precision, and energy efficiency, a capable solution of the fitness function is encouraged, incorporating explicit penalties for overshoot, settling time, and energy consumption. Incorporating these parameters could lead to more balanced solutions capable of reducing transient peaks without compromising response agility.

A promising direction is the development of hybrid schemes in which the NEAT controller operates alongside a classical layer to combine the evolutionary network's agility with the well-known stability of traditional regulators. Complementarily, implementing limiters and filters saturation limits, rate limiters, and transient filters will protect the physical system by controlling command peaks and attenuating unwanted transient effects.

To maintain long-term performance in the face of dynamic environmental changes, it is necessary to integrate incremental learning mechanisms that allow the network to continuously readjust to slow variations in dynamics or operating conditions. In parallel, implementing monitoring degradation detection will ensure an acceptable performance level is maintained and will reduce the risk of unexpected failures, that could lead to an efficient cost reduction.

REFERENCES

[1] M. Anbazhagan, R. Danus, S. B. Suriya, and S. Raagul, "AI based Adaptive Speed Control of BLDC Motor," Proc. 6th Int. Conf. Electronics and Sustainable Communication Systems (ICESC), pp. 40–46, September 2025..

[2] D. Kanase, S. Chavan, and H. Mehta, "AI-Enhanced Control and Simulation of BLDC Motors for Autonomous Vehicle Applications," Proc. 2025 Global Conf. Emerging Technology (GINOTECH), pp. 208–212, May 2025.

[3] M. G. Basha, K. Chenchireddy, S. A. Sydu, W. Sultana, V. Kumar, and L. B. Ganesh, "Performance Improvement of DC Motor by using ANFIS Controller," Proc. 4th Int. Conf. Pervasive Computing and Social Networking (ICPCSN), pp. 932–938, May 2024.

[4] K. O. Stanley and R. Miikkulainen, "Evolving Neural Networks through Augmenting Topologies," MIT Press Journals, vol. 10, pp. 99–127, June 2002.

[5] P. Verbancsics and J. Harguess, "Generative NeuroEvolution for Deep Learning," arXiv preprint arXiv:1312.5355, December 2013.

[6] C. V. Kumar, G. M. Rao, and A. R. Ram, "AI based Vector Control Method for BLDC Motor with Multi Switch Three-Phase Topology," Psychology and Education, vol. 58, no. 1, pp. 3132–3141, January 2021.

[7] Y. Lee and S. Makonin, "BLDC Hall Sensor Displacement Dataset (BLDC-HSD)," IEEE Data Descriptions, vol. 1, pp. 22–26, October 2024.

[8] E. H. Al-Zaidi, "Supervised Learning for Adaptive BLDC Motor Control: Integrating Classical PID with Neural Networks," Eurasia Proc. Science, Technology, Engineering and Mathematics (EPSTEM), vol. 37, pp. 180–193, November 2025.

[9] S. Banerjee, S. Pramanik, "AI-Integrated sensor less Control of BLDC Motors for Energy-Efficient Electric Vehicles", International Research Journal on Advanced Engineering and Management, vol. 3, pp. 2869-2876, September 2025.

[10] R. Dhanasekar, S. G. Kumar, and M. Rivera, "Improved Speed Control of BLDC Motor using Luo Converter by Sliding Mode Control," IEEE Conf. Proc. (CHILECON), pp. 1–6, November 2019.