International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 11 | Nov - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Autonomus Systems Control Design Using Neuro Evolution

Ms. Rajeshwari S ,Thilak T M, Mahanthesh J R

Assistant Professor at Channabasveshwara Institute of Technology ,Gubbi
Under Graduate Student at Channabasveshwara Institute of Technology ,Gubbi
Under Graduate Student at Channabasveshwara Institute of Technology ,Gubbi

………………………………*********…………………………………

ABSTRACT:

This paper explores the application of neuro-evolution techniques in the design of control systems for autonomous systems. Neuro-evolution merges neural networks with evolutionary algorithms to develop adaptive, real-time control strategies for autonomous platforms. This approach is particularly effective in dynamic and complex environments, where traditional control methods struggle to ensure optimal performance. By evolving neural network structures, neuro-evolution provides robust and flexible control solutions, with promising applications in fields such as robotics, autonomous vehicles, and drone navigation.

The design of autonomous systems has rapidly advanced, fueled by the need for intelligent, adaptive control in environments that are complex and unpredictable. Traditional control systems often lack the flexibility required to handle dynamic changes, making them less effective in highly variable conditions. Neuro-evolution, which combines neural networks with evolutionary algorithms, offers a promising solution by enabling autonomous systems to develop and refine control strategies on their own. This approach leverages evolutionary principles to optimize neural network architectures, producing control systems that adapt in real-time and improve through experience. Neuro-evolution has gained significant attention for applications in robotics, self-driving vehicles, and aerospace, where autonomous, robust control is essential. This paper explores the role of neuro-evolution in enhancing control design, focusing on its advantages over conventional methods and its potential to drive further innovations in autonomous systems.

INTRODUCTION: The design of autonomous systems has rapidly advanced, fueled by the need for intelligent, adaptive control in environments that are complex and unpredictable. Traditional control systems often lack the flexibility required to handle dynamic changes, making them less effective in highly variable conditions. Neuro-evolution, which combines neural networks with evolutionary algorithms, offers a promising solution by enabling autonomous systems to develop and refine control strategies on their own. This approach leverages evolutionary principles to optimize neural network architectures, producing control systems that adapt in real-time and improve through experience. Neuro-evolution has gained significant attention for applications in robotics, self-driving vehicles, and aerospace, where autonomous, robust control is essential. This paper explores the role of neuro-evolution in enhancing control design, focusing on its advantages over conventional methods and its potential to drive further innovations in autonomous systems. Neuro-evolution works by iteratively improving neural network-based controllers using evolutionary techniques, which can involve tuning parameters or even restructuring the network itself to enhance functionality. This adaptive capability makes neuro-evolution particularly appealing for autonomous applications in fields such as robotics, aerospace, and automotive engineering, where precision, robustness, and real-time decision-making are essential. As the demand for fully autonomous and self-improving systems grows, neuro-evolution offers a promising path forward, enabling systems to perform complex tasks with minimal human intervention. This paper investigates the principles, methodologies, and practical applications of neuro-evolution in autonomous systems control, highlighting its advantages over conventional control approaches and its potential for future advancements in autonomous technologies.

METHODOLGY:

1. **Dataset Selection:** The methodology begins with the selection of appropriate datasets that reflect the complexities of the autonomous environment. Datasets can include simulated scenarios generated from environments like OpenAI Gym or Unity ML-Agents, which provide rich, diverse data for training. Additionally, real-world datasets from sources such as the Udacity self-driving car dataset or robotic control benchmarks may be utilized. These datasets typically consist of various sensory inputs (e.g., LIDAR, camera images, IMU data) and corresponding control outputs (e.g., steering angles, throttle commands) that are crucial for effective model training.
2. **Neuro-Evolutionary Algorithm Implementation:** The neuro-evolutionary approach involves the integration of a neural network architecture with an evolutionary algorithm. A population of neural networks is initialized, each representing a different control policy. The evolutionary algorithm iteratively selects, recombines, and mutates the network architectures based on their performance in the simulated or real-world tasks. Key components of this process include fitness evaluation, selection mechanisms (such as tournament selection or roulette wheel selection), and genetic operators (crossover and mutation) to promote diversity within the population.
3. **Training and Optimization:** During the training phase, the selected datasets are used to optimize the neural networks. Each network in the population is evaluated based on its ability to perform the specified control tasks, such as navigating through obstacles or maintaining stability. The fitness function is designed to quantify performance, considering factors such as task completion time, error rates, and energy efficiency. Over successive generations, the best-performing networks are selected for reproduction, and the process continues until a satisfactory level of performance is achieved.
4. **Performance Evaluation Metrics:**

To assess the effectiveness of the neuro-evolutionary control design, several performance evaluation metrics are employed. These metrics include:

- **Mean Squared Error (MSE)**: Measures the average squared difference between predicted and actual outputs, providing an indication of prediction accuracy.

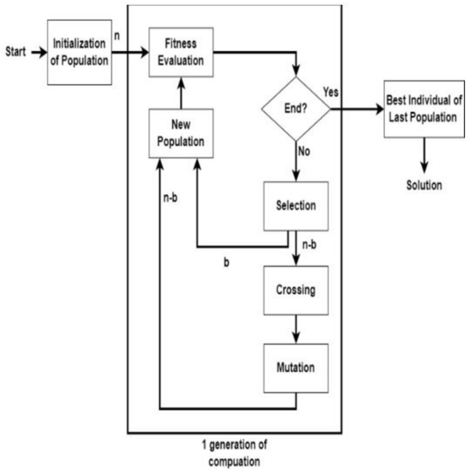
- **Success Rate**: The percentage of successful task completions, which reflects the reliability of the control system.

- **Response Time**: The time taken for the system to react to stimuli, which is critical for real-time applications.

- **Energy Efficiency**: The amount of energy consumed during operation, important for autonomous systems with limited resources.

- **Robustness**: The ability of the control system to perform well under varying conditions and disturbances, assessed through scenario testing and stress tests.

FLOW CHART:



Challenges and Limitations:

1. Computational Complexity

One of the primary challenges associated with neuro-evolution is its computational complexity. The process of evolving neural networks through iterative evaluations and optimizations can be resource-intensive, requiring significant computational power and time. This complexity is particularly pronounced in high-dimensional state spaces, where the number of possible solutions increases exponentially, leading to longer training times and higher costs in terms of computational resources.

2. Convergence Issues: Achieving convergence in neuro-evolution can be problematic, especially in environments with high variability or noise. The stochastic nature of evolutionary algorithms can result in suboptimal solutions, where the evolved network may not fully exploit the potential of the feature space. Furthermore, premature convergence can occur, leading to a lack of diversity in the population and hindering the exploration of potentially better solutions.

3. Interpretability

Neuro-evolutionary models, particularly those based on deep learning architectures, can be challenging to interpret. Understanding how the evolved networks make decisions can be difficult, which poses significant obstacles in safety-critical applications, such as autonomous driving or healthcare. The lack of transparency can hinder trust and acceptance among users and stakeholders, making it essential to develop methods for better interpretability of the models.

4. Real-World Implementation

Transitioning from simulation to real-world applications presents its own set of challenges. Neuro-evolutionary algorithms may perform well in controlled environments, but their effectiveness can diminish when faced with real-world complexities and uncertainties. Factors such as sensor noise, actuator delays, and unmodeled dynamics can significantly impact performance, necessitating extensive testing and adaptation to ensure reliability in practical scenarios.

To train and test the ANN, two different maps with start and finish were created. On the map, there is a sufficiently wide road through which the vehicle must be able to pass. The environment outside the road is defined as the obstacle, so any leaving of the road is considered a collision. On each map, there are several so-called checkpoints, for reaching which the individual is rewarded after their passing.

Fig. 9. is showing the proposed training tracks, the left track is No. 1, the right track is No. 2. The map area defined as the obstacle is shown in light green, the allowed path is shown in grey. The green/red bars show the checkpoints, for which the vehicle receives rewards, and the target is shown in blue.

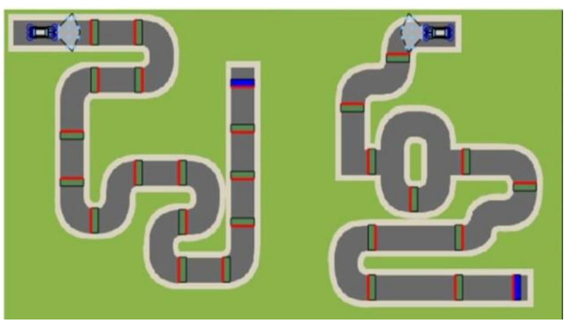


Fig. 9.Example of designed training tracks

Fig. 10. shows the test tracks, on the left is a track No. 4 and on the right is track No. 3. The track No. 4 is similar to the training track, however, the direction is reversed, and the checkpoints are removed on each runway, which provides enough change to use this track for testing.

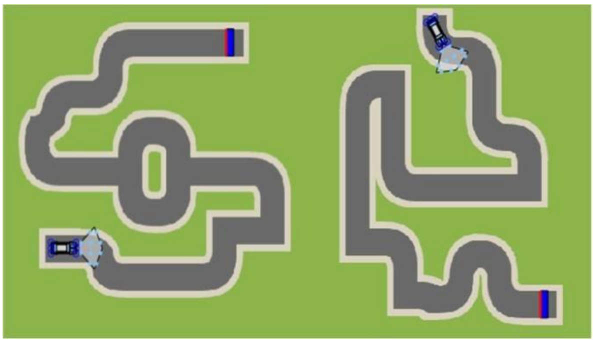


Fig. 10.Testing tracks

A.Experiment with radar sensor

The first experiment was to train and then test the vehicle using a radar sensor only. The ANN was trained on Track 2. It was then tested on Track 3 and Track 4. The results on Tracks 3 and 4 can be seen in Fig. 11. Although ANN was only trained on track 2, it was able to pass both test tracks and do so without any major problems, which can be considered as a good generalization ability of the ANN. In Fig. 12 and 13, the graphs of the fitness evolution are depicted.

Fig. 11.Resulting vehicle motion on test tracks 3 and 4 for the radar type sensor

RESULTS:

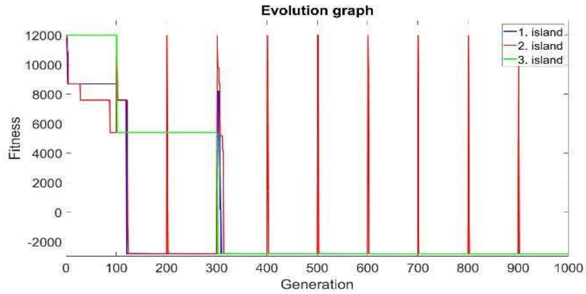


Fig. 12.The evolution of fitness during training using the radar sensor

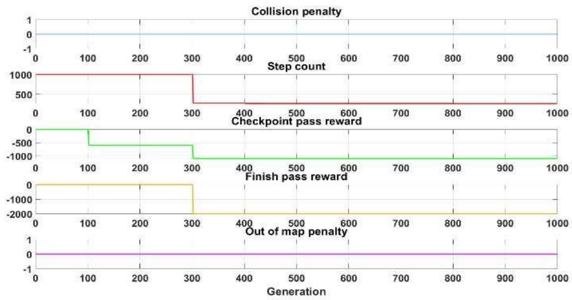


Fig. 13.The graphs of fitness sub-values during training with radar sensor

B.Experiments with camera sensor

The second experiment was to train and then test the ANN using only the camera sensor. For the training, track 2 has been used and for the testing, tracks 3 and 4 (Fig. 14. and 15.) have been used. Again, it can be concluded that the network was able to pass both test track 3 and track 4 without much difficulty and hence the ANN is able to generalize the learned control ability and subsequently apply it to other tracks.

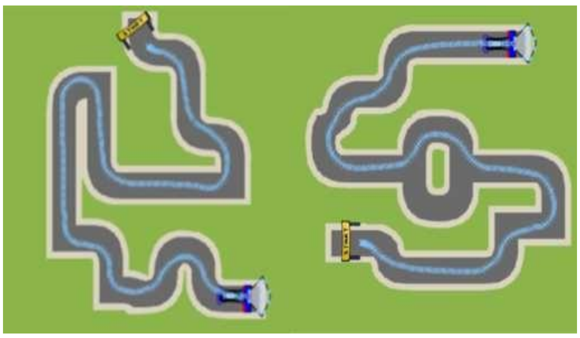


Fig. 14.Resulting motion of the trained vehicle using only the camera on tracks 3 and 4

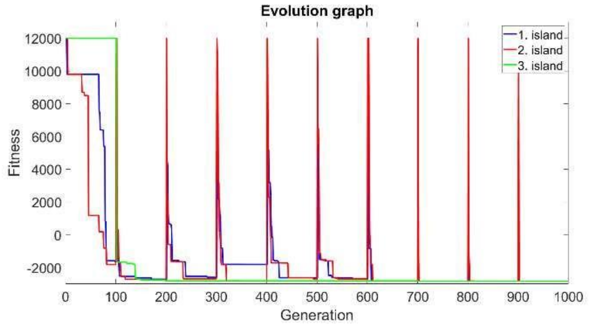


Fig. 15.The evolution of fitness during training using the camera

C. Experiment with radar + camera
The last experiment was the training and subsequent testing of the ANN using the fusion of sensors: radar and camera. This approach achieves slightly better results than using radar or a camera alone. However, since in this case, we consider more input variables, it was necessary to use a different ANN architecture (Fig. 6.). The latter contained more parameters and is thus more complex to train. Nevertheless, the ANN with sensor fusion was able to reach the finish usually a few steps earlier.

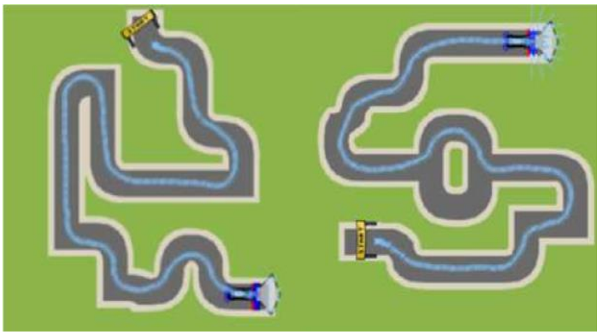


Fig. 16. Movement of trained ANN with camera and radar on tracks 3 and 4

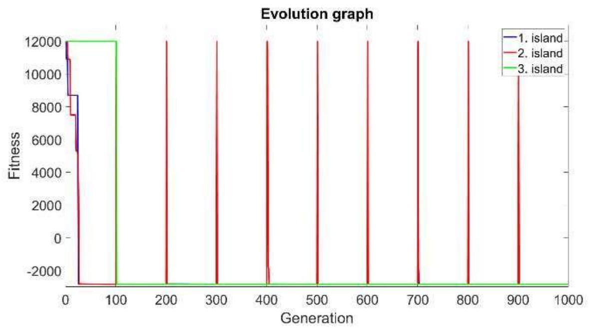


Fig. 17. The evolution of fitness for the radar + camera

CONCLUSION: while neuro-evolution presents a transformative approach for designing control systems in autonomous applications, it is essential to recognize the significant challenges and limitations that accompany this methodology. The computational complexity inherent in neuro-evolution raises concerns regarding resource requirements and training durations, particularly in environments characterized by high-dimensional state spaces and dynamic conditions. This complexity can hinder the practical deployment of neuro-evolutionary algorithms, especially in applications where real-time performance is critical.

Moreover, convergence issues pose another significant challenge. The risk of premature convergence can limit the ability of evolutionary algorithms to explore diverse solutions, potentially resulting in suboptimal performance. To mitigate this, ongoing research is required to develop robust strategies that enhance solution diversity and ensure convergence toward optimal or near-optimal control policies.

Overfitting remains a pressing concern in machine learning, including neuro-evolution. The balance between model complexity and generalization capabilities must be carefully managed to avoid models that perform well on training data but fail to adapt to new, unseen scenarios. Implementing effective validation techniques and regularization methods will be crucial in addressing this limitation.

Interpretability of neuro-evolutionary models also stands as a barrier to wider acceptance, especially in safety-critical domains where understanding the decision-making process is paramount. Developing frameworks that enhance model transparency and provide insights into how decisions are made will be essential in building trust among users and stakeholders.

Lastly, the transition from simulation to real-world implementation is fraught with challenges. Neuro-evolutionary algorithms must be rigorously tested and adapted to handle the complexities of real-world environments, which can differ significantly from simulated scenarios. This may involve incorporating robustness checks, environment modeling, and adaptive mechanisms to ensure reliable performance in practical applications.

ACKNOWLEDGMENT:

We would like to express our sincere gratitude to all those who contributed to the completion of this research on autonomous systems control design using neuro-evolution. First and foremost, we extend our appreciation to our academic advisors and mentors for their invaluable guidance and support throughout the research process. Their expertise and insights have been instrumental in shaping our understanding of neuro-evolution and its applications.

We also wish to thank our peers and colleagues for their collaboration and constructive feedback, which helped refine our methodologies and enhance the overall quality of our work. Special thanks to the research groups and institutions that provided access to datasets and resources essential for this study.

Furthermore, we acknowledge the contributions of the broader academic community and researchers whose prior work laid the foundation for our exploration of neuro-evolutionary methods in autonomous systems. Their dedication to advancing this field has inspired us and provided critical context for our findings.

Lastly, we appreciate the support of our families and friends, who have encouraged us throughout this journey. Their understanding and motivation have been a source of strength, enabling us to persevere through challenges and celebrate our achievements in this research endeavor.

REFERENCES:

[1] Y. Chauvin and D. E. Rumelhart, “Backpropagation: Theory, Architectures, and Applications”, Lawrence Erlbaum Associates, Publishers, Hillsdale, New Jersey, 576 p, 1995.

[2] T. P. Lillicrap, A. Santoro, L. Marris et al.: “Backpropagation and the brain”, Nat Rev Neurosci 21, pp. 335–346, 2020, [online]. Available at: https://doi.org/10.1038/s41583-020-0277-3.

[3] R. S. Sutton and A. G. Barto, “Reinforcement Learning, second edition: An Introduction”, MIT Press, ISBN: 978-0-262-35270-3, 552 p., 2018.

[4] B. Zoph, and Q. V. Le, “Neural Architecture Search with Reinforcement Learning”,2017, [online]. Available at: https://arxiv.org/abs/1611.01578

[5] I. Sekaj, I. Kenický and F. Zúbek, ”Neuro-Evolution of ContinuousTime Dynamic Process Controllers”, MENDEL. 27 , 7-11, 2 Dec.

2021 [online]. Available at: https://doi.org/10.13164/mendel.2021.2.007.

[6] I. Sekaj, L. Cíferský and M. Hvozdík: “Neuro-Evolution of Mobile Robot Controller”, MENDEL. 25., 2019. pp. 39-42. [online]. Available at: 10.13164/mendel.2019.1.039.

[7] A. Melichar.: “Autonomous Vehicle Control Design Using Neuroevolution”, Bachelor thesis, FEI STU Bratislava, 2022

[8] M. Chýlik: “Neuro-evolution of Autonomous Vehicle”, Diploma thesis, FEI STU Bratislava, 2022

[9] V. Kvasnička, J. Pospíchal and P. Tiňo: “Evolučnéalgoritmy”, STU, Bratislava, 2000 (in slovak)

[10] I. Sekaj: “Evolučnévýpočtya ich využitie v praxi”, iris, Bratislava. ISBN 80-89018-87-4, 158 p., 2005 (in slovak)

[11] A. E. Eiben and J. E. Smith : “Introduction to Evolutionary Computing”, Springer, 2003 16/j.medi.2020.05.008