AN INTEGRATIVE APP PRODUCING AN OPTIMAL PATH FOR THE VESSEL IN ORDER TO REDUCE THE IMPACTS OF CARGO SHIPS ON THE ENVIRONMENT

Chenyu Zuo[1], Yu Sun[2]

[1]Sage Hill School, 20402 Newport Coast Dr, Newport Beach, CA 92657
[2]California State Polytechnic University, Pomona, CA, 91768,
Irvine, CA 92620

ABSTRACT

Almost every business in the world relies in some way on the shipping industry, whether it is to ship goods or natural resources, the shipping industry is undeniably the global industry, However, these very ships that drive the economy also produce close to 1 billion metric tons of carbon dioxide per year. In this project, we explore the use of machine learning to improve the performance of cargo ships in the ocean by implementing a genetic algorithm AI and a virtual simulation environment. An app was made based on using the training developed by the AI to be able to be deployed on cargo ships as part of their navigation system. Once sufficient data regarding a vessel's environment was collected, the algorithm could then produce an optimal path for the vessel. Experiments show that the AI system could sufficiently adjust to varying conditions and produce optimal paths for vessels.

KEYWORDS

Machine Learning, AI, Mobile APP, environment

1. INTRODUCTION

With the expanding global supply chain that supports the world economy, cargo ships have become an important aspect of everyday life. Almost 80% of world trade is handled by cargo ships which are equal to around 70% of the world trade value each year [1]. Goods from clothing to cars, from food to oil are all transported by sea, holding up the global economy and supporting the lives we live every day. Moreover, the global shipping value in 2019 reached 14 trillion US Dollars [2]. Despite the importance and growth of this industry, its environmental impact on cargo ships is often ignored. Cargo ships released 1,000 Mt CO2 per year, 3% of global CO2 emissions in 2020, and the rate of pollution is set to increase by 120%. While most of this pollution is created through the transport of goods, another area of concern is the pollution caused by cargo ships when they sit idle in traffic jams or backups along trade routes and ports. An example of this phenomenon could be seen in The Ports of Long Beach and Los Angeles, which handled around 10.7 million TEUs in CY 2021 each, are producing 100 tons of smog, or particulate-forming nitrogen oxides per day [3][4]. This level of pollution equals to around more than 6 million cars and has led to an increased rate of asthma and an increased rate of fatality during the Covid-19 pandemic in regions near or next to the ports [5]. This leaves our society in a dilemma, while the shipping industry is tremendously important to both the global economy and daily life, but also is actively destroying the environment and the health of everyday people.

David C. Wyld et al. (Eds): CCNET, AIMLA, CICS, IOTBS, NLTM, COIT - 2023
pp. 57-69, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130405

Existing methods of reducing the impacts of cargo ships on the environment exist in two forms [6]. One, certain shipping companies have started to adopt a strategy referred to as "slow steaming", where ships are purposely sailed at a lower speed than the max speed to save money on fuel, which could slash fuel conception by 59%. [7]. However, this method is not very economically sustainable for most companies, as slower ships mean longer trips, which causes crew wages, maintenance costs, and other expenses to rise [8]. Meaning that slow steaming sometimes becomes financially unsustainable for shipping companies, who are already losing money due to the Covid-19 pandemic. Another solution was to create alternative-powered boats, such as boats relying on wind power or green fuel. Although these technologies do exist, the problem is that investing in these technologies is not commercially viable for many companies, meaning most of these technologies are not mature yet for commercial use. Another challenge is making sure that these solutions don't inherently cause more pollution, as in the case of alternative sources of fuel, which could produce less immersion initially, but produce more dangerous pollutants or be riskier to use on a large scale [9]. For example, some alternative sources of fuel suggested are hydrogen, ammonia, or biofuel, however, each of these fuels has its issues regarding safety such as toxicity or increased risk of fires and price, and are just generally more expensive to use for shipping companies. Moreover, even if these innovations do catch on, it would take years for new ships to be built and adapted by most of the world. In the meantime, a solution is needed that is both low risk both financially and in real life, well also effective enough to significantly reduce pollution caused by cargo ships.

In this paper, we will follow the line of research of "slow steaming" by making ships use less fuel by being efficient. However, instead of going slower, we are exploring if ships could be more efficient by using an Artificial Intelligence model to be more efficient in the water to go faster. By taking existing data that ships have on board regarding the conditions of the weather the AI system is applied to the current conditions to generate the most efficient pass for the boat to take. Inspired by the automated cars developed by some of the biggest companies in the world, the use of artificial intelligence to direct a transportation system is not something new. But, these systems are not designed to be the most efficient, nor are they on the water. First, sensors currently already installed in the modern cargo ship gather data regarding conditions which allows for the system to generate a simulation model reflecting conditions the boat is facing in real-time. Then, the reinforced learning system is applied to the simulation to find the best path for the boat. Then, the system sends the path generated to the control system, where the Captain could apply the path manually through a guidance system, or allow the autopilot to apply the path. Through this, we believe that fuel use could be cut drastically through a financially convenient and easy-to-install system that just needs a computer.

To prove our results, we will demonstrate the effectiveness of the above combination of techniques by comparing it to a regular path used by cargo ships and afterward, compare the distance and average speed to calculate the amount of fuel used during the trip to measure the fuel savings. First, we would find a set route for the boats to travel. Then, by first letting the simulation run the route using our simulation, and without it, we can gather data on the average speed and time needed for travel. Afterward, using the distance and average speed, the amount of fuel used could be gathered by using the average fuel used by cargo ships. Finally, the comparison of the fuel use would allow the demonstration of the effectiveness of the system.

This paper is organized as follows: Section 2 gives details and a list of challenges that we faced during the experiment, design, and creation of the product; Section 3 focuses on the details of our solutions corresponding to challenges that we mentioned in Section 2; Section 4 presents the relevant details and regarding the experiment along with data gathered and conclusions we drew through the data. Following section 4, section 5 presents the related work in Section 5. Finally,

Section 6 gives the conclusion remarks regarding the project and the future work and direction of this project.

2. CHALLENGES

To build the project, a few challenges have been identified as follows.

2.1. How to Build a Realistic Boat Simulation Environment

The first step to building an AI simulation is to build the simulation itself [13]. In our case, that means constructing an accurate boat model and an accurate model that could represent the ocean using data that average ship sensors could gather. However, figuring out how to utilize the tools available in Unity to represent certain conditions and variables in the water is the most important part. The issue is the number of conditions to take into account makes influencing the simulation immensely complicated. The main conditions we took into account are wind conditions, currents, wave conditions, and length of the path. To overcome this, we have combined the effects of the conditions into one single angular tilt of the simulation, which allows us to simulate accurately.

2.2. How to Integrate AI Algorithms in the Simulation Environment

The second challenge encountered when developing the simulation is integrating the AI algorithms into the simulation environment. In the simulation environment, there are two sets of coordinates, the global coordinate and the local coordinate. The global coordinate is the coordinate system of the whole environment, while local coordinates represent the individual simulations. For the simulations to function, the coordinate systems have to match with the AI algorithm, and by adjusting the position of the individual simulations, we connect the AI algorithm with the simulation environment, which allows for the program to function.

2.3. How to Improve the AI Algorithms without Depending on the Specific Environment Settings

Once the AI algorithm is integrated, however, more challenges arise, as through testing and data gathered, there seems to be no significant improvement in terms of paths the AI is generating. After investigating, it seems that the equation used to generate the punishment of the path was not sufficient to improve the performance of the path in a significant and timely manner. The issue with the old equation is that it only takes into account the vessel's final x position rewarding the target location, and not the x coordinate. So when we generate a new equation that does take into account both values, the system is able to better differentiate which path's and good and which are not, drastically improving performance.

3. SOLUTION

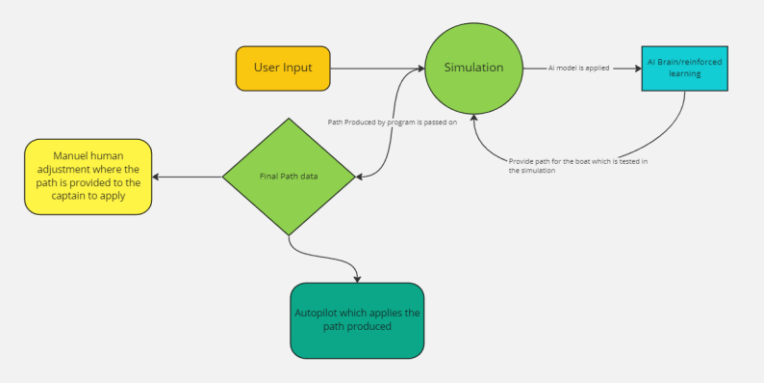

Figure 1. Overview of the solution

This diagram starts at the user input, where data such as wind speed, wave height and length, and current is entered into the system. Then, the data travels to the simulation, where they are used to adjust the simulation through different tilts and lengths of the simulation environment. Next, after the environment is finalized, the AI brain, (produced through reinforced learning), is applied to the simulation environment where it produces a path that is efficient and sufficient for the vessel. This path data is produced in the form of angles and speeds for the vessel to apply. Then there are two options, one where the path is automatically applied by the autopilot, which controls the speed and the direction of the vessel and could adjust the boat according to the new path directions, or the second option is a manual override, where the captain or senior officer of the ship could choose to apply the path manually or override it in favor of a different path.

The Components come together as a sort of cycle of information. Meaning that each passing and transformation of data directly affects the data produced in the next component. The entire system starts with the collection of real-world data in the form of weather and surface conditions. which is converted into a simulation environment built using Unity software. Then the AI brain is applied to the simulation which focuses on finding the optimal path for the particular environment. Afterward, that information is passed in the form of directions for autopilot or a captain.

The Unity Simulation environment in particular allows for the accurate simulation of conditions on the water, however, instead of directly simulating the conditions on the water which would have needed years of research, we are able to instead represent the effects of conditions in other ways such as tilts and speeds of the vessel. As seen in figure 2, the boat is represented as a white sphere that is launched at a constant speed across the platform. The platform is surrounded by white bars, which represent the boundaries and finish of the route. If the ball contacts the three walls behind it and to the left or right, it is counted as leaving the route meaning that it failed. The wall in front represents the finish or goal, which means that the path was able to reach the requested location efficiently. Using the application which we built, the system is able to take input, as shown in figure 3, in the form of boat speed (in knots), wave height and length, and travel distance, then translate the input into ball speed, platform tilt, and distance between the goal and ball respectively. This allows for both human input or a direct connection to an automated system that would collect data through sensors.

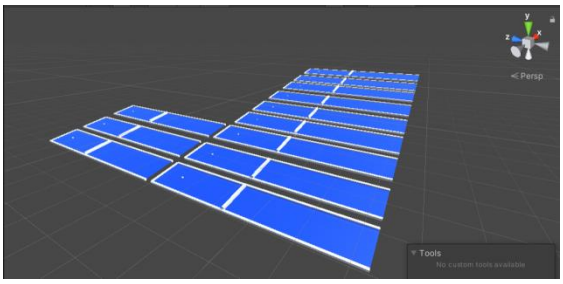

Figure 2. View of the simulation environments

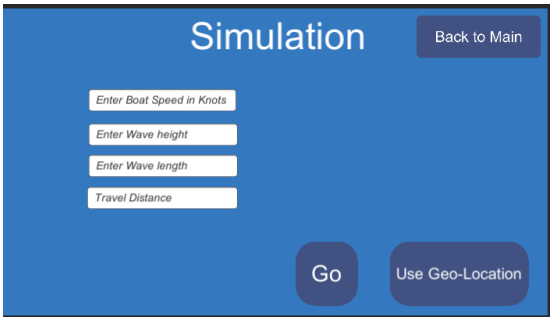

Figure 3. The input format for the application

When simulating, the ball would be launched through input produced by the Ml agents AI which would be further explained in the next section. This input comes in the form of two decimal numbers ranging from 0 to 1. This represents the angle at which the ball (vessel) would be launched, simulating the change in heading a vessel would undergo, then the speed would be adjusted according to the input. This can be seen in figure 4, first, the speed of the ball is set according to the vessel's speed, then the force is applied using input from the Genetic algorithm.

```
public override void OnActionReceived(ActionBuffers actions)
{
    //Debug.Log("OnActionReceived: " + " X: " + actions.ContinuousActions[0] + "  Y: " + actions.ContinuousActions[1]);
    // Debug.Log("X: " + actions.ContinuousActions[0]);
    // Debug.Log("Y: " + actions.ContinuousActions[1]);
    float moveX = actions.ContinuousActions[0];
    float moveZ = actions.ContinuousActions[1];

    // float moveX = 0f;
    // float moveZ = 1f;

    float moveSpeed = infos.b*100f;
    // Debug.Log("Foced: " + isForced);
    // Debug.Log(DateTime.Now - startTime);
    if (DateTime.Now > startTime.AddSeconds(10))
    {
        Debug.Log("Too slow! EndEpisode");
        SetReward(-1f);
        floorMeshRenderer.material = loseMaterial;
        EndEpisode();
    }
    if (!isForced)
    {
        Debug.Log("Applied the force!");
        Debug.Log("OnActionReceived: " + " X: " + actions.ContinuousActions[0] + "  Y: " + actions.ContinuousActions[1]);
        m_Rigidbody.AddForce(new Vector3(moveX, 0, moveZ) * moveSpeed);
        isForced = true;
    }
    // transform.localPosition += new Vector3(moveX, 0, moveZ) * Time.deltaTime * moveSpeed;
}
```

Figure 4. Code for action Simulator in accordance with input data

In order to construct the AI brain for the system, an AI simulation would be needed to be implemented first. A simulation allows the AI brain to "develop" in a way, or in more technical terms, gather data regarding paths for different types of simulated conditions which it can use to apply to paths it may or may not have seen before.

The most important part of this simulation would be the model of Artificial Learning chosen for this project. Due to the goal and needs of the simulation, a Reinforcement learning Algorithm was chosen to be created using the Ml agent AI system. This choice was made due to two factors. One, the Algorithm would smoothly integrate into the existing simulation model as described in section 3.2.1. Two, the system of the algorithm, which in simple terms is an award/punishment, means that we can quickly separate efficient and inefficient paths in the simulation. In order to interpret the algorithm, a couple of changes need to be made to the algorithm. First, the Ml Agents AI produces a path in the value of one x and on the y value using a t first random data. Then, that produced path is passed to the simulation which is able to two important pieces of data.

A key piece of this process is the ml agent Ai, which handles the actual learning aspect of the AI. In simple terms, ml-agents take environmental feedback in terms of awards and punishments to improve its future decisions. So at first, the agent might move randomly with no particular direction. However, as soon as the agent commits an action with leads to an award or punishment, it is able to shape its decision-making quickly. By going through multiple interactions of awards/punishments, the agent is able to quickly make decisions based on past “experiences”. Even if the environment changes, through past experience, the AI is able to still make decisions based on environmental data and improve with each new environment.

What makes this AI unique is its use of Reinforcement learning, which allows it to remember past actions and improve based on results. By assigning a positive value to desired behaviors and negative values to undesirable behaviors or outcomes, the algorithm is designed for seeking long terms and maximum awards. This allows the algorithm to focus on finding the optimal behavior quickly and works best in an environment where the agent is allowed to “explore” the environment on its own and create data regarding the environment. Which makes it perfect for our use. By making early mistakes that may lead the agent in the complete opposite direction as what the user wants, the vessel is able to quickly correct itself using feedback and find the optimal route.

In order to integrate into the AI algorithm, first the agent needs to be created, this agent would act as the vessel for our use and would carry out the testing and creation of the optimal path. In order to incorporate ml agents, there are several things we would first need to adjust. For one, we would need to decide what type of space type we want for our action, essentially, this decides whether the action produced is the whole number or a float (decimal). For this project, we choose a continuous space type, which allows getting a more diverse and accurate range of angles for the vessel to take. We also would adjust the max step to 15000, the max steps represent how many steps or paths in our case the AI is allowed to generate to find the optimal path. Then, by setting the actions to heuristic mode, we allow the AI to fully run on its own and begin testing.

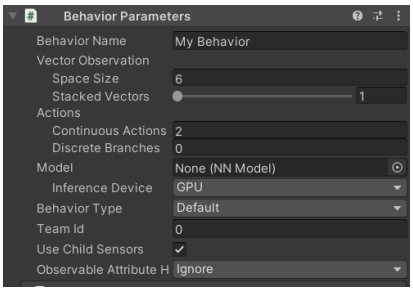

Figure 5. Behavior Parameters of the Agents

Once everything is set up, the agent still needs the reinforcement learning algorithm to learn from the simulation. This relies on the code seen in figure 6, which uses data collected from each

simulation, regarding both the position of the ball, and whether or not the path worked is adequate for the path specified from the start. Once the data is imputed, the code assigns an award or punishment based on the performance of the vessel, is the path is optimal and does reach the target location, it is assigned the award value of 1, however, if it does not reach the target location, then the punishment is calculated using the equation

$$P = -1 * ((z + L/2)/(L/2)) * ((x + w/2)/(w/2))$$

Where z and x represent the vessel's final position along the path z-axis and x-axis respectively, while L and W represent the goal's distance away from the initial position of the vessel and the width of the acceptable final location zone respectively.

The purpose of the equation is to further refine the process of training the AI to further develop better paths. Although all of the paths deemed unacceptable would not reach the target location, in the initial phases of training, there is a very low chance that the path generated would be successful, meaning that to speed up training, the AI would need to be able to improve upon the initially unsuccessful path. What the equation allows us to do is assign different levels of punishment to different levels of failure. For example, if the vessel ends up closer to the target than in previous attempts, then it would receive less punishment, essentially promoting that path above others. The initial separation of success would mean that the computer can more quickly create paths that have a higher chance to be successful.

```
private void OnTriggerEnter(Collider other)
{
    Debug.Log("Triggered!");
    if (other.TryGetComponent<Goal>(out Goal goal))
    {
        SetReward(+1f);
        floorMeshRenderer.material = winMaterial;
        Debug.Log("Win!");
        EndEpisode();
    }
    if (other.TryGetComponent<Wall>(out Wall wall))
    {
        float distance = (transform.localPosition.z + 31.3f) / 31.3f;
        float distance2 = ((transform.localPosition.x + 4) / 4f);
        distance2 = Math.Abs(distance2);
        // Debug.Log("Distance: " + distance);
        SetReward(-1f*distance*distance2);
        floorMeshRenderer.material = loseMaterial;
        Debug.Log("Lose!");
        EndEpisode();
    }
}
```

Figure 6. Reinforced Learning Algorithm

4. EXPERIMENT

In order to test the efficiency and accuracy of our system, we would need to ensure that the system works for all types of conditions on the water. This means simulating conditions commonly seen on the ocean, calm seas, and rough conditions. Furthermore, to test the flexibility of the system, we would also be testing the viability of the system to generate paths for longer distances. For those reasons, we would be testing the algorithm under three situations, a flat and short course, then we would be testing a longer course, and finally a longer course along with a larger degree of elevation. By running each condition through 15000 steps, we will gather data on the Accuracy and efficiency of the path.

4.1. Experiment 1:Flat and Short Course

For the first of our experiments, we used the standard design to simulate calm seas with a short distance of travel as seen in Figure 7. While not challenging for the human operator, the importance of the AI to be operated in any condition means even the calmest of easy need to be tested.

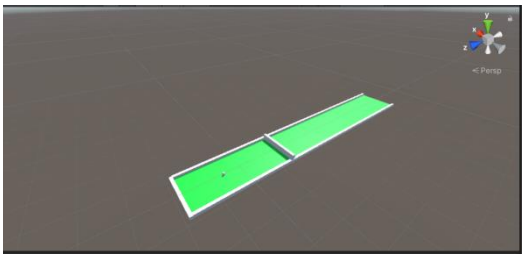

Figure 7. Flat and short course

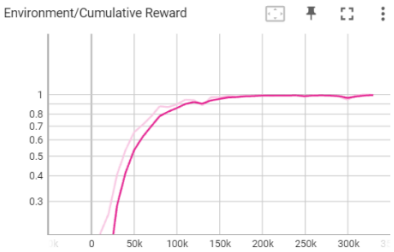

Figure 8. Environment/Cumulative reward

A. Accuracy

Despite an initial poor performance with an average reward of less than 0.3, the AI quickly finds an optimal path according to the judging system, reaching the upper limit of 1. This initial experiential growth demonstrates the AI quickly identifying which paths are optimal and which are not. Then, the AI was able to further pick out the most optimal path out of the chosen ones which can be seen through a curve from 50k to 100k step value. The resulting accuracy of close to 1 shows that the AI has successfully found the correct path in less than 150k steps.

B. Duration (steps)

A step, which is an atomic change of the engine that happens between Agent decisions, serves as our way of measuring the amount of work an agent or AI has spent developing an optimal path for a certain environment. Since the AI works very fast, steps instead of seconds are a better way for us to measure the performance of the agent.

C. Irregularities in the data

Certain irregularities can be found in the data for many reasons such as a slight decrease in performance. Such irregularities can be explained by the slow down of the computer or as the AI searches for more optimal paths, it can try to develop its own through existing paths, which can lead to improved performance or decreased performance as seen in these bumps.

4.2. Experiment 2

For the design of the second type of course, the width of the course and the position of the ball has not changed. However, instead of the finish being 31.5m away from the agent, the finish is moved to 51.5 m away shown in figure 9. This course is designed to test the flexibility of the program regarding distance and if it can adjust to the different needs of the captain, such as a longer target location.

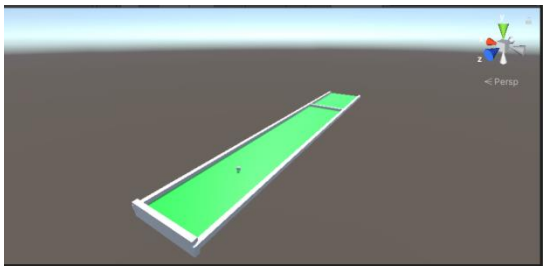

Figure 9. Flat and long course

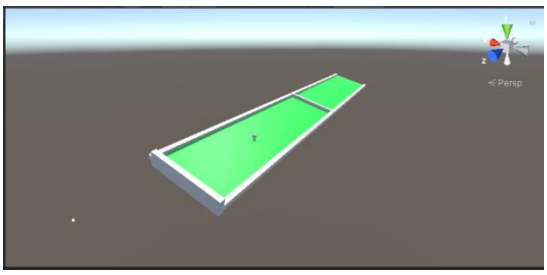

Figure 10. Tilted and Short Course

After the Simulation, the data provided in figure 10 showed another efficient performance. With a quick improvement, although slower than type 1, the AI is able to adapt to the different environment in roughly 150k steps, which as mentioned in section 4.1 are atomic changes made by the agent to attempt to adapt to the environment. The upper limit of 1 reached at 300k means that the AI has been able to find the most optimal path, which means that the AI is able to adapt to the environment adequately.

4.3. Experiment 3: Tilted Course

Type 3 courses represent rough conditions on the ocean which the AI would need to be able to adapt to in order to be adequate for service on vessels for use anywhere on the ocean. In order to simulate the tilt experienced cargo ships, the platform is tilted by 15 degrees as seen in figure 14.

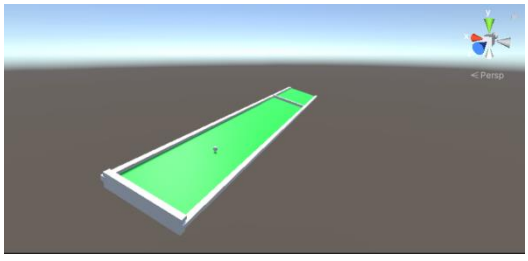

Figure 11. Tilted and long course

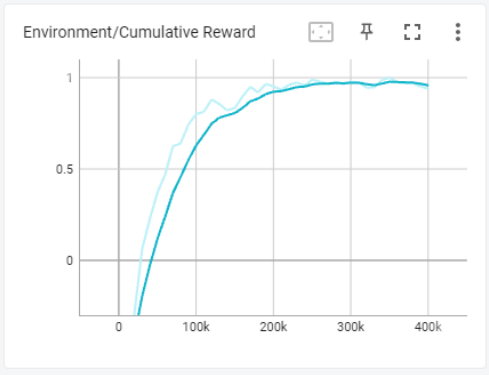

Figure 12. Environment/Cumulative Reward for type 3 simulation

After this simulation, the data in figure 11 shows that the AI was actually able to adjust to the tilted platform relatively quickly, finding optimal paths after 100k steps and reaching the upper limit of 1 around 225k steps. Again, the initial experiential growth represents the separation of non-optimal and optimal paths, while the curve before entering the upper limit represents the final improvement to the remaining paths. the Ai's ability to quickly adjust to different tilt would mean that it would be proficient in a real world setting.

4.4. Experiment 4

Type 4 being the last type is a combination of both type 2 and 3. This particular simulation serves as a stress test for the simulation. By combining both a further target distance and tilt, the environment tests if the simulation could efficiently adapt to both types of environments at the time time, and if the combination of conditions affect performances. In order to simulate the rough conditions and longer travel distances, the simulation environment has been tilted and extended to match the environments the experiment requires. So with a distance of 51.5 to cover and a 15 degree tilt as seen in figure 13, the environment would provide an adequate challenge for the ai

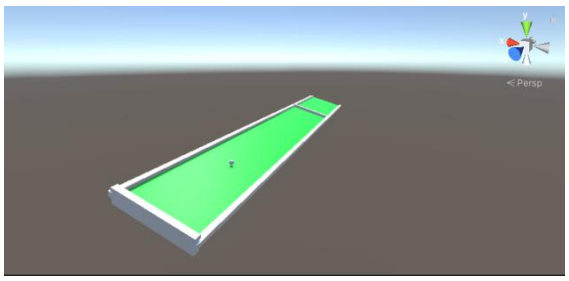

Figure 13. Tilted and long course

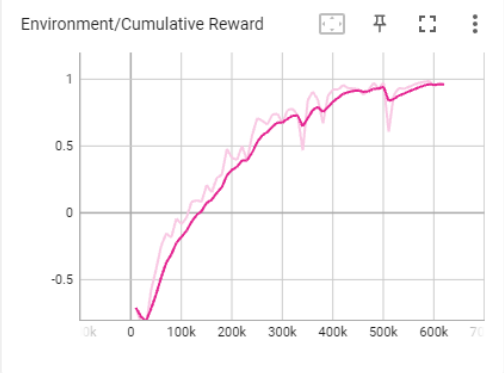

Figure 14: Environment/Cumulative Reward for type 1 simulation

As could be seen in Figure 14, the performance of the AI is certainly not as efficient as ones seen before, however, the AI is still able to find optimal paths after 250k steps and reach the upper limit of 1 around 450k. Which is performance expected for the complexity of the environment compared to previous ones. Although more irregularities in the data do exist due to the simulation's complexity, the ai's ability to reach the upper limit means that the AI is optimal and adequate for use on a real-world level.

Through the result shown through data collected on the four simulations, it becomes apparent that the machine learning algorithm can adequately respond to different types of environments. By finding the most optimal paths for a cargo ship, the ship is able to minimize its pollution impact but to use the same or even less fuel than before. Moreover, by being more efficient, the travel time gets cut dramatically over time, which further limits the pollution impact and lessens the strain on the global economy by reducing the chance of traffic jams due to late arrivals and allowing for better planning.

The design of the experiment with 4 different simulations not only ensures we cover all real world possibilities, but also ensures that we have enough samples to test the viability of the solution scientifically. By collecting trends in the graphs, we are able to notice the general trend of the growth for each environment. By measuring through their performance and steps, we are able to both evaluate general performance and irregularities that allow opportunities or improvement.

5. RELATED WORK

Zhang, M.et al presented a probability model that assesses the effects of human errors when utilizing an autonomous vessel [10]. They chose to focus on the relation between the human component of the human-autonomous interaction as part of an autonomous system. At the same time, our project focuses on the autonomous element. Moreover, while their paper focused on the probability of events that have already happened and analyzed performance, our paper focused on the creation of paths for a vessel to take which would be the most optimal. While the analysis of performance is essential when judging the performance, their work's focus on probability does not provide much insight into how the system could be improved but only highlights the problem through probability. Our project on the other hand focuses on finding a solution to the issue of inefficiency in relation to the environmental impact.

Kooj, C. et al investigated different effects regarding automated ships, and that is the effects on the crewing level due to the system [11]. Since an automated system would affect the tasks that the crew would need to handle, this paper investigates how an automated system would affect the

crew during sailing time, arrival, and departure of the cargo ships. This paper is an important aspect of developing new technology, especially regarding the shipping industry, as even though technology might be developing quickly through research like ours regarding the autonomous system itself, the effects of its implementation on ships and the shipping system as a whole can not be ignored.

Sui, S. et al investigate alternative methods of reducing the environmental impact and emissions of cargo [12]. The main methods they chose to focus on were alternative fuels such as liquefied natural gas (LNG) and the use of hybrid engines. While the focus of both of our papers is essentially on researching ways to reduce the impact of the shipping industry on the environment, the methods chosen are radically different. Their investigations focused on the idea of replacing emission-producing ships entirely with more environmentally friendly options. On the other hand, mine focuses on finding a way to improve current usable ships through an algorithm-driven AI which would make ships more efficient through their path.

6. CONCLUSIONS

In this project, we have proposed an algorithmic solution to the issue of environmental impact through cargo ships using machine learning and environmental simulation [14][15]. A platform designed to be used directly on shipboard computers has been proposed and created. Once a sufficient amount of data and training have been processed by the genetic algorithm, the system is able to take available data regarding conditions surrounding a vessel, and generate a path that would provide optimal efficiency, thus reducing fuel use, which leads to fewer emissions overall. Experiments show that the program can indeed adapt to different environments efficiently, and produce stable results quickly.

However, it is important that the technology produced in this paper has not yet been widely tested in real-world situations or adopted widely. Moreover, the technology does rely on a simulation that could be described as simple, which could be improved upon to be more accurate.

To address these issues, future work would be targeted toward improving the accuracy of the simulation and demonstrating the viability of our methods in real-world situations. In addition, improvements to the code and application to get them ready for real-world use would be needed. Features such as an interactive simulation or implementation of fluid dynamics for different models of ships would be needed.

REFERENCES

[1] Jacks, David S., and Krishna Pendakur. "Global trade and the maritime transport revolution." The Review of Economics and Statistics 92.4 (2010): 745-755.
[2] International Chamber of Shipping. "Shipping and World Trade: Driving Prosperity." (2021).
[3] Rosoff, Heather, and Detlof Von Winterfeldt. "A risk and economic analysis of dirty bomb attacks on the ports of Los Angeles and Long Beach." Risk Analysis: An International Journal 27.3 (2007): 533-546.
[4] National Research Council. Clean ships, clean ports, clean oceans: Controlling garbage and plastic wastes at sea. 1930.
[5] Mitchell, E. A. "International trends in hospital admission rates for asthma." Archives of disease in childhood 60.4 (1985): 376-378.
[6] Lipsitt, Jonah, et al. "Spatial analysis of COVID-19 and traffic-related air pollution in Los Angeles." Environment International 153 (2021): 106531.
[7] Wiesmann, Andreas. "Slow steaming–a viable long-term option." Wartsila Technical Journal 2 (2010): 49-55.

[8] Meyer, Jasper, Robert Stahlbock, and Stefan Voß. "Slow steaming in container shipping." 2012 45th Hawaii International Conference on System Sciences. IEEE, 2012.
[9] Kim, Hyungju, Kwi Yeon Koo, and Tae-Hwan Joung. "A study on the necessity of integrated evaluation of alternative marine fuels." Journal of International Maritime Safety, Environmental Affairs, and Shipping 4.2 (2020): 26-31.
[10] Zhang, Mingyang, et al. "A probabilistic model of human error assessment for autonomous cargo ships focusing on human–autonomy collaboration." Safety science 130 (2020): 104838.
[11] Kooij, Carmen, and Robert Hekkenberg. "Towards Unmanned Cargo-Ships: The Effects of Automating Navigational Tasks on Crewing Levels." Available at SSRN 3438144 (2019).
[12] Sui, Congbiao, et al. "Fuel consumption and emissions of ocean-going cargo ship with hybrid propulsion and different fuels over voyage." Journal of Marine Science and Engineering 8.8 (2020): 588.
[13] Fu, Daniel, and Ryan Houlette. "Putting AI in entertainment: An AI authoring tool for simulation and games." IEEE Intelligent Systems 17.4 (2002): 81-84.
[14] Ronen, David. "Cargo ships routing and scheduling: Survey of models and problems." European Journal of Operational Research 12.2 (1983): 119-126.
[15] Marans, Robert W., and Daniel Stokols, eds. "Environmental simulation: Research and policy issues." (2013).

AUTHORS

Chenyu Zuo is a second-year High school student at Sage Hill School located in Newport Beach, California. He is currently studying Multivariable Calculus and Physics. He is intreated in Coding, Sailing and wishes to one day explore the stars.