Dynamic Resource Allocation in Space-Air-Ground Integrated Networks Using Deep Reinforcement Learning

Project Report

Tewodros Abdishu Mamo

AALBORG UNIVERSITET

Department of Electronic Systems
A. C. Meyers Vænge 15, 2450 København

Copyright © Aalborg University 2015

Here you can write something about which tools and software you have used for typesetting the document, running simulations and creating figures. If you do not know what to write, either leave this page blank or have a look at the colophon in some of your books.

Electronics and IT
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Dynamic Resource Allocation in Space-Air-Ground Integrated Networks Using Deep Reinforcement Learning

Theme:
SAGIN, Resource optimization,DQN

Project Period:
Fall Semester 2024

Project Group:

Participant(s):

Tewodros Abdishu Mamo

Supervisor(s):
Reza Tadayoni
Yan Kyaw Tun

Copies: 1

Page Numbers: ??

Date of Completion:
June 20, 2025

Abstract:

As the number of users continues to grow in the telecommunication sector, the resources to serve the growing customers can be limited. One solution for this is to create a resource allocation mechanism that can effectively allocate the limited resources without interrupting the connectivity. However, handling all these users manually can not be effective and fast. The emergence of Reinforcement learning has revolutionized this allocation challenge. Applying Deep Reinforcement Learning in SAGIN (Space-Air-Ground Integrated Networks) network resource allocation enhances the management of the resource allocation in the dynamic environment. This project's scope is to apply resource allocation for SAGIN using Deep Q-Networks (DQN). The proposed model dynamically optimizes the resource allocation It uses SAGIN architecture for serving connectivity to an area where communication infrastructure is not available. The scenario is composed of two LEO satellites and three UAVs serving as base stations and the goal is to optimize the resources for the user on the ground. With this paper, it provides an algorithm that optimizes resource allocation. DQN SAGIN RESOURCE Allocation

The content of this report is freely available, but publication (with reference) may only be pursued due to agreement with the author.

Contents

Chapter 1

Introduction

The rapidly growing world population will tremendously increase the need for communication technology to accommodate this growth. This technology is evolving rapidly, and we are currently at a time when we can engage with the latest communication technology, 5G. However, 5G has limitations that could hinder future communication complexity, requiring broader coverage and higher speed rates than what 5G currently offers. Additionally, the latency of current 5G networks needs to be lower. Besides this, current communication technology makes some areas on our planet inaccessible. For this reason, future communication technology, namely 6G, is expected to address these limitations and is also aimed at being the new innovative technology that can elevate communication technology to a higher level. This groundbreaking technology has a transformative impact on the coverage area of communication. The 6G technology is already starting to be standardized by the responsible standardization bodies of communication technology, and the outcome of this wireless communication technology is expected by 2030 [8]. But no matter which future communication generation we use, the rapidly increasing connectivity of communication devices and busy networks cannot be adequately served solely by terrestrial network infrastructures. To address this growth in network traffic effectively, the best solution will be to implement an infrastructure that integrates space, aerial, and terrestrial communication technologies [66]. Scholars have agreed that integrating these three network architectures is the future of 6G. This will replace the current infrastructure, which is mainly based on terrestrial infrastructure.In addition, it will be suitable for future intelligent technologies such as smart cities and self-driving vehicles, which hold many devices to control traffic and the Internet of Things (IoT). To serve all these technologies, there is still a challenge with the resource allocation and user association in the Space-Air-Ground Integrated Network (SAGIN). Furthermore, user association and radio resources must be allocated efficiently and intelligently to optimize the throughput performance for overall network efficiency. Thus, the purpose of this thesis is proposing intelligent and innovative algorithm for the dynamic resource allocation in SAGIN while maximizing spectrum efficiency. [89]. This thesis consists of the works that has been done for the fulfillment of the master thesis in ICTE at the Aalborg University.

1.1 Motivation

It has been a high agenda to expand communication coverage worldwide as part of the United Nations 2030 Agenda, which aligns with the International Telecommunication Union to provide telecommunication services that include everyone regardless of their location. The plan aims to bring more people into digital societies [1]. To realize this plan, one of the mechanisms to be used is to emerge new communication technology generations such as 6G, and to combine different types of communication technologies to achieve the sustainable goal more easily.
One of the main resources that help to bringing this agenda in to reality is the combination of terrestrial, aerial, and space communication technologies. By integrating these three technologies, connectivity in rural areas, deserts, and other locations lacking communication facilities can be significantly improved. However, achieving this goal requires intensive research efforts to support this fantastic, ambitious 2030 agenda. Implementing these technologies will play a crucial role in enhancing the social and economic development of various parts of the societies, by creating a more connected and sustainable world. For this reason, doing research that focuses on this goal is important. SAGIN (Space-Air-Ground Integrated Network) is a promising alternative solution for this agenda. Since SAGIN has not yet been fully implemented, contributing research to help realize this connectivity is crucial. This thesis aims to address the problem definitions by studying SAGIN resource allocation and user association problems using reinforcement learning, which is one part of machine learning. People living in areas with limited communication access will benefit from this new technology integration when fully applied. Furthermore, this technology is not only used for rural areas; it can be applied to urban areas during emergencies such as fires or floods. Additionally, it can facilitate transport between cities and rural areas by serving communication connectivity. By doing this thesis, we can bring more parts of our society into the digital world. Thus, this research could serve as a motivation for further studies on this topic.

1.2 Problem definition

In the future, Sixth Generation (6G) wireless communication systems will facilitate higher throughput, lower latency, and seamless wireless connectivity in different types of environments, such as deserts and sparsely populated areas. Achieving global connectivity can be realized through the integration of satellite, air, and ground networks. Beyond this, the expectation for the goals of 6G wireless communication systems extends to the deep sea as well [18] [69].Even though space-air-ground integrated networks (SAGIN) have assumed it play crucial role in the next-generation 6G wireless communication systems, they still have limitations that need to be addressed. These include issues such as radio resource management, allocation ,user association, transmit power controls, and mobility management of aerial base stations. Consequently, this thesis will focus on dynamic radio resource management and user association in SAGIN to minimize energy consumption and maximize throughput while ensuring adherence to backhaul link constraints. Subsequently, this thesis will develop a fundamental algorithm based on a

well-known deep reinforcement learning algorithm (DRL) to address the energy minimization and throughput maximization problem. How can an algorithm be developed to allocate the above-mentioned resources efficiently? Addressing this question will be the main problem of this thesis. The problem can be further divided into smaller sub-questions, as mentioned below:

Research questions:

1. What is the current research on applying deep reinforcement learning algorithm (DRL) to SAGIN dynamic radio resource management and user association?
2. What kind of deep reinforcement learning algorithm (DRL) has to be applied to optimize the throughput of the SAGIN?
3. How to keep the backhaul link constraint while optimizing the throughput?
4. Is it possible to minimize energy consumption?

1.3 Limitations

The project can be influenced by various limitations of resources. For example, the limitation of time can affect the opportunity to research different types of deep reinforcement techniques. The complex nature of the project, since it involves new emerging technology, makes it important to research and test different types of deep learning techniques. The reason for doing this is to improve performance by comparing different techniques and to facilitate optimal resource allocation algorithms. However, due to time constraints, only one type of Deep reinforcement learning is proposed, deep Q-learning.

Because of the complex nature of communication optimization, the shortage of time to review and understand these concepts in a short period will be difficult, and this can affect the ability to grasp the very complex mathematical formulations in different previously researched literature. Understanding all these formulations within a short period can be challenging, and going in-depth through the existing research publications can be even more difficult. Most of the mathematical concepts that express the optimization theory require an enormous amount of time to fully grasp the core ideas. Therefore, this complexity will also be another challenge for the project.

The other challenge can be a lack of deep knowledge on how communication satellites work, such as in different weather conditions, their mobility, and the types of obstacles that can hinder communication with users on the ground. This kind of problem can be also a challenge in UAVs. The communication fields are evolving constantly. Therefore, a lack of knowledge of all these factors that can affect communication can influence the project.

Besides this, the evolving nature of these emerging technologies can introduce uncertainties regarding the methods and techniques chosen for this thesis. This is because both SAGIN and 6G are still in the early stages of development, which may present unforeseen challenges that

could impact the thesis. Despite these challenges, the thesis will explore deep reinforcement learning techniques, namely deep Q-learning, to solve the optimization problem in the SAGIN architecture and ensure good communication coverage for inaccessible areas.

1.4 History of machine Machine Learning

In 1946, the first electronic general-purpose computer was developed. While there are controversies on the part innovated by whom. It is a general fact that ENIAC is the pioneer of computing technology.ENIAC has the capability to solve high numerical problems at a high speed. The machine has to work with some experts. The machines were not independent like computers nowadays, and most of the operations needed manual help. The starting of this machine pushed the concept of innovating a computing machine that rendered logically the human thinking and learning capability. Alan Turing was one of the influential people in machine learning. Alan tries to implement the concept of learning and algorithm into his machine, Turing machine. Alan tried to address the question of "Can machines think?" with his Turing test experiment. Alan articulated his concept in 1950. The concept behind this test is based on the idea if machine communication reaches a level not determined by others from human communication, then we can say the machine is intelligent. For many machines, it was difficult to pass this test. Arthur Samuel built among the pioneer self-learning machines and it is considered as one of the basic foundations for the field of machine learning. Arthur Machine was a game player program for checkers that had a small memory but challenged many champions. To reduce the memory problem, he used Alpha-beta pruning. Using this algorithm, reduce the computation time and optimize the recursive algorithm for selecting the next step in the game. Arthur Samuel is also known for his introduction of the word machine learning. In 1957, Frank Rosenblatt created perceptron, which is considered to be the leading innovation in neural networks.The perceptron is a linear classifier learning algorithm that is used to categorize the input datasets. It is one of the most challenging algorithms for solving complex problems, but it is still good for solving simple problems. Perceptron has four arts, namely the input data, net sum, activation function, and the last are the weight and bias. Perceptron does adjust its parameters unless there are errors. In 1967 Marcello Pelillo invented the concept of "nearest neighbour rule." This invention is considered the Pillar foundation for recognizing patterns from the input data. NN does not train the machine rather it stores all the data in one and all works based on the stored data and the measurement distance between the input data. The algorithm starts to classify data based on a comparison with every data in the store. At the end of the 1970s and the start of the 1980s, there was a schism between artificial intelligence and machine learning. Machine learning starts focusing only on learning from data, while AI focuses on decision-making based on the implemented program in advance. In the 2000s, speech recognition and facial recognition became a reality [26].

Chapter 2

Methodology

This part focuses on the approaches that will be used to conduct the thesis from introduction to conclusion. Different methods and techniques that will be applied to make this work more manageable and productive will be discussed here. The approach will play a role in managing the project, and tools such as the Gantt chart will be used for scheduling the time to complete the different tasks of the thesis. By applying these approaches and tools, this part aims to provide a good roadmap for completing the thesis, ensuring that all aspects of the problem definition are analyzed properly and that the right solution is provided. More or less, this part of the project deals with proposing methods for managing the time needed to complete the entire thesis and the approach to executing the project. Both of these features are very important for finishing the work on time and incorporating all feedback and requirements so that they will be executed in the project. The approaches and tools that will be used in this thesis will serve as a means to track the progress of the project, to easily adjust the plan when there is new idea pops up or something that diverted the original plan, and establish a structured workflow throughout the project. By doing this, they can help ensure that the project is completed on time.

2.1 Project management

This thesis uses Agile methodology as the primary approach to manage the entire project. The iterative nature of agile makes it preferable for this project, as it allows for the incorporation of new ideas and ultimately enhances manageability [20]. Agile's flexibility helps to adapt to changed requirements while the project is being executed. This plays a big role in incorporating new ideas that are brainstormed from someone or due to new innovations in the principles or technologies used in this thesis. With this method, it enables us to continuously improve the structure and quality of the project.

In addition to this, Agile software development methodology has the capability of making the project more transparent to the stakeholders. This helps to make the work process of the project visible to all, and it will be more easy to update the stakeholders on the new changes that are adapted to the project. Another advantage of Agile is, its ability to detect errors

at early stages, which can ensure the delivery of a high-quality project. Additionally, it boosts communication among team members, including the supervisor and myself or other scholars [48]. Therefore, by choosing this methodology, it increases the quality of the project. The nature of Agile methodology involves working with a time-boxed iteration called a sprint. Tasks are taken from the backlog, a list of tasks, and done within the specified time to achieve the goal. Sprints provide a more organized and managed approach [60]. In this project, the high-level Gantt chart serves as a support to the sprints. The sprints will be written simply in personal notes. Therefore, no drawing or chart will be used to depict each sprint in the report. Tasks may be changed, removed, or added as necessary throughout the project. 2.1 depicts the agile steps

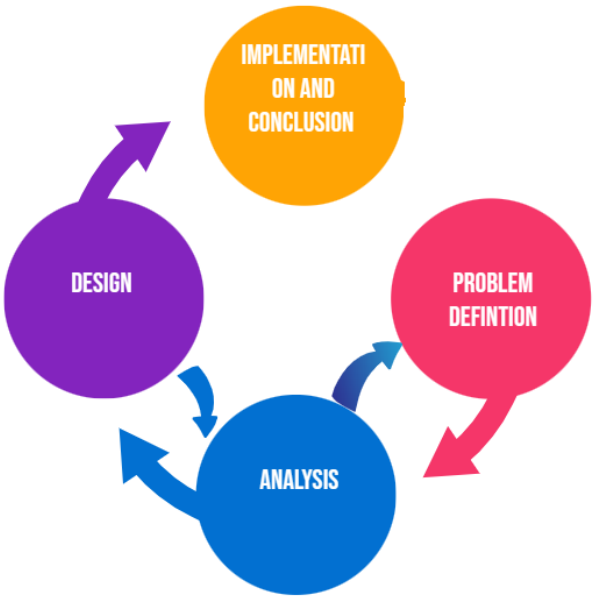


Figure 2.1: Agile steps of the project

that show the iteration among the problem definition, development process analysis, design, implementation, and conclusion, highlighting the structured approach taken throughout the project.

2.2 Gantt Chart of the report

To organize and plan the schedule for each task, I have used a Gantt chart. I applied this chart due to time constraints and to be more organized and ensuring the tasks are completed on schedule. If the tasks are well planned, a Gantt chart can be considered one of the crucial tools for managing projects [65]. Knowing the start and end times for each task is one of the functionalities provided by the Gantt chart. This helps me to complete tasks within the allocated time without conflicting with the thesis deadline. The Gantt chart is located in Appendix A and serves as a guideline for thesis time planning. However, due to the Agile methodology, there may

be deviations from the chart's schedule. This can occur due to new feedback or modifications to project.

Chapter 3

State of the art

In this chapter, we will introduce the fundamental concept of the Space-Air-Ground Integrated Network (SAGIN). This will include taking a closer look at the main components of SAGIN and the way it operates with the technology that supports its architecture for communication. In addition to this, it will cover machine learning techniques, specifically focusing on deep reinforcement learning (DRL), which will be used to optimize resource allocation. We will explore the nature of DRL and its importance in resource allocation, It will go through how it can improve the efficiency and effectiveness of SAGIN's communication systems. In addition, this section will give insights into the challenges and opportunities that arise when the DRL is applying to manage the SAGIN resource allocation and the user association with its resources. At the end of this section, related works that have been previously applied to SAGIN will also be explored, offering a comprehensive overview of the current state of research and advancements in this field.

3.1 SAGIN

The emerging wireless network technology generation, 6G, will incorporate the architecture of SAGIN as an important component. As it aims to increase the coverage area of the communication, it is critical to create a network infrastructure by integrating satellite, air, and ground [14].Collaborating with all three platforms' base stations,increases the network connectivity that was not reached in previous cellular wireless network generations. This will be especially crucial in rural areas or other areas that do not have sufficient telecommunications networks to serve the necessary access. By choosing different base stations according to their suitability to the service area, the coverage will be broad. The different parts of the SAGIN can function individually. For example, the satellite segment can provide service without relying on the airborne station, and vice versa. When necessary, all three segments can work together as a single network to meet user demand [14].

3.1.1 space

One of the main components of the SAGIN is the space, which is located at the top of the architecture. Unlike the ground-based stations, which are fixed to the ground, they serve as communication channels for different types of satellites that navigate in orbit. This can be a satellite that can work alone or a group of satellites that work together to achieve the same mission or coverage of an area.

Among the advantages of satellite communication systems are their wider coverage compared to terrestrial communication infrastructures, their ability to provide good mobility for accessing services from any location, their global reach which facilitates communication across national and continental boundaries, and their high data transmission capacity since it has wide frequency spectrum available [22]. All these advantages make satellite technology superior to traditional earth-based communication technologies. Therefore, in realizing the goals of the 2030 agenda, satellites play a key role by creating seamless connectivity.
The types of satellites in space can be divided into three, namely geostationary (GEO), low Earth orbit (LEO), or medium Earth orbit (MEO). The classification is based on how high the satellites are from our planet [22].

Geo

GEO satellites orbit at approximately 36,000 kilometers above the equator, since they rotate as equal to the Earth the position of this satellite remains fixed relative to something positioned on Earth. This nature allows them to give continuous coverage to wide but fixed regions of the earth. Due to this nature, they are preferred for applications such as television broadcasting, metrology, and other services that need fixed satellite service. Since GEO covers a wide area and is relatively fixed it has the advantage of eliminating handover processes that are very important in LEO and MEO. Therefore, users can experience low service interruption. But GEO has also its challenges due to the higher altitude position it can lead to higher latency and make it vulnerable to attenuation relative to the other types of satellites [22] [77].

Meo

MEO satellites rotate the earth at altitudes between 5000 and 12000 kilometers,this helps to get the balanced advantage between the advantages of LEO and GEO systems. They need fewer satellites than LEO to cover a wider area of the earth.Due to its lower altitude,it is expected to have lower latency than GEO.They are preferable for applications such as satellite-based navigation that help to locate anyone on earth,namely GPS.It is useful for communication also.Compared to LEO it experienced lower handover,this is due to one MEO covering more than another LEO.Therfore there will not be frequent handovers among the satellite to keep the connectivity.These characteristics help to reduce some problems that occur during handovers such as call drop or delay while changing to the other satellite [22] [77].

LEO

LEO satellites orbit at altitudes ranging from 500 to 900 kilometers. Due to their proximity to Earth, they facilitate communication with lower latency and reduced attenuation. These features make LEO satellites practical for applications that require low latency, such as real-time video and audio communications. Besides this, LEO satellites offer higher frequency reuse, which can highly improve communication capacity, especially for mobile communications. LEO satellites also have their own drawbacks. Since they cover a smaller area compared to higher orbit satellites, they need frequently to hand over between satellites. The high speed of LEO satellites' orbits also leads to creating high Doppler effects, that lead to changes of frequency every time. To adjust this, it requires advanced techniques for the adjustment of the frequency [22] [77].

Table 3.1: Altitude range of different types of satellites [22]

Name of satellite	Altitude range
GEO	36000km
MEO	5000-12000km
LEO	500-900km

Satellite Handover

Handover is the process of transitioning user equipment from one satellite to another [25]. Since satellites keep rotating, they will not stay fixed in one place except the GEO. As the satellite moves out of reach from the coverage area to the user, another satellite within the user's vicinity takes over to provide service. This process is very important to keep continuous connectivity for the end. When there are many LEO satellites, managing the handovers between them can be challenging and complex. This is due to the high speed of the LEO satellites and the relatively compared to other types of satellites they cover small areas. This problem can affect the overall performance of the communication. Some of the good solutions to confront these problems are to incorporate MIMO technology and the Kuhn-Munkres(KM) algorithm [25]. Since Mimo technology enables the satellite to send and receive multiple data at the same time [62] and the KM gives weight to each node, which weighs the quality of the communication, both play a role in maintaining good performance [25]. By applying the MIMO the whole system can enhance the capacity of the satellite network which helps to enable high throughput and better data traffic handling even during the interference of both noise and interference. This will help to ensure multiple users can get good service simultaneously. In another way the KM algorithm's ability to help the satellite can decide which communication channel should be given priority. Therefore the integration of km and MIMO technology generally high impact on enhancing the overall communication network performance.

3.1.2 Air

The second segment of the SAGIN architecture is the air network. As shown in Fig 3.1, the location of the air component is between the space and the ground network. This part consists of aerial vehicles such as balloons, Unmanned Aerial Vehicles (UAVs), and planes that can transmit signals and process to provide access for communications. This portion of SAGIN has greater mobility compared to the ground, which is fixed. Therefore, it can cover areas that are not accessible by terrestrial stations. In addition to this,it can cover a wide network range [14].

UAV

UAVs can be applied in communication technology by serving as flying base stations to facilitate connectivity coverage and enhance communication capabilities. Their mobility allows them to adaptively position themselves for optimal service in different areas. This feature enables them to play a significant role in emerging Next Generation Wireless [15]. As flying base stations, UAVs have several features that improve communication capabilities. Firstly, their mobility allows them to dynamically position themselves closer to users, improving channel reliability and increasing data communication rates. However, this mobility also has its drawbacks, as it can lead to security vulnerabilities. Therefore, when UAVs are applied as base stations, it is important to consider their related security challenges [11]. UAV base stations can help ensure better Quality of Service in communication. Although most terrestrial networks are designed based on long-term traffic patterns, there are times when events are crowded with a large number of people, and the underutilized network cannot properly serve the users. In such kind of situation, UAVs help in achieving good coverage and capacity by working side to side with the terrestrial network. This feature helps to facilitate big events such as the world cup, big music festivals, or elections smoothly [15].

Balloon base station

The other option for base stations in the air is Balloon-based base stations, which can have good potential for flexible and mobile communication services in areas where terrestrial communication infrastructure is unavailable or interrupted. In areas where geographical and logistical barriers balloon with base stations can reach and serve seamless connectivity for users.
The balloon-based system is easy to deploy. After the balloon is mounted with a telecommunication base station it can be connected directly to mobile base stations, from mobile wireless vehicles, it can also be easily using the satellite communication channels. According to the application we need the balloon can serve to any operational needs. The balloon was used for simple communication tasks such as text and audio communication, but now it can be used for high data rates for addressing different communication scenarios. This feature makes Balloon one of the potential air communication infrastructures that can enable the integration of SAGIN [63]. Unlike the other air vehicles High-altitude balloon flights cost less money and we can get them easily [47].

Aircraft

The third option for air borne base station is aircraft. Aircraft can serve as base station as UAV and Balloon by offering connectivity for various purposes.

One good example of applying aircraft as a communication base station is, a recent research made by Japanese researchers who use a small Cessna plane to serve internet access for people on the ground. The aircraft was outfitted with a base station that operates with 38 GHz frequency 5G technology. This helped to communicate with ground stations successfully and was a significant achievement in providing base station services from an altitude of 4 km. Since the aircraft is powered by solar energy, it can stay in the air for days without landing. This research shows how using aircraft as a base station is emerging as a potential communication hub [34].
The successful result of the research demonstrates the aircraft's potential role in creating seamless connectivity. Even if the research was conducted with 5G, it lays the groundwork for a crucial role in 6G networks. This will help to realize SAGIN integration networks with many alternatives and extended flight durations, by mounting the aircraft with solar or any other renewable energy.

3.1.3 Ground

The ground segment represents the terrestrial infrastructure capable of communicating on the ground with all components of the communications system. The ground communication infrastructure can be composed of three networks cellular networks, mobile ad hoc networks, and wireless local area networks [13]. This helps in facilitating the space and aerial network of the SAGIN and collaborates to form the complete architecture. Inside the networks, there are core components such as user equipment, base stations, and other supporting equipment for ground connectivity [14].
In general, the farther a segment is from Earth, the higher the latency compared to terrestrial segments. Therefore, satellites have a higher transmission delay than ground-based base stations. However, the ground infrastructure also has its disadvantages. For instance, it can be more easily affected by natural hazards than the air and space components. Satellites can cover larger areas and a broader range than stations in the ground. As mentioned above, all components have their strengths and weaknesses. By complementing each other, they form the structure of SAGIN. This combination helps us achieve seamless connectivity. According to the user's location, SAGIN uses the best combination to provide optimal service.

User Equipment

User Equipment (UE) is the device that helps us to interact with a variety of end-user devices that interact with communication networks. These include smartphones, tablets, IoT devices, and others [2]. UE devices are not only used to interact with communication infrastructures, but they can also provide various services through different offline applications that can help

solve problems for the users. When we come to the context of communication, they are key devices for displaying the results in a convenient way for the user.
The current user equipment (UE) technology is becoming more advanced and complex. It integrates many advanced technologies that are able to enhance the overall performance of communication and data transmission. It combines different components to able to process the advanced multimedia data, these components are antenna, sensor, and various software. Even if the current user equipment is advanced there are still constraints that challenge the communication process these are battery efficiency, circuit, and system optimization. This is due to the performance of the communication depends on the advancement of addressing these constraints. For example in circuit design, more compact chips can hold many digital electronic components that can increase the overall quality [36]. Therefore, the design of the User equipment is important to overcome problems like human blockage and high path loss. This can be optimized by strategic antenna positioning after the architecture of the mobile handset architecture is changed to distributed phased array MIMO (DPA-MIMO) architecture. This architecture helps to boost both the efficiency and adaptability of new technology that will be integrated into the network in the future. The DPA architecture is also the best solution for communication between user equipment (UE) on the ground and unmanned aerial vehicles (UAVs) serving as a base station in the air. The architecture supports both single-user and multi-user from the ground to connect with the UAV base station simultaneously. This design enhances the effectiveness of communication in 5G and beyond networks [36] [37]. The DPA technology enhances throughput in the communication between arial base stations such as UAV and user equipment, this indicates that it will have an enormous effect on the SAGIN architecture in general.

Base Station

Base stations are used to relay the information coming from different types of communication infrastructure. By using base stations the multimedia data is capable of reaching the users [76]. Each base station has its own communication coverage capability this feature can limit the communication connectivity if it is not used in the proper base station. Therefore, the performance of the base stations affects the area covered by the communication service. Due to this, multiple base stations can be implemented to transmit data effectively [51]. Increasing the base stations means we are reducing the cell size in communication networks. This offers different types of advantages. It grows the bandwidth per user, since there will be fewer users per station it helps to allocate greater bandwidth for each user. This helps to have lower transmit power since the users are near the station [70].

3.2 Throughput

The term throughput refers to the data rate received and transmitted from one device or station to another within the communication system. This can be quantified by measuring the data

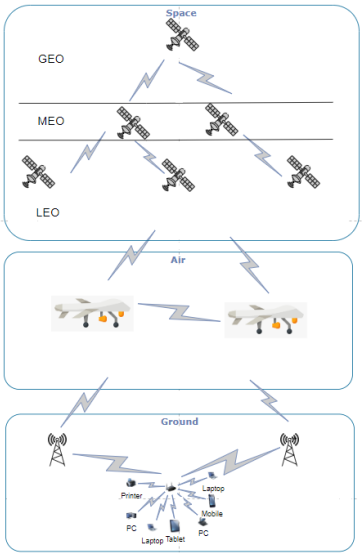


Figure 3.1: Architecture of space-air-ground integrated network [14]

transferred within some time. The data can serve as a measurement of the data transmission performance. The quality of the data transmission can be interfered with with unintended signals such as noise, therefore knowing the signal-to-noise ratio (SNR) can help to quantify the influence of the unintended signal on the throughput [86]. There is a significant relationship among radio resource allocation methods and the throughput result in communication networks. It is important to implement effective resource management strategies that consider the different demand types. This helps to use resources efficiently [41]. In the different multi-service environments it is important to integrate different types of resources, which need addressing the resource allocation for enhancing the throughputs or other types of efficiency metrics that show the resource is utilized effectively. Each service in the communication network needs resources to accomplish its task, these resources can be communication links, data repositories, or other hardware that connects and manages the system's connection. Understanding these interactions between the services and the shared resources is very important since it helps to implement strategies and control mechanisms that optimize the response to multiple service requests [41]. A good understanding of the available resources and the requests from different services such as user equipment are directly related to the throughput optimization in the network. When the demanded resources are well mapped based on the demand size then it reduces the delay time also to reach the destination. Throughput is affected not only by the strategies to allocate resources but also by bandwidth, which is a big factor. The nature of the channel bandwidth is one of the key factors that influence the overall data transmission of the network. Therefore, it is important to study the relationship between bandwidth and channel capacity (throughput), since they are fundamental to the optimization of the data rate.

$$C = B \log_2(1 + \text{SNR})$$

Where:

- C is the **channel capacity** (measured with bps)
- B is the **bandwidth** of the channel in Hertz (Hz)
- SNR is the **Signal-to-Noise Ratio**(The ratio of desired signal to the noise)

SNR unit here is in decibels (dB)

$$\text{SNR (dB)} = 10 \cdot \log_{10} \left(\frac{\text{Desired signal}}{\text{noise}} \right)$$

SNR quantifies the quality of data transmission by comparing the strength of the desired signal to the noise. While Signal-to-Interference-plus-Noise Ratio (SINR includes interference in addition to the noise. In wireless networks when the measurement of the SNR is high, it means the data has less noise which is a sign of better data quality. The ratio expresses the difference between the desired data and the unnecessary noise. The recommended values of SNR lead to better data quality and fewer retransmissions. The recommended SNR for WLANs has to be at least 20 dB, 25 dB for voice applications, and 30 dB when it is needed higher data quality [35].

As we can see from the formula the capacity of the network is influenced by the channel width of the link where the data is transmitted. This shows when the bandwidth is increased it can serve many users at the same time. This enhances the overall throughput in the network integration. Due to some factors, these improvements in the throughput are not strictly proportional to the bandwidth increase. Some of these factors are the number of physical resource blocks that are allocated resource either time or frequency can be decreased after a certain amount and the other factor is the modulation scheme of the data. Since the higher order modulation is finite, whether we increase bandwidth the modulation scheme has limitations at some point. Therefore, these two concepts contribute to not utilizing the available bandwidth resource to increase linearly the data capacity rate [87]. All communication network designs tend to have high throughput and low latency. One good example of a network with high throughput is SAGIN. This is due to its integration of different powerful communication layers, such as UAVs and satellites. SAGIN is more favorable in terms of the quality of data it transmits compared to other traditional networks [55].

Even though SAGIN has relatively high throughput, getting fixed latency is not always possible. Many factors can cause the throughput to become unstable. One technique used to address such kinds of problems is network coding (NC) [55]. It is a new approach where different data packets will combine to transmit from the transmission source to the destination. One widely used network coding is randomized network coding [79]. However, this technique is also not free of drawbacks. It has vulnerabilities that can be exploited. Using the statistical signal

processing method called Blind Source Separation (BSS), can easily attack the vulnerability. These attacks can affect the effectiveness of the throughput needed for our task.

Therefore, robust security measures must be incorporated to safeguard packets and reduce breaches or theft attempts when trying to enhance throughput using this method.

3.3 User Association

In integrated networks such as SAGIN, it needs to manage the connection between the user equipment and the limited resources of the network in a way that can not disturb the connectivity. Sometimes the shortage of network resources cannot serve all the demands that come from the users. Therefore, it needs to have some sort of mechanism that arranges the connection of the request coming from the user to its corresponding network resource. This is due to modern networks being more complex and distribution of the resources has to have some management system to serve the service optimally. This fundamental process that maps a user to the base station is called user association [71] [56]. This important concept of communication has a key role in resource allocation. Traditionally, the connection between the base station and the UE is based on the strength of the signal. This method is more suitable for homogeneous networks, where the different network components have similar capacities and characteristics. But when the communication networks are becoming heterogeneous networks, the criteria for the connection start to consider many factors. Formerly mobile devices would connect to the cell with the strongest signal. However, this approach has changed as networks have become more sophisticated and integrated different types of networks to serve users. While connection based on signal strength is simple, additional factors must be considered such as the capacity of different base stations, the cost it costs when the UE connects to various stations, and the load created by these connections [46].

It is important to apply effective user association techniques in a network where a satellite is integrated into connectivity for larger coverage areas. Applying good user association techniques has a significant impact on improving handover times [85]. Even if Low Earth Orbit (LEO) satellites have lower latency and higher throughput when compared to other types of satellites since the handovers are frequent between the satellite and the user it leads to lower Quality of Service (QoS). This challenge can be reduced if it is managed with good user association techniques [85] [52] .In complex networks that include satellites like SAGIN, which adds another communication layer in the air such as ballon and UAVS, the influence of the handovers will make it more complex. Therefore the user association mechanism in the satellite will play a critical role in maintaining the optimal QOS in the whole SAGIN.

3.4 Latency

Latency is the time it takes for data transmission from one point of a communication network to another. Even though the thumb rule of communication assumes that data transmission should

occur at the speed of light, this idea is generally more theoretical and far from reality. The reason for this is that many factors can hinder the data rate, such as the distance between two communication points, the type of infrastructure used, network congestion, etc. The time it takes for data to reach the endpoint, considering the mentioned factors, is what we call latency. When we say higher latency, we mean that the data takes more time to reach its destination. This is not desirable in communication [32]. Each communication network aims to have lower latency as much as possible. With growing services that demand lower latency technologies, such as telemedicine, the lower latency is very critical for doing many tasks. Now day Low latency is becoming achievable because of advanced data center technology, next-generation Ethernet switching chips, and faster network interface controllers (NICs).This technology can lead us to the assumption that data transmission has to be transmitted through light speed.From time to time the time it takes to traverse data through the network is reduced. The emerged technologies indicate that the round-trip time for the data transfer will take less than 1µs [68]The traditional satellite is known for simply transmitting data to the destination point of communication. This means the data remains unchanged from the status it had at the sender station. This will result in lower latency. But in SAGIN the transmission will not be as straightforward as in the traditional systems. This is because in the future the satellites integrated into SAGIN will be equipped with various types of computing powers that can perform different types of tasks on the data content [19]. The delay caused by this additional computing capability requires the implementation of a mechanism to reduce the transmission time. This time has to satisfy the demands of future communication generations, such as 6G. One of the mechanisms that can enhance the latency is by using the optimal positions of the relay and the receiver UAVs. This optimization can be done by a convex optimization mathematical model. To achieve this optimization it is necessary to study the previous maneuvering history of the UAVs and analyze the corresponding latency result for each positioning. The optimization has to consider different critical such as the interference level, the user equipment positioning, and other communication metrics that can affect the overall performance of the SAGIN. This best positioning result from the optimization will contribute to getting the lower latency. One of the mechanisms that can enhance the latency is by using the optimal positions of the relay and the receiver UAVs. This optimization can be done by a convex optimization mathematical model. To achieve this optimization it is necessary to study the previous maneuvering history of the UAVs and analyze the corresponding latency result for each positioning. The optimization has to consider different critical factors such as the interference level, the user equipment positioning, and other communication metrics that can affect the overall performance of the SAGIN. This best positioning result from the optimization will contribute to getting the lower latency [7]. This is just one method that helps to improve the latency in the SAGIN communication. However, this research is mainly focusing the positioning of the UAVs therefore there is other research also that can explore different techniques and models that contribute to reducing the time of the data delivery. .

3.5 Reinforcement Learning (RL)

Reinforcement learning is a part of machine learning that uses an agent to make decisions by learning through trial and error within a dynamic environment that consists of all the components of the states of the system. This method allows the agent to learn and update its decision-making capabilities iteratively. This helps to achieve more effective decision-making in different cases. The process is based on the feedback that the agent receives from the environment to improve itself and send back a reward for suitable states and a penalty for bad ones. By doing this, it creates the best optimal solution [42] [81].
Reinforcement learning as a system can be expressed using the mathematical framework called the Markov decision process. This framework can formalize the whole process of the reinforcement learning task. By doing this, all the components and characteristics of the process can be depicted mathematically [81].

This simplifies the quantification and expression of the process. This mathematical formulation can have a more systematic approach to execute, understand, and optimize the learning algorithms. Depending on what type of policy we choose this provides a clear structure for assessing the policies and strategies in the entire system. The essential components of RL are:

- **Agent:** The main goal of reinforcement learning is to create an automated system that decides the optimal decision. This is done by learning one of its components which is called agent[3]. The agent has interaction with the other components of the RL, the environment, to learn to make a list of decisions. The state of the environment has to be transferred to the agent so that the agent can explore the status of the Environment. Learning from this the agent starts to act[3]. Besides the state, the agent gets a reward from the environment for the response of the action it takes on the state in the environment. This can be either positive or negative based on the agent's action it is taken. This process continues until the agent learns to make decisions optimally. The end goal of reinforcement learning is to train the agent to find the optimal solution. After many trials and errors, the agent develops the ability to make optimal decisions[82].
  The agent itself has two main components that help it to take the right action. These are the policies and the algorithms used to learn it[3]. The policy dictates the characteristics of the agent, like how it should behave toward the next action. Therefore, it plays the main role in deciding the actions to be taken about the states that are observed or sent to the agent. To incorporate the policy into the agent, a function approximator with adjustable parameters can be used so that the learning process can improve the agent's decision-making ability. The algorithm will improve the policy itself over time. The main goal of the algorithm is to develop a policy that achieves the highest cumulative reward from the environment based on the actions taken by the agent[3]. This helps the agent reach the best version of itself to make optimal decisions.

- **Environment:** The environment is also a main component of reinforcement learning. It is the external system with which the agent interacts to obtain states, based on which it

decides the actions to take by the agent[67]. The interaction loop between the environment and the agent improves the agent's efficiency. The environment is where the hardware components are defined. Each state of this communication infrastructure must be declared. Therefore, the state has to be forwarded, and in response, it must accept the action to modify the state. This interaction continues in a loop until the agent finds the optimal solution[3][67].

- **State :**The state comprises summarized data of the events that have occurred in the environment. This information is sufficient to describe the nature of the resources outside of the agent and serves as input for the agent. The data it contains can range from variables to various characteristics of the components in the environment. This data is crucial for the system to act optimally.[81]. This enables the agent to have all the necessary data for decision-making.

- **Action (a):** An action is the decision taken by the agent to interact with the environment. It is a finite set that is kept within the agent. The agent's main goal is to learn which actions should be taken for different scenarios. This is used to optimize the total reward over time. The action that has more reward, the more chance it has to be chosen for interaction. Not all actions apply to every state. Every action changes the state from one state to the next.[81].

- **Reward (r):** The reward is a signal generated by the environment as feedback for the action it is taken. The agent decided to take. The reward can be either a punishment or a positive reinforcement. Initially, the agent does not have a clue which action from the list it should take. Therefore, the concept of reward helps us train the agent by giving positive reinforcement for good decisions and punishment for bad actions. The agent then learns to favor actions that result in more rewards[81].

- **Policy (π):** A policy is just a function that dictates which action has to be taken for each state that comes out from the environment. A deterministic policy directly maps the states coming from the environment to the appropriate actions. On the other hand, in a stochastic policy rather than deciding directly, it assigns probability numbers to each action. The number is assigned to each state that emerges from the environment. The total sum of the probabilities can reach up to 1. This helps ensure the actions that must be taken are decided by comparing these numbers. The policy is implemented using a mathematical framework for incorporating the policy into the agent[67][81].

This will be formulated in how it looks to represent the enforcement learning with the Markov Decision Process (MDP) framework. The variables it uses to encapsulate for working its decision-making on the action it has to be taken in the environment. An MDP is normally defined by is defined by a five-tuple (S, A, P, R, γ): The meaning of each variable and its expression is listed below.

- **Set of states** S:The collection of the available states in the environment
- **Set of actions** A: $A(s)$ this represents all the potential action that can be taken in the state s.
- **Transition probabilities** $T(s,a,s')$: This dictate the transition of one state s to the next state s' based on the distribution probablity when the action a is excuted. The transition of the states can be expressed with the function as this $T : S \times A \times S \rightarrow [0,1]$.
- **Reward function** $R(s,a,s')$: Then the transition of state initiates the reward function for its corresponding action. This will be formulated with this expression, $R : S \times A \times S \rightarrow \mathbb{R}$.
- **Discount factor** γ: This value is in the boundary of $\gamma \in [0,1]$ and it is used as a balance of the current rewards with the future rewards.

In Markov's assumption, the future state can depend only on the current action it is taking, rather than the events that happened in the past. This assumption can be expressed in the formulation below. This concept serves as the basis for reinforcement learning. Based on this, the problem we want to execute can be transformed into a model.

$$P(s_{t+1}|s_t,a_t,s_{t-1},a_{t-1},\ldots) = P(s_{t+1}|s_t,a_t) = T(s_t,a_t,s_{t+1})$$

- s_{t+1}- future state at time $t+1$.
- s_t- current state at time t.
- a_t-action taken at time t.
- $s_{t-1}, a_{t-1}, \ldots$- past states and actions at a time $t-1$.
- $T(s_t,a_t,s_{t+1})$- transition probability function.

The main object of the Markov decision process(MDP) is to get a policy π that can give as a maximum reward after the action in the environment. This can be expressed with the following formula.

$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

- $\pi^*(s)$: The optimal policy at state s, which is the action that should be taken to maximize future rewards.
- $\arg\max_a$: The argument of the maximum, which returns the value of a that maximizes the expression $Q^*(s,a)$.
- $Q^*(s,a)$: The optimal action-value function, representing the maximum expected return (sum of rewards) obtainable by taking action a in state s and thereafter following the optimal policy.

where

$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$

$$V^*(s) = \max_a Q^*(s,a)$$

Here, $Q^*(s,a)$ is the optimal action-value function, representing the expected return of taking action a in state s and following the optimal policy thereafter.

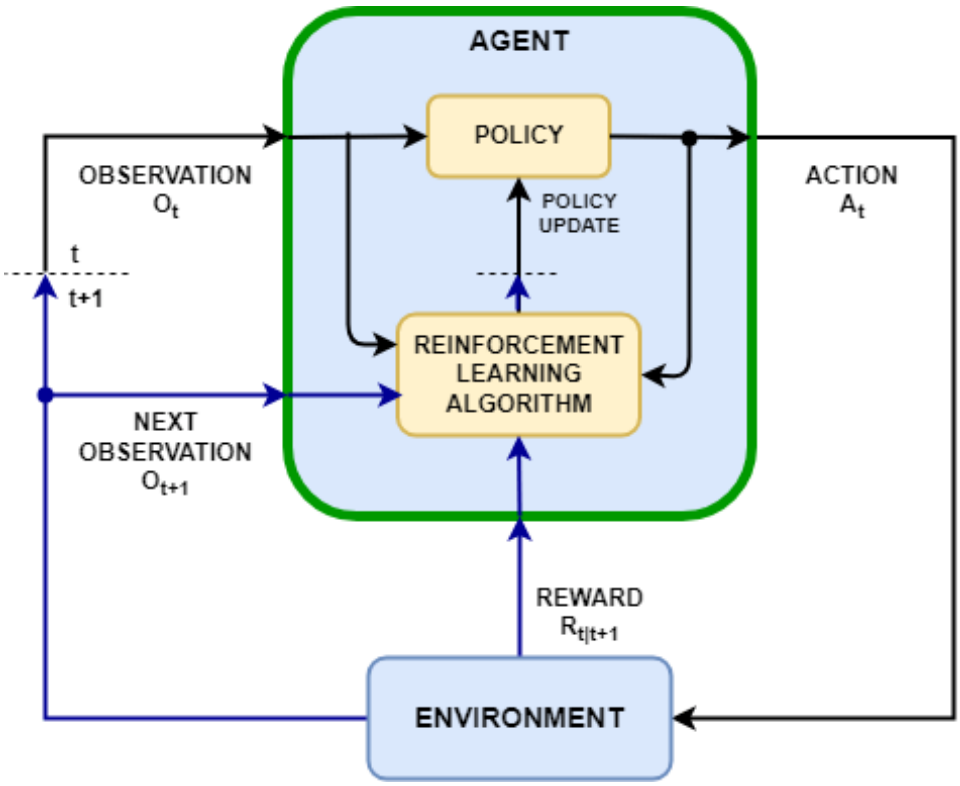


Figure 3.2: Reinforcement Learning Model [67]

3.5.1 Deep reinforcement learning (DRL)

Traditional machine learning methods were used to solve simple problems with large amounts of data and played a significant role in their time. For example, linear regression can easily predict the future based on the historical data. This is done by creating a straight line based on the interaction between the input and output historical data. By making this straight line, the model can predict future output just by inserting the new

input. This kind of machine learning method is most effective when the data pattern is linear or approximate to linear. The same is true for classification using Support Vector Machines (SVM). These traditional methods are structured with simple input and output layers. Therefore, using these two methods cannot learn from all the data, which means the data that is not aligned with the street line will be discarded and the model will not learn from it. This model will not effectively work with the real model scenario. Therefore, to handle such kind f complex data it is added a hidden layer in the middle. This layer adds a feature for the network the capability to incorporate diverse data for learning. This kind of learning is called deep learning[53] [27] [6]

Deep Reinforcement Learning (DRL) is a powerful artificial intelligence method that incorporates the concept of reinforcement learning principles with the architecture of deep neural networks. The neural network is composed of both input, neurons, and output that are able to create a model capable of solving complex tasks. Reinforcement learning is a broader concept and DRL can be considered as one of its branches that applies reinforcement learning with the core idea of deep learning. The core idea behind deep reinforcement learning is, that it does not rely on previously recorded data to train a model for solving a problem. The agent's interaction with the environment enables DRL to learn independently and define an optimized model for solving the problem. This distinguishes DRL from unsupervised learning, where learning is based on some sort of statistical data or other types of previously recorded data. Its ability to learn independently without needing input data makes it a powerful tool for training artificial intelligence models. This method is especially very good in scenarios where no previously recorded data exists[10].

The result of merging the traditional reinforcement with the powerful concept of the deep neural network has created an advanced mechanism with integrates different foundational concepts such Markov Property, the Markov Decision Process, and the Bellman Equations.

[58] These three key concepts are important when somebody needs to design DRL algorithms that solve problems in dynamic and high-dimensional environments.[21]. One of the key pillar ideas that is a foundational base for the DRL is the Markov property that expresses the characteristics of DRL the future state of the environment only depends on the current state of the model rather than external data that are recorded from previous events. This property able DRL to not have memory for recording the bulk data that have been recorded for many years, as is seen in other types of machine learning methods. This reduces the amount of data stored for solving the problem. This simplifies both the data handling and computing which means it also for producing an efficient model.

The implementation of the Markov property into reinforcement learning leads to the concept called the Markov Decision Process is a more detailed model that integrates the Markov property to handle the decision-making process in DRL. This includes the whole process of each state that is waiting for a decision and all the interactions between the environment and the states. The decision-making is accomplished by the agent. Each action accomplished transfers the current state into a future state. Each action of the

interaction gets a reward that motivates or discourages the model based on how the result is aligned to be achieved. This iteratively rewarding process helps to guide the agent to the goal. The whole process of this interaction is described with the broad concept of MDP[5]. The other main component of the DRL is the Bellman equation. It supports the MDP by calculating the values associated with each state. It also provides a framework that helps in the process of rewarding which actions have to be awarded a positive reward or negative consequence. Therefore, it assists to get the optimal solution for the problem. This is done by relating the current state values with the expected future reward using the partial differential equations. This helps to take the action that can give the highest reward. In the end, it creates a model for making informed decisions. By doing this it avoids uncertainty, and the result will be more predictable.[45].

The expansion of the Internet of Things (IoT), where every electronic and mechanical device is connected to the Internet creates a very complex and dynamic environment. To manage billions of devices connected where it needs real-time decision making it demands to apply DRL which can optimize resource allocation and interaction management can be divided into Deep Q-Network (DQN), Double Deep Q-Learning (DDQL), and Deep Deterministic Policy Gradient (DDPG). the network of all of them use neural networks but the approach how they solve the problem is different[64].

Artificial Neural Networks

Artificial Neural Networks (ANNs) is a powerful machine learning computational algorithm that is inspired by the biological neural system of our body. Our body is composed of a nervous system that serves as a path for the message transferred from one part of the body to another. This foundational element of this complex nerve system is called a neuron. The nerve system of our body is believed to have around 10^{11}. The biological neurons accept electronic signals both from the internal and external environment and process the signal. The structure of the natural neuron can be classified into three according to the way how it is organized namely nuclei, ganglia, and Lamin. In the first structure, the neurons are collected together as a group while in the last one, it has a layout as a sheet. The artificial one is a network connected with a neuron that has input and output. Each data has to pass through the input layer and based on the result it gives to the output it has to be adjusted the neuron with the right weight that can reduce the error compared to the objective goal.

ANNs have revolutionized the optimization techniques of many complex problems. It can solve problems easily and with good speed. But it is not completely without drawbacks of the biggest challenge in applying ANNs to get the optimal weight for the neuron while without causing overfitting. Getting overfitting can lead to resistance to learning when a new pattern is coming that is completely different from the data used to train the model. It is taking measures to avoid this kind of overfitting for example by using regularization

[78] [44] [23].Regularization is a technique that helps machine learning increase the performance of the model toward data that is not used in the model training. This helps to increase the generalizability of the model. While it is important to apply regularization for avoiding overfitting it is important not to use too much regularization to avoid underfitting. Different types of regularization can be applied to make the model work with any unfamiliar data. One of the methods to regularize is data augmentation which is applied by deforming the existing data nature to create more unseen format data. The second one is early stopping before it reaches the overfitting.The third one is just dropping out different neurons during training which ale to adjust its weights without taking into consideration the dropped neuron and the last one is by reducing the weight.These four methods will be a constraint to the model so that it can not learn the data perfectly[61]. These constraints able the model to retain its ability to perform well on new input data. That means the model can work also better with the data that is not similar with the used in the training. By doing this, it is able to use in the real-world scenarios where the nature of the input data is unpredictable. Neural networks are becoming the focal point of different type of technological research. Their applications where to apply are increasingly diversified.One of the main sectors where neural networks are applied include image recognition, speech identification, and pattern recognition. The size of the neural network and the number of neurons in the network depend on the complexity of the problem. The hidden layer of the network determines to what extent the model has to learn from the training data. For this reason, using too few hidden layers has its own impact. This will lead the algorithm into difficulty to learn well from data. If too many hidden layers are used also it has its own challenge that leads to fit the model to fit with the training data and it will be difficult to work properly when there is little change in the data[75]. The process of learning the model is by forwarding data samples iteratively. The size of this data is very important for the overall performance of the model. This amount of data that is used to pass through the network is normally measured by Batch size. This size can be higher or smaller. It is based on the nature of the problem. Adjusting this hyperparameter based on the errors it generates is very important to increase generalization. By tuning the right learning, the batch size serves to have a controlled model. The learning rate and the batch size are interdependent. For better performance, it is important to increase or decrease them proportionally. The higher batch size gives good results with the highest learning rate and vice versa[43].

Q-Learning

Q-learning is a model-free reinforcement learning algorithm. The agent of Q-learning learns through each interaction with the environment and stores the Q-value of the interaction in its table.Each time an action is taken in a state, the Q-value from the table is evaluated to choose the action that has highest q-value.The action taken in the state leads to either positive or negative rewards. Unlike DQN, Q-learning does not use a neural network,

which limits its scalability[88].

The Q-value represent the reward given to the action applied a_t on the current state s_t [88].

After the action is applied the new Q-value is updated with the Bellman equation, which helps to formalize the Markov property[88].

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

where:

- $Q(s_t, a_t)$: Q-Function .
- α: Learning rate .
- r_t: Reward either positive or negative.
- γ: Discount rate ($0 < \gamma \leq 1$).
- $\max_a Q(s_{t+1}, a)$: The maximum future reward s_{t+1}.

[88].

The learning rate is important parameter in reinforcement learning that helps to define the rate to which percent should update the new information one the previous.The rate is defined from 0 to 1.when the whole information is updated on the new it means the learning point is one and when it is zero the system is not learning[31].

The discount rate is an important parameter in reinforcement learning that helps to balance between the future and current reward. This saves the model from choosing repeatedly one action and helps to explore different strategies by exploring more. The lower discount is focusing more on the current reward, while the higher is focusing on the long term. Therefore, if there is no balance between this the system will focus either only on the immediate reward or the long term. Lowering the discount does not mean completely bad. Some situations need a lower discount rate, such as tasks that need immediate real-time control. [4].To improve performance in Q-learning, it is important to adjust both the learning rate and the discount rate by controlling the results. Therefore, selecting the appropriate values is crucial. As a rule, applying higher learning and discount rates leads to faster convergence. Starting with higher values and gradually reducing them to the optimal rate helps the model to solve the problem optimally. this not the definition[93]. The optimal policy of Q-learning is represented with the Bellman equation, does not mean the Q learning strictly follow this optimal result to learn itself-learning is off-policy method where sometimes it explores the environment to learn itself, even if there is optimal solution from the table. This helps to increase the model to learn more about the environment. The off-policy can be implemented in practice using the e-greedy policy. [39]

Deep Q-Network (DQN)

Is based on the concept of the Q-learning but with the neural network replacing the table.It was innovated by Deep Mind in 2015[74].DQN emerged because Q-learning uses a table for each Q-value result coming from the iteration.This makes it impractical in many real-world scenarios. First and foremost, the table becomes larger and larger for bigger scenarios. This can also affect speed. Therefore,DQN was introduced. It only needs to iteratively adjust to obtain the optimal weights for the neural network[74].

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')\sim D}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i)\right)^2\right]$$

Target Q-value: $r + \gamma \max_{a'} Q(s', a'; \theta_i^-)$

Predicted Q-value: $Q(s, a; \theta_i)$

Sampling from Replay Buffer: $(s, a, r, s') \sim D$

This equation is important in optimizing the DQN algorithm for Q-learning. It calculates the loss using mean squared error (MSE) between the target Q-value and the predicted value. The loss serves to correct the weights of the neurons in the network. The data is extracted from the previous experience storage or replay buffer[59].DQN gathers information from the environment and state interactions. This information is stored in a memory to train the model by mixing the sampled experience data with the current update data. Therefore, every time the model starts learning it combines samples from the replay buffer. This contributes to the stability of the model. If it does not use the stored experience, the model can stop or it will have difficulty handling complex scenarios. The replay buffer capacity has an impact on the performance of the algorithm. But this depends on which type of RL is applied because for some RL models, it increases the performance for the other it is the opposite. In the case of DQN, it shows good improvement when the capacity is increased. Even if some RL algorithms increase the performance proportionally with replay capacity, at some point, it reaches a limit where the performance is reduced [24]. The picture in Fig. 3.3 depicts the layout of the DQN. It shows the interaction among different parts of the algorithm. DQN updates the weights every time it receives error feedback. This is to reduce the error as much as possible. However, sometimes DQN can struggle with a shortage of feedback rewards. This can be receiving negative rewards frequently. Another challenge with DQN is its sensitivity to hyperparameters such as learning rate, batch size, etc. To address this issue, Double DQN was introduced. This algorithm uses two separate networks for action selection and value estimation. This helps prevent the algorithm from overfitting and improves its performance [40].

Double Deep Q-Networks (DDQN)

In the above, we have discussed both Q-learning and DQN, which can yield good results in solving complex problems. However, these methods have their own drawbacks, since they tend to select

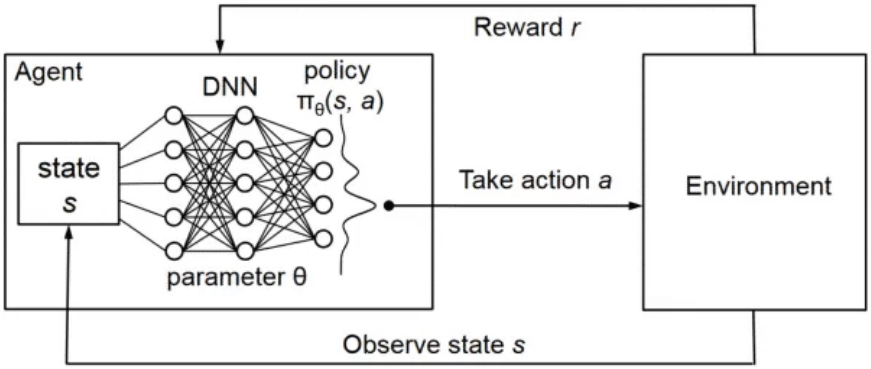


Figure 3.3: Deep q-learning layout [40]

only the action that has the highest Q-value. This method is based on the Bellman equation, which aims to maximize the expected future reward. This can lead to suffer from overestimate bias. Therefore,to solve such kind of problems,DDQN is created [30].Unlike Q-learning and DQN, DDQN uses two separate Q networks one for selecting actions and the other for evaluating them. Using one network for both evaluation and selection leads to an overoptimism problem. Therefore,the separation of the main Q network and the target Q network helps to have a stable and more accurate algorithm [30].The way how the Q-value is updated is the basic difference between the double deep Q-learning and the previous methods. Therefore, the decoupling of the two networks leads to learning stability and ultimately better policy performance in different scenarios [57] .The DDQN solves the overestimation bias that was the main problem in DQN. But this leads to focusing only on reducing the Q-value without putting great emphasis on how valuable each state and action are.

Dueling DQN

The improvement in DDQN, helps to stabilize and increase learning efficiency compared to its predecessor DQN. However, it still has a significant problem that can reduce the efficiency of the learning process for the algorithm. Each Q-value result is a single number without explicitly defining the state. This single number is just told as, in general, it is the result of the convergence of the action and the state. This can make it difficult for the agent to precisely assess which states are more valuable, regardless of the action taken in the state [28]. This limitation can lead the policy learning process to suffer in environments since it is hard to distinguish between different state-action pairs. During this situation, the agent has difficulty in distinguishing actions that lead to the same outcome. This makes it difficult to assume that all these outcome states are identical, even if they are not identical in reality. To overcome this problem, Dueling DQN was introduced. This approach has the same architecture as both DQN and DDQN, but the network

structure is different. Dueling DQN decouples the estimation of Q-values into two components. The first part represents the overall worth value of being in a particular state, and the second is the advantage of taking a specific action in that state. This separates learn value and advantage, which is very important to improve overall performance [28]. As shown in Figure 3.4, the main

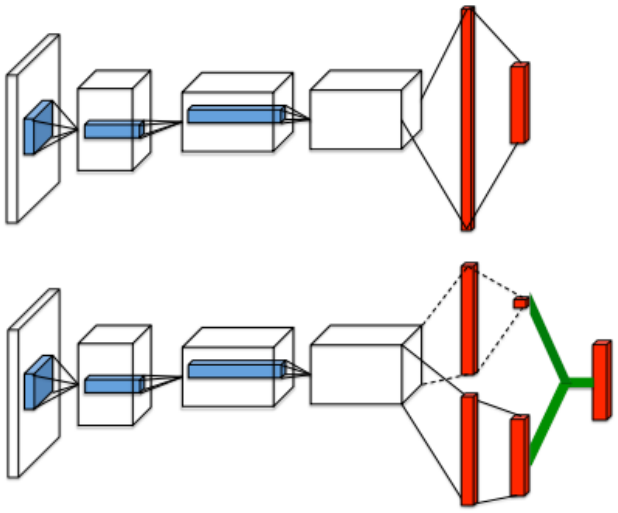


Figure 3.4: Single stream Q-network and Dueling DQN architecture [80]

architectural difference between a standard deep Q-network and the proposed dueling network architecture. In the traditional Q-network, the network consists of a single stream that processes input through shared convolutional layers. The flow continues into fully connected layers that produce results in the form of Q-values for every action. In contrast to this, the dueling network architecture has the same initial convolutional feature extractor as the traditional one, but then splits into two independent streams in the middle. These two output results are then combined through a special module to give the final Q-values. This decoupling of the two streams enables the dueling network to learn which states are valuable. This helps achieve potentially more efficient and robust learning even when the precise effect of individual actions is uncertain or irrelevant [80] [28].

Deterministic Policy Gradient (DDPG)

The previous methods are effective for fixed actions, such as left, right, up, and down. However, it can be difficult to find the optimal solution in continuous action domains due to the infinite set of possible actions. To solve such problems, DDPG was introduced in 2015 by Lillicrap.DDPG was specifically created for environments where actions are continuous [72].In methods that are suitable for discrete actions, such as Q-learning, it learns only the action-value function. But DDPG learns simultaneously both the policy, which we call the actor, and the critic. The first one determines the continuous action, and the second one evaluates those actions. The main

purpose of the separation is to efficiently optimize the policy in continuous space. Unlike the discrete method, it is difficult to find the best action in continuous action spaces. Therefore, DDPG has to apply gradients to solve this. The approach is built on the capability of Q-learning, where the Q-function finds a policy that maximizes the total reward. The concept is depicted with the action value function, denoted as $Q(s, a)$, which represents the expected return by taking an action in a state, governed by the policy. This idea is modeled mathematically using the formula below. The ideas behind Q-learning and Deep Deterministic Policy Gradient (DDPG) focus on how they learn the Q-function and determine the optimal policy

[84] [72]. The optimal action $a^*(s)$ in state s is the action that maximizes the Q-value:

$$a^*(s) = \arg\max_a Q^*(s, a)$$

Where:

- $a^*(s)$ is the optimal action for state s
- $Q^*(s, a)$ is the optimal action-value function
- $\arg\max_a$ means choosing the action a that gives the highest Q-value

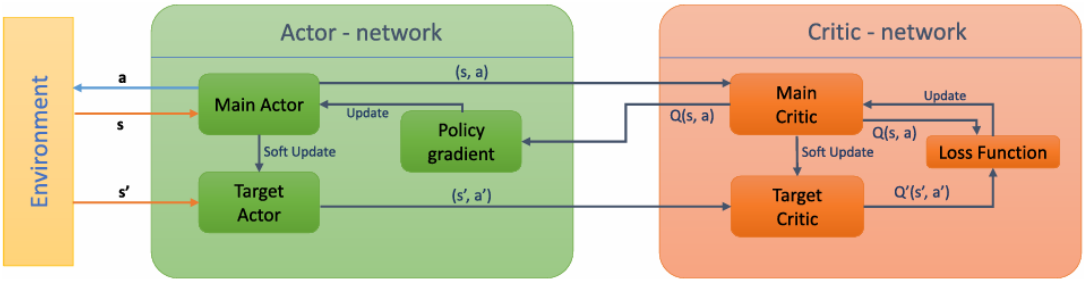


Figure 3.5: Architecture of DDPG [80]

The diagram represents the training process of the Deep Deterministic Policy Gradient (DDPG) algorithm. As depicted in the figure, it consists of the actor-critic architecture for effective learning in continuous action spaces. The process begins with the environment giving the actor the current state and accepting the action that should be taken, which creates the next state for the actor. The critic evaluates the quality of the selected action by estimating the Q-value [83] [84].

3.5.2 Resource allocation method in SAGIN

The SAGIN system architecture is advanced and can be the best solution for creating connectivity in areas with low communication network infrastructure However, this feature also has

its own drawbacks. Due to the integration of different types of networks namely air, ground, and space it create high complexity in managing and achieving optimal traffic transmission [54]. As a result, resource allocation and user association require well innovative methods that can enhance the overall network performance. Therefore by increasing throughput and minimizing power consumption we can utilize and allocate resources effectively. To get these results there are various methods that can be applied to manage this resource allocation. Some of the main approaches used in managing are listed in the below table.

Category	Methods/Techniques
Mathematical Optimization	**Classical Optimization**: This is a traditional way of resource allocation that can reduce the delay and make the overall network stable, it looks simple but it is an effective way of handling the complex resources of SAGIN.Some of the techniques are Convex optimization, Lagrange dual decomposition, Successive convex approximation (SCA) **Nonlinear Optimization**: This method helps to handle when there are functions that are not linear some of the methods are Integer nonlinear programming (INLP), Mixed integer nonlinear programming (MINLP), Geometric programming **Heuristic/Metaheuristic**: This method is mostly used when we are looking for a solution that is near to optimal during some time frame some of the methods are Greedy algorithms, Genetic algorithms, Tabu search, Simulated annealing, Adaptive particle swarm optimization
Dynamic Optimization	**Dynamic Programming (DP)**: This method is applied by chopping the problem into small and manageable parts and each small part is optimized through a small time slice. Some of the methods are Optimization through time-varying models for satellite and UAV communication **Markov Decision Process (MDP)**: A mathematical approach that helps to allocate resources using an agent's interactions with an environment by determining the future by the current state and action.
Game Theory	**Matching Theory**: Matching theory algorithm helps to group resource requester based on their preference this helps to use the resource effectively. **Auction Theory**: Auction theory is an algorithm type that allocates various communication resources based on the bid and assigns the resource to the highest. **Differential Games**: Stackelberg games for continuous-time dynamic resource allocation **Bargaining Game**: Creating a strategy based on negotiation among different entities. These strategies create a management where all the entity share the resources fairly. **Coalition Games**: This method mainly focuses on grouping teams that collaborate to achieve the goal in SAGIN and by doing this can reduce the expense and optimize file delivery.
AI-Based Methods	**Machine Learning**: It use for effective resource allocation using K-means clustering, Reinforcement learning (RL), Monte Carlo Markov decision processes (MC-MDP) **Deep Learning**:It is good choice for dynamic and complex networks where it needs Offloading and caching optimization, SAGIN performance improvement, Heterogeneous resource orchestration **Deep Reinforcement Learning (DRL)**:In SAGIN it is always faces challenges such as fast-changing network conditions and resource limitations especially when the users are high and the base station is UAV in case of IOT data gathering and processing. Currently, DRL is the preferred solution to such kind of challenges by autonomously optimizing scheduling, resource allocation, and energy consumption. **Multi-Agent Learning (MAL)**: Different agents Cooperative by updating each other information about the environment for decision-making

Table 3.2: Different types of Resource Allocation Methods for SAGIN [54]

3.6 Related works

This part revolves around the review of the previous works that are related to this thesis. Therefore it will give insights into the work related to Resource allocations and user associations in the SAGIN communication architecture. Many papers talk about the related issue. This section aims to provide insight into some of these works.

3.6.1 Current State of SAGINs

SAGIN architecture is emerging as a good solution for creating seamless connectivity for covering inaccessible areas. This will be realized by the integration of satellite, air, and ground communication infrastructure. This integration plays a big role in covering a wider area. The architecture of these networks is composed of different types of communication layers, therefore it requires various protocols to work together for its functionality [17]. This can make it more complex and it needs high expertise. SAGIN is currently a potential and advanced technology for the future emerging 6G offering lower latency and more flexibility than the former communication architecture. Compared with the previous generations such as 4G that primarily relies on the cellular network. The traditional terrestrial network has main limitations in terms of communication coverage and data transmission performance. SAGIN's is addressing these challenges and shortcomings that appear in the former technologies [12].To address this kind of problem, researchers are actively working in the full realization of SAGIN to support the emerging 6G and the exponentially growing number of communication devices. In the future, the emergence of 6G will introduce various innovative applications that demand high data transmission and low latency. These features will be characterized by ubiquitous coverage, the Internet of Things (IoT), and holographic communication [73]. Even though SAGIN has its own strengths, it also comes with limitations that can challenge its effective application. One of the main challenges is security. When applications involving 6G-SAGIN are implemented, they must be evaluated against standard security frameworks. Trust mechanisms need to be established within the SAGIN architecture before data transmission can occur. This is particularly difficult since the technology integrates three different layers and is vast in scope.

Another major concern is vulnerability, especially in airborne base stations such as UAVs (Unmanned Aerial Vehicles), which are often not fully resistant to potential cyber-attacks [92]. Reliability is also a key concern. To meet user expectations, the system must ensure that all applications consistently function as expected and fulfill user requirements. Agility and timeliness are equally critical.

The speed at which the system's components are deployed and perform their tasks to deliver the service is another issue that must be emphasized. This should not be compromised by new components that are not adaptable to the system. Therefore, by increasing agility, the system must maintain the timeliness of overall architectural performance [66].(It has to e written more)

Chapter 4

Analysis

This section goes deep into the insight of the nature of the system to be developed. By exploring the requirements of the objectives and the stakeholders in general, along with the interaction among each other, the role of each stakeholder will be examined in detail. This part serves as a roadmap to the design that provides a solution for the problem mentioned in the problem definition.

4.1 Modeling Scenario for the SAGIN

The scenario chosen for this project involves using the SAGIN architecture to solve communication problems, where the ground communication infrastructure is poorly implemented. SAGIN plays a crucial role in areas where there is no communication network or during events that make the network congested. Using satellites and UAVs ensures reliable communication.

The goal of this scenario is to enable secure and seamless connectivity for obtaining communication services in areas where there is no connectivity. Therefore the aim is to create seamless connectivity in parts of the where there is no access to communication by optimizing throughput. Three UAVs and two LEO satellites are coordinating to serve as base stations for the users on the ground. Resource allocation to the users will be done in an optimal way. Each base station is equipped with 20 communication channels. Therefore, the scenario has a total of 100 channels available for users. The number of users in the simulation is 30, but in real life, the capacity can go up to 1,000.

Table 4.1: Resource for the scenario of the desert

SAGIN component	number
LEO	2
UAVs	3
User	30

4.2 Problem formulation

The problem formulation will start by formulating the specific problem to be solved. This helps us to have a clear goal of what we are going to achieve. By doing this, it enables us to clearly define the objectives of this thesis. In our case, two main objectives are identified namely.

- Optimize user association
- Maximize Throughput

The first objective is to create a mechanism where the allocation of resources will be delivered with maximal throughput. Even if there is a shortage of communication infrastructure or maximum traffic of users. It aims to create a solution where the allocated resources use an optimized communication flow.

Bandwidth: This resource plays in the overall transmission speed of the SAGIN communication. It is the amount of frequency spectrum assigned to each user who demands the service. Therefore the allocated frequency serves as for data transmission to the UE.
The bandwidth allocation is important to maximizing the overall data rate of the communication and to increase network efficiency. Therefore, it is crucial to estimate the necessary frequency for different services and allocate the resources to give appropriate bandwidth for different services. By doing this it optimizes the performance of the transmission [49] [90].

Power: The algorithms for power resources help to manage energy efficiently. By doing this, aerial vehicles such as UAVs can operate for long periods without interrupted due to the shortage of power. This is crucial especially when there are harsh environments that are affected by either floods or any area that experiencing high fire. In these kinds of difficult situations where it needs to maintain connectivity to communicate with people in bad situations, it is very important to conserve the power that is necessary for the long-term work of aerial-based base stations. Managing the power Effectively ensures that the aerial vehicles remain operational in the air without going to the ground. This helps to improve the overall communication performance. The overall result of this resource management also increases the network's reliability [90].

User association : The other thing that should be optimized in the SAGIN architecture is User association which impacts the overall network performance, and is key for the existence of seamless connectivity. The main goal of the user association is to create an effective optimal allocation method for the user device connected to the base stations in the air or space. Due to the complex non-convex nature of the mathematical model that represents the SAGIN, we cannot solve it using only traditional optimization methods [38]. The traditional method can such as simply locate the optimal point on the model graph to determine where the maximum or minimum value is found. But for complex mathematical representations of networks that cannot be easily converged the traditional method cannot be used to get the optimal value that is used for the resource association. The combinatorial nature of the user association can also make it more difficult to handle with simple optimization algorithms. It is important to address

these challenges using more advanced techniques such as DRL for the decision-making process for user device associations with base stations. DRL helps the mathematical formulation to be converged so that we get optimal solutions to our non-convex problems. Achieving the optimal combination of the UE with the aerial base stations able us to use the resource efficiently and increase the overall network performance [90].

The other objective that should be formulated is to consider the energy consumed during resource allocation. The mechanism has to implement a method aimed at reducing energy consumption by the network components while providing a service for the requested service. By doing this, unnecessary energy consumption will be reduced by properly managing the components of the network and the energy consumed to operate it. Therefore, the resource allocation for the service must prioritize communication efficiently with maximum flow and reduce the energy consumed in the network components that operate to provide the necessary service to the requester of the service. To search for the algorithm that must address these problems, the first step will be to precisely formulate the problem to help in devising optimized strategies for providing communication facilities for areas that have shortages of communication resources.
The formulas below are key tools to model and optimize the SAGIN network. The path loss through the way of one communication node, the data transmission rate, and the energy lost during the communication have to be quantified Knowing this helps in optimal resource utilization. Therefore, when one user wants to connect to the base station the algorithm decides both on the power strength toward the user and the radius between them. Therefore, these formulas will be the backbone of our design of the algorithm[91].

Path Loss Formula:

$$L_{i,j} = 20\log_{10}\left(\frac{4\pi f c(d_{i,j}^2 + r_{i,j}^2)}{c}\right) + PLoS_{i,j}\gamma_{LoS_{i,j}} + (1 - PLoS_{i,j})\gamma_{NLoS_{i,j}} \tag{4.1}$$

Line-of-Sight Probability Formula:

$$PLoS_{i,j} = \frac{1}{1 + b1\exp\left(-b2\arctan\left(\frac{d_{i,j}}{r_{i,j}}\right) - b1\right)} \tag{4.2}$$

Transmission Rate Formula:

$$R_{i,j} = B_{i,j}\log_2\left(1 + \frac{P_{i,j}10^{-\frac{L_{i,j}}{10}}}{\sigma^2}\right) \tag{4.3}$$

$$\text{Energy Efficiency (EE)} = \frac{\sum_i \sum_j R_{i,j}}{\sum_i \sum_j P_{i,j}}$$

$$\text{Resource Utilization } (U) = \frac{\sum_i \sum_j \sum_k \phi_{i,j,k} T_i}{\sum_j \sum_k f_{j,k}}$$

Optimization Function

$$O = \beta_i \omega_3 \sum_i \sum_j \sum_k \phi_{i,j,k} + \omega_2 U + \omega_1 \phi_{i,j,k} R_{i,j,k}$$

Constraints

1. **Power Constraint**:
$$P_i \leq P_{MAX} \quad \forall i$$
2. **Rate Constraint**:
$$R_{i,j,k} \geq R_{MIN} \quad \forall i, \forall j, \forall k$$
3. **Single Assignment Constraint**:
$$\sum_j \sum_k \phi_{i,j,k} \leq 1 \quad \forall i$$
4. **Station Capacity Constraint**:
$$\sum_i \phi_{i,j,k} T_i \leq f_{j,k} \quad \forall j, \forall k$$
5. **Resource Constraint**:
$$\sum_i \phi_{i,j,k} \leq \gamma_{j,k} \quad \forall j, \forall k$$
6. **Binary Assignment Variable**:
$$\phi_{i,j,k} \in \{0, 1\} \quad \forall i, \forall j, \forall k$$

Variable	Description
$R_{i,j}$	Transmission rate between user i and SAGIN j
$B_{i,j}$	Transmission bandwidth
T_i	Task
$P_{i,j}$	Transmission power
$L_{i,j}$	Channel gain
σ^2	Noise power

Initialization:

- Initialize a neural network with appropriate architecture to approximate the action-value function $Q(s, a)$.
- Set hyperparameters: learning rate (α), discount factor (γ), exploration rate (ϵ).

Variable	Description
$L_{i,j}$	Path loss between entities i and j
f	Frequency of the signal
c	Speed of light
$d_{i,j}$	Distance between entities i and j
$r_{i,j}$	Radius between entities i and j
$PLoS_{i,j}$	Line-of-sight probability between entities i and j
$\gamma_{LoS_{i,j}}$	Line-of-sight path loss exponent for entities i and j
$\gamma_{NLoS_{i,j}}$	Non-line-of-sight path loss exponent for entities i and j
$b1$	Constant parameter for line-of-sight probability formula
$b2$	Constant parameter for line-of-sight probability formula

Table 4.2: Description of variables

4.3 Selecting parameter

This part revolves around the selection of parameters used to run the code. The code has two types of configuration parameters. The first part is used to configure the environment (scenario), and the second part is used to define the decision-maker agent. This helps to define the simulation according to real-world characteristics. Therefore, parameters play a crucial role in defining the behavior of the simulation. These configuration parameters are used here based on real-world parameters that define the communication technology. The goal of this project is to enable the user equipment on the ground to stay connected with the air base stations in the sky, namely the UAES and the LEO. The integration network requires ensuring seamless connectivity. One of the obstacles that can hinder seamless connectivity is when the SNR falls below the communication threshold. Obtaining the right SNR ensures obtaining a powerful signal. If the SNR is very low, it can lead to a poor signal. In such kind of situations, UE may not receive reliable service from the base station. For this reason, it is important to know the appropriate SNR threshold for the base stations[50].

Outage probability is an important metric to measure the strength of the signal between two communication points. This metric helps to determine the right SNR threshold when designing a communication network. Due to many factors that interfere with the signal, it is difficult to quantify all the factors that contribute to the poor quality of the signal. Therefore, outage probability is most of the time calculated with some assumptions [33].

As shown in Figure 4.1, when the different parameters are kept constant the outage probability decreases as the SNR increases. At 30 dB, the probability of signal failure is almost zero. This means during communication the signal between the UE and the station becomes strong.T his able the UE to transmit different types of multimedia data without any problems. After 15 dB, the probability of receiving a weak signal sharply decreases. Starting from around 5 dB,

the outage probability reaches almost %100. Since below below 5db there is no communication this serves as the threshold for communication. This thesis uses UAVs and satellites as a base station, therefore considering the distance from the UE it is better to choose 30 db. The figure in 4.1 uses Radio Frequency (RF) and Free Space Optics (FSO) as communication links. Therefore, for simulation purposes, the SAGIN is linked with due to RF and FSO. Based on this the simulation uses 30db.

$$P_{\gamma BM\,out,\infty} \approx (G\bar{\gamma})^{-D}$$

- $P_{\gamma BM\,out,\infty}$: Outage probability.
- G: Coding gain.
- $\bar{\gamma}$: Average SNR.
- D: Diversity order.

[29]

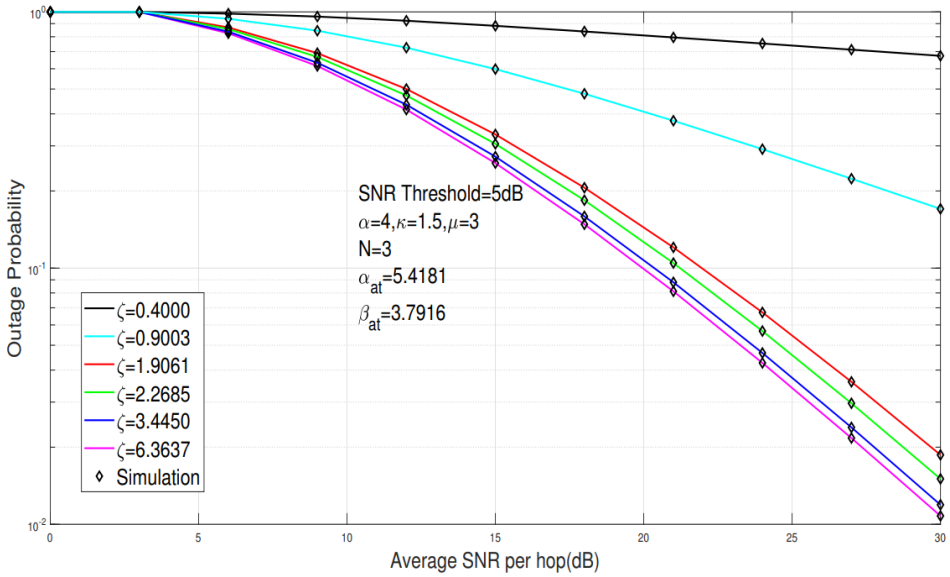


Figure 4.1: Outage probability compare to the SNR[50]

Chapter 5

Design

In this section, we will outline the comprehensive conceptual design of the SAGIN system, based on the preceding analysis from the previous chapter. It will provide a high-level architectural overview along with the data flow among the system components. The data flow will emphasize all the processes of resource requesting, starting from the requesting of resources by the service requester, up to the allocation of resources within the system.

5.1 System Overview

In this section, the high-level components of the SAGIN system are depicted, as shown in Figure 5.1. This visualization aids in understanding the overall system architecture and the interactions among its main components. The diagram illustrates the flow of data using labelled arrows, indicating the transfer of information between components. The process begins with a service requester sending a request for communication services to the Environment, which contains the Network infrastructure. The Environment then sends the state of its resources to an agent, allowing the agent to assess resource availability. Upon successful allocation, the agent receives a positive reward. Using deep reinforcement learning, the agent learns from the environment and optimizes the resource allocation. If it is successful, resources are allocated to the service requester. Otherwise, the agent receives a negative reward, that helps to improve for its next strategies. Figure 5.2 will depict each data flow in detail, facilitating a comprehensive understanding of the system's operation.

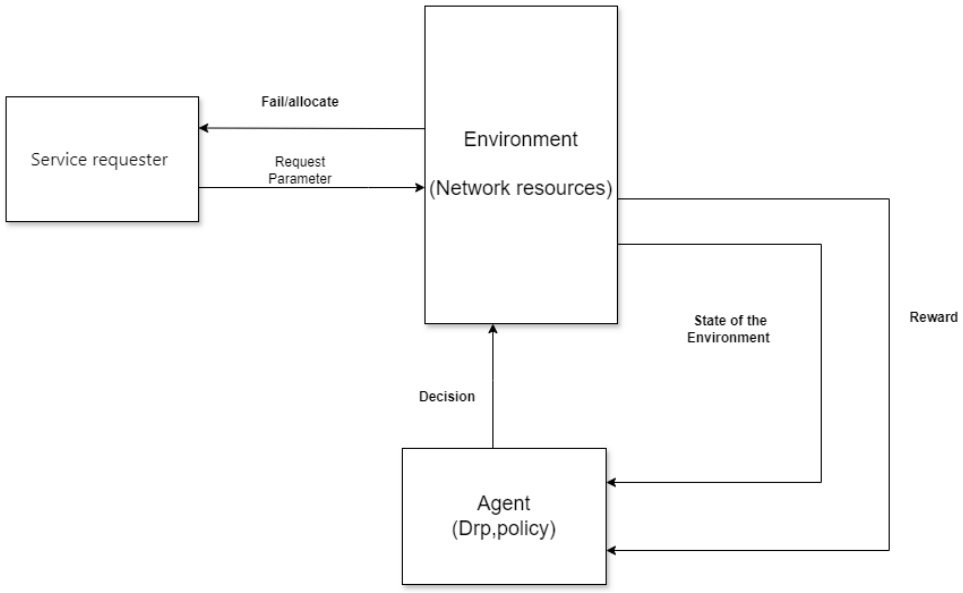


Figure 5.1: System overview of the SAGIN

Table 5.1: Variables in the Agent (GEO Satellite)

Variable	Definition
`name`	Name of the agent (LEO satellite).
`position`	Current position of the agent (latitude and longitude).
`communication_range`	Range within which the agent can establish communication.
`energy_level`	Current energy level of the agent.
`task_queue`	Queue of tasks assigned to the agent.
`state`	Current state of the agent (e.g., idle, communicating).

Table 5.2: Variables for Deep Q-Learning in the Agent

Variable	Definition
`state`	Current state of the agent (observation from the environment).
`action`	Action taken by the agent in response to the current state.
`reward`	Reward received by the agent for taking the action in the current state.
`next_state`	Next state of the agent after taking the action.
`done`	Flag indicating whether the episode has ended (terminal state reached).
`Q-table`	Table storing Q-values for state-action pairs.
`epsilon`	Exploration rate determining the probability of taking a random action.
`learning_rate`	Rate at which the Q-values are updated based on new information.
`discount_factor`	Discount factor determining the importance of future rewards.

Chapter 6

Implementation

This part will cover the implementation of the SAGIN system based on the conceptual design presented in Chapter 5. The implementation will always be oriented towards meeting the requirements exposed in the problem definition, aiming to be useful in answering the research question presented at the beginning of the report. This section will consist of explanations for each of the steps performed in the implementation.

6.1 Programs and Libraries

The SAGIN system is being developed with Python as its primary programming language. Python was selected for its compatibility with deep reinforcement learning and the abundance of existing resources dedicated to this field. Moreover, Python boasts an extensive array of reinforcement learning and network libraries, which will be leveraged extensively throughout our implementation. This choice ensures seamless integration of advanced techniques and facilitates the incorporation of state-of-the-art methodologies into our system.

Library	**Description**
`Gym`	Provides the environment interface required by the training algorithm.
`matplotlib`	Used for various visualizations.
`numpy`	Used for various calculations.
`tensorboard`	Used for visualizing training and testing performance.

6.2 Algorithms for the implementation

In complex communication systems that integrate space, air, and ground components, it's essential to first design a well-structured algorithm. Diving into development with out having a clear plan can be time-consuming and costly. This algorithm should serve as a clear blueprint to guide the coding process. Algorithms are systematic procedures that are used to solve specific

problems or accomplish defined objectives. They serve as the backbone of computer programming. Then, programmers can easily follow the guidelines to code the algorithms. This will help to give a clear image of the goal and the process. In this context, the algorithm is divided into three main parts, namely: the agent, the environment, and the main function part. Accordingly, the algorithms are depicted below. [9].

6.2.1 Algorithms for the Agent

Initialization

```
Initialize environment (UAVs, LEOs, agents, channels)
Initialize each agent:
    - Assign a random location near a UAV or LEO
    - Initialize replay memory
    - Initialize policy network (DNN)
    - Initialize target network (copy of policy network)
    - Initialize optimizer
Set  (exploration rate),  (discount factor), and training parameters
```

Main Training Loop

```
for each episode do
    Reset environment and agent locations

    for each time step in the episode do
        for each agent do
            Observe current state
            Select action using -greedy strategy:
                if random >  then
                    action ← random valid action
                else
                    action ← argmax Q-value from policy network
                end if
        end for

        Apply all selected actions in the environment
```

```
        for each agent do
            Observe next state and reward
            Store (state, action, next_state, reward) in replay memory
        end for

        for each agent do
            if memory contains enough transitions then
                Sample a batch of transitions
                For each transition:
                    Compute current Q-value using policy network
                    Compute target Q-value using reward +  × max Q(next_state) from target
                    Compute loss between current and target Q-values
                end for
                Perform backpropagation and update policy network
            end if
        end for

        if step is a target update interval then
            for each agent do
                Copy policy network weights to target network
            end for
        end if

        Decay  if using a decay schedule
    end for
end for
```

Reward Computation

```
function Compute_Reward(action_vector, agent_index)
    Determine the station and channel selected by agent
    Compute distance between agent and station
    Compute received power

    if received power = 0 then
        reward ← negative cost
        QoS ← not satisfied
```

```
    else
        interference ← 0
        for each other agent do
            if using the same channel then
                Compute received power from that agent's signal
                Add to interference
            end if
        end for

        Remove self signal from interference
        Compute SINR ← received power / (interference + noise)
        Compute data rate ← bandwidth × log2(1 + SINR)

        if SINR  QoS threshold then
            reward ← +1
            QoS ← satisfied
        else
            reward ← negative cost
            QoS ← not satisfied
        end if
    end if

    return reward, QoS, data_rate
end function
```

Experience Replay

```
function Store_Transition(state, action, next_state, reward)
    Add (state, action, next_state, reward) to memory
    If memory exceeds capacity, overwrite oldest
end function
```

Optimization Step

```
function Optimize_Model()
    if memory size < batch size then
        return
    end if

    Sample a batch of transitions
    for each transition in batch do
        current_Q ← Q-value from policy network for selected action
        next_Q ← max Q-value from target network for next state
        target_Q ← reward +  × next_Q
        loss ← difference between current_Q and target_Q
    end for

    Backpropagate loss
    Clip gradients if needed
    Update policy network weights
end function
```

Target Network Synchronization

```
function Update_Target_Network()
    Copy weights from policy network to target network
end function
```

6.2.2 Algorithms for the Environment

Initialization

```
Initialize scenario parameters (number of UAVs, LEOs, channels, radius)
Initialize environment:
    - Generate UAV locations with altitude and position
    - Generate LEO locations with orbital positions
Initialize stations (UAVs and LEOs) with location and coverage radius

Initialize each agent:
```

```
    - Assign random initial location near UAV or LEO
    - Initialize replay memory
    - Initialize policy network (DNN)
    - Initialize target network (copy of policy network)
    - Initialize optimizer

Set  (exploration rate),  (discount factor), and training parameters
```

Main Training Loop

```
for each episode do
    Reset environment:
        - Reset channel states at all stations
        - Reset agent locations near UAVs or LEOs

    for each time step in the episode do
        for each agent do
            Observe current state (location, channel occupancy, etc.)
            Select action (station and channel) using -greedy policy:
                if random >  then
                    action ← random valid station-channel pair
                else
                    action ← argmax Q-value from policy network
                end if
        end for

        Apply all agents' selected actions:
            - Update channel occupancy in stations
            - Compute received power based on distance and station type
            - Compute interference from other agents on same channel

        for each agent do
            Compute reward using:
                - Received power and interference
                - SINR and QoS threshold
            Store (state, action, next_state, reward) in replay memory
        end for
```

```
        for each agent do
            if replay memory size  batch size then
                Sample batch transitions
                Compute loss between policy and target networks using Q-learning update
                Optimize policy network via backpropagation
            end if
        end for

        if time to update target network then
            for each agent do
                Copy policy network weights to target network
            end for
        end if

        Decay  according to schedule
    end for
end for
```

Reward Computation

```
function Compute_Reward(agent_action, agent_index)
    Extract selected station and channel from agent_action
    Calculate distance between agent and station
    Compute received power based on station type and distance

    if received power = 0 then
        reward ← negative penalty (out of range)
        QoS ← not satisfied
    else
        interference ← sum of received power from other agents using same channel
        Compute SINR ← received power / (interference + noise power)
        Compute data rate ← bandwidth × log2(1 + SINR)

        if SINR  QoS threshold then
            reward ← positive reward (+1)
            QoS ← satisfied
        else
            reward ← negative penalty (QoS not met)
```

```
            QoS ← not satisfied
        end if
    end if

    return reward, QoS, data rate
end function
```

Experience Replay

```
function Store_Transition(state, action, next_state, reward)
    Append transition (state, action, next_state, reward) to replay memory
    If memory size exceeds capacity, remove oldest transition
end function
```

Optimization Step

```
function Optimize_Model(agent)
    if replay memory size < batch size then
        return
    end if

    Sample batch transitions from replay memory
    for each transition in batch do
        current_Q ← Q-value for action from policy network
        next_Q ← max Q-value for next_state from target network
        target_Q ← reward +  × next_Q
        Compute loss between current_Q and target_Q
    end for

    Backpropagate loss and update policy network weights
    Optionally clip gradients to stabilize training
end function
```

Target Network Synchronization

```
function Update_Target_Network(agent)
    Copy policy network weights to target network
end function
```

6.2.3 Algorithms for Main

Initialization

```
Import necessary libraries: copy, json, argparse, matplotlib.pyplot, torch
Import custom modules: Scenario, Agent, DotDic
Set device to CUDA if available else CPU

Load scenario config and options from JSON files
Initialize scenario environment (UAVs/LEOs)
Create agents list:
    for i in range(number_of_agents):
        Initialize Agent(i) with scenario, device, and hyperparameters

Initialize global training parameters:
    - episode counter = 0
    - global step counter = 0
    - initialize epsilon parameters (eps_min, eps_max, eps_decay)
    - discount factor
    - batch size, update frequency, etc.
Initialize data structures to store results:
    - episode_rewards, episode_steps, episode_store, episode_data_rates
```

Main Training Loop

```
while episode < total_episodes do
    Reset environment states for all agents: state, next_state = zeros
```

```
Initialize cumulative_reward = 0
Initialize cumulative_data_rate = 0
step = 0

while step < max_steps_per_episode do
    Compute epsilon threshold:
        eps_threshold = max(eps_min, eps_max - eps_decay * global_step)

    for each agent i do
        action[i] = agent[i].Select_Action(state, eps_threshold)
    end for

    for each agent i do
        QoS[i], reward[i], data_rate = agent[i].Get_Reward(action, action[i], state, sce
        next_state[i] = QoS[i]
        cumulative_reward += reward[i]
    end for

    Compute average data rate across all agents at current step:
        average_rate = sum(data_rate) / number_of_agents
        cumulative_data_rate += average_rate

    for each agent i do
        agent[i].Save_Transition(state, action[i], next_state, reward[i], scenario)
        agent[i].Optimize_Model()
        if step % target_update_frequency == 0 then
            agent[i].Target_Update()
        end if
    end for

    Update state = next_state
    Increment step and global_step

    if QoS for all agents satisfied then
        break from step loop
    end if
end while

Append cumulative_reward to episode_rewards
Append step count to episode_steps
Append episode number to episode_store
```

```
    Log episode summary: episode number, steps, total reward, avg data rate

    Increment episode counter
end while
```

Agent Action Selection (ε-greedy)

```
function Select_Action(state, epsilon)
    if random() > epsilon then
        return random valid action
    else
        return argmax Q-value from policy network given state
    end if
end function
```

Reward Computation

```
function Get_Reward(action_vector, agent_action, state, scenario)
    Determine station and channel selected by agent_action
    Compute distance between agent and station
    Compute received power

    if received power == 0 then
        reward = negative penalty
        QoS = False
    else
        Calculate interference from other agents using the same channel
        Compute SINR = received_power / (interference + noise)
        Compute data_rate = bandwidth * log2(1 + SINR)

        if SINR >= QoS_threshold then
            reward = +1
            QoS = True
```

```
        else
            reward = negative penalty
            QoS = False
        end if
    end if

    return QoS, reward, data_rate
end function
```

Experience Replay and Model Optimization

```
function Save_Transition(state, action, next_state, reward, scenario)
    Add (state, action, next_state, reward) to replay memory
    If memory capacity exceeded, overwrite oldest transitions
end function

function Optimize_Model()
    if memory size < batch size then
        return
    end if

    Sample batch of transitions
    for each transition in batch do
        current_Q = Q-value for action from policy network
        next_Q = max Q-value from target network for next_state
        target_Q = reward +  * next_Q
        loss = difference(current_Q, target_Q)
    end for

    Backpropagate loss through policy network
    Clip gradients if needed
    Update policy network parameters
end function
```

Target Network Update

```
function Target_Update()
    Copy weights from policy network to target network
end function
```

Result Logging and Visualization

```
After all episodes:
    Plot steps per episode vs episode number
    Plot cumulative rewards vs episode number
    Save plots to file
    Optionally, save logs to CSV or text file
```

6.3 Coding

The implementation code is modified based on this work [16] to transform it into SAGIN. I want to thank the authors for making their code freely available, which reduced the challenge to my work, and it also helped me to learn more about this field. The code was modified from terrestrial base stations to SAGIN by transforming the ground base stations into UAVs and LEO. I have followed Agile methodology to handle my coding process. The contents of the code are composed of a scenario that defines the environment, and there is also user equipment that is represented as an agent in the code. I used 3 UAVs and 2 Leo satellites, which are defined in the configuration, and it continues as a fixed number throughout the project. The snippet of code in Fig 6.2 is used to calculate the distance of the agent from any station in the air either from satellites or the UAVs.The formula below is used to calculate the distance(d). To get the distance, it uses the three-dimensional coordinates of the agent and the base station in the air. The distance of the agent helps in the reward process because if the agent is outside of the base station's area, there is no point in getting resources.

$$d = \sqrt{(x_{\text{agent}} - x_{\text{station}})^2 + (y_{\text{agent}} - y_{\text{station}})^2 + (z_{\text{agent}} - z_{\text{station}})^2}$$

6.4 Environment

The interaction of neural, the number of agents, configuration

```python
def Set_Location(self, scenario):
    Loc_UAV, Loc_LEO = scenario.UAV_Location(), scenario.LEO_Location()
    Loc_agent = np.zeros(3)
    Loc_type = choice(["UAV", "LEO"])

    if Loc_type == "UAV":
        LocU = choice(Loc_UAV)
        r = self.sce.rUAV * random()
    else:
        LocU = choice(Loc_LEO)
        r = self.sce.rLEO * random()

    theta = uniform(-pi, pi)
    Loc_agent[0] = LocU[0] + r * np.cos(theta)
    Loc_agent[1] = LocU[1] + r * np.sin(theta)
    Loc_agent[2] = LocU[2]

    return Loc_agent
```

Figure 6.1: Snippet of code to get the position of the user equipment

6.5 Simulation Results

This chapter depicts the graphical result of the simulation tested with different learning rates, neural network structures, and the number of hidden. The different learning rates were tested to see how they influenced the result's performance. The range of learning rates for the simulation range from 0.01 to 0.00025. The range of the learning rate is from 0 to 1. A higher learning rate enables the model to learn more, on the other hand, a lower rate leads the algorithms to learn little[31]. For this simulation, three learning rates were chosen. First, it was tested with 0.01 which is a slightly higher learning rate compared to the others. The results are depicted in Fig 6.1 below, it does not converge until it reaches 300 episodes. Next, it was tested with 0.00005, which is slightly lower than the first learning rate. with this on,e the result was improved but to see further improvement it tested using 0.0025. The third result was more or less the same as the second learning rate. Based on these findings, 0.0025 was selected for testing the rest of the simulation. This rate was chosen for two reasons. First, it performed almost the same as the smaller rates, and second, choosing a slightly higher learning rate empowers the capability to learn more. By carefully choosing this result it contributes to the algorithm's overall performance and stability.

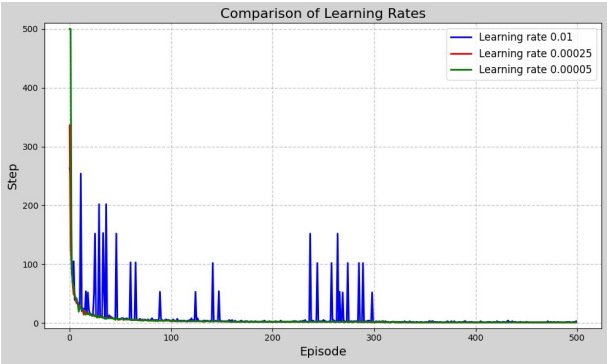


Figure 6.2: The training step with different Learning rate

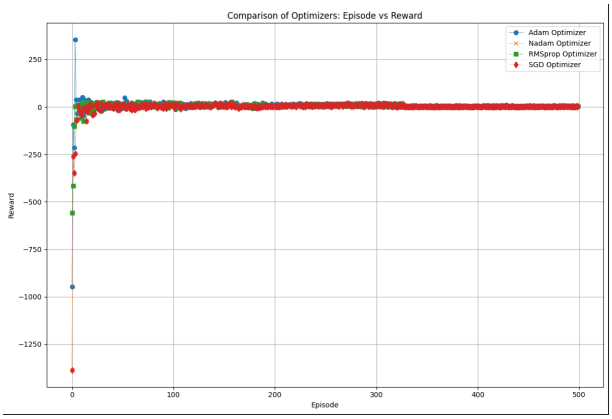


Figure 6.3: Simulation result that shows Epsoide with the reward

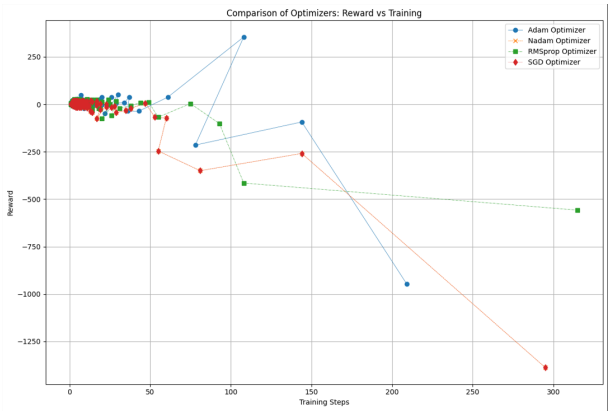


Figure 6.4: Simulation result that shows Reward with training steps

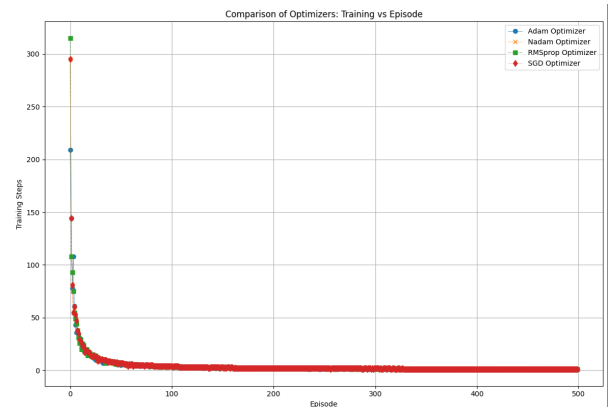


Figure 6.5: Simulation result that shows Episode with training steps

Chapter 7

Discussion

Chapter 8

Conclusion and Future work

8.1 Conclusions

8.2 Future Research Directions

Bibliography

[1] URL: https://www.itu.int/en/mediacentre/backgrounders/Pages/connect-2030-agenda.aspx#%3a~%3atext=The%20%27Connect%202030%20Agenda%20for%2cGoals%20%28SDGs%29%20by%202030.&text=ITU%27s%20Connect%202030%20Agenda%20%28PP-18%20Resolution%20200%2c%20Rev. (visited on 05/06/2024).

[2] https://telcomatraining.com/what-is-ue-user-equipment-the-3gpp-name-for-the-mobile-terminal/. [Accessed 16-09-2024].

[3] Accessed: (26/05/2024). URL: https://www.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html.

[4] EITCA Academy. "What is the significance of the discount factor (γ) in the context of reinforcement learning, and how does it influence the training and performance of a DRL agent?" In: (2024). Accessed: 2024-12-14. URL: https://eitca.org/artificial-intelligence/eitc-ai-arl-advanced-reinforcement-learning/deep-reinforcement-learning/deep-reinforcement-learning-agents/examination-review-deep-reinforcement-learning-agents/what-is-the-significance-of-the-discount-factor-gamma-in-the-context-of-reinforcement-learning-and-how-does-it-influence-the-training-and-performance-of-a-drl-agent/.

[5] Amrin Maria Khan Adawadkar and Nilima Kulkarni. "Cyber-security and reinforcement learning — A brief survey". In: *Engineering Applications of Artificial Intelligence* 114 (2022), p. 105116. ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2022.105116. URL: https://www.sciencedirect.com/science/article/pii/S0952197622002512.

[6] Kai Arulkumaran et al. "Deep reinforcement learning: A brief survey". In: *IEEE Signal Processing Magazine* 34.6 (2017), pp. 26–38.

[7] Sudhanshu Arya, Jingda Yang, and Ying Wang. "Towards the designing of low-latency sagin: Ground-to-uav communications over interference channel". In: *Drones* 7.7 (2023), p. 479.

[8] Riccardo Bassoli, FH Fitzek, and E Calvanese Strinati. "Why do we need 6G?" In: *ITU Journal on Future and Evolving Technologies* 2.6 (2021), pp. 1–31.

[9] Nicholas Bennett. *Introduction to Algorithms and Pseudocode*. Tech. rep. Working paper in Project "Exploring Modelling and Computation, 2015.

[10] Muhammad Zohaib Butt, Nazri Nasir, and Rozeha Bt ARashid. "A review of perception sensors, techniques, and hardware architectures for autonomous low-altitude UAVs in non-cooperative local obstacle avoidance". In: *Robotics and Autonomous Systems* 173 (2024), p. 104629. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2024.104629. URL: https://www.sciencedirect.com/science/article/pii/S0921889024000125.

[11] Sang-Yoon Chang et al. "Securing UAV flying base station for mobile networking: A review". In: *Future Internet* 15.5 (2023), p. 176.

[12] Jiming Chen, Han Zhang, and Zhe Xie. "Space-air-ground integrated network (sagin): A survey". In: *arXiv preprint arXiv:2307.14697* (2023).

[13] Qian Chen et al. "A survey on resource management in joint communication and computing-embedded SAGIN". In: *IEEE Communications Surveys & Tutorials* (2024).

[14] Nan Cheng et al. "6G service-oriented space-air-ground integrated network: A survey". In: *Chinese Journal of Aeronautics* 35.9 (2022), pp. 1–18.

[15] Cihan Tugrul Cicek et al. "UAV base station location optimization for next generation wireless networks: Overview and future research directions". In: *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*. IEEE. 2019, pp. 1–6.

[16] Cly1994jlu. *UARA-DRL*. Accessed: 2025-01-15. July 19, 2020. URL: https://github.com/cly1994jlu/UARA-DRL.

[17] Huanxi Cui et al. "Space-air-ground integrated network (SAGIN) for 6G: Requirements, architecture and challenges". In: *China Communications* 19.2 (2022), pp. 90–108.

[18] Shuping Dang et al. "What should 6G be?" In: *Nature Electronics* 3.1 (2020), pp. 20–29.

[19] Xinhang Ding, Zhilong Zhang, and Danpu Liu. "Low-delay secure handover for space-air-ground integrated networks". In: *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE. 2020, pp. 1–6.

[20] Charles Edeki. "Agile software development methodology". In: *European Journal of Mathematics and Computer Science* 2.1 (2015).

[21] Dhivya Elavarasan et al. "Forecasting yield by integrating agrarian factors and machine learning models: A survey". In: *Computers and Electronics in Agriculture* 155 (2018), pp. 257–282. ISSN: 0168-1699. DOI: https://doi.org/10.1016/j.compag.2018.10.024. URL: https://www.sciencedirect.com/science/article/pii/S0168169918311529.

[22] Bruce R Elbert. *The satellite communication applications handbook*. Artech house, 2004.

[23] OS Eluyode and Dipo Theophilus Akomolafe. "Comparative study of biological and artificial neural networks". In: *European Journal of Applied Engineering and Scientific Research* 2.1 (2013), pp. 36–46.

[24] William Fedus et al. "Revisiting fundamentals of experience replay". In: *International conference on machine learning*. PMLR. 2020, pp. 3061–3071.

[25] Lang Feng et al. “A Satellite Handover Strategy Based on MIMO Technology in LEO Satellite Networks”. In: *IEEE Communications Letters* 24.7 (2020), pp. 1505–1509. DOI: 10.1109/LCOMM.2020.2988043.

[26] Keith D. Foote. “A Brief History of Machine Learning”. In: *DATAVERSITY* (2021). Accessed: 2025-01-15. URL: https://www.dataversity.net/a-brief-history-of-machine-learning/.

[27] Vincent François-Lavet et al. “An introduction to deep reinforcement learning”. In: *Foundations and Trends® in Machine Learning* 11.3-4 (2018), pp. 219–354.

[28] Xiangwei Fu. “A Comparison of DQN and Dueling DQN in A Super Mario Environment”. In: *2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*. Atlantis Press. 2024, pp. 221–231.

[29] Jiayan Gan et al. “SAS-SEINet: A SNR-Aware Adaptive Scalable SEI Neural Network Accelerator Using Algorithm–Hardware Co-Design for High-Accuracy and Power-Efficient UAV Surveillance”. In: *Sensors* 22.17 (2022), p. 6532.

[30] Langxi Gao. “Comparison of dqn and double dqn reinforcement learning algorithms for stock market prediction”. In: *2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*. Atlantis Press. 2024, pp. 169–177.

[31] Tad Gonsalves and Jaychand Upadhyay. “Chapter Eight - Integrated deep learning for self-driving robotic cars”. In: *Artificial Intelligence for Future Generation Robotics*. Ed. by Rabindra Nath Shaw et al. Elsevier, 2021, pp. 93–118. ISBN: 978-0-323-85498-6. DOI: https://doi.org/10.1016/B978-0-323-85498-6.00010-1. URL: https://www.sciencedirect.com/science/article/pii/B9780323854986000101.

[32] Michael Goodwin. *What Is Latency?* Accessed: 2024-11-12. 2023. URL: https://www.ibm.com/topics/latency.

[33] Jing Guo, Salman Durrani, and Xiangyun Zhou. “Outage probability in arbitrarily-shaped finite wireless networks”. In: *IEEE Transactions on Communications* 62.2 (2014), pp. 699–712.

[34] Tim Hornyak. *This Japanese aircraft became a 5G base station*. 2024. URL: https://spectrum.ieee.org/high-altitude-platform-station-5g.

[35] Huawei. *SNR (Signal-to-Noise Ratio) Encyclopedia*. Accessed: 2025-01-06. 2025. URL: https://info.support.huawei.com/info-finder/encyclopedia/en/SNR.html.

[36] Yiming Huo, Xiaodai Dong, and Wei Xu. “5G cellular user equipment: From theory to practical hardware design”. In: *IEEE access* 5 (2017), pp. 13992–14010.

[37] Yiming Huo, Xiaodai Dong, et al. “Multi-Beam Multi-Stream Communications for 5G and Beyond Mobile User Equipment and UAV Proof of Concept Designs”. In: *arXiv preprint arXiv:1909.13040* (2019). URL: https://arxiv.org/abs/1909.13040.

[38] Prateek Jain, Purushottam Kar, et al. “Non-convex optimization for machine learning”. In: *Foundations and Trends® in Machine Learning* 10.3-4 (2017), pp. 142–363.

[39] Beakcheol Jang et al. "Q-Learning Algorithms: A Comprehensive Classification and Applications". In: *IEEE Access* PP (Sept. 2019), pp. 1–1. DOI: 10.1109/ACCESS.2019.2941229.

[40] Dilith Jay. *Deep Q-Networks (DQN) Explained*. Accessed: 2025-01-14. 2020. URL: https://dilithjay.com/blog/dqn.

[41] Scott Jordan and Pravin P Varaiya. "Throughput in multiple service, multiple resource communication networks". In: *IEEE Transactions on communications* 39.8 (1991), pp. 1216–1222.

[42] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. "Reinforcement learning: A survey". In: *Journal of artificial intelligence research* 4 (1996), pp. 237–285.

[43] Ibrahem Kandel and Mauro Castelli. "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset". In: *ICT Express* 6.4 (2020), pp. 312–315. ISSN: 2405-9595. DOI: https://doi.org/10.1016/j.icte.2020.04.010. URL: https://www.sciencedirect.com/science/article/pii/S2405959519303455.

[44] John Kang et al. "Machine Learning Approaches for Predicting Radiation Therapy Outcomes: A Clinician's Perspective". In: *International Journal of Radiation Oncology*Biology*Physics* 93.5 (2015), pp. 1127–1135. ISSN: 0360-3016. DOI: https://doi.org/10.1016/j.ijrobp.2015.07.2286. URL: https://www.sciencedirect.com/science/article/pii/S0360301615030783.

[45] Hilbert J Kappen. "Optimal control theory and the linear Bellman equation". In: (2011).

[46] Changkyu Kim, Russell Ford, and Sundeep Rangan. "Joint interference and user association optimization in cellular wireless networks". In: *2014 48th Asilomar Conference on Signals, Systems and Computers*. 2014, pp. 511–515. DOI: 10.1109/ACSSC.2014.7094497.

[47] Haklin Kimm et al. "Real time data communication using high altitude balloon based on cubesat payload". In: *Journal of Advances in Computer Networks* 3.3 (2015), pp. 186–190.

[48] Gaurav Kumar and Pradeep Kumar Bhatia. "Impact of agile methodology on software development process". In: *International Journal of Computer Technology and Electronics Engineering (IJCTEE)* 2.4 (2012), pp. 46–50.

[49] Kevin Lai and Mary Baker. "Measuring bandwidth". In: *IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*. Vol. 1. IEEE. 1999, pp. 235–245.

[50] Deepika Latka, Mona Aggarwal, and Swaran Ahuja. "Outage Probability Analysis and Altitude Optimization of a UAV-Enabled Triple-Hop Mixed RF-FSO-Based Wireless Communication System". In: *Photonics*. Vol. 11. 12. MDPI AG. 2024, p. 1145.

[51] Seunghyun Lee and Kaibin Huang. "Coverage and economy of cellular networks with many base stations". In: *IEEE Communications Letters* 16.7 (2012), pp. 1038–1040.

[52] Jian Li et al. “A User-Centric Handover Scheme for Ultra-Dense LEO Satellite Networks”. In: *IEEE Wireless Communications Letters* 9.11 (2020), pp. 1904–1908. DOI: 10.1109/LWC.2020.3007818.

[53] Yuxi Li. “Deep Reinforcement Learning: An Overview”. In: *arXiv preprint arXiv:1701.07274* (2017).

[54] Hui Liang et al. “Resource Allocation for Space-Air-Ground Integrated Networks: A Comprehensive Review”. In: *Journal of Communications and Information Networks* 9.1 (2024), pp. 1–23.

[55] Jiajia Liu et al. “Space-air-ground integrated network: A survey”. In: *IEEE Communications Surveys & Tutorials* 20.4 (2018), pp. 2714–2741.

[56] Yue Meng et al. “User association in heterogeneous networks: A social interaction approach”. In: *IEEE Transactions on Vehicular Technology* 65.12 (2016), pp. 9982–9993.

[57] Denisz Mihajlov. “Explanation and comparison of deep learning models and improvement approaches used in Reinforcement Learning.” In: (2022).

[58] Shaili Mishra and Anuja Arora. “Intelligent computational techniques for physical object properties discovery, detection, and prediction: A comprehensive survey”. In: *Computer Science Review* 51 (2024), p. 100609. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2023.100609. URL: https://www.sciencedirect.com/science/article/pii/S157401372300076X.

[59] Volodymyr Mnih et al. “Human-level control through deep reinforcement learning”. In: *nature* 518.7540 (2015), pp. 529–533.

[60] Ernest Mnkandla and B Dwolatzky. “A survey of agile methodologies”. In: *Transactions of the South African Institute of Electrical Engineers* 95.4 (2004), pp. 236–247.

[61] Jacob Murel and Eda Kavlakoglu. “What is regularization?” In: (2023). Accessed: 2025-01-13. URL: https://www.ibm.com/think/topics/regularization.

[62] Muddasar Naeem, Giuseppe De Pietro, and Antonio Coronato. “Application of reinforcement learning and deep learning in multiple-input and multiple-output (MIMO) systems”. In: *Sensors* 22.1 (2021), p. 309.

[63] Junichi Nakajima et al. “A Balloon-Based Wireless Relay System for Disaster Response”. In: *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*. 2015, pp. 1–6. DOI: 10.1109/VTCFall.2015.7390783.

[64] Zahra Makki Nayeri, Toktam Ghafarian, and Bahman Javadi. “Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey”. In: *Journal of Network and Computer Applications* 185 (2021), p. 103078. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2021.103078. URL: https://www.sciencedirect.com/science/article/pii/S1084804521000989.

[65] KK Ramachandran and KK Karthick. “Gantt chart: An important tool of management”. In: *Int. J. Innov. Technol. Explor. Eng* 8.7 (2019).

[66] Partha Pratim Ray. "A review on 6G for space-air-ground integrated network: Key enablers, open challenges, and future direction". In: *Journal of King Saud University-Computer and Information Sciences* 34.9 (2022), pp. 6949–6976.

[67] *Reinforcement Learning Environments - MATLAB & Simulink — mathworks.com.* `https://www.mathworks.com/help/reinforcement-learning/ug/reinforcement-learning-environments.html`. [Accessed 28-05-2024].

[68] Stephen M Rumble et al. "It's time for low latency". In: *13th Workshop on Hot Topics in Operating Systems (HotOS XIII)*. 2011.

[69] Adeeb Salh et al. "A Survey on Deep Learning for Ultra-Reliable and Low-Latency Communications Challenges on 6G Wireless Systems". In: *IEEE Access* 9 (2021), pp. 55098–55131. DOI: 10.1109/ACCESS.2021.3069707.

[70] Sanglap Sarkar, Radha Krishna Ganti, and Martin Haenggi. "Optimal base station density for power efficiency in cellular networks". In: *2014 IEEE International Conference on Communications (ICC)*. IEEE. 2014, pp. 4054–4059.

[71] Ararat Shaverdian, Jagadish Ghimire, and Catherine Rosenberg. "Simple and efficient network-aware user association rules for heterogeneous networks". In: *Computer Networks* 156 (2019), pp. 20–32.

[72] Ebrahim Hamid Sumiea et al. "Deep deterministic policy gradient algorithm: A systematic review". In: *Heliyon* (2024).

[73] Yaohua Sun et al. "Integrated satellite-terrestrial networks: Architectures, key techniques, and experimental progress". In: *IEEE Network* 36.6 (2022), pp. 191–198.

[74] TensorFlow. *Introduction to Reinforcement Learning with TensorFlow Agents*. Accessed: 2024-11-4. URL: `https://www.tensorflow.org/agents/tutorials/0_intro_rl`.

[75] Muhammad Uzair and Noreen Jamil. "Effects of hidden layers on the efficiency of neural networks". In: *2020 IEEE 23rd international multitopic conference (INMIC)*. IEEE. 2020, pp. 1–6.

[76] Peter A Valberg, T Emilie Van Deventer, and Michael H Repacholi. "Workgroup report: base stations and wireless networks—radiofrequency (RF) exposures and health consequences". In: *Environmental health perspectives* 115.3 (2007), pp. 416–424.

[77] Francesco Vatalaro et al. "Analysis of LEO, MEO, and GEO global mobile satellite systems in the presence of interference and fading". In: *IEEE Journal on selected areas in communications* 13.2 (1995), pp. 291–300.

[78] Steven Walczak and Narciso Cerpa. "Artificial Neural Networks". In: *Encyclopedia of Physical Science and Technology (Third Edition)*. Ed. by Robert A. Meyers. Third Edition. New York: Academic Press, 2003, pp. 631–645. ISBN: 978-0-12-227410-7. DOI: `https://doi.org/10.1016/B0-12-227410-5/00837-1`. URL: `https://www.sciencedirect.com/science/article/pii/B0122274105008371`.

[79] Mea Wang and Baochun Li. "How practical is network coding?" In: *200614th IEEE International Workshop on Quality of Service*. IEEE. 2006, pp. 274–278.

[80] Ziyu Wang et al. "Dueling network architectures for deep reinforcement learning". In: *International conference on machine learning*. PMLR. 2016, pp. 1995–2003.

[81] Marco A Wiering and Martijn Van Otterlo. "Reinforcement learning". In: *Adaptation, learning, and optimization* 12.3 (2012), p. 729.

[82] Florentin Woergoetter and Bernd Porr. "Reinforcement learning". In: *Scholarpedia* 3.3 (2008), p. 1448.

[83] Dong Xie and Xiangnan Zhong. "Semicentralized deep deterministic policy gradient in cooperative StarCraft games". In: *IEEE transactions on neural networks and learning systems* 33.4 (2020), pp. 1584–1593.

[84] Huaqing Xiong et al. "Deterministic policy gradient: Convergence analysis". In: *Uncertainty in Artificial Intelligence*. PMLR. 2022, pp. 2159–2169.

[85] Wenli Yan et al. "Joint User Association and Handover Strategy for Large Scale LEO Satellite Constellation Networks". In: *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*. IEEE. 2023, pp. 880–884.

[86] Taesang Yoo et al. "Throughput optimization using adaptive techniques". In: *IEEE Communication Letters* (2006), p. 18.

[87] Sri Yusnita et al. "Analysis on The Effect of Channel Bandwidth Occupying in LTE Frequency Band on Throughput". In: *International Journal Of Advanced Science Computing And Engineering* 5.1 (2023), pp. 8–14.

[88] Meng Zhang et al. "Top ten intelligent algorithms towards smart manufacturing". In: *Journal of Manufacturing Systems* 71 (2023), pp. 158–171. ISSN: 0278-6125. DOI: https://doi.org/10.1016/j.jmsy.2023.09.006. URL: https://www.sciencedirect.com/science/article/pii/S0278612523001887.

[89] Ming Zhao et al. "Orbital collaborative learning in 6G space-air-ground integrated networks". In: *Neurocomputing* 497 (2022), pp. 94–109.

[90] Nan Zhao et al. "Deep reinforcement learning for user association and resource allocation in heterogeneous cellular networks". In: *IEEE Transactions on Wireless Communications* 18.11 (2019), pp. 5141–5152.

[91] Zhouxiang Zhao et al. "Energy-Efficient Probabilistic Semantic Communication over Space-Air-Ground Integrated Networks". In: *arXiv preprint arXiv:2407.03776* (2024). DOI: 10.48550/arXiv.2407.03776.

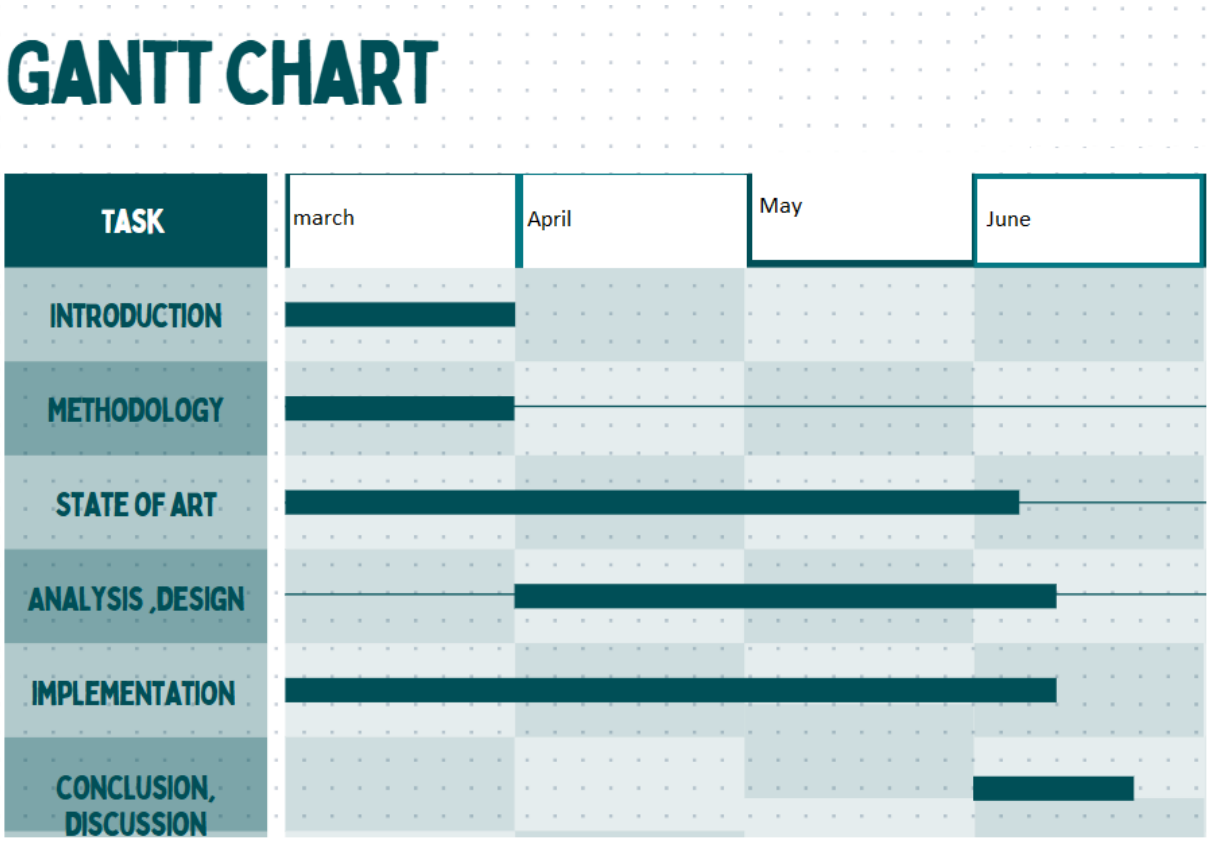
[92] Yueyan Zhi et al. "Security and privacy issues of UAV: A survey". In: *Mobile Networks and Applications* 25.1 (2020), pp. 95–101.

[93] Xiaolin Zhou. "Optimal Values Selection of Q-learning Parameters in Stochastic Mazes". In: *Journal of Physics: Conference Series* 2386 (2022), p. 012037. DOI: 10.1088/1742-6596/2386/1/012037. URL: https://iopscience.iop.org/article/10.1088/1742-6596/2386/1/012037.

Appendix A

Appendix

A.1 Appendix A: Gantt



The Gantt chart depicted in Appendix A shows the timeline of the project tasks, months, and the time it took to complete. It outlines the sequence of the activity with their corresponding starting and ending times.

A.2 Appendix B: x