Advancing Autonomous Vehicle Navigation through Hybrid Fuzzy-Neural Network Training Systems

Abstract:

In the evolving realm of autonomous vehicle navigation, the integration of fuzzy logic and neural networks presents a formidable challenge, particularly in the context of real-time, on-the-fly neural network training. This paper addresses the gap in dynamic and adaptable training methods necessary for navigating unpredictable environments with limited computational resources. The primary objective of our study is to empirically validate a hybrid training approach that combines fuzzy logic with back-propagation learning algorithms, aiming to optimize neural network performance under hardware constraints. Our methodology leverages a fuzzy logic trainer to provide initial training sets dynamically, which guide the neural network in adjusting its weights in real time, thus facilitating adaptive learning during navigation tasks. The findings reveal that this integrated approach not only enhances the learning efficiency of neural networks but also significantly improves navigation accuracy in real-time scenarios. These advancements contribute to the field by demonstrating the feasibility of deploying more adaptable and robust autonomous navigation systems, potentially expanding their application in more diverse and challenging environments.

Keywords: Autonomous Vehicle; Navigation; Fuzzy Logic; Neural Network; Optimization.

1. Introduction

Crafting an autonomous vehicle apt for navigating uncertain terrains remains an intricate puzzle, laden with multifaceted challenges such as adept path planning and nimble obstacle evasion. Given the intricate dynamics governing vehicle control juxtaposed with the unpredictable nature of uncharted territories, many theoretical navigation strategies find their application restrained. The sphere of intelligent solutions has witnessed the proliferation of methods harnessing the prowess of fuzzy logic and neural networks, targeting the conundrums of autonomous exploration.

The research problem at the heart of this study is the effective integration of fuzzy logic with neural network training in real-time scenarios for autonomous vehicle navigation. Despite significant advancements in both fields, autonomous systems still struggle with dynamic and unpredictable environments when limited computational resources are available. This paper aims to address these challenges by proposing a hybrid training methodology that enhances both the adaptability and efficiency of neural networks in real-world navigation tasks.

Systems rooted in fuzzy logic, as evidenced in references [1-8], employ a semantically rich rule base epitomized by IF {antecedent} THEN {consequent} rule structures. These guide vehicle actions steered by fuzzified sensory interpretations. Remarkably, such systems circumvent the intricate nuances of comprehensive vehicle dynamic modeling and yet endow control mechanisms resilient to sensory discrepancies. However, the static essence of their fuzzy rule architecture curtails adaptability, rendering them less equipped to counter unforeseen environmental challenges.

In juxtaposition, neural network-centric methods dismantle this adaptability barricade, empowered by their intrinsic learning algorithms. The conceptual synthesis of fuzzy-neural networks, as

illustrated in studies [9-16], amalgamates the merits of both paradigms, laying the groundwork for a potentially more robust autonomous navigation apparatus. Yet, even as fuzzy-neural networks emanate promise as a beacon for navigating the unknown, tangible impediments hinder their ubiquitous embrace in real-world applications.

The universe of neural networks often grapples with the twin challenges of curating pertinent, actionable training datasets and deploying real-time learning algorithms. While a diverse array of learning algorithms, as showcased in references [17-20], [21-23], and [24-27], have been introduced to train neural networks, their application is constrained, especially in scenarios demanding rapid vehicular responses to unforeseen obstacles with scant online training data. Independent of the algorithmic blueprint, the herculean task of sculpting an actionable training dataset remains. It is an arduous endeavor for specialists to create the multitudinous input-output data vectors crucial for training the network. While there are methodologies, as highlighted in [28-30] and [30-33], to curate impactful training data, their generalizability across varied scenarios remains challenging.

A unique remedy to this training data conundrum emerges in the form of a fuzzy logic trainer that offers the requisite data for training the vehicle's neural network, as illustrated in reference [34-35]. Herein, a vehicle steered by a neural network, initialized with randomized weights, is maneuvered within a racetrack. Concurrently, the fuzzy logic trainer churns out data, instructing the neural network via the classic back-propagation learning algorithm. This dynamic on-the-go training, with ideal outputs emanating from the fuzzy logic trainer, fine-tunes the neural network to closely mirror the performance of the trainer. The slight mathematical discrepancies between the fuzzy logic trainer and the neural network metamorphose into unique, and often superior, navigational performances by the latter. Additionally, the inherent noise in the training dataset potentially wards off the back-propagation algorithm from stagnating in local minima. Thus, crafting the perfect training dataset transforms into the art of designing the apt racetrack.

This study specifically addresses the challenge of real-time neural network training in autonomous vehicle navigation. Despite the advancements in fuzzy logic and neural network technologies, the integration of these systems in dynamic and uncharted environments remains a significant gap. Our research aims to bridge this by optimizing neural network training effectiveness and efficiency on limited hardware setups, a common constraint in practical autonomous navigation systems.

The primary objectives of this study are twofold: First, to empirically validate the feasibility of real-time neural network training for autonomous vehicle navigation using fuzzy logic. Second, to determine the optimal neural network configuration that balances training speed and navigation accuracy in real-time scenarios.

This paper dives into an experimental exploration of this innovative training paradigm. At its core is a robotic vehicle, dictated by an untrained neural network, governed by a fuzzy logic trainer. The vehicle's steering and speed are orchestrated by a feed-forward neural network, sporting a singular hidden layer. This study sets out with dual objectives: firstly, to empirically validate the feasibility of real-time neural network training for autonomous navigation, and secondly, to pinpoint the optimal neuron count for the hidden layer tailored to the experimental design. The neuron selection mechanism, driven more by empirical observations than theoretical musings, possesses a versatility extending beyond mere autonomous navigation. Furthermore, the paper sheds light on the implications of adding more neurons to the hidden layer concerning training efficacy and driving velocity. This real-time training blueprint, proven effective on this experimental platform, anticipates scalability to

accommodate evolving system complexities. Thus, the insights garnered are poised to be relevant to real-time training ecosystems encompassing diverse neural network architectures and hardware paradigms.

Autonomous vehicle navigation remains a formidable challenge in the field of robotics, particularly in real-time adaptive training of neural networks within dynamic and unpredictable environments. The primary aim of this study is to address these challenges by developing a hybrid system that integrates fuzzy logic with neural networks to optimize real-time training on constrained hardware platforms. This research is significant as it explores a novel approach to enhance the learning capabilities of autonomous vehicles, which is crucial for navigating complex environments. By achieving this, we contribute to the broader goal of deploying autonomous vehicles in more varied and unstructured settings than currently possible.

Having established the context and significance of integrating fuzzy logic with neural network training for autonomous navigation, we now explore how these technologies have been applied historically, highlighting the gap our study aims to fill.

2. Brief Previous work Discussion

Introduction: In recent years, the realms of autonomous vehicle navigation have witnessed a resurgence in interest, owing to technological advancements and the quest for more efficient, real-time solutions. Central to this discourse is the integration of neural networks and fuzzy logic systems, both robust computational models, designed to mimic human thinking and decision-making.

Neural Networks in Autonomous Navigation: Neural networks, specifically feed-forward neural networks, have been a mainstay in the field of machine learning and have demonstrated their prowess in numerous applications, including autonomous navigation. Their architecture, which mirrors the human neural structure, allows them to recognize patterns, learn from data, and make decisions. While their efficacy is recognized, the key challenge lies in training these networks efficiently to respond in real-time scenarios, especially when the environment is unpredictable, such as uncharted racetracks.

Fuzzy Logic Systems: Fuzzy logic systems operate on the principle of handling degrees of uncertainty, rather than absolute truths. This characteristic makes them particularly suited for complex decision-making tasks where conventional logic systems might falter. In autonomous navigation, fuzzy logic controllers (FLCs) aid in translating these degrees of uncertainties into actionable control decisions. While FLCs have been successfully deployed in several autonomous systems, their potential as trainers for neural networks has been an area of latent exploration.

Real-time Training Challenges: Real-time training remains a coveted milestone for autonomous vehicle systems. The challenges include but are not limited to computational limitations, ensuring initial feasibility, optimizing error rates, and scaling neural networks to align with specific tasks. Prior research primarily revolved around optimizing post-training performance, with limited emphasis on the training phase itself, especially in real-world, dynamic environments.

Impact of Neural Network Size: The relationship between the size of a neural network and its performance has been a recurrent theme. Previous studies have underscored that increasing the number of neurons could lead to enhanced performance. However, the challenge lies in balancing performance enhancements against computational costs, especially in applications demanding real-time responses.

The Uniqueness and Novelty of the Current Study: The reviewed study presents a distinct approach by amalgamating fuzzy logic trainers with neural networks in the context of real-time

autonomous vehicle navigation. While individual merits of neural networks and fuzzy logic have been extolled in the literature, their synergistic application is relatively nascent.

The root of the study's novelty lies in its methodology:

- It showcases the efficacy of a minimalistic fuzzy logic implementation in training neural networks, even under significant hardware constraints.
- Instead of a traditional approach that emphasizes reducing final error rates, this study shifts its focus on the feasibility of initial training.
- The establishment of a comprehensive criterion for determining the optimal size of the neural network, factoring in real-world challenges and constraints, offers a more pragmatic approach than purely theoretical models.

Furthermore, the study's emphasis on real-world, dynamic environments, such as navigating unknown racetracks, offers valuable insights into the application-driven aspects of neural network optimization, as opposed to simulations that might not always replicate real-world intricacies.

In essence, while the individual components (neural networks, fuzzy logic, autonomous navigation) have been discussed extensively in literature, the present study's unique amalgamation and its approach towards real-time, practical application-driven optimization signify its novelty in the domain of autonomous navigation research.

Our review of existing literature reveals a fragmented landscape where studies focus either on fuzzy logic or neural networks in isolation. This paper bridges this gap by synthesizing insights from both domains, demonstrating how integrated approaches can surpass the limitations of singular systems. For instance, while neural networks offer adaptability through learning, they require extensive data and computational power. Fuzzy logic, conversely, provides robust decision-making with less data but lacks adaptability. By combining these approaches, our study seeks to harness the strengths of both, facilitating a more nuanced understanding and application in autonomous navigation.

The literature review underscores the need for innovative approaches that combine the robust decision-making capabilities of fuzzy logic with the adaptive learning of neural networks. This need brings us to the methodological framework of our study.

3. Experimental Platform

In this section, we detail our integrated methodology, which harnesses both fuzzy logic and neural networks to train autonomous vehicles in real-time, a method poised to overcome the limitations discussed in the previous sections.

A state-of-the-art robot vehicle was engineered for adept navigation on indoor racetracks. Equipped with a premier Hokuyo URG-04LX Lidar, it channeled precise sensory data to a pair of Atmega1280-based microcontrollers. This duo was meticulously connected: one to a singular steering servo motor and the other to a Sabretooth 2x10 motor controller, orchestrating the vehicle's movements. Both the fuzzy logic and neural network blueprints were sculpted with a prime focus on

real-time autonomous navigation, even under the computational constraints levied by the 8-bit microcontrollers.

The implementation of the fuzzy logic trainer and neural networks spanned the twin microcontrollers. The first microcontroller served as the nexus to the sensor, ingeniously utilizing the fuzzy logic trainer and neural networks to craft control instructions for both steering angle and velocity. These directives, originated from the primary microcontroller, were relayed to its counterpart, which seamlessly interfaced with the steering servo motor and the velocity motor. This bifurcated microcontroller architecture was strategized to foster scalability, especially with potential integrations of additional sensors, thanks to a surplus of processing prowess and I/O ports. The confluence of the fuzzy logic trainer and neural networks on a singular microcontroller was predominantly for streamlined data transmission, with negligible performance variations had the neural network resided on the secondary microcontroller.

Our methodology encompasses a unique integration of fuzzy logic with neural network training through back-propagation. The fuzzy logic trainer provides initial training data dynamically, guiding the neural network in adjusting its weights in real-time. This setup leverages the strengths of fuzzy logic's handling of uncertainties and the adaptive learning capabilities of neural networks. The core innovation lies in using fuzzy logic not only for control but also as a dynamic trainer, which constantly adapts the neural network during navigation tasks.

The crux of the platform's performance challenge pivoted around its reflexes. Although the Lidar consistently delivered data at a brisk rate of 10 Hz, the vehicle's aggregate response time, factoring in computation durations, lingered between 5-7 Hz. To truly embody real-time navigation, it was imperative for the vehicle to discern and react to impediments at a pace congruent to its velocity. Thus, navigational speeds in the ensuing experiments were deliberately tapered to guarantee the vehicle's astute obstacle detection and reaction capability. Amplifying the response rate would inevitably have propelled both the average cruising speed and the pace during the neural network's training phase.

Our methodology integrates fuzzy logic with a neural network via a novel training protocol that uses fuzzy logic outputs to dynamically adjust the neural network's weights during real-time operation. This setup allows the system to adapt swiftly to changes in the environment, enhancing navigation accuracy. The dual-layered approach—initial fuzzy logic inference followed by neural adaptation through back-propagation—ensures that our system not only learns from its immediate context but also adjusts its learning process based on ongoing performance feedback.

4. Fuzzy Logic Trainer

The underpinning navigation principle for our fuzzy logic trainer draws inspiration from the electrostatic potential fields-based navigation paradigm. This method analogizes both the vehicle and proximate obstacles as entities radiating an electrostatic charge. The direction indicative of the vehicle's movement is illustrated by the negative gradient of this synthesized electrostatic field.

In a parallel vein, our fuzzy logic trainer discerns distinct forces which aid in sculpting the terrain's profile. Comprising of an assembly of four forces, these energies serve as the precursor to the fuzzy logic speed formulation, which subsequently, after computing a singular cumulative force, lays the foundation for the steering angle calculation via fuzzy logic. The lifeline for the fuzzy logic trainer

remains the sensory data, generously provided by the Lidar. This data-driven control tactic perfectly aligns with the prerequisites needed to adeptly train a neural network. Vehicle kinematics are also factored into the equation, mainly through the meticulous crafting of the fuzzy logic membership functions and its rule base. It's worth noting that while the augmentation of the fuzzy logic's navigational prowess can be achieved with more sensor integrations or advanced filtering, such enhancements might inflate computational demands and intricacy without unequivocally refining the neural network's training dynamics.

Upon data receipt from the Lidar, a preliminary data treatment is executed. With its robust 180° field of vision, the Lidar bequeaths a dataset encompassing 256 points. Each point narrates the distance of identified obstacles, uniformly spaced at nearly 0.70° angular intervals. This dataset undergoes a refinement process to filter out anomalies, setting a floor value at 10 cm to sidestep exceedingly low readings and a ceiling at 70 cm to circumvent reacting to distant, non-threatening obstacles.

The polished dataset then metamorphoses into four pivotal forces, each epitomizing the obstacle closeness within the dedicated 45° quadrants (namely Left-Extreme, Left, Right, and Right-Extreme). The transformational process is orchestrated by aggregating the inverse of each culled data point confined within the designated quadrant. This process can be mathematically delineated by the following equation (Where d_i is a data point):

$$Force = \sum_i \frac{1}{d_i}$$

Derived force values were meticulously refined by multiplying them with an empirically determined constant, epitomizing the Lidar's desired sensitivity to ambient structures. This normalization strategy constrained force values within a spectrum of 0 to 255. Interpreting these values, a score of 0 signifies a quadrant devoid of any proximal obstacles, whereas a score nearing 255 flags the proximity of a potential impediment.

This systematic scaling streamlines subsequent development phases, particularly in crafting the fuzzy logic membership functions. These recalibrated force quadrants—Left-Extreme (LE), Left (L), Right (R), and Right-Extreme (RE)—then become integral inputs for the fuzzy speed determinants. A subsequent analysis amalgamates these four distinct forces into a singular representative force, optimizing it for steering angle calculations via fuzzy logic.

For both the neural network and the fuzzy logic speed computation, these four forces act as pivotal inputs. The further enhancement to determine the fuzzy logic's steering angle, however, demands the conversion of these quartet inputs into a solitary force, emblematic of the most dominant obstacle's trajectory. This synthesis is accomplished by gauging the differential between the Left-Side (LS) and Right-Side (RS) forces. Crafting the LS force follows a distinct logical path, detailed as follows:

$$LS\,Force = \begin{cases} \frac{LE+L}{2} & if\ LE \geq \alpha\ and\ L \geq \alpha \\ \max(LE, L) & if\ LE < \alpha\ or\ L < \alpha \end{cases}$$

In determining the driving force dynamics, α acted as a foundational threshold value. Crafting the Right-Side (RS) force mirrored the logic employed for the Left-Side (LS), albeit factoring in Right-Extreme (RE) and Right (R) forces in lieu of Left-Extreme (LE) and Left (L) forces. The consequential net force—derived from the differential between LS and RS forces—depicted a holistic view of obstacle distribution. A pronounced positive net force denoted a more significant obstacle alignment to the left, while its negative counterpart signified a rightward bias.

The magnitude of this consolidated force encapsulated not only the obstacle's proximity but also its relative positioning vis-a-vis the vehicle—be it clearly oriented to one side or directly ahead. This nuanced force calibration originated from the data point averaging system and the chosen methodology of leveraging a single differential force over individualized force components.

Transitioning to the fuzzy logic computations, exponential curves were harnessed for their membership functions, resonating more aptly with the nonlinear dynamics of the input forces than their triangular counterparts. For the crucial defuzzification stage, the centroid method was the method of choice, promising a more holistic coverage of the decision arena compared to alternate strategies, such as the mean of maximum approach. Yet, the computational heft of the centroid method posed a challenge for the experimental setup. To circumvent this while preserving efficiency and curtailing numerical aberrations, the method's precision was strategically curtailed, reflecting in the utilization of fewer points to represent the active functions within the computational sums.

4.1 Fuzzy Steering Angle Controller

The fuzzy rule base governing the steering angle can be comprehensively delineated as follows:

1) If (force) is NEGATIVE, go LEFT
2) If (force) is ZERO, go STRAIGHT
3) If (force) is POSITIVE, go RIGHT

The fuzzy variables in the input domain are depicted as NEGATIVE, ZERO, and POSITIVE, as illustrated in Fig. 1. Meanwhile, the output domain is defined by the fuzzy variables LEFT, STRAIGHT, and RIGHT, as visualized in Fig. 2. A significant streamlining of the rule architecture was achieved by leveraging a solitary net force as the input, instead of the earlier quadruple force system. Experimentally, opting for the comprehensive four-force model to craft the fuzzy rule base didn't offer discernible enhancements in performance. Thus, the more focused, net force-driven approach was both efficient and equally effective.

4.2 Fuzzy Speed Controller

For the speed controller, the fuzzy rule base is detailed as follows:

1) If (LE) AND (L) AND (RE) AND (R) are LOW, go FAST
2) If (LE) AND (RE) are HIGH AND If (L) AND (R) are LOW, go MEDIUM
3) If (LE) AND (L) AND (RE) AND (R) are HIGH, go SLOW

The input space is characterized by the fuzzy variables LOW and HIGH, as visually represented in Fig. 3. Conversely, the output spectrum incorporates the fuzzy variables FAST, MEDIUM, and SLOW, showcased in Fig. 4. The intersection of these fuzzy variables was achieved using the 'minimum' operator, serving as the fuzzy AND gate.

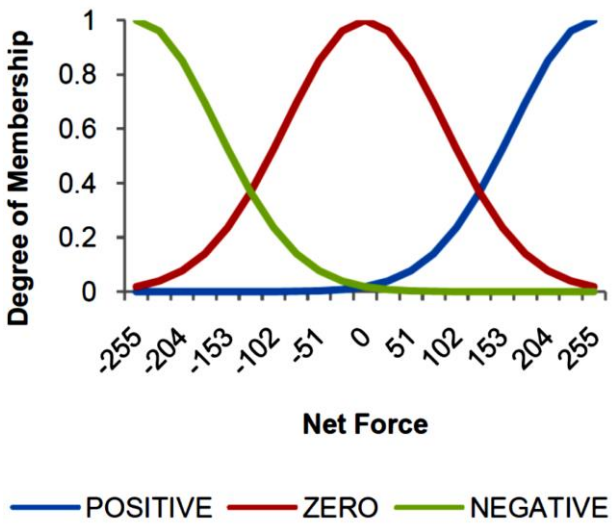

Fig. 1. Membership Functions for Steering Angle Input

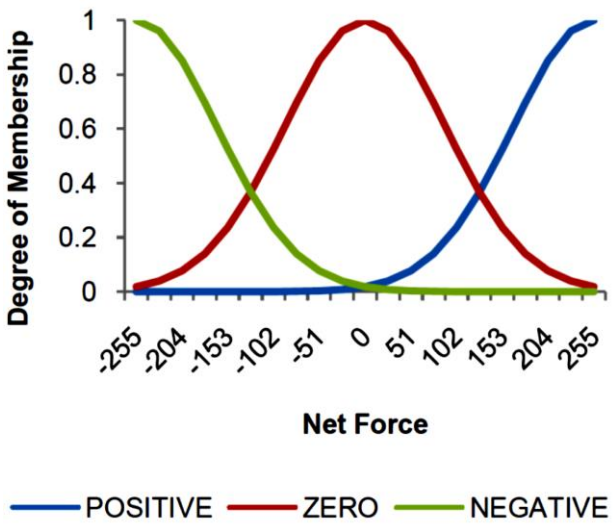

Fig. 2. Membership Functions for Steering Angle Output

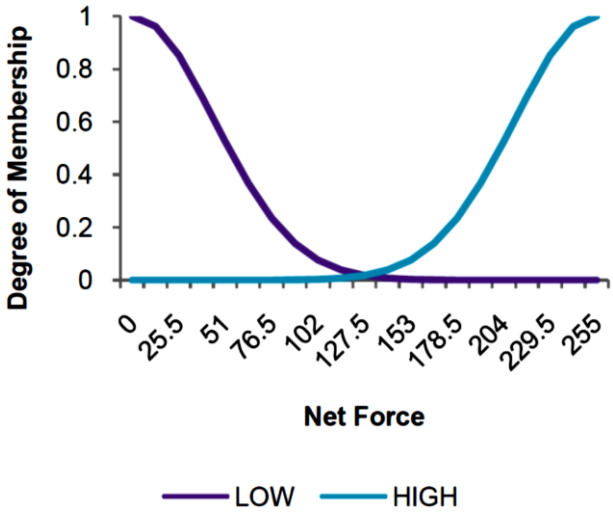

Fig. 3. Membership Functions for Velocity Input

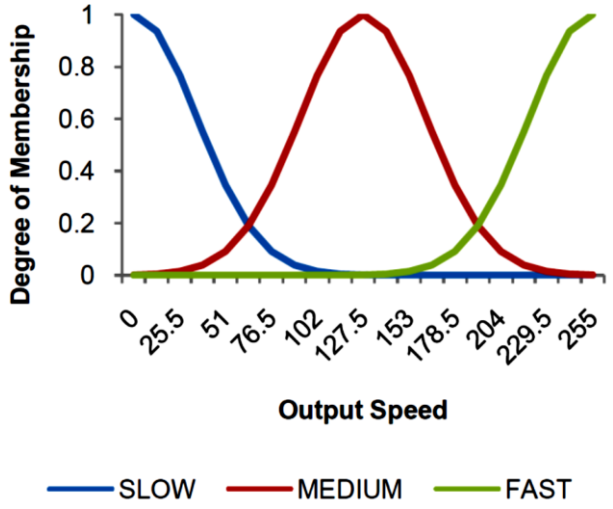

Fig. 4. Membership Functions for Velocity Output

The governing principle was straightforward: In denser obstacle scenarios, the vehicle should adopt a cautious pace, allowing ample time for responsive navigation—essentially, to decelerate during sharper turns or when faced with a multitude of obstructions. The vehicle's pace oscillated between predefined maximum and minimum thresholds, with the fuzzy speed metric determining the precise speed within these bounds. The framework of the fuzzy logic trainer was tailored for maneuvering tracks adorned with gentle curves and rudimentary barriers, leading to the creation of a harmonized fuzzy rule base.

While deriving the speed metric, both the vehicle's kinematics and its current velocity were intentionally omitted, introducing a layer of simplicity that scarcely affected the vehicle's functional

efficacy. To counterbalance the vehicle's delayed reflexes, an upper cap on speed was set. Simultaneously, a raised floor limit on speed ensured effortless friction combat and speed sustenance during turns. Given these speed constraints, the scope for drastic speed alterations was minimal, hence even basic or not entirely optimal speed derivations didn't notably dent the performance. Such calibrated constraints, coupled with the concise set of fuzzy rules, empowered the vehicle to adeptly navigate the experimental tracks.

5. Neural Network

Artificial Neural Networks (ANNs) serve as sophisticated non-linear statistical models adept at recognizing and emulating patterns bridging input and output data. Characterized by their dynamic learning capacity, ANNs discern patterns from a distinctive set of data points using learning algorithms. Conceptually, ANNs are visualized as interconnected neurons, facilitated by specific weights. These neurons, upon activation, produce an output ranging from 0 to 1. Each neuron's input undergoes weight multiplication, and this matrix, influenced by the activation function, dictates the neuron's output. The essence of learning resides in the strategic adaptation of the neural network's weights.

Optimization Protocol: To address the challenge of finding the most suitable neural network configuration for autonomous vehicle navigation, our methodology introduces a detailed optimization protocol. This protocol iteratively tests various neural network architectures, adjusting neuron counts and layer configurations to find the optimal balance between computational efficiency and navigation accuracy. The protocol considers specific problem contexts and hardware constraints, using a combination of simulation and real-time training on physical platforms to validate each configuration.

For this project, a rudimentary two-layer feed-forward neural network, depicted in Fig. 5, was entrusted with the autonomous vehicle's controls. This structure is amongst the most fundamental neural topologies fitting for autonomous navigation.

The employed feed-forward neural network orchestrates the vehicle's operations. Receiving the four force metrics (LE, L, R, and RE) as inputs, it generates steering angle and speed as outputs. Comprising an output layer with two neurons and a concealed layer housing 1-10 neurons, its simplicity is affirmed by the single hidden layer—adequate for mirroring the fuzzy logic trainer with unparalleled precision [36-41]. Neuron activations were governed by the sigmoid function, and their state discerned from the summative product of weights and inputs. The classic back-propagation technique, devoid of ancillary enhancements like momentum terms, undertook neural network training. After meticulous simulations, an optimal global learning rate bracketing 0.5-1.5 was chosen, settling on a definitive rate of 1 for all neural networks under test.

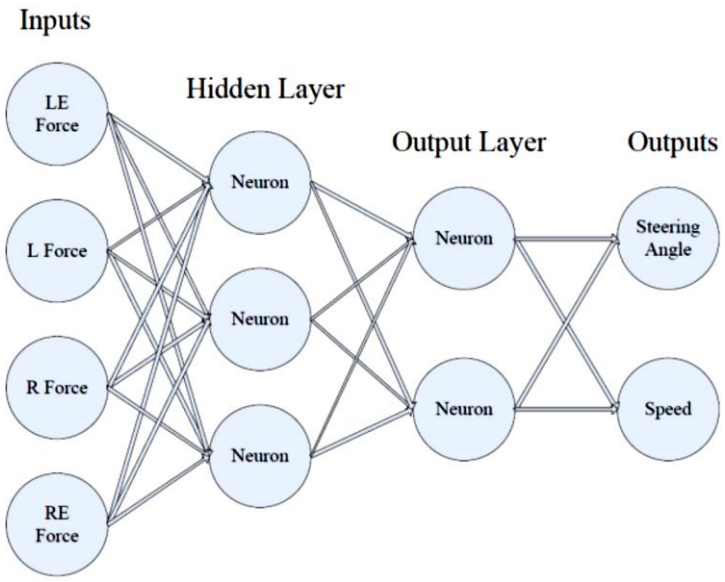

Fig. 5. Representative Feedforward Neural Network Design for Automotive Navigation

The collaborative workflow between the fuzzy logic trainer and the neural network unfolded as follows:

1. **Data Intake:** The quartet of force metrics was dispatched to both the neural network and the fuzzy logic trainer. Post normalization, these values bolstered the neural network's learning efficacy.

2. **Fuzzy Logic Determination:** The fuzzy logic trainer, based on its input, deduced the ideal steering angle and speed, which subsequently trained the neural network.

3. **Forward Propagation:** This algorithm ascertained the steering angle and speed as discerned by the neural network.

4. **Error Correction via Back-propagation:** Outputs from both systems fed into the error back-propagation algorithm, adjusting the neural network's weights to curtail error:

$$e_{steering} = e_{fuzzy\ steering} - e_{neural\ network}$$

$$e_{speed} = e_{fuzzy\ speed} - e_{neural\ network}$$

5. **Iterative Refinement:** Steps 3 and 4 were reiterated twice, culminating in three complete back-propagation cycles per input force set. This optimized iteration count struck a balance between resource efficiency and experimental performance. Superfluous iterations would only escalate processing demands without proportionate improvements in training.

6. **Final Deployment:** The rectified neural network output steered the vehicle, ensuring its training bore relevance to its own performance rather than just relying on the fuzzy logic navigation. This synergistic method created a feedback mechanism: the fuzzy logic trainer educated the neural network based on vehicular surroundings and the divergence between the neural and fuzzy

trainer outputs. The dynamic training approach was empirically validated as the navigational strategy evolved in real-time.

6. Results and Discussion

Before delving into real-world racetrack experiments, a rigorous simulation was orchestrated to decipher the epochs—a lap around the racetrack, necessary for the neural network's convergence to the least possible error threshold. Additionally, these simulations gauged the error patterns manifested during the learning journey. The simulation framework encompassed 10 distinct neural networks, each distinguished by the neuron count in its hidden layer. These were subjected to 256 unique training vectors. With each input force residing within the 0-255 spectrum, the training vectors ensured comprehensive coverage of all potential input permutations. The mean squared error, calibrated against the fuzzy logic output and neural network output across epochs, was integral to both the simulated and experimental scenarios. This uniformity in metric ensured the simulations' efficacy as a precursor to real-world experiments.

As depicted in Fig. 6, the simulation unveiled the neural networks' propensity to stabilize at a minimum error within a 10-epoch window. Interestingly, more expansive neural networks exhibited both reduced inception error and quicker convergence to minimized post-training error. Conversely, networks with fewer neurons in their hidden layer showcased expedited convergence rates. These findings established a foundational hypothesis: a cap of 10 epochs would suffice for training each neural network in subsequent experiments.

Transitioning to real-world applications, two distinct racetracks were conceptualized to rigorously evaluate the neural networks' learning acumen. The first—a simplistic design showcased in Fig. 7, aimed to validate real-time neural network training feasibility. In contrast, the latter, a more intricate layout displayed in Fig. 8, was crafted to affirm the consistency of experimental outcomes across varied terrains. Each racetrack contributed approximately 200-300 unique training vectors per epoch, mirroring the simulation's vector volume, hence ensuring parallelism between simulated predictions and real-world validations.

6.1 Experiment #1

Positioned at the starting line of Racetrack #1, the vehicle embarked on a 10-epoch journey, piloted by an untouched neural network. Networks, differentiated by the neuron count—2, 4, 6, 8, and 10 in their hidden layers, were put to the test. This exercise aimed to not only ascertain the real-time trainability of a neural network in a dynamic environment but also to contrast simulated predictions against tangible outcomes.

Fig. 9 paints a vivid picture: each neural network, guided by the fuzzy logic trainer, astutely learned the art of racetrack navigation. They seemingly gravitated towards similar error baselines, exhibiting convergence rates that mirrored those witnessed during simulations.

The tangible intrigue unfolded when, post the initial 5 epochs, the networks appeared to have attained full training. Leveraging this insight, the experimental baton was handed over to the neural network, post its 5-epoch training. This phase excluded the guiding hand of the fuzzy logic trainer, challenging the neural network to autonomously navigate Racetrack #1.

Results, captured in Fig. 10, stood testament to the efficacy of the training process. Post-training error margins across neural networks, irrespective of their size, were strikingly uniform. A deeper dive into the performance of a network with 6 neurons, as illustrated in Fig. 11, showcased minimal

divergence between trajectories steered by the fuzzy logic trainer and the neural network. What's more, the post-training neural network path exuded finesse, tracing a steadier course closer to the racetrack's median, in stark contrast to its earlier erratic runs. Remarkably, when pitted against the seasoned fuzzy logic trainer, the neural networks' performances were neck and neck, underscoring the profound impact of training.

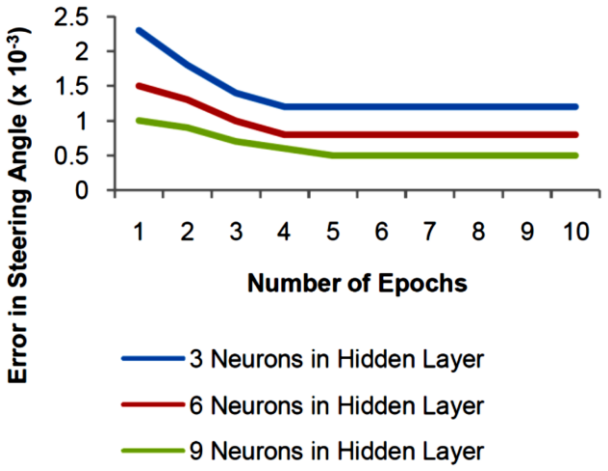

Fig. 6. Steering Angle Error Analysis in Simulation

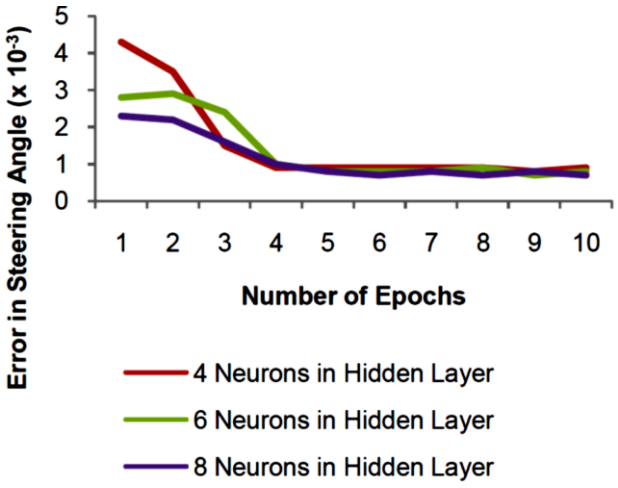

Fig. 9. Steering Angle Error Distribution Across 10 Epochs

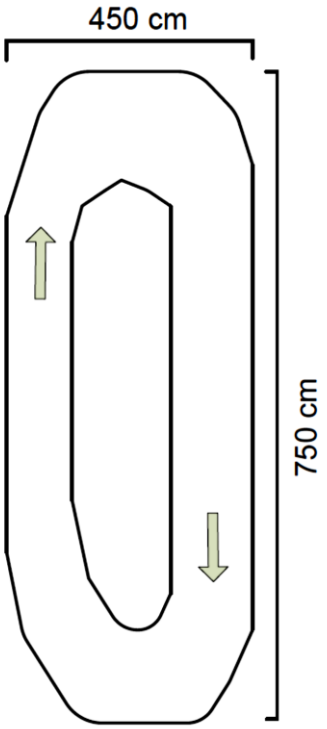

Fig. 7. Illustration of Race Circuit No. 1

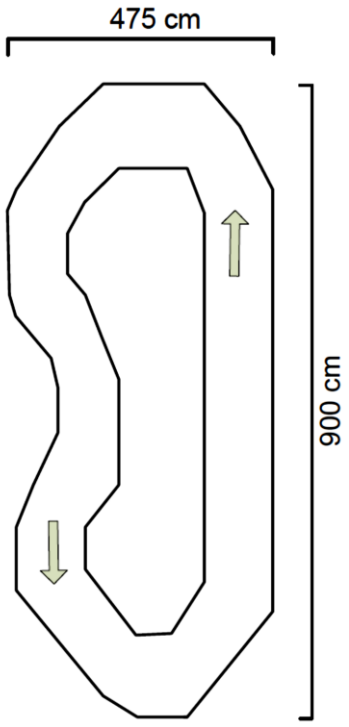

Fig. 8. Illustration of Race Circuit No. 2

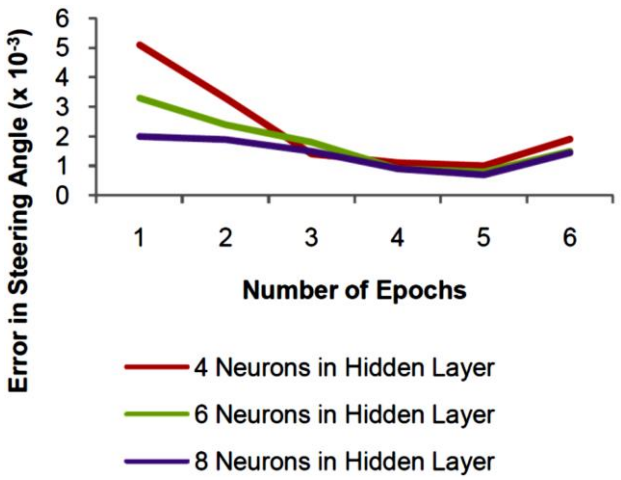

Fig. 10. Steering Angle Error Analysis Over 5 Epochs

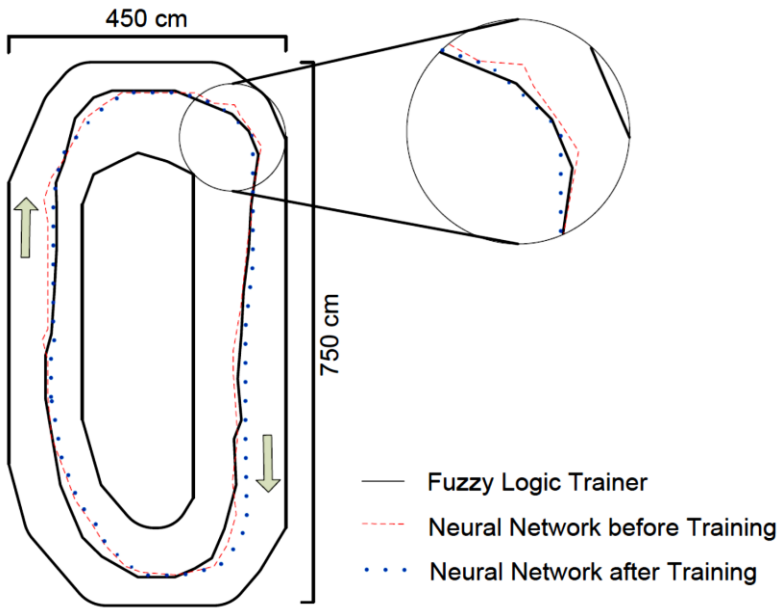

Fig. 11. Hypothetical Vehicle Trajectory in Second Experiment

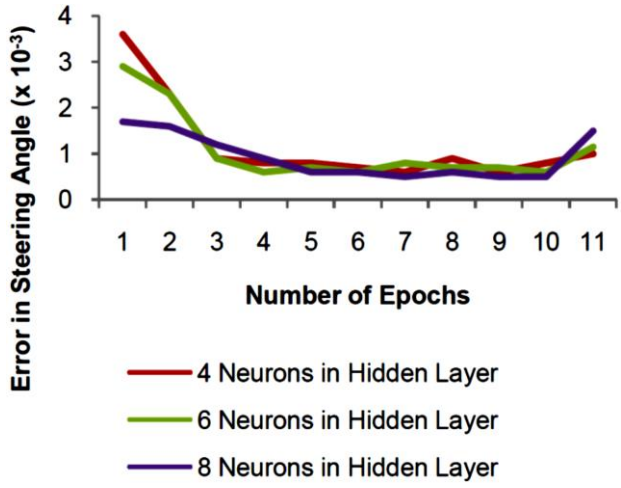

Fig. 12. Steering Angle Error During Second Experimental Run

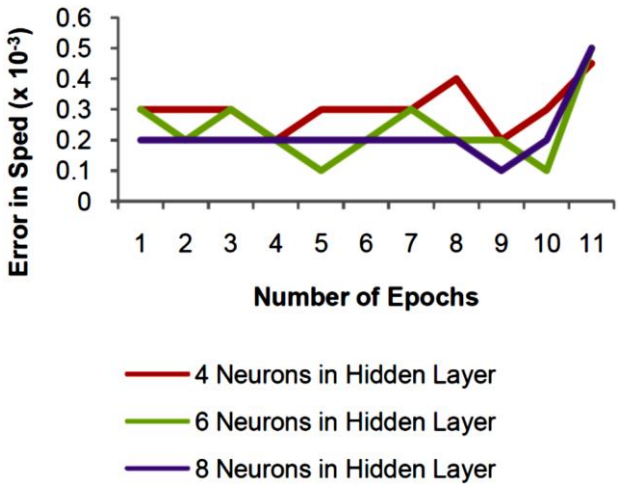

Fig. 13. Speed Error Analysis in Second Experiment

6.2 Experiment #2

To discern the optimal dimensions of a neural network, it became imperative to understand how the intricacy of a racetrack affects the neural networks' learning prowess. The autonomous vehicle, deployed on Racetrack #2, underwent 10 epochs under the guidance of an uninitiated neural network. Following this training, an independent epoch was embarked upon, with the vehicle solely maneuvered by the trained neural network, devoid of the fuzzy logic controller's input. Analyzing these outcomes against prior tests enabled the drawing of broader inferences about neural network dimensions.

The derived data posited that the racetrack's intricacy bore minimal influence on training behaviors, as illustrated in Fig. 12 and Fig. 13. The introduction of extra neurons in the hidden layer curtailed the error in the initial epoch. Yet, this also extended the epochs needed for the network to settle at its minimum error. Remarkably, post-training performance across various neural networks appeared homogenous, suggesting limited advantages in enlarging the network to further reduce error.

Specifically, steering angle error stabilized within roughly 5 epoch cycles, while speed error attained higher accuracy within the inaugural epoch. This can be attributed to the architecture of the fuzzy speed rules and the racetrack's design. Consequently, the back-propagation algorithm, aiming to curtail the overarching error across both speed and steering outputs, induced variations in speed error while focusing on rectifying the steering angle error. Segregating these outputs into distinct neural networks might have circumvented such pronounced error oscillations.

Interestingly, the vehicle's velocity during neural training was approximately half of that achieved using only the fuzzy logic approach. However, post-training speeds, when employing the neural network, mirrored those attained with the fuzzy logic trainer. It's noteworthy that the neural network, being computationally leaner than its fuzzy logic counterpart, offered a marginally better reaction time, potentially allowing for an uptick in average vehicular speed.

Our findings reveal that the optimization protocol successfully identifies neural network configurations that perform best under different environmental complexities and hardware limitations. For instance, simpler network architectures yielded faster response times in less complex environments, whereas more sophisticated networks were necessary to navigate more challenging terrains effectively. This adaptability demonstrates the protocol’s capability to tailor neural network setups according to specific operational needs and constraints.

A comparative reanalysis of Experiment 1 and Experiment 2 highlights the robustness of our methodology across different scenarios. In Experiment 1, conducted on a simpler racetrack, the neural network quickly adapted to the environment, demonstrating rapid error reduction within five epochs. Conversely, Experiment 2, featuring a more complex track, required more epochs to achieve similar error rates, underscoring the influence of environmental complexity on learning efficiency. This comparison not only validates our model's efficacy but also illustrates its scalability and adaptability to varying complexities in real-time navigation tasks.

7. Selection of Optimal Neural Network Size

Several factors contribute to the determination of the ideal neuron count within a neural network's hidden layer:

1. **Fidelity to Fuzzy Logic Trainer:** The hidden layer should proficiently emulate the fuzzy logic trainer. Experimental data suggested that this wasn't a dominant concern since diverse sizes of the hidden layer seemed to converge to comparable error levels.

2. **Computational Efficiency:** The computational demands of the neural network should not compromise its real-time applicability, either during the training phase or autonomous operation. All assessed configurations complied with this standard.

3. **Initial Training Precision:** During the training phase, the initial error should be sufficiently low to empower the neural network to proficiently navigate unfamiliar racetracks. While all assessed

models satisfied this, networks with more neurons in the hidden layer exhibited higher initial test reliability, whereas those with fewer neurons demonstrated occasional unpredictability.

4. **Rapid Learning Curve:** A minimal epoch count should suffice for the neural network to effectively adapt to a given racetrack. Such efficiency enhances the neural network's utility, especially since undergoing numerous training epochs, as seen in simulations, isn't feasible in real-world settings. This criterion suggests a predilection for smaller neural network configurations.

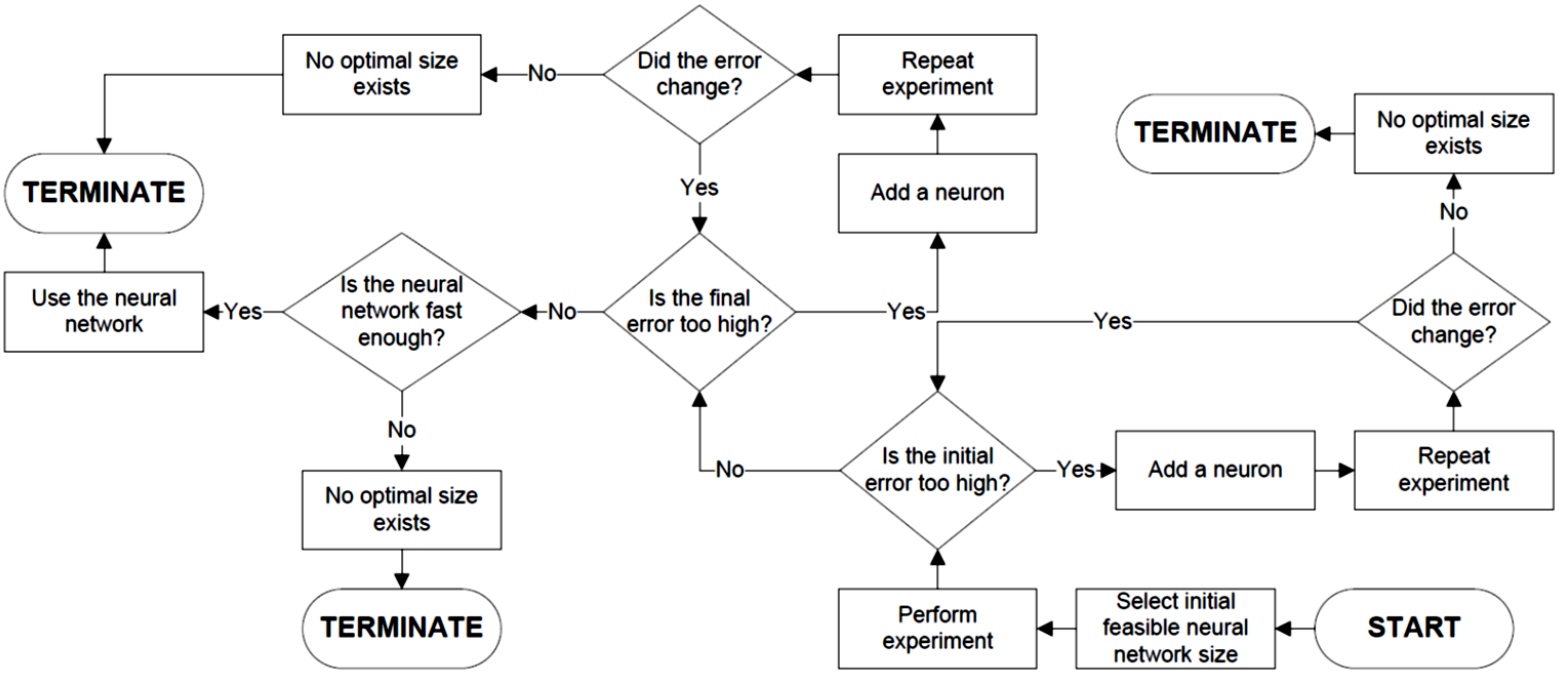

Fig. 14. Flowchart for Neural Network Dimension Determination

Building on these principles, an optimization framework was crafted, visualized in Fig. 14, aiming to pinpoint an optimal neural network dimension. Operating under the presumption of real-time training within a bounded epoch count in tangible environments, an introductory neural network size was posited. This size was calibrated to be as compact as feasible yet sufficiently expansive to navigate the challenge at hand. Each added neuron was hypothesized to diminish both the preliminary and concluding errors, echoing the patterns observed in trial networks. Upon identifying a neural network satisfying all criteria, the search would conclude, earmarking the configuration as optimal. The absence of such a configuration would signal the need for refining the learning methodology, reassessing the error bounds, or considering the computational constraints. Through this meticulous framework, a neural network with 4-6 neurons in its hidden layer emerged as the most balanced choice. Networks with less than 4 neurons exhibited elevated initial errors, while the computational overhead rendered configurations exceeding 6 neurons suboptimal. The precise dimensions were contingent on the racetrack complexities and the density of impediments.

5. Conclusion and Summary

The feasibility of employing a fuzzy logic trainer to train a neural network in real-time while directing a vehicle through an unfamiliar racetrack was conclusively established. Remarkably, even with a basic fuzzy logic framework and pronounced hardware limitations, the neural networks exhibited a commendable learning capacity, effectively navigating the tested scenarios. These findings highlight the potential for implementing real-time autonomous navigation without necessitating intricate control mechanisms or substantial computational prowess.

Furthermore, the research revealed that the neuron count within the hidden layer chiefly influenced the neural network's preliminary precision and its convergence speed towards minimal error levels. While augmenting the neural network size showcased enhanced results during simulations, tangible experimental enhancements remained marginal. Impressively, the neural networks exhibited competitive performance relative to the fuzzy logic trainer, even when subjected to a sparse set of training epochs. While these findings might not be universally applicable across diverse neural network and fuzzy logic configurations, the focal shift from ultimate error mitigation to initial feasibility signifies a departure from conventional neural network size determination paradigms.

Drawing from these insights, a methodical procedure was instituted to ascertain the optimal neuron count for the specific neural network in question. Tailored for application-driven contexts where traditional theoretical optimization techniques might fall short, this approach aims to provide practical guidance. For the tested scenario, the optimal neuron range within the hidden layer was discerned to be approximately 4-6. Although this methodology might not guarantee universally optimal outcomes across all neural network designs, its efficacy for similar feed-forward neural network challenges can offer pragmatic, near-optimal configurations.

The practical implications of our findings are significant for the field of autonomous vehicle navigation. By demonstrating that neural networks can be trained in real time on constrained hardware, we provide a scalable method that can be applied in diverse operational environments. This approach reduces the dependency on pre-trained models and allows autonomous systems to adapt to new environments swiftly. Furthermore, our study contributes to the ongoing discussion about the balance between computational overhead and real-time response capabilities in neural network training, advancing the deployment of intelligent autonomous vehicles in more dynamic and unpredictable settings.

The development of our detailed optimization protocol significantly advances the efficiency and effectiveness of autonomous vehicle navigation systems. By enabling tailored configurations of neural networks, our approach allows for enhanced adaptability and performance in diverse navigation scenarios. These contributions are pivotal for the ongoing evolution of autonomous vehicle technologies, potentially leading to more robust and reliable navigation solutions that can operate effectively across a broad spectrum of real-world conditions.

Disclaimer (Artificial intelligence)

Option 1:

Author(s) hereby declare that NO generative AI technologies such as Large Language Models (ChatGPT, COPILOT, etc.) and text-to-image generators have been used during the writing or editing of this manuscript.

Option 2:

Author(s) hereby declare that generative AI technologies such as Large Language Models, etc. have been used during the writing or editing of manuscripts. This explanation will include the name, version, model, and source of the generative AI technology and as well as all input prompts provided to the generative AI technology

Details of the AI usage are given below:

1.
2.
3.

6. Acknowledgements

The author is appreciative of the support given by King Fahd University of Petroleum & Minerals – Deanship of Research (Project: DF181030).

7. References

[1] Ishikawa, Shigeki. "A Method of Indoor Mobile Robot Navigation by Using Fuzzy Control." IEEE International Workshop on Robots and Systems, November 1991, pp. 1013-1018.

[2] Chen, Weihua & Zhang, Tie. (2017). An indoor mobile robot navigation technique using odometry and electronic compass. International Journal of Advanced Robotic Systems. 14. 172988141771164. 10.1177/1729881417711643.

[3] Fahima, Benyounes & NEMRA, Abdelkrim. (2021). Multispectral Visual Odometry Using SVSF for Mobile Robot Localization. Unmanned Systems. 10. 10.1142/S2301385022500157.

[4] Sabrina, Mohandsaidi & Mellah, Rabah & Fekik, Arezki & Azar, Ahmad Taher. (2022). Real-Time Fuzzy-PID for Mobile Robot Control and Vision-Based Obstacle Avoidance. International Journal of Service Science, Management, Engineering, and Technology. 13. 1-32. 10.4018/IJSSMET.304818.

[5] Reignier, Patrick. "Fuzzy logic techniques for mobile robot obstacle avoidance." Elsevier - Robotics and Autonomous Systems, vol. 12, 1994, pp. 143-153.

[6] Yusubov, Mahabbat & Sadigli, Ismayil. (2023). A FUZZY CONTROLLER FOR A MOBILE ROBOT WITH OBSTACLE AVOIDANCE. PIRETC-Proceeding of The International Research Education & Training Centre. 27. 150-155. 10.36962/PIRETC27062023-150.

[7] Singh, R & Bera, Tarun. (2019). Obstacle Avoidance of Mobile Robot using Fuzzy Logic and Hybrid Obstacle Avoidance Algorithm. IOP Conference Series: Materials Science and Engineering. 517. 012009. 10.1088/1757-899X/517/1/012009.

[8] Singh, Rajmeet & Bera, Tarun. (2018). Fuzzy Logic Controller for Obstacle Avoidance of Mobile Robot. International Journal of Nonlinear Sciences and Numerical Simulation. 20. 10.1515/ijnsns-2018-0038.

[9] Tsoukalas, L.H., Houstis, E.N., and Jones, G.V. "Neurofuzzy Motion Planners for Intelligent Robots." Journal of Intelligent and Robotic Systems, vol. 19, 1997, pp. 339-356.

[10] Jiwei, Han. (2023). Intelligent Motion Control Technology of Industrial Robot. 10.1007/978-981-99-1983-3_18.

[11] Fried, Inbar & Akulian, Jason & Alterovitz, Ron. (2022). A Clinical Dataset for the Evaluation of Motion Planners in Medical Applications. 10.48550/arXiv.2210.10834.

[12] Zhao, Yongyong & Wang, Jinghua & Cao, Guohua & Yuan, Yl & Yao, Xu & Qi, Luqiang. (2023). Intelligent Control of Multilegged Robot Smooth Motion: A Review. IEEE Access. PP. 1-1. 10.1109/ACCESS.2023.3304992.

[13] Mbede, Jean Bosco, et al. "Intelligent mobile manipulator navigation using adaptive neuro-fuzzy systems." Elsevier - Information Sciences, vol. 171, 2005, pp. 447-474.

[14] Karray, Amal & Njah, M. & Feki, Moez & Jallouli, Mohamed. (2016). Intelligent mobile manipulator navigation using hybrid adaptive-fuzzy controller. Computers & Electrical Engineering. 56. 10.1016/j.compeleceng.2016.09.007.

[15] Han, Jiangyi & Wang, Fan & Sun, Chenxi. (2023). Trajectory Tracking Control of a Manipulator Based on an Adaptive Neuro-Fuzzy Inference System. Applied Sciences. 13. 1046. 10.3390/app13021046.

[16] Haider, Muhammad & Ali, Hub & Khan, Abdullah & Zheng, Hao & Bhutta, M. Usman Maqbool & Usman, Shaban & Zhi, Pengpeng & Wang, Zhonglai. (2022). Autonomous Mobile Robot Navigation using Adaptive Neuro Fuzzy Inference System. 93-99. 10.1109/IDITR54676.2022.9796495.

[17] Li, Shunming, et al. "The algorithm of obstacle avoidance based on improved fuzzy neural networks fusion for exploration vehicle." WSEAS Transactions on Systems and Control, vol. 3, iss. 4, March 2009, pp. 140-150.

[18] Lv, Jiliang, et al. "Research on Obstacle Avoidance Algorithm for Unmanned Ground Vehicle Based on Multi-Sensor Information Fusion." Math Biosci Eng, vol. 18, no. 2, 2021, pp. 1022-1039, doi: 10.3934/mbe.2021055.

[19] Al-Sagban, M., and R. Dhaouadi. "Neural Based Autonomous Navigation of Wheeled Mobile Robots." Journal of Automation, Mobile Robotics & Intelligent Systems, vol. 10, 2016, pp. 64-72.

[20] Chwa, D. "Fuzzy Adaptive Tracking Control of Wheeled Mobile Robots with State-Dependent Kinematic and Dynamic Disturbances." IEEE Transactions on Fuzzy Systems, vol. 20, 2012, pp. 587-593, doi: 10.1109/TFUZZ.2011.2176738.

[21] Sanger, Terence D. "Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural Network." Neural Networks, vol. 2, 1989, pp. 459-473.

[22] Demin, Vyacheslav & Nekhaev, Dmitry & Surazhevsky, I.A. & Nikiruy, Kristina & Emelyanov, Andrey & Nikolaev, Sergey & Rylkov, V. & Kovalchuk, M.V.. (2021). Necessary conditions for STDP-based pattern recognition learning in a memristive spiking neural network. Neural Networks. 134. 64-75. 10.1016/j.neunet.2020.11.005.

[23] Sanger, Terence. (2011). OPTIMAL HIDDEN UNITS FOR TWO-LAYER NONLINEAR FEEDFORWARD NEURAL NETWORKS. International Journal of Pattern Recognition and Artificial Intelligence. 05. 10.1142/S0218001491000314.

[24] Huynh, Hieu Trung, and Won, Yonggwan. "Online training for single hidden-layer feedforward neural networks using RLS-ELM." IEEE International Symposium on Computational Intelligence in Robotics and Automation, December 15-18 2009, pp. 469-473.

[25] Satoh, Seiya & Yamagishi, Kenta & Takahashi, Tatsuji. (2023). Comparing feedforward neural networks using independent component analysis on hidden units. PloS one. 18. e0290435. 10.1371/journal.pone.0290435.

[26] Alemu, Habtamu & Wu, Wei & Zhao, Junhong. (2018). Feedforward Neural Networks with a Hidden Layer Regularization Method. Symmetry. 10. 525. 10.3390/sym10100525.

[27] Wong, Hiu Tung & Leung, Chi-Sing & Kwong, Sam. (2017). A Generalized I-ELM Algorithm for Handling Node Noise in Single-Hidden Layer Feedforward Networks. 424-433. 10.1007/978-3-319-70087-8_45.
[28] Baluja, Shumeet. "Evolution of an Artificial Neural Network Based Autonomous Land Vehicle Controller." IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, vol. 26, iss. 3, June 1996, pp. 450-463.
[29] Rana, Kritika & Gupta, Gaurav & Vaidya, Pankaj & Tomar, Abhishek & Kumar, Nagesh. (2023). Evolution of Autonomous Vehicle: An Artificial Intelligence Perspective. 10.1007/978-981-19-9876-8_6.
[30] Geng, Guoqing & Lu, Sinan & Duan, Chen & Jiang, Haobin & Xiang, Huarong. (2023). Design of autonomous vehicle trajectory tracking controller based on Neural Network Predictive Control. Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering. 095440702211500. 10.1177/09544070221150023.
[31] Pomerleau, Dean A. "Efficient Training of Artificial Neural Networks for Autonomous Navigation." Neural Computation, vol. 3, iss. 1, Spring 1991, pp. 88-97.
[32] Neto, G. & Campos Velho, Haroldo & Shiguemori, Elcio. (2023). Data Selection for Training the Neural Fuser Applied to Autonomous UAV Navigation. Trends in Computational and Applied Mathematics. 24. 159-175. 10.5540/tcam.2022.024.01.00159.
[33] Wijesinghe, Rukshan & Tissera, Dumindu & Vithanage, Mihira & Xavier, Alex & Fernando, Subha & Samarawickrama, Jayathu. (2023). An Advisor-Based Architecture for a Sample-Efficient Training of Autonomous Navigation Agents with Reinforcement Learning. Robotics. 12. 133. 10.3390/robotics12050133.
[34] Hara, Kazuyuki, and Nakayama, Kenji. "Selection of Minimum Training Data for Generalization and On-line Training by Multilayer Neural Networks." IEEE International Conference on Neural Networks, vol. 1, June 3-6 1996, pp. 436-441.
[35] Luan, Dianxin & Thompson, John. (2023). Achieving Robust Generalization for Wireless Channel Estimation Neural Networks by Designed Training Data. 10.48550/arXiv.2302.02302.
[36] Freisleben, Bernd, and Kunkelmann, Thomas. "Combining Fuzzy Logic and Neural Networks to Control an Autonomous Vehicle." Second IEEE International Conference on Fuzzy Systems, vol. 1, 1993, pp. 321-326.
[37] Selma, Boumediene & Chouraqui, S.. (2013). Neuro-fuzzy controller to navigate an unmanned vehicle. SpringerPlus. 2. 188. 10.1186/2193-1801-2-188.
[38] Alsuwian, Turki & Usman, Mian & Amin, Arslan Ahmed. (2022). An Autonomous Vehicle Stability Control Using Active Fault-Tolerant Control Based on a Fuzzy Neural Network. Electronics. 11. 3165. 10.3390/electronics11193165.
[39] Cybenko, G. "Approximation by Superpositions of a Sigmoidal Function." Mathematics of Control, Signals and Systems, vol. 2, 1989, pp. 303-314.
[40] Lewicki, Grzegorz & Marino, Giuseppe. (2004). Approximation by Superpositions of a Sigmoidal Function. Appl. Math. Lett.. 17. 1147-1152. 10.1016/j.aml.2003.11.006.
[41] Costarelli, Danilo & Spigler, Renato. (2013). Constructive Approximation by Superposition of Sigmoidal Functions. Analysis in Theory and Applications. 29. 169-196. 10.4208/ata.2013.v29.n2.8.