Department of Computer Science

Submitted in part fulfilment for the degree of MSc

Autonomous Navigation System for Boats in Narrow Water Channels

Mohit Bidikar

13 September 2023

Supervisor: Dr. Poonam Yadav

ACKNOWLEDGEMENTS

I want to express my gratitude to the University of York for creating an environment that made it possible to conduct this research. I also want to express my gratitude to my supervisor, Dr. Poonam Yadav, for her advice and mentoring throughout this project. Lastly, I would like to thank my friends and family who offered support in one way or the other to keep me going.

STATEMENT OF ETHICS

I hereby attest that the project was conducted with the highest ethical standards in mind. The essay represents my original work. It has not been previously submitted elsewhere, and was solely authored by me. The entire process of executing this project adhered to ethical guidelines, ensuring no harm to any individuals. This report is a truthful representation of the results. Any contributions or perspectives from others have been duly acknowledged and accurately cited. Furthermore, no significant ethical concerns arise from the project's findings.

ABSTRACT

Despite significant advancements in autonomous road vehicles, marine navigation in confined waterways remains a challenging domain for automation. This dissertation introduces an innovative approach to autonomous boat navigation in narrow channels by integrating neural networks with genetic algorithms. A unique 2D simulation environment was developed, serving as a platform for training, testing, and refining the autonomous navigation system. The research successfully demonstrated that boats, equipped with a minimal sensor count, can navigate narrow channels with high accuracy and reliability. The system achieved consistent results, covering over 95% of the length on both testing channels across multiple runs. Key findings indicate a marked improvement in navigation capabilities over successive generations and the potential for safer, more efficient, and cost-effective marine transportation. This research not only sets a technological precedent in the marine domain but also offers promising avenues for future exploration in real-world scenarios and broader autonomous navigation contexts.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

Executive summary

Marine navigation, particularly in confined waterways, represents a challenging frontier for automation. As the era of autonomous vehicles dawns on roads and in the air, the waters remain a realm where technology has yet to fully chart its course. This research seeks to address this gap, aiming at a transformation in the way boats navigate narrow channels.

Road-based vehicles have seen significant strides in autonomous navigation, yet ships and boats, especially in tight channels, remain reliant on human expertise. There's an urgent need to provide reliable, precise, and cost-effective solutions for marine vessels navigating ports, rivers, and other narrow channels.

The aim of this research is the development of an autonomous navigation system specifically tailored for boats in narrow channels. By harnessing the combined power of neural networks and genetic algorithms, the study seeks to elevate the safety, reliability, and efficiency of waterway transportation.

A unique approach is employed in this research, wherein a simulated 2D environment replicating narrow channels is developed. This 2D environment, a significant contribution of this research, offers a customizable setting and compatibility with machine learning models, making it an invaluable tool for training autonomous boats, evaluating navigation algorithms, and serving as an educational resource.

In this environment, boats equipped with a reduced number of sensors, measure proximity to channel boundaries. This sensor data serves as input for neural networks, which attempt to navigate the channel. Subsequently, these neural networks are then refined through genetic algorithms, enhancing their navigation capabilities.

This work presents two scenarios: first, an agent is trained on a training channel, and then it is challenged to navigate completely unfamiliar testing channels.

The research culminates in the successful design of an autonomous navigation system adept at navigating narrow channels. Notably, by using fewer sensors than traditional methods, the proposed system achieves a marked reduction in costs. Preliminary results, showcasing adaptability and collision avoidance, underline the potential of this system to redefine marine navigation.

Throughout the course of this research, no significant legal, social, ethical, professional, or commercial issues were identified. This work serves as a foundational step before the actual autonomous boats are built and tested. As these technological advancements transition from simulation to real-world application, a thorough evaluation across the aforementioned domains will be essential.

In summary, this research paves the way for a new era of marine transportation, one where narrow channels are navigated with precision, safety, and cost-efficiency, all thanks to the amalgamation of neural networks and genetic algorithms.

1 Introduction

From ancient times, waterways have been a significant mode of transportation for both people and goods. Historically, ships and boats have relied on experienced human pilots to navigate them through treacherous waters, unpredictable weather, and narrow channels. As technology advanced, various navigation systems were introduced, such as sonar, GPS, and radar, which aided captains in making informed decisions. However, the human factor remained central to the navigation process, and with it came the potential for error.

In recent years, there has been a significant thrust towards the automation of vehicles. Self-driving cars, once the stuff of science fiction, are now on the brink of commercial viability. Autonomous drones, too, are finding myriad applications from delivery services to surveillance. Yet, when it comes to marine vehicles, especially in constrained waterways like ports, rivers, and narrow channels, automation hasn't kept pace [1]. The challenges posed by the dynamic nature of water, the variability of marine environments, and the constraints of narrow banks make it a challenging domain for automation.

In 2021, the world witnessed the perils of manual navigation in confined waterways with the incident of the Ever Given in the Suez Canal [2]. Such a blockage caused enormous economic repercussions and brought global attention to the pressing need for more reliable navigation systems. While the maritime domain is vast and varied, narrow channels stand out as particularly challenging due to their constrained nature and the high risk of collisions.

Traditional navigation systems in these contexts rely on a combination of technology and human expertise. However, the complexity of narrow channels, influenced by factors like narrow banks and traffic can make navigation difficult even for the most seasoned pilots. Furthermore, the traditional systems often employ a large number of sensors, making them costly and maintenance-intensive.

Against this backdrop, the motivation for this research arises. It seeks to harness the power of artificial intelligence, specifically neural networks evolved with genetic algorithms, to develop an autonomous navigation system tailored for boats in narrow channels. Such a system not only has the potential to significantly reduce the margin of error but also promises to be more cost-effective by leveraging fewer sensors.

In the rapidly evolving landscape of marine transportation, our primary aim stands clear and ambitious: to develop an autonomous navigation system proficient in navigating any channel configuration. This vision is rooted in the broader aspiration to revolutionize how boats traverse narrow channels, ensuring safety, reliability and cost efficiency.

Objectives Breakdown:

Development of a 2D Simulation Framework:

The first step in our journey is to create a robust 2D simulation. This simulation will serve as the foundational framework upon which our autonomous system will be built and refined. It will not only provide a visual representation of the boat's navigation capabilities but also offer a platform for iterative testing and improvement. This framework can be used as a playground before the real autonomous boats are built and tested.

Neural Network Training:

With the simulation in place, the next objective is to train our neural network on a designated training channel. This process will involve evolving the neural network with genetic algorithms, allowing it to learn and adapt to various navigation scenarios. The culmination of this phase will be the exportation of the trained neural network.

Testing and Validation:

The trained neural network will be put to the test on multiple testing channels. Our benchmark for success is stringent yet achievable: we aim for the system to cover more than 95% of the length on both testing channels during each of the 10 runs. This metric ensures that our solution is not only accurate but consistently reliable across different channel configurations.

Furthermore, we also want to demonstrate that efficient navigation can be achieved with a minimal sensor count, underscoring our solution's cost-effectiveness.

The rest of this thesis contains the following.

Section 2 provides a comprehensive review of the existing literature on autonomous navigation systems for boats in narrow channels, establishing a foundation for the research.

Section 3 describes the methodology employed in this research, including the simulation environment, neural network architecture, and genetic algorithms used for evolution.

Section 4 presents the results and analysis of the experiments conducted to evaluate the performance of the autonomous navigation system.

Section 5 offers concluding remarks, summarizing the main findings, discussing their implications, and suggesting avenues for future research.

2 Literature review

Navigating in intricate marine environments has always been a challenge, and narrow channels exemplify these complexities. These confined spaces demand a high level of precision in navigation to avoid collisions and guarantee safety. The past two decades have seen a surge in leveraging advanced computational techniques, particularly machine learning, to address such challenges. This literature review attempts to chronicle the evolution of marine navigation and spotlight the promising intervention of neural networks and genetic algorithms in enhancing autonomous navigation in narrow channels.

The primary methodology employed for literature collection involved an in-depth analysis of both old and modern navigation systems, complemented by a focus on the latest research articles, conference proceedings, and authoritative texts on machine learning and autonomous marine navigation.

2.1 Evolution of marine navigation

In ancient times, marine navigation primarily relied on the stars, sun, and moon, along with rudimentary tools like astrolabes and quadrant. As time progressed, the magnetic compass and sextant came into use, significantly enhancing navigation precision [3].

Modern marine navigation experienced a technological leap with the introduction of sensors such as the Global Positioning System (GPS), sonar, radar, and the Automatic Identification System (AIS) [6] [7] [10] [11] [16]. These sensors enhanced both the safety and precision of navigation but they also escalated costs and demanded rigorous calibration for accuracy.

Subsequently, we transitioned from a system of manual navigation, where human intervention was crucial, to an advanced mode of autonomous navigation, which relies on self-guided mechanisms and technologies to direct the course without continuous human input.

Recent years have witnessed significant advancements in the field of autonomous navigation of boats, driven by the application of machine learning and deep learning techniques. These techniques enable ships to navigate autonomously by leveraging intelligent analytics and training algorithms based on stored data about the vessel's behaviour in various sailing environments [4].

Deep learning has been successfully applied in various maritime applications, including anomaly detection, ship classification and collision avoidance [4]. Deep learning methodologies have also been utilized to process data collected from onboard sensors such as optical, infrared, radar images, point clouds, and more [9].

One proposed approach for autonomous navigation is the use of hierarchical deep reinforcement learning for maritime autonomous surface ships (MASSs). This approach involves a scene division layer that quantifies sub-scenarios based on international regulations and an autonomous navigation decision-making layer that employs deep Q-learning to train navigation strategies [5]. Another approach involves the development of an autonomous navigation system for unmanned surface vessels, incorporating collision avoidance and manoeuvre auto-negotiation using multi-agent systems [6].

Researchers have also explored the integration of multiple sensor modalities and artificial intelligence algorithms to enhance autonomous navigation [7]. Additionally, studies have also investigated the utilization of neural networks, fuzzy logic, genetic algorithms, and evolutionary algorithms [19] [23].

In addressing the challenges specific to navigation in narrow channels, several approaches have been proposed. The Roboat project aims to develop a logistics platform for people and goods in water cities with narrow water channels like Amsterdam and Venice. It suggests that deep learning based frameworks could be a viable option for Autonomous Surface Vessel (ASV) control and obstacle avoidance in these challenging urban environments [9].

Another approach involves the use of different switching controllers for high and low-speed motion, which could be applicable to navigation in narrow channels [14].

Hybrid approaches, such as Fast Marching Square and velocity obstacles, have also been utilized for global path planning of autonomous ships in unknown and unpredictable environments [23]

While significant progress has been made in autonomous navigation of boats, several challenges and limitations persist.

The complexity and variability of navigational scenarios present a challenge, compounded by the slow manoeuvrability response of autonomous boats [5].

Ensuring compliance with the International Regulations for Preventing Collisions at Sea (COLREGs) presents another significant hurdle [8] [20]. The subjectivity of COLREGs and the embodiment of good seamanship in autonomous navigation systems developed using traditional methods pose challenges in achieving compliance [10] [11].

Another major challenge is the designing of reward functions for Deep Reinforcement Learning (DRL) networks, which are usually based on experience and practice rather than mathematical reasoning. [9]

Safety, reliability assurance, and the interaction between manned, remotely controlled, and unmanned vessels in a mixed navigational environment are additional concerns [12] [13].

Path planning algorithms for dynamic environments and the limitations of existing methods also remain open research problems [23].

Environmental factors such as weather conditions, sea state, and visibility can affect the performance of autonomous navigation systems [13].

Another limitation is the significant cost of the sensors used in marine navigation.

2.2 Sensors used in marine navigation

Boats are typically equipped with a combination of different sensors, including long-range radars, stereo vision systems, short-range radars, lidars, sonar, gyrocompass, a speed log, an echo sounder, hydrophones, and the Global Positioning System (GPS) [6] [7] [10] [11] [16]. They also rely on Automatic Identification System (AIS) data for state estimation and behaviour prediction [9]. Inertial Measurement Units (IMUs) have been used to measure the orientation and acceleration of the boat [16], and Global Navigation Satellite Systems (GNSS) have been utilized for position estimation [7]. Additionally, some boats employ laser range finders to report the relative distance to perceived obstacles within their field of view [18].

The cost of a marine radar system can range from £1,000 to £10,000 or more, depending on its capabilities and range. [27]

GPS devices come at varying costs based on their precision. Basic marine GPS units can cost as low as £100, while high-end systems with advanced features can be priced at £10,000 or more [28]. GNSS, which includes systems like Galileo and GLONASS [29], offer higher precision and reliability. The integration of GNSS receivers adds an additional cost, typically ranging from £500 to £5,000.

Sonar devices, which are used to detect underwater objects and gauge water depth, come at varied costs based on their range and capabilities. Basic sonar systems can start from £1000, while high-end multi-beam sonar systems can cost upwards of £20,000. [30]

The AIS, critical for vessel tracking and collision avoidance, costs between £500 to £5,000 based on its data processing capabilities and integration features [31].

The laser range finder which will be simulated in this research are relatively inexpensive costing around £100 to £2000 based on it's range [32].

The cost of acquiring sensors is compounded by the expenses related to their maintenance and periodic calibration. These processes are crucial to ensure the accuracy and reliability of the sensors. The combined annual maintenance and calibration costs can add an additional 5-10% of the original cost of the sensors.

2.3 Gaps in the current research

While the literature provides ample information on autonomous navigation, there is a noticeable gap concerning navigation in narrow channels. Narrow channels present unique challenges due to limited space and the presence of obstacles. Navigating in narrow channels requires precise manoeuvring to avoid the banks, which may be challenging for autonomous vessels. Existing approaches may not adequately address these challenges.

The few existing approaches for autonomous navigation in narrow channels also require a lot of sensors and therefore are very expensive but still don't guarantee hundred percent safety and collision avoidance. These sensors also require frequent maintenance. Failure of one sensor could bring down the entire system as this system requires them to work in tandem.

The limited availability of material on autonomous navigation in narrow channels indicates a research gap that needs to be addressed to ensure safe and efficient navigation in these constrained environments. Neural networks evolved with genetic algorithms provide a promising solution.

2.4 Neural networks evolved with genetic algorithms

While the approaches mentioned above employ various neural network architectures, including Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), Autoencoders (AE), Deep Reinforcement Learning (DRL), Recurrent Neural Networks (RNNs), and Deep Q Network [9] [17] [21], these neural networks are not optimized.

To overcome these limitations, the concept of neural networks evolved with genetic algorithms can be explored. This approach combines the learning capabilities of neural networks with the optimization abilities of genetic algorithms to develop navigation strategies specifically tailored to narrow channel scenarios.

In Evolutionary Algorithms (EA) like Genetic Algorithms (GA), when a population is created, an artificial neural network (ANN) is considered as an individual. Each ANN can vary in terms of the weights and biases that are incorporated into the individual agents within an environment [24]. This environment, used as a platform to investigate the agent's response, can be set up either in a physical environment or through virtual simulations.

Virtual simulation is an effective method to simulate real-world circumstances and test intelligent algorithms while examining common real-world scenarios.

By using genetic algorithms to optimize the parameters of neural network models, the system can learn from past experiences, extract relevant information, and make informed decisions in real-time. This integrated approach can improve the manoeuvrability and collision avoidance capabilities of autonomous boats in narrow channels.

By leveraging the learning and optimization capabilities of these techniques, autonomous vessels can navigate with increased precision, ensuring compliance with COLREGs and avoiding collisions.

While current autonomous boats heavily rely on a large number of sensors, utilizing neural networks evolved with genetic algorithms offers a cost-effective solution for navigation and obstacle avoidance tasks by reducing the number of required sensors.

However, it is important to note that further research and experimentation are necessary to validate the effectiveness of this concept in addressing the limitations of current approaches in narrow channels. The scarcity of available literature on autonomous navigation in narrow channels emphasizes the need for more exploration and development in this specific area.

In conclusion, current approaches for autonomous navigation of boats have made significant progress through the use of artificial intelligence. However, challenges and limitations remain, particularly in narrow channels where specific navigational constraints come into play. The concept of neural networks evolved with genetic algorithms offers a promising cost effective solution to address these limitations. In the next sections we explore potential of this integrated approach and develop its application.

3 Methodology

Given the intricate nature of real-world narrow channels with their varying widths, obstructions, and unpredictable bends, it was essential to adopt an approach that captured these complexities in the simulation.

The following technologies were used to build the simulation:

JavaScript: The primary programming language used is JavaScript. JavaScript is often used for dynamic and interactive features on web pages. The object-oriented features of JavaScript are being used to define several classes (DNA, Checkpoints, NeuralNetwork, etc.) that encapsulate the logic and data for various components of the simulation.

TensorFlow.js: This is a JavaScript library for machine learning. In this project, it's used for defining, training, and using the neural networks that control the boats. The NeuralNetwork class is based on the 'Sequential' model from TensorFlow.js, which is a type of model where layers are stacked linearly on top of each other. This class includes methods for creating a model, making predictions, copying a model, and mutating a model's weights.

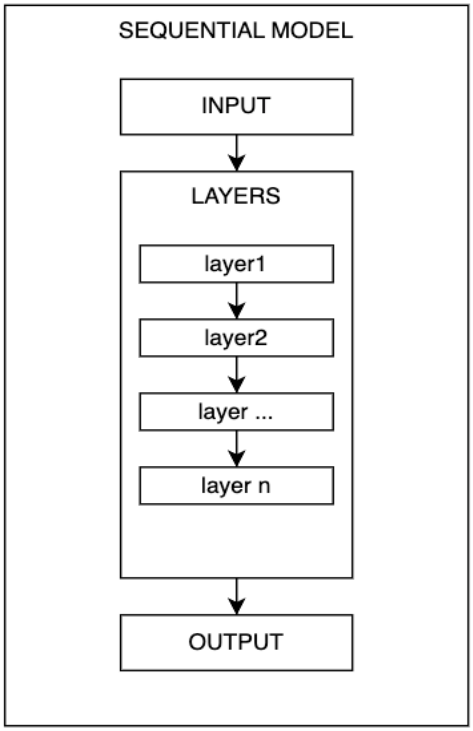

***Figure 1**: Sequential model from TensorFlow.js*

p5.js: This is a JavaScript library that helps with creating visualizations and graphical content. It is used here for rendering the visual elements of the simulation and it also handles the main animation loop.

JSON: JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax. It is used here to export a trained neural network.

Additionally, custom code and algorithms were developed to facilitate the simulation and performance evaluation.

3.1 Building the simulation

A 2D simulation environment was developed, depicted by a user-defined channel image. The simulation represented a realistic narrow channel with various configurations, including twists, turns, varying bank distances, rock outcrops, and docks to make it as close as possible to real-life water channels. In this simulation, each pixel's height and width are considered to be 12 centimeters (0.12 meters) each.

The environment allowed boats to move forward, turn left or right at a user defined angle. The boat's size in the simulation is represented by a radius of 15 pixels. Given that the height and width of 1 pixel are 0.12 meters each, the boat would be 3.6 meters long. This is a reasonable size for a typical boat, making the simulation more realistic and applicable to real-world scenarios.

A high level pseudo code for the entire project with all it's classes and functions is shown in Figure 2:

```
Initialize constants: TURN_MAX, POPULATION_COUNT, SPEED, mutationRate

Load assets:
    Load track image
    Load boat image

Setup simulation:
    Create canvas of specified dimensions
    Initialize population of boats
    Set TensorFlow backend to CPU

Main loop (draw):
    Draw background and track
    Check for wall collisions for each boat
    Update and draw each boat in the population
    If all boats are "dead":
        Calculate fitness for each boat
        Perform natural selection to create a mating pool
        Generate a new population of boats
        Increment generation count

Boat class:
    Properties: position, velocity, neural network (brain), etc.
    Methods:
        Update position and angle based on neural network decisions
        Draw boat and its sensors/reference points on canvas
        Calculate fitness based on progress
        Die if hit a wall or obstacle

DNA class:
    Properties: genes (array of turn values)
    Methods:
        Create genes (initialize with random turn values)
        Calculate fitness based on distance covered
        Crossover with another DNA to produce child DNA
        Mutate genes based on mutation rate


NeuralNetwork class:
    Properties: model, number of input, hidden, and output nodes
    Methods:
        Create a new neural network model
        Predict output given some inputs
        Copy the neural network (clone)
        Mutate the neural network's weights

Event handlers:
    On key press: Save the best neural network model if 'X' key is pressed
```

***Figure 2**: A high-level pseudocode of the project*

Initially, several constants are set, including the maximum turn angle for boats (TURN_MAX), the number of boats in the simulation (POPULATION_COUNT), their movement speed (SPEED), and the mutation rate for the genetic algorithm (mutationRate). The simulation then loads visual assets of the environment, specifically images of the channel and the boats.

Upon setting up the simulation, a canvas of specified dimensions is created to visually display the simulation. A population of boat objects is initialized to navigate the channel, and TensorFlow, a machine learning library, is configured to use the computer's CPU for its computations. The main loop of the simulation continually updates, displaying the environment and each boat. It checks if any boat has collided with a wall or obstacle and updates and displays each boat's position based on its neural network-driven decisions.

If all boats in the simulation become inactive due to collisions, the simulation evaluates their performance by calculating their fitness, which is based on the distance they travelled. The best-performing boats are then selected to serve as "parents" for the next generation, using a process called natural selection. A new set of boats is created for the next round of the simulation using the genetic algorithm, and the generation count is incremented to keep track of the number of simulation rounds completed.

The Boat class in the pseudocode represents individual boats in the simulation, each with properties like position, speed, and a neural network that determines its decisions. The DNA class represents the genetic information of each boat, containing an array of genes that influence the boat's decisions. Functions within the DNA class manage this genetic information, such as evaluating performance, combining genes from two parent boats, and introducing random changes or mutations. The NeuralNetwork class represents the decision-making mechanism of each boat, with functions that allow it to make decisions, replicate itself, and mutate. Lastly, the simulation includes event handlers that listen for user input, allowing users to save the best-performing neural network by pressing a specific key.

Next, we dive deeper in the neural network and the genetic algorithm used in the simulation.

3.1.1 Neural network and genetic algorithm

```
1. Define NeuralNetwork class:

2. Initialize constructor(a, b, c, d):
  - If 'a' is an instance of tf.Sequential (TensorFlow.js Sequential model):
    - this.model = a
    - this.inputNodes = b
    - this.hiddenNodes = c
    - this.outputNodes = d
  - Else:
    - this.inputNodes = a
    - this.hiddenNodes = b
    - this.outputNodes = c
    - this.model = call createModel()


3. Define createModel() method:
   - Create a new tf.sequential() model
   - Define a hidden layer with 'sigmoid' activation and 'this.inputNodes' as inputShape
   - Add hidden layer to the model
   - Define an output layer with 'softmax' activation
   - Add output layer to the model
   - Return the model

4. Define predict(inputs) method:
   - Create a tensor from the inputs
   - Predict the output by passing the input tensor to the model
   - Get the outputs data from the tensor
   - Return the outputs

5. Define copy() method:
   - Create a copy of the model using createModel()
   - Get the weights of the original model
   - Clone these weights
   - Set the weights of the copied model to these cloned weights
   - Return a new NeuralNetwork object with the copied model and original parameters

6. Define mutate(mutationRate) method:
   - Get the weights of the model
   - Iterate over each weight tensor:
     - Get a clone of the tensor's values
     - Iterate over each value:
       - If a random number is less than the mutation rate:
         - Add a random gaussian number to the value (this introduces variation)
     - Create a new tensor from the updated values
     - Replace the corresponding weight in the model with this new tensor
```

***Figure 3**: Pseudocode for the Neural Network class*

The provided code shows the structure of the NeuralNetwork class, crafted to harness the capabilities of TensorFlow.js. When an instance of this class is initiated via its constructor, it checks if the first parameter is a pre-existing TensorFlow model (tf.Sequential). If so, the model and its parameters are directly integrated. Otherwise, the parameters are assumed to represent the number of nodes in the

input, hidden, and output layers, and a new model is constructed using the createModel() method. This method crafts a sequential neural network, embedding the hidden layer with a sigmoid activation function and an output layer with a softmax activation function, ensuring the output values sum up to 1, making them interpretable as probabilities. The predict() method is designed to process an array of inputs, converting them into a tensor, and subsequently returning the model's predictions. To facilitate the evolution of neural networks, the copy() method clones the current network, producing a new instance with identical weights and parameters. The mutate() method introduces genetic variability by adjusting the network's weights based on a given mutation rate. If a random value falls below this rate, a weight is slightly altered by adding a random gaussian value. Collectively, the NeuralNetwork class not only provides tools for creating and managing neural networks but also integrates genetic algorithm principles, allowing for the evolution of these networks over time.

To explain further, this project uses a feed-forward neural network (also known as a Multilayer Perceptron (MLP)) architecture as part of the decision-making process for the boats. The network is structured as follows:

Input Layer: Receives data from the boat sensors. We have 4 input nodes. Data from sensor 1, sensor 2, Sensor 3 and the speed of the boat.

Hidden Layers: Multiple layers that process the input through weighted connections, learning and adapting over time. We have 8 nodes in this layer. The activation function for this layer is the sigmoid function. It maps input values to a range between 0 and 1, enabling non-linear transformations. The hidden layer's purpose is to transform the input into a format suitable for the output layer. The sigmoid activation function is calculated using the following equation:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

***Figure 4**: Sigmoid activation function*

where σ is the sigmoid function, *x* is the input to the sigmoid function and *e* is Euler's number [26].

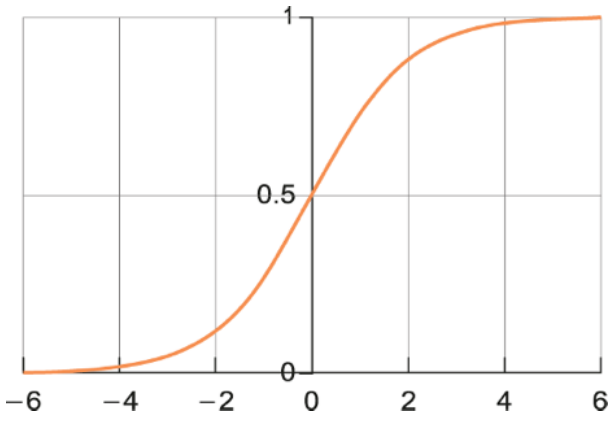

***Figure 5**: Graphical representation of sigmoid function*

Output Layer: Determines the boat's response, such as adjusting its course. The activation function used is softmax, which converts raw scores into probabilities that sum up to 1. We have 2 nodes in this layer.

The softmax activation function is calculated with the following equation:

$$f(y) = \frac{e_i^y}{\sum_{k=1}^{K} e_i^y}$$

***Figure 6**: Softmax activation function*

Where *f* is softmax activation method, *y* is the input value, e_i^y is the standard exponential function of the input value, and K is the number of classes in the given dataset [26].

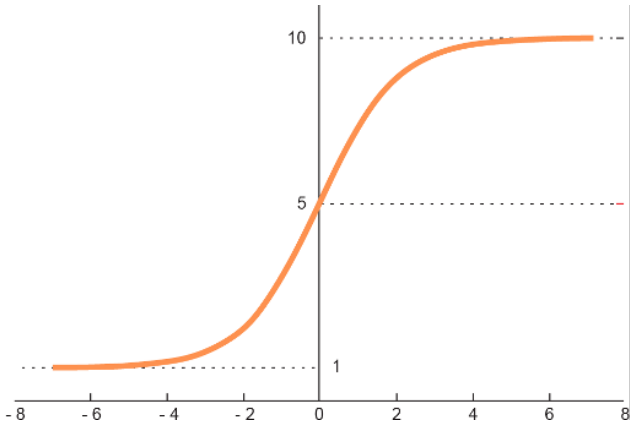

***Figure 7**: Graphical representation of softmax function*

Training the neural network

This project takes a unique approach to training neural networks. Instead of the conventional method of iterative weight updates based on prediction error, a genetic algorithm is employed to train the neural networks across generations.

At the start, when a new neural network is created, the weights are randomly initialized. Unlike traditional training, the selection of neural networks for reproduction is based on their fitness scores. In our simulation, the fitness of each boat is determined by its progress through a predefined sequence of decisions, represented by its DNA genes. Specifically, the index of the current gene serves as a measure of the boat's progress. This index is normalized to a range between 0 and 1. To emphasize the differences in performance, the normalized fitness value is raised to the power of 4. As a result, boats that have travelled further are considered significantly more fit than their counterparts. This fitness metric ensures that, over generations, boats that are better at navigating the environment are more likely to pass their traits to the next generation.

During the creation of a new generation of boats, the "child" boats inherit the neural network weights of their "parents." However, there is also a chance for mutation to occur. Mutation plays a crucial role in genetic algorithms as it introduces small perturbations to the neural network weights. This perturbation involves adding a value drawn from a normal (Gaussian) distribution to the weights. The probability of mutation is determined by the mutationRate parameter.

The Genetic algorithm flows sequentially:

1. Initialization: The algorithm initializes a population of "Boats" each having a DNA and a Neural Network (acting as a "brain"). The population size is set by POPULATION_COUNT.

2. Fitness Calculation: Each boat in the population is evaluated for their performance. Here, the fitness is calculated as the boat's progress through the channel. More the distance covered by a boat, the higher the fitness score. This is achieved in the calcFitness() method in the Boat class.

3. Selection: This process starts by creating a "mating pool". The boats with higher fitness are more likely to get selected for reproduction, as they are added to the mating pool proportionally to their fitness score. The naturalSelection() function handles this process.

4. Crossover: The new generation is created by randomly selecting parents from the mating pool and performing crossover to form a new child. The crossover process is performed in the crossover() method in the DNA class, where a new DNA (child) is created by merging two parents' genes at a random midpoint.

5. Mutation: To maintain diversity within the population and avoid premature convergence, mutation is introduced. The mutate() method within the DNA class alters random genes in the DNA by introducing slight deviations in the gene values.

6. Replacement: The old population is then replaced by the new population (children), and the cycle continues until all the boats are dead (either by hitting an obstacle or reaching the goal). Then a new generation is created.

7. Saving the Best Boat: The best performing boat from each generation is saved and given a greater chance to pass its traits to the next generation.

This genetic algorithm, in tandem with the Neural Network, allows the boats to learn to navigate the channel over time. Each generation should get progressively better at this task as the beneficial traits that allow boats to go further are passed down.

3.2 Simulation environment

3.2.1 Boats (agents)

A "Boat" represents an individual entity in the genetic algorithm's population. The Boat's behavior is influenced by two main factors: its "brain" (a Neural Network), and its "DNA" (a sequence of genes that instructs the boat how to turn at each frame).

Each Boat keeps track of its current position (pos), current distance (currentCheckpoint), and whether it is still alive (alive). These variables are updated as the boat moves around the channel and collides with obstacles.

The speed of the boats is defined by the variable SPEED. The variable TURN_MAX controls the maximum turning angle, which affects how sharply the boats can change direction.

The values of SPEED and TURN_MAX are kept constant in this scenario to ensure that the differences observed in the performance of different boats are due solely to the genetic algorithm and the learning of the neural network. By keeping these variables constant, the experiment ensures that the variation in boat performance is a result of the different "brains" (i.e., the neural networks) of the boats rather than differences in their basic capabilities. To explain further:

SPEED: Keeping speed constant ensures that all boats are moving at the same pace. This way, a boat doesn't have an unfair advantage just because it is faster. Instead, success depends on how well the boat's neural network has been trained to navigate the course.

TURN_MAX: This variable determines how sharply a boat can change direction. By keeping this constant, it ensures that all boats have the same manoeuvrability. If some boats could turn more sharply than others, they might be able to navigate the course more effectively, regardless of how well their neural network is trained.

Keeping these values constant effectively sets the "physical" parameters of the boat equal for all instances. Therefore, the genetic algorithm's performance in training the neural network to navigate the boats effectively can be evaluated in isolation.

Please note that in different simulations or real-life scenarios, these values could be varied, and the genetic algorithm could also optimize for the best speed and turn angle. But in this case, for the sake of simplicity and clarity, these physical attributes have been kept constant.

Sensors on boats

As shown in Figure 8, the boat is equipped with 3 laser range sensors. Their primary function is to determine the distance between the boat and any obstacle. These sensors are strategically placed for maximum coverage. This reduced configuration not only prioritizes cost-effectiveness but also demonstrates that effective navigation is achievable with a minimal sensor count. To explain further:

s0 sensor is pointing directly ahead of the boat

s1 sensor is pointing to the left of the boat

s2 sensor is pointing to the right of the boat

The red dots indicate the maximum range of these sensors

***Figure 8**: Three laser range sensors on the boat*

Each sensor emits a ray from the boat in the direction it is facing. The sensor then measures the distance to the first pixel of a different colour that the ray intersects with. This process effectively gauges the distance between the boat and any obstacle, providing valuable information for safe navigation.

It is important to note that these laser range sensors can only measure distances within a certain range. This range has been carefully calibrated to provide optimal and reliable results. By focusing on distances within this predefined range, the sensors deliver accurate readings.

These readings are then provided as input to the boat's neural network. The neural network then uses these inputs to make informed decisions about steering adjustments. For example, if a sensor detected that the boat was close to the channel bank, the boat might decide to steer away from the bank.

3.2.2 Narrow channels

The setup includes two distinct channel types, the training channel where the boats are trained and the testing channels where the best boat from the training channel is tested.

Training channel

***Figure 9**: Training channel*

The training channel presents a narrow 844 meters long water passageway with a multitude of sharp twists and turns, providing a challenging environment for navigation. The channel's width changes at different locations, further adding to the complexity.

Within the channel, several obstacles are strategically placed. Notably, rock outcrops are marked in black and the deck is displayed in dark brown. These obstacles demand careful manoeuvring and decision-making from the boats navigating through the training channel.

To aid in the training process, a dynamic section is included below the simulation. This section continuously displays essential information, such as the best distance achieved overall by the boats, the current generation number in the genetic algorithm, the initial population of the boats, and the mutation rate applied during the genetic algorithm's evolution.

Together, these features and elements create an engaging and challenging training environment, enabling boats to learn and optimize their navigation strategies.

Testing channels

The testing channels offers a distinctly different channel configuration compared to the training channel. Their unique layout and challenges provide a new and separate environment for evaluating the boats' performance.

Following the completion of the training channel, the best-performing boat is selected for further assessment. This exceptional boat is then put to the test on the testing channels, where it faces the novel channel configuration and obstacles.

Testing channel 1

***Figure 10**: Testing channel 1*

Testing channel 1, with a length of 629 meters, features a deck, rock outcrops, and shallow banks that are highlighted in light brown.

Testing channel 2

***Figure 11**: Testing channel 2*

Testing channel 2 is 600 meters long. It contains swamps, a deck and some rock outcrops.

The outcomes of this testing phase, along with relevant performance metrics, are meticulously documented and presented in the next section. This presentation of results offers valuable insights into how the trained boat adapts to new and unfamiliar environments.

4 Experiments and results

Results and analysis of the experiments conducted to evaluate the performance of the autonomous navigation system are shown below. Our initial efforts focused on determining the optimal parameter values for the training channel.

4.1 Training channel

Several notable observations were made as we meticulously experimented with various parameter values on the training channel.

Parameter	Value	Description
TURN_MAX	0.01	Boats turn very little, making them unable to navigate turns effectively.
	0.05	Optimal value. Boats navigate turns effectively without excessive zig-zagging and without moving in circles
	0.1	Boats start rotating significantly, causing zig-zag movement. Some boats also fall in a circular infinite loop.

***Table 1**: TURN_MAX values*

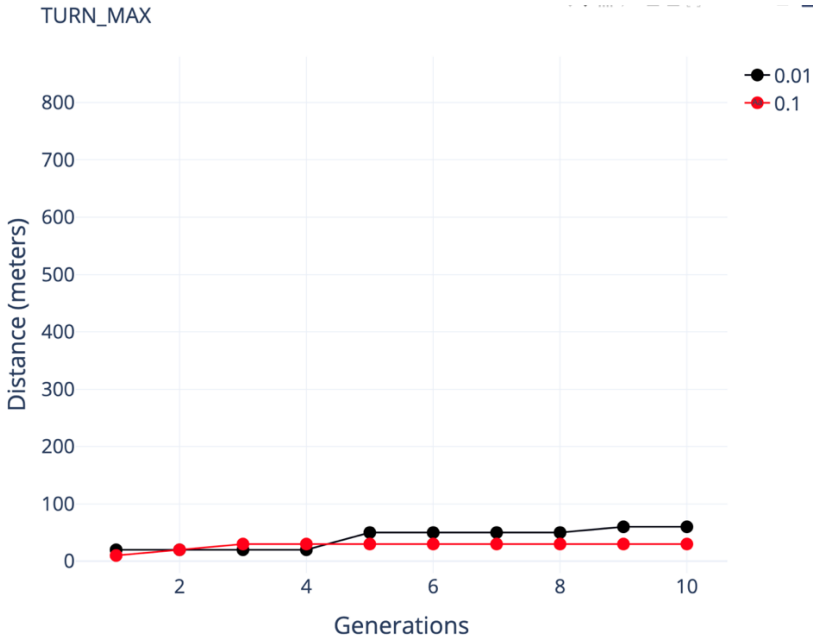

***Figure 12**: TURN_MAX plot for non-optimal values*

Parameter	Value	Description
mutationRate	0.01	A low mutation rate leads very little observable progress across subsequent generations and the boats finally end up in deadlock
	0.05	Optimal mutation rate that effectively promotes genetic diversity. This rate ensures a balanced evolution without leading to any unpredictable or erratic behaviour in the population
	0.1	High mutation rate results in erratic behaviour within the population. While there are instances where it can lead to superior performance, there are equally frequent occurrences where it underperforms or behaves poorly

***Table 2**: mutationRate values*

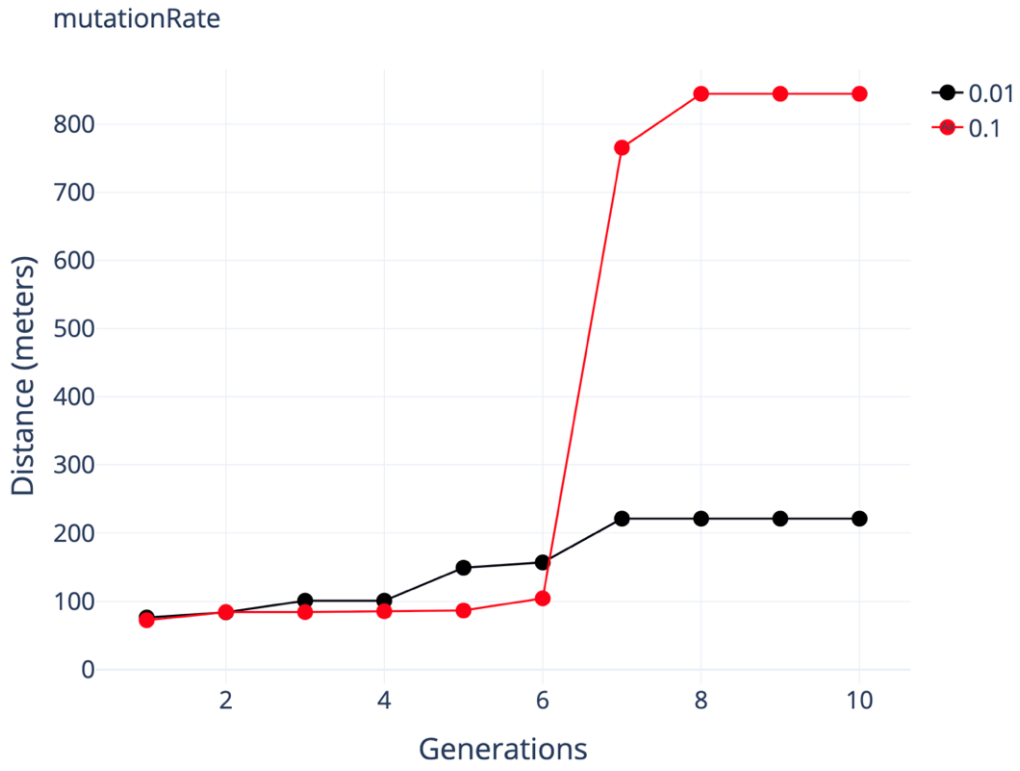

***Figure 13**: mutationRate plot for non-optimal values*

Parameter	Value	Description
POPULATION_COUNT	50	Too few agents for significant mutations. Steady progress is made over the generation but this requires a greater number of generations to cover the entire channel.
	100	Optimal population size for computing resources. Good balance between diversity and computational efficiency.
	200	Agents learn to navigate faster and cover the entire length of the channel in very few generations. Limited by computing resources.

***Table 3**: POPULATION_COUNT values*

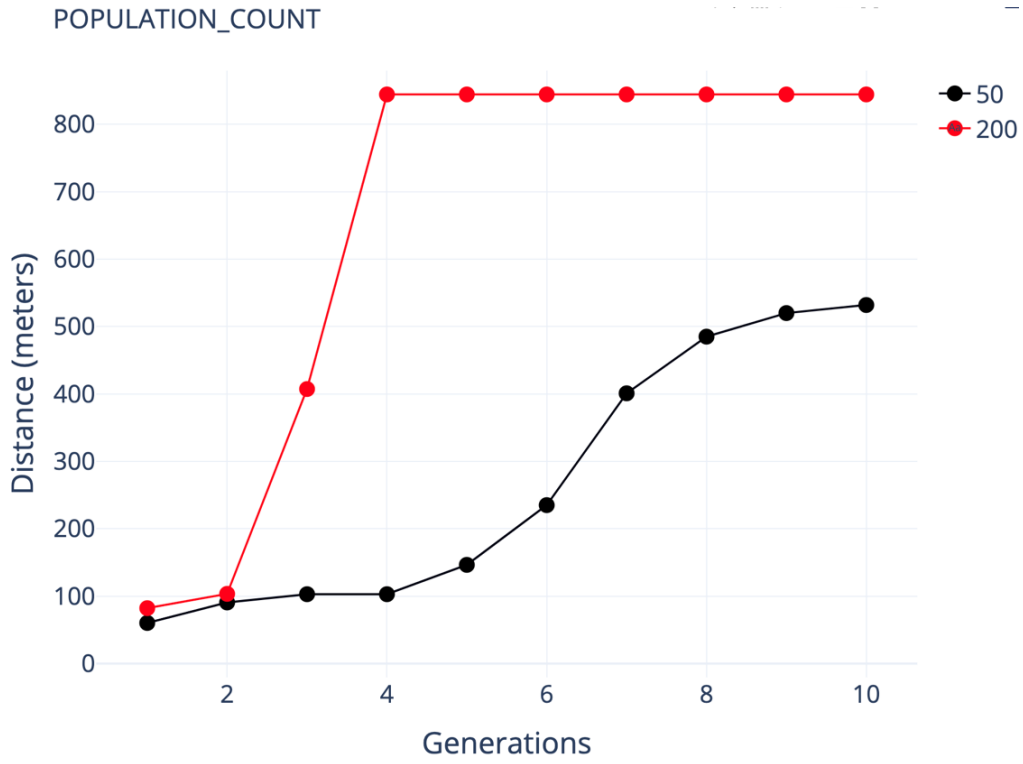

***Figure 14**: POPULATION_COUNT plot for non-optimal values*

Parameter	Value	Description
SPEED	1	Boats cover very less distance and start rotating in loops. The looping behaviour was unexpected
	1.5	Optimal speed. Boats navigate the channel effectively without getting stuck in loops.
	2	Boats cover good distance initially but cannot handle the sharp turn in the middle of the training channel due to ineffective turning at high speeds.

***Table 4**: SPEED values*

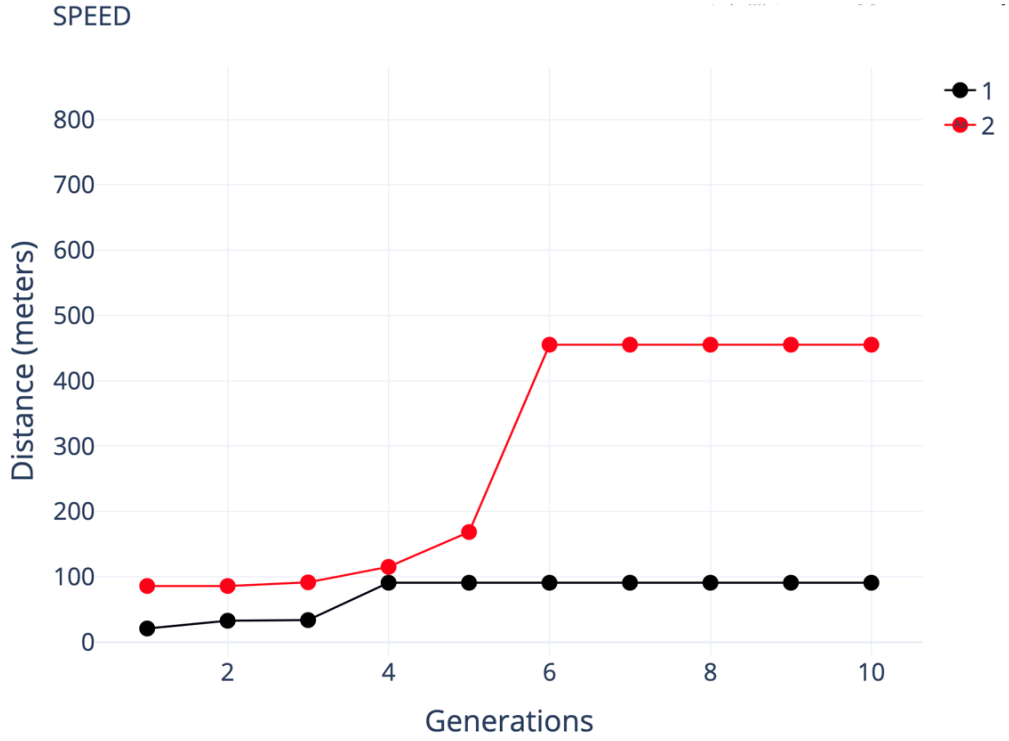

***Figure 15**: SPEED plot for non-optimal values*

These observations highlight the importance of carefully selecting the parameters for the simulation. The optimal values depend on the specific requirements of the simulation, the characteristics of the channel, and the available computing resources. In the next section we run the simulation with the optimal values and showcase the results.

Running with optimal parameters values

TURN_MAX = 0.05
POPULATION_COUNT = 100
SPEED = 1.5
mutationRate = 0.05

Over multiple iterations of the simulation, we observed a clear improvement in the performance of the boats. In the initial generations, the boats exhibited seemingly random behaviour, often running aground or failing to navigate the channel successfully. However, as the genetic algorithm took effect, the navigation improved markedly. By the sixth generation, the majority of the boats were able to successfully negotiate turns and avoid obstacles. The boats were able to predict the best steering angle to avoid collisions. Majority of the boats covered the entire channel (844 meters) by the seventh generation.

Generation	Distance (meters)
1	116.82
2	211.86
3	445.86
4	447.12
5	766.98
6	840.60
7	844.20
8	844.20
9	844.20
10	844.38

***Table 5**: Training channel results*

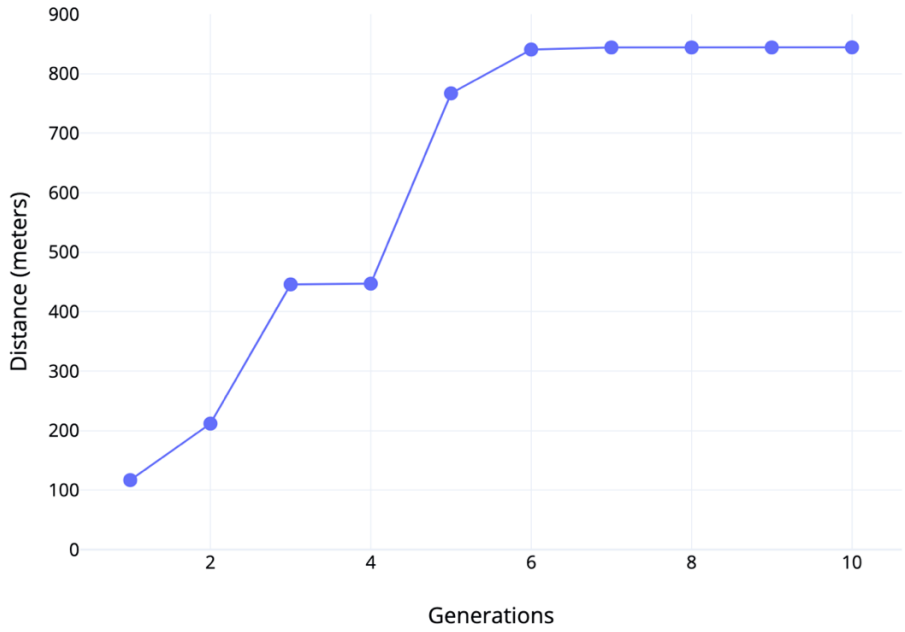

***Figure 16**: Distance covered for optimal values*

We exported this trained neural network and ran it on the testing channels.

4.2 Testing channels

Testing channel 1

The trained boat was tested on channel 1 for a total of 10 runs. In each of these runs, it successfully covered the entire channel distance of 629 meters.

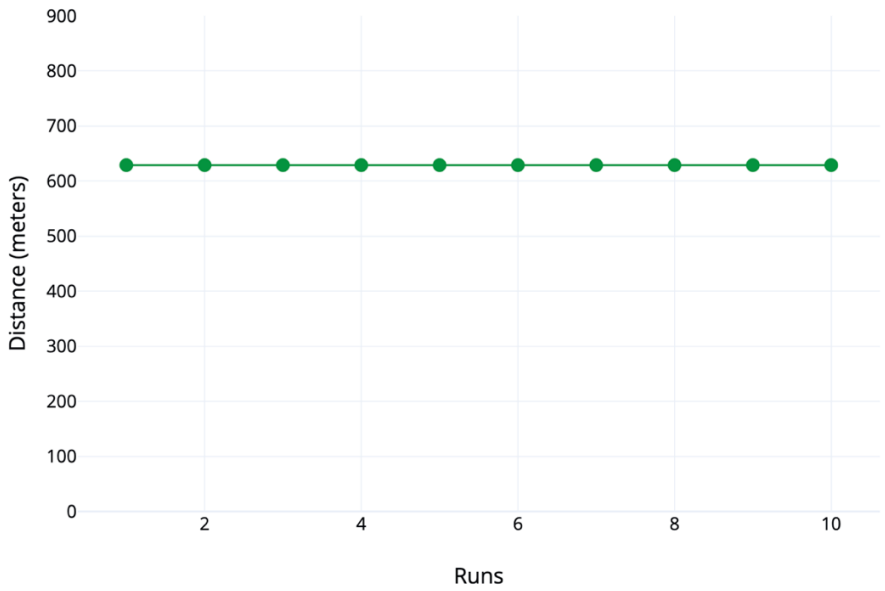

***Figure 17**: Distance covered over 10 runs on testing channel 1*

Testing channel 2

The trained boat was tested on channel 2 for a total of 10 runs. In each of these runs, it successfully covered the entire channel distance of 600 meters.

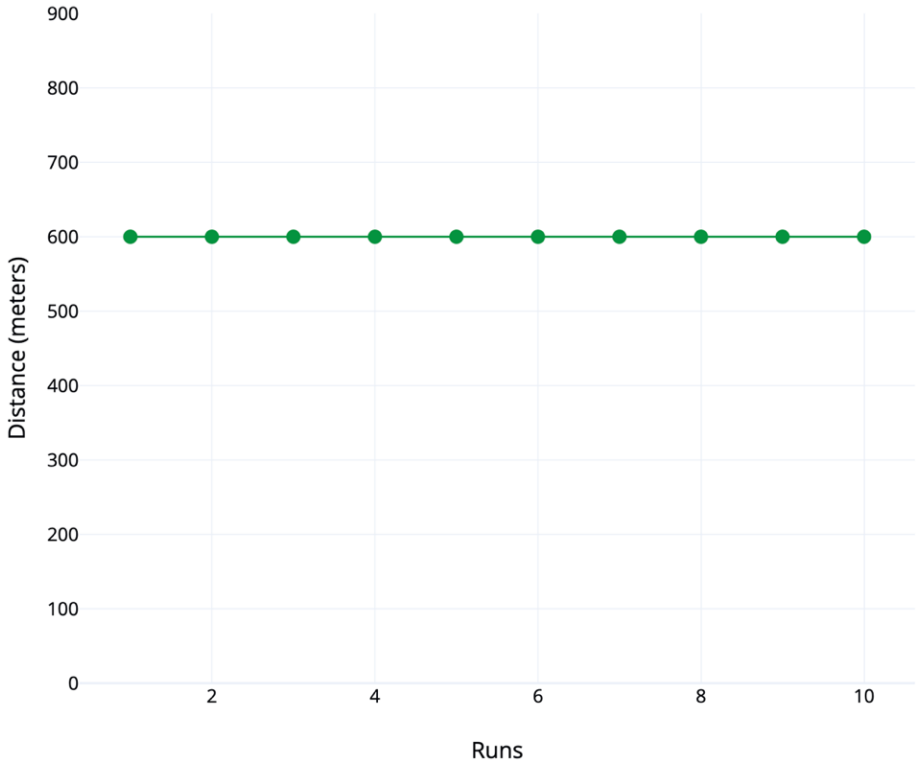

***Figure 18**: Distance covered over 10 runs on testing channel 2*

4.3 Discussion and summary of results

The primary objective of this research was to explore the effectiveness of combining a genetic algorithm with a neural network for autonomous boat navigation. The results have been surprisingly good, with the system meeting the >95% success criteria set at the research's outset. Over generations, there was a noticeable improvement in performance, reinforcing the idea that the system can adapt and learn from its environment. This adaptability suggests potential applications of this model to other challenges within the realm of autonomous navigation.

During the experimental phase, an unexpected looping behaviour of boats at lower speeds was observed. Contrary to the expectation that boats would exhibit greater control at reduced speeds, they seemed to overcompensate for their turns, leading to looping. This observation underscores the significance of speed in determining effective navigation and emphasizes the need for a fine-tuned balance between various parameters.

The experiments successfully pinpointed the optimal parameters for training boats to navigate through channels. The most effective parameters identified were TURN_MAX at 0.05, POPULATION_COUNT of 100, SPEED at 1.5, and a mutationRate of 0.05. When trained with these parameters, boats consistently navigated the entire length of the testing channels, highlighting the training's efficacy.

However, it's crucial to acknowledge the limitations encountered. The constraints on computational resources meant that the population size was capped at 100, even though a population of 200 demonstrated quicker learning. The boats were also tested on only two channels, which might not represent the myriad challenges faced in real-world scenarios. Furthermore, while the parameters identified were optimal for the given conditions, they might not be universally so. Different environments or channels might necessitate parameter adjustments.

While the findings provide a solid foundation for boat navigation in controlled environments, further research and testing in diverse conditions will be crucial for broader applications.

5 Conclusion

The research embarked on a journey to understand the potential of combining genetic algorithms with neural networks to train boats to navigate narrow channels autonomously. A significant milestone of this research is the successful achievement of all the set objectives:

Development of a 2D Simulation Framework: A robust 2D simulation was created, serving as the foundational framework for the autonomous system. This simulation not only provided a visual representation of the boat's navigation capabilities but also offered a platform for iterative testing and improvement, ensuring a solid foundation before transitioning to real-world scenarios.

Neural Network Training: The neural network was effectively trained on a designated training channel, with the integration of genetic algorithms allowing it to learn and adapt to various navigation scenarios. The culmination of this phase was the successful exportation of the trained neural network, showcasing the efficacy of the training process.

Testing and Validation: The trained neural network was rigorously tested on multiple channels, consistently achieving the benchmark of covering more than 95% of the length on both testing channels during each of the 10 runs. This not only validated the accuracy of the system but also its reliability across different channel configurations. Furthermore, the research demonstrated that efficient navigation could be achieved with a minimal sensor count, emphasizing the solution's cost-effectiveness.

The primary findings from achieving these objectives include:

The genetic algorithm, when combined with a neural network, showed a progressive improvement in boat navigation capabilities over successive generations.

Boats trained with the optimal parameters consistently navigated the entire length of the testing channels, showcasing the efficacy of the training.

The research highlighted the importance of balancing various parameters to achieve efficient navigation and the potential pitfalls of not doing so, such as looping behaviours at lower speeds.

The successful training of boats to navigate narrow channels autonomously has several implications:

Safety: The ability to navigate autonomously can reduce human error, potentially leading to safer waterways.

Efficiency: Autonomous navigation can optimize routes and speeds, leading to faster and more fuel-efficient journeys.

Cost-effectiveness: Over time, autonomous boats can reduce operational costs by using fewer sensors.

Technological Precedence: This research sets a precedent for the integration of genetic algorithms with neural networks in other areas of autonomous navigation, not just limited to boats.

While the current research provides a robust foundation, there are several avenues for future exploration:

Real-world Testing: Transitioning from simulations to real-world scenarios to understand how the system responds to dynamic challenges like water currents, changing weather conditions, and other boats.

Alternative Fitness Metrics: Exploring other fitness metrics, such as energy efficiency or time taken, could provide insights into optimizing different aspects of navigation.

Integration of Additional Sensors: Incorporating sensors like depth sensors, LIDAR, or cameras could provide a richer input dataset for the neural network, enhancing navigation capabilities.

Alternative Machine Learning Models: Exploring other machine learning models or optimization techniques to compare and potentially enhance performance.

The dissertation delved deep into the realm of autonomous boat navigation in narrow channels, leveraging the strengths of genetic algorithms and neural networks. The results were promising, showcasing the potential of this combined approach in navigating intricate water channels. While the simulation results provide optimism, the real challenge and opportunity lie in translating these findings to real-world scenarios. The waters ahead are uncharted, but with continued research and innovation, the horizon looks promising for autonomous boat navigation in narrow channels.

Bibliography

[1] J. Lisinska, “Autonomous vehicles: automotive, maritime and aerial A policy landscape review,” 2021. DOI: https://doi.org/10.18742/pub01-064.

[2] “2021 Suez Canal obstruction,” Wikipedia. Available at: https://en.wikipedia.org/wiki/2021_Suez_Canal_obstruction.

[3] “History of navigation,” Wikipedia. Available at: https://en.wikipedia.org/wiki/History_of_navigation.

[4] A. Noel, S. K, K. G. S. Kumar, and S. Bm, "Autonomous Ship Navigation Methods: A Review," in International Conference on Marine Engineering and Technology Oman, November 2019. DOI: 10.24868/icmet.oman.2019.028.

[5] X. Zhang, C. Wang, Y. Liu, and X. Chen, "Decision-Making for the Autonomous Navigation of Maritime Autonomous Surface Ships Based on Scene Division and Deep Reinforcement Learning," in Sensors, vol. 19, no. 18, p. 4055, September 2019. DOI: 10.3390/s19184055.

[6] A. Lazarowska and A. Żak, “A Concept of Autonomous Multi-Agent Navigation System for Unmanned Surface Vessels,” Electronics, vol. 11, no. 18, p. 2853, Sep. 2022. DOI: 10.3390/electronics11182853.

[7] R. G. Wright, "Intelligent Autonomous Ship Navigation using Multi-Sensor Modalities," in TransNav the International Journal on Marine Navigation and Safety of Sea Transportation, vol. 13, no. 3, pp. 503-510, September 2019.

[8] L. P. Perera, "Autonomous Ship Navigation Under Deep Learning and the Challenges in COLREGs," in Proceedings of the 37th International Conference on Ocean, Offshore and Arctic Engineering, Madrid, Spain, June 2018.

[9] Y. Qiao, J. Yin, W. Wang, F. Duarte, J. Yang, and C. Ratti, “Survey of Deep Learning for Autonomous Surface Vehicles in Marine Environments,” in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 4, pp. 3678–3701, Apr. 2023. DOI: 10.1109/TITS.2023.3235911.

[10] L. Hu, H. Hu, W. Naeem, and Z. Wang, "A review on COLREGs-compliant navigation of autonomous surface vehicles: From traditional to learning-based approaches," Journal of Automation and Intelligence, vol. 1, no. 1, pp. 100003, 2022.

[11] B. Karakostas, "Perspective Chapter: Training Autonomous Ships for Safe Navigation," 2023. DOI:10.5772/intechopen.1001355

[12] B. S. Rivkin, "Unmanned ships: Navigation and more," Gyroscopy and Navigation, vol. 12, pp. 96-108, 2021. DOI: 10.1134/S2075108721010090.

[13] T. E. Kim, L. P. Perera, M. P. Sollid, B. M. Batalden, and A. K. Sydnes, "Safety challenges related to autonomous ships in mixed navigational environments," in WMU Journal of Maritime Affairs, vol. 21, no. 2, pp. 141-159, 2022.

[14] A. Alessandri, S. Donnarumma, M. Martelli, and S. Vignolo, "Motion Control for Autonomous Navigation in Blue and Narrow Waters Using Switched Controllers," in Journal of Marine Science and Engineering, vol. 7, no. 6, p. 196, 2019.

[15] L. Elkins, D. Sellers, and W. R. Monach, "The Autonomous Maritime Navigation (AMN) project: Field tests, autonomous and cooperative behaviors, data fusion, sensors, and vehicles," in Journal of Field Robotics, vol. 27, no. 6, pp. 790-818, 2010.

[16] J. Shepard, “How are sensors improving maritime navigation?,” Sensor Tips, Available at: https://www.sensortips.com/featured/how-are-sensors-improving-maritime-navigation-faq/.

[17] F. Ferreira, A. Quattrini Li, and Ø. J. Rødseth, "Navigation and perception for autonomous surface vessels," Frontiers in Robotics and AI, vol. 9, pp. 918464, 2022. DOI: 10.3389/frobt.2022.918464.

[18] S. W. Chen, N. Atanasov, A. Khan, K. Karydis, D. D. Lee, and V. Kumar, "Neural network memory architectures for autonomous robot navigation," arXiv preprint arXiv:1705.08049, 2017.

[19] J. Balicki and Z. Kitowski, "Neural-genetic techniques for ship control systems," in WIT Transactions on The Built Environment, vol. 27, 1970.

[20] P. N. Hansen, T. T. Enevoldsen, D. Papageorgiou, and M. Blanke, "Autonomous Navigation in Confined Waters-A COLREGs Rule 9 Compliant Framework," IFAC-PapersOnLine, vol. 55, no. 31, pp. 222-228, 2022. DOI: 10.48550/arXiv.2207.08227.

[21] A. Asgharpoor Golroudbari and M. H. Sabour, "Recent Advancements in Deep Learning Applications and Methods for Autonomous Navigation: A Comprehensive Review," University of Tehran, Faculty of New Sciences and Technologies, June 2023. DOI: 10.22541/au.168664884.43899660/v2.

[22] W. T. Kearney, "Using genetic algorithms to evolve artificial neural networks," 2016.

[23] J. R. Sanchez-Ibanez, C. J. Perez-del-Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," in Sensors, vol. 21, no. 23, pp. 7898, 2021. DOI: 10.3390/s21237898.

[24] X. Yao, "Evolving artificial neural networks," Proceedings of the IEEE, vol. 87, no. 9, pp. 1423-1447, 1999. DOI: https://doi.org/10.1109/5.784219

[25] M. C. ter Brake, E. Iperen, D. Looije, and Y. Koldenhof, "Unmanned ship simulation with real-time dynamic risk index," in Zeszyty Naukowe Akademii Morskiej w Szczecinie, 2015.

[26] F. Es-Sabery, A. Hair, J. Qadir, B. Sainz-De-Abajo, B. García-Zapirain and I. D. L. Torre-Díez, "Sentence-Level Classification Using Parallel Fuzzy Deep Learning Classifier," in IEEE Access, vol. 9, pp. 17943-17985, 2021, DOI: 10.1109/ACCESS.2021.3053917.

[27] “Boat Radar,” Pirates Cave Chandlery. Available at: https://www.piratescave.co.uk/electronics/marine-navigation/boat-radars/

[28] “Marine GPS Chart Plotters & Depth Sounders | Force 4,” www.force4.co.uk. Available at: https://www.force4.co.uk/department/electronics/gps

[29] “GS25 Professional GNSS (GPS+GLONASS+GALILEO+BEIDOU),” cpe.leica-geosystems.com. Available at: https://cpe.leica-geosystems.com/ch/gs25-professional-gnss-gps-glonass-galileo-beidou.html

[30] “Sonars,” www.echomasterdirect.co.uk. Available at: https://www.echomasterdirect.co.uk/Sonar

[31] “Marine AIS - Transponders, Transceivers & Receivers,” www.force4.co.uk. Available at: https://www.force4.co.uk/department/electronics/ais

[32] “Optics | Rangefinders & Monoculars | Rangefinders,” Uttings.co.uk. Available at: https://www.uttings.co.uk/c511-rangefinders/show=all/