Swarm Driving with Deep Learning

Md Limon Apu
Dept. Electrical Engineering
Hamm-Lippstadt University of Applied Sciences
Lippstadt , Germany
md-limon.apu@stud.hshl.de

***Abstract*—Swarm driving is a technology that involves multiple autonomous vehicles (AVs) working together to achieve a common goal. Deep learning is a type of machine learning that can enable AVs to make informed decisions based on data from their surroundings. In swarm driving, deep learning can be used to enable AVs to work together as a coordinated team using multi-agent systems or decentralized control in distributed systems. This can improve the efficiency and safety of the swarm. Overall, the use of deep learning in swarm driving can enhance the capabilities of AVs to work together and achieve a common goal.**

***Index Terms*—Swarm driving, Deep learning,Platooning.**

I. Introduction

There has been a long history of swarm-based algorithms used in robotics and autonomous system, and they continue to be used today. The concept of "swarm intelligence" was first proposed by Gerardo Beni and Jing Wang in 1988 as a way of describing cellular robotic systems. This term was chosen by Beni because he had found that the group of cellular robots he was working on had some special characteristics, in fact, they are found in swarms of insects, such as centralized control, lack of synchronicity, and simple and (quasi) identical components. [2] And also Swarm driving is a transportation concept that involves the use of a large number of autonomous vehicles that are coordinated to move as a single unit, or "swarm," in order to achieve a specific goal or objective. This concept has the potential to revolutionize the way we think about transportation, as it could enable the efficient and safe movement of people and goods over long distances.
Deep learning algorithms are ideal for swarm driving applications because they can process large amounts of data and learn to recognize patterns and make decisions based on that data. This enables the swarm's vehicles to adapt to changing environments and respond to new situations in real time. [3] Deep Learning has many advantages, but there are nuances to be aware of, and techniques like Neural Network can be fiddly at times. In this paper, we will look at how Deep Learning techniques can be used in swarm systems to create fast, efficient, and reliable systems that can solve very complex problems. To accomplish this, we will explain what swarm systems are and where they are used in the first section. Following that, in the second and third sections, we will introduce various communication methods and use cases. In the final section, we will discuss the use of Deep Learning in swarm systems, as well as some proposed solutions. Finally, we are going to discuss an implementation of said techniques in a simulation of a swarm system.

II. Methods

There are several communication methods that can be used to implement swarm-driving technology, including the following:

A. Vehicle-to-vehicle (V2V) communication

Vehicle-to-vehicle (V2V) communication is a key technology used in swarm driving to enable the coordination of multiple autonomous vehicles (AVs). V2V communication allows AVs to communicate with each other in real time, sharing information about their location, speed, and other sensor data. This enables the AVs to work together as a coordinated group to achieve a common goal, such as delivering goods or performing search and rescue operations. [7]

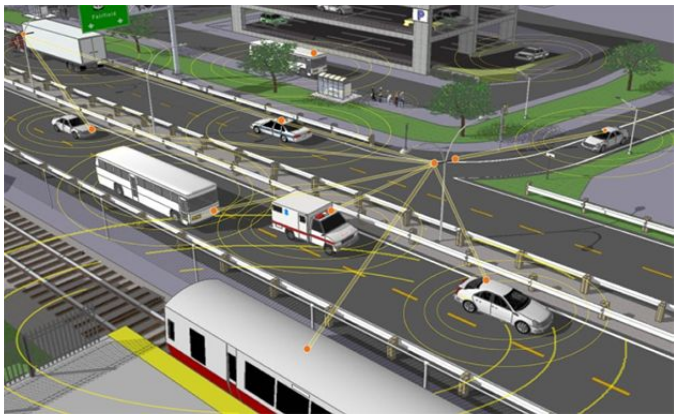


Fig. 1. V2V Communication system [5]

There are several different V2V communication technologies that can be used in swarm driving such as -

1) Dedicated short-range communication (DSRC): This technology uses radio frequency (RF) communication to enable direct communication between AVs over short distances.
2) Cellular communication: This technology uses cellular networks to enable AVs to communicate with each other and with the broader internet.
3) Ad-hoc networking: This technology allows AVs to form a temporary network without the need for infrastructure,

using wireless communication protocols such as WiFi or Bluetooth.

B. Vehicle-to-infrastructure (V2I) communication

Vehicle-to-infrastructure (V2I) communication is an important technology for enabling swarm driving, as it allows the vehicles to exchange information with nearby infrastructure and adjust their behavior accordingly.

In a swarm driving system, V2I communication can be used to transmit information about the location, speed, and direction of the vehicles to nearby infrastructures, such as traffic lights and road signs. The infrastructure can then use this information to adjust the traffic signals or provide other guidance to the vehicles. [4]

Fig. 2. Decision tree

V2I communication can also be used to transmit information about traffic conditions, weather, and other factors that may affect the vehicles' behavior. For example, if a vehicle detects an obstacle in the road, it can use V2I communication to alert the other vehicles in the swarm and the surrounding infrastructure, allowing the vehicles to adjust their speed or change lanes to avoid the obstacle.

C. Centralized control

This involves the use of a central control system to coordinate the movements of the vehicles in the swarm. The central control system can receive input from the vehicles, analyze the data, and generate commands that are transmitted back to the vehicles to guide their movements.

D. Decentralized control

This control system of decentralized algorithms enables the vehicles in the swarm to coordinate their movements without the need for a central control system. This can be achieved using techniques such as consensus algorithms, which enable the vehicles to reach a common decision based on the information they have available.

III. Algorithms For Swarm Driving

Swarm driving is a type of autonomous driving in which a group of vehicles operates collaboratively to achieve the accuracy and results that are expected. There are a variety of algorithms that can be used to enable swarm driving to operate to its best capabilities in various use cases.

- Behavior-based algorithms: Behavior-based algorithms are a type of algorithm that can be used in swarm driving to allow vehicles to work together to accomplish a common goal. The algorithm is based on simple rules that govern how each vehicle should behave in various situations, such as following the vehicle in front of it, avoiding collisions with other vehicles or objects, and keeping a safe distance from other vehicles. The algorithms are easy to implement, and they do not require a lot of data or computational resources. However, they can be limited in their ability to adapt to complex or dynamic environments, as they rely on predefined rules that may not always be sufficient to handle all possible scenarios.

  To implement a behavior-based algorithm for swarm driving, first, define the rules that each vehicle should follow, and then program the vehicle to follow those rules based on its sensors and other inputs. For example, you could program the vehicle to detect when the vehicle in front of it stops and apply the brakes accordingly.

- Multi-agent systems: Multi-agent systems can be more adaptable and flexible than behavior-based algorithms because they allow vehicles to respond to changes in the environment in real-time and adjust their behavior as needed. They do, however, necessitate more data and computational resources because they rely on vehicle communication and frequently involve more complex decision-making processes. [9]

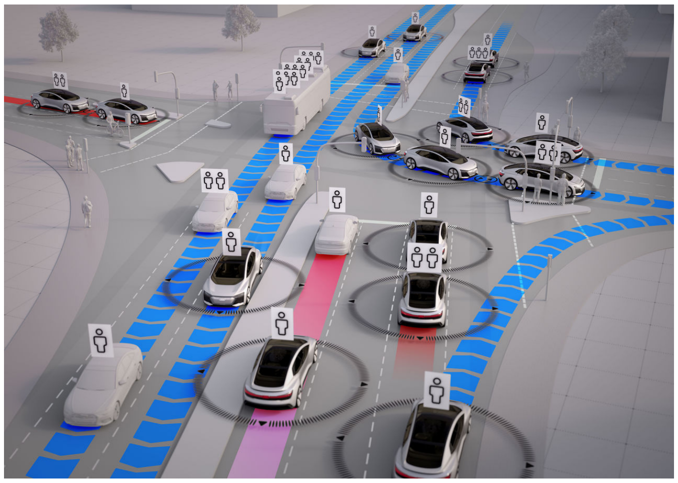


Fig. 3. Agent base algorithm

  To implement a multi-agent system for swarm driving, the system must first define the communication protocols and decision-making rules that each vehicle must adhere to, and then program the vehicles to execute those rules based on information received from other vehicles. Also need to consider communication range and reliability, as well as the impact of communication delays on overall system performance.

- Machine learning algorithms: In order for vehicles to learn from data and gradually improve their performance, machine learning algorithms are a type of algorithm that can be used in swarm driving. These algorithms use data to learn how to perform tasks like predicting other vehicles' actions or identifying hazards in the environment. A machine learning algorithm, for example, could be trained to recognize pedestrians or other objects on

the road and adjust the vehicle's behavior accordingly. A machine learning algorithm, on the other hand, could be used to predict the likely trajectory of other vehicles based on previous observations, and to adjust the vehicle's own trajectory to avoid collisions.
Because they can learn from a variety of data sources and adapt to changing environments over time, machine learning algorithms can be extremely powerful and adaptive. However, it requires a large amount of data and computational resources to train and execute effectively, and it may be more difficult to implement than other types of algorithms.
To implement a swarm-driving machine learning algorithm, the system would need to collect and prepossess a large data set of relevant data, such as sensor readings or road images. This information would then be used to train a machine learning model, such as a neural network or decision tree, to perform a specific task, such as object recognition or trajectory prediction.

- Optimization algorithms: Optimization algorithms can be a powerful tool for enabling efficient and coordinated forms of swarm driving, but they may require more sophisticated hardware and software infrastructure and may be more complex to implement than other types of algorithms. Swarm driving can make use of optimization algorithms to find the best solution to issues like minimizing fuel consumption or maximizing throughput. Given a set of constraints and objectives, these algorithms use mathematical optimization techniques to find the best solution to a problem. [10]
  An optimization algorithm, for example, could be used to determine the best route for a group of vehicles to take in order to deliver goods to a set of destinations in the shortest amount of time. An optimization algorithm could also be used to coordinate the movements of a group of vehicles in order to reduce fuel consumption or congestion on the road. Given a set of constraints and objectives, optimization algorithms can be very effective at finding the best solution to a problem. However, they can be complex and computationally intensive to implement, and they may necessitate a large amount of data and computational resources to function properly.
  To implement an optimization algorithm for swarm driving, first define the problem you want to solve, such as finding the best route for a group of vehicles or minimizing fuel consumption. The problem would then need to be formulated as a mathematical optimization problem using techniques like linear programming or mixed-integer programming. Finally, an optimization solver would be required to find the best solution to the problem.

IV. Use Cases

Swarm driving, also known as platooning, connected, and automated driving, is a technology that enables a group of vehicles to travel in unison. These vehicles can travel safely and efficiently as a group by communicating with one another and with infrastructures such as traffic lights and road signs. Long-distance trucking is one possible application for swarm driving. Fuel efficiency can be increased by allowing a convoy of trucks to travel closely together, as the lead truck can create a draft that reduces drag on the following trucks. This not only saves fuel but also lowers emissions and may save money. [1]

There are several potential use cases for swarm driving, including:

- Transportation: Swarm driving could be used to improve the efficiency and reliability of transportation systems, such as buses or delivery trucks, by allowing vehicles to work together to optimize routes and avoid congestion.
- Agriculture: Swarm driving could be used in agriculture to automate tasks such as planting, watering, and harvesting crops, potentially reducing the need for manual labor.
- Environmental monitoring: Swarm driving could be used to deploy a fleet of sensors or other monitoring devices to track environmental conditions over a large area, such as air quality or water quality.
- Disaster response: Swarm driving could be used to quickly and efficiently deploy resources to disaster-stricken areas, such as delivering aid or supplies, or searching for survivors.

V. Implementation in Deep Learning

Truck platooning, also known as swarm driving, is a new technology that enables a convoy of trucks to drive in formation, with a single driver at the front in command. This technology is intended to reduce fuel consumption while also improving safety and efficiency.

Truck platooning employs a wireless communication system and radar sensors to enable trucks to follow the lead vehicle in a convoy. The lead driver controls the speed and direction of the other trucks in the platoon, while the other drivers act as followers with no direct control over their speed or direction. The trucks can stay extremely close together, with the lead vehicle controlling the convoy's speed and direction and the other vehicles following.

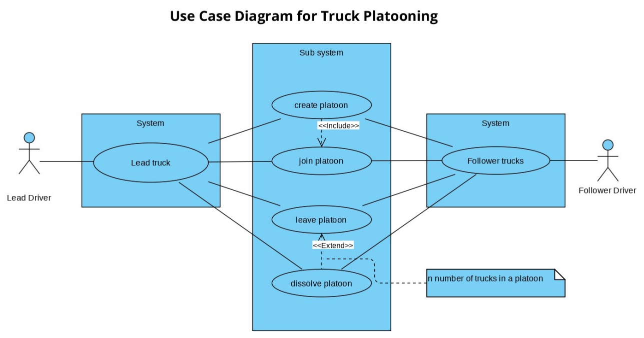
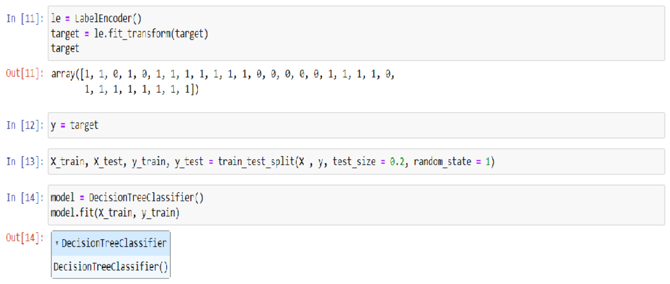


Fig. 4. Use case for a truck platooning

In this use case the driver which is connected to the lead truck at right side and the follower driver which is connected to

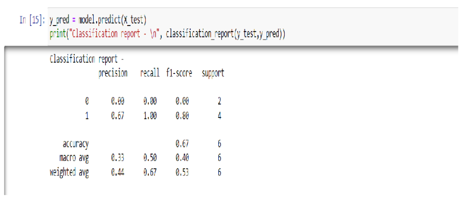
followers trucks on the other side. Which are further connected to the subsystem in which everything is happening from creating platoon to dissolving platoon. The lead truck will ask following truck drivers to create platoon they will join the platoon and vice versa.
For implementing the platooning scenario, scikit learning algorithm has been implemented to determine the leader vehicle and other possible actions. To implement all those actions creating a dataset of various factors such as Distance, Fuel, Body characteristics, sensors, match of the route, and efficiency in platoon. Also the algorithm used the analytic to make some basic analysis from that data set. After this, keeping the goal in mind, creating a machine learning model and training it to using the decision trees algorithm. [8]



```
In [11]: le = LabelEncoder()
         target = le.fit_transform(target)
         target
Out[11]: array([1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,
                1, 1, 1, 1, 1, 1, 1, 1])

In [12]: y = target

In [13]: X_train, X_test, y_train, y_test = train_test_split(X , y, test_size = 0.2, random_state = 1)

In [14]: model = DecisionTreeClassifier()
         model.fit(X_train, y_train)
Out[14]: ▾ DecisionTreeClassifier
         DecisionTreeClassifier()
```

Fig. 5. Decision tree for training dataset

For creating a machine learning model, it uses label encoder. Here, for this, it must fit some features on x-axis and some on y-axis. It depends on what features we need to explore,we need to fit that on y-axis. Here, we fitted efficiency feature on y-axis and rest of the features on x-axis. Then declared the train and test size for the model. We used 80-20 ratio for train and test sample size. And after this we declared the algorithm for model that is Decision trees classifier. Decision Tree is a Supervised Machine Learning Algorithm that uses a set of rules to make decisions, similarly to how humans make decisions. This is how we created and trained the model.
Using model.predict to predict the accuracy of the model. In the figure 6 we can see all the statistical details of the models including accuracy. The accuracy for the model is 67% which is way less than expected. For decision trees average expected accuracy should be more than 90%.



```
In [15]: y_pred = model.predict(X_test)
         print("Classification report - \n", classification_report(y_test,y_pred))

Classification report -
               precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.67      1.00      0.80         4

    accuracy                           0.67         6
   macro avg       0.33      0.50      0.40         6
weighted avg       0.44      0.67      0.53         6
```

Fig. 6. Modeling Decision tree for training data set

The decision tree which includes gini score, samples, values and class. This is how a machine learning model works and also trained it using an appropriate Decision trees classifier algorithm. We used concepts of machine learning, python and data science libraries to create, test and train a machine learning model with accuracy of 67%. We used the feature match of the route and other features which will determine the leader truck. The python code are just snippets from the code. Entire source code and dataset can be found uploaded on github.

```
plt.show()
```

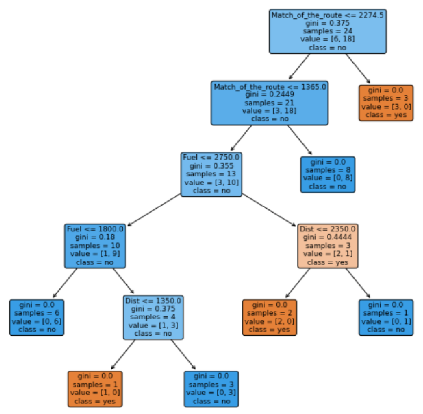
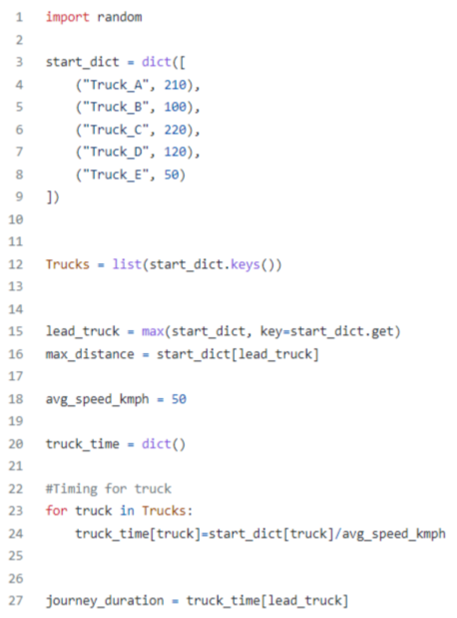


Fig. 7. Decision tree

A system was created(Fig.8) with trucks utilizing the random module in Python. All trucks in the platoon traveled at a speed of 50 kilometers per hour. The timing for the trucks was also established, with a half-hour break period for each truck. The system used a list to store the values of the trucks which had been defined using dictionaries and assigned a distance to cover.

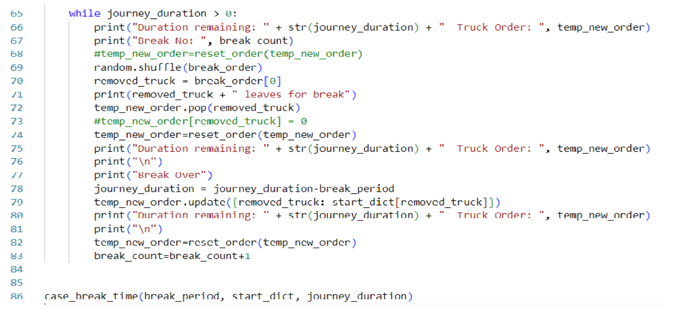


```
import random

start_dict = dict([
    ("Truck_A", 210),
    ("Truck_B", 100),
    ("Truck_C", 220),
    ("Truck_D", 120),
    ("Truck_E", 50)
])


Trucks = list(start_dict.keys())


lead_truck = max(start_dict, key=start_dict.get)
max_distance = start_dict[lead_truck]

avg_speed_kmph = 50

truck_time = dict()

#Timing for truck
for truck in Trucks:
    truck_time[truck]=start_dict[truck]/avg_speed_kmph


journey_duration = truck_time[lead_truck]
```

Fig. 8. Trucks data

VI. Simulation

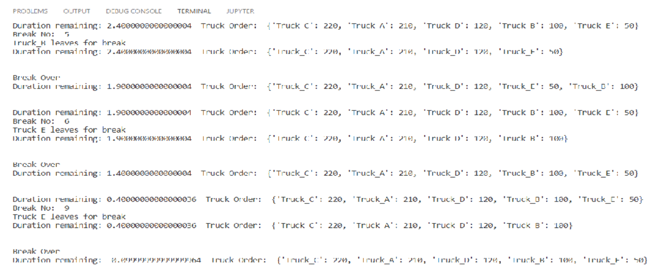
For this simulation the trucks in a platoon are always arranged in the order of the distance left to cover. [6] The truck with the longest remaining distance is always the lead truck, with the others following in descending order of remaining distance.Truck A, Truck B, Truck C, Truck D, and Truck E are the names of the trucks. Assume truck C wants to take a break and leave the platoon. Using the random shuffle function to remove truck C at random from the platoon, and we removed it with '.pop'. The remaining trucks in a platoon then re-arrange themselves based on the distance left. Assume that if the left truck C wants to rejoin the platoon, it will match the platoon's speed and join them at the end. Then it will rearrange itself based on the distance left to cover.



```
    while journey_duration > 0:
        print("Duration remaining: " + str(journey_duration) + "  Truck Order: ", temp_new_order)
        print("Break No: ", break count)
        #temp_new_order=reset_order(temp_new_order)
        random.shuffle(break_order)
        removed_truck = break_order[0]
        print(removed_truck + " leaves for break")
        temp_new_order.pop(removed_truck)
        #temp_new_order[removed_truck] = 0
        temp_new_order=reset_order(temp_new_order)
        print("Duration remaining: " + str(journey_duration) + "  Truck Order: ", temp_new_order)
        print("\n")
        print("Break Over")
        journey_duration = journey_duration-break_period
        temp_new_order.update({removed_truck: start_dict[removed_truck]})
        print("Duration remaining: " + str(journey_duration) + "  Truck Order: ", temp_new_order)
        print("\n")
        temp_new_order=reset_order(temp_new_order)
        break_count=break_count+1


case_break_time(break_period, start_dict, journey_duration)
```

Fig. 9. Determine the lead truck

If the distance to cover for truck C is which is somewhere between the distances left for truck A and truck B, it will go places up in a platoon and be between truck A and truck B. It is assumed that every half hour, one or more trucks in a platoon will take a break. After each break, the distance remaining for all trucks is recalculated, as is the time required to arrive. Assume that if the left truck C wants to rejoin the platoon, it will match the platoon's speed and join them at the end. Then it will rearrange itself based on the distance left to cover. If truck C has a distance to travel of It will go places up in a platoon and be between truck A and truck B somewhere between the distance left for truck A and truck B. It is assumed that every half hour, one or more trucks in a platoon will take a break. After each break, the distance left for each truck and the time it will take to arrive at its destination are recalculated.



```
Duration remaining: 2.4000000000000004  Truck Order:  {'Truck C': 220, 'Truck A': 210, 'Truck D': 120, 'Truck B': 100, 'Truck E': 50}
Break No:  5
Truck_B leaves for break
Duration remaining: 2.4000000000000004  Truck Order:  {'Truck_C': 220, 'Truck_A': 210, 'Truck_D': 120, 'Truck_E': 50}


Break Over
Duration remaining: 1.9000000000000004  Truck Order:  {'Truck_C': 220, 'Truck_A': 210, 'Truck_D': 120, 'Truck_E': 50, 'Truck_B': 100}


Duration remaining: 1.9000000000000004  Truck Order:  {'Truck C': 220, 'Truck A': 210, 'Truck D': 120, 'Truck B': 100, 'Truck E': 50}
Break No:  6
Truck E leaves for break
Duration remaining: 1.9000000000000004  Truck Order:  {'Truck C': 220, 'Truck A': 210, 'Truck D': 120, 'Truck B': 100}


Break Over
Duration remaining: 1.4000000000000004  Truck Order:  {'Truck_C': 220, 'Truck_A': 210, 'Truck_D': 120, 'Truck_B': 100, 'Truck_E': 50}


Duration remaining: 0.40000000000000036  Truck Order:  {'Truck_C': 220, 'Truck_A': 210, 'Truck_D': 120, 'Truck_B': 100, 'Truck_E': 50}
Break No:  9
Truck E leaves for break
Duration remaining: 0.40000000000000036  Truck Order:  {'Truck C': 220, 'Truck A': 210, 'Truck D': 120, 'Truck B': 100}


Break Over
Duration remaining:  0.09999999999999964  Truck Order:  {'Truck_C': 220, 'Truck_A': 210, 'Truck_D': 120, 'Truck_B': 100, 'Truck_E': 50}
```

Fig. 10. Results of lead truck

The results are then displayed, along with the remaining distance and truck order until they all arrive at their destination. Every 0.5 hours, the order is updated. Furthermore, this solution can be improved further. We can add a new user input function where the user can provide details of the features to give out certain information and simulate the environment. The entire swarm system can be simulated, with the user providing feature inputs and output.

VII. Conclusion

Swarm driving is a relatively new field that has recently received a lot of attention in academia and industry. Swarms enable more stable, safer, and secure systems. with a lower use of resources and money. Indeed, swarm driving is being used in fields such as autonomous driving and smart farming, which have historically been expensive and high-performance demanding. However, in real-world scenarios, the complexity of these systems can reach very high levels, making their actual deployment and development impossible due to physical resource constraints.
In this paper, we examined the vast field of swarm robotics, describing the numerous advancements it can bring, the fields that would benefit the most, and the challenges that must still be overcome.In addition, we have been able to develop a simulation of a swarm driven technique called truck platooning and manage to build a flexible, scalable and robust system.

VIII. Declaration of Originality

I,Md Limon Apu, herewith declare that I have composed the present paper and work by myself and without use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form has not been submitted to any examination body and has not been published. This paper was not yet, even in part, used in another examination or as a course performance.

References

[1] H Anil, KS Nikhil, V Chaitra, and BS Guru Sharan. Revolutionizing farming using swarm robotics. In *2015 6th International Conference on Intelligent Systems, Modelling and Simulation*, pages 141–147. IEEE, 2015.
[2] Gerardo Beni. From swarm intelligence to swarm robotics. In *International Workshop on Swarm Robotics*, pages 1–9. Springer, 2004.
[3] Ahmad Reza Cheraghi, Sahdia Shahzad, and Kalman Graffi. Past, present, and future of swarm robotics. In *Proceedings of SAI Intelligent Systems Conference*, pages 190–233. Springer, 2022.
[4] Maanak Gupta, James Benson, Farhan Patwa, and Ravi Sandhu. Secure v2v and v2i communication in intelligent transportation using cloudlets. *IEEE Transactions on Services Computing*, 2020.
[5] John Harding, Gregory Powell, Rebecca Yoon, Joshua Fikentscher, Charlene Doyle, Dana Sade, Mike Lukuc, Jim Simons, Jing Wang, et al. Vehicle-to-vehicle communications: readiness of v2v technology for application. Technical report, United States. National Highway Traffic Safety Administration, 2014.
[6] GR Janssen, J Zwijnenberg, IJ Blankers, and JS de Kruijff. Truck platooning: Driving the future of transportation. 2015.
[7] Mrs Khairnar, D Vaishali, and Dr SN Pradhan. V2v communication survey wireless technology. *arXiv preprint arXiv:1403.3993*, 2014.
[8] Oliver Kramer. Scikit-learn. In *Machine learning for evolution strategies*, pages 45–53. Springer, 2016.

[9] Charles M Macal and Michael J North. Agent-based modeling and simulation. In *Proceedings of the 2009 winter simulation conference (WSC)*, pages 86–98. IEEE, 2009.

[10] Dongshu Wang, Dapei Tan, and Lei Liu. Particle swarm optimization algorithm: an overview. *Soft computing*, 22(2):387–408, 2018.