SAPIENZA
UNIVERSITÀ DI ROMA

Data–Driven Control of Terrestrial and Satellite Communication Networks

Dipartimento di Ingegneria Informatica, Automatica e Gestionale (DIAG)
"Antonio Ruberti"

Dottorato di Ricerca in **Automatica**, Bioingegneria e Ricerca Operativa (XXXVI cycle)

Andrea Wrona
ID number 1699966

Advisor
Prof. Alessandro Di Giorgio

Academic Year 2022/2023

Data–Driven Control of Terrestrial and Satellite Communication Networks
PhD thesis. Sapienza University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: wrona@diag.uniroma1.it

Ai miei genitori e a mio nonno

Contents

Abstract

Artificial Intelligence and the pervasive presence of Big Data have vigorously become the technological protagonists of the new millennium.

In the interconnected world in which we live, millions of virtual interactions take place, thanks to the development of increasingly sophisticated information, electronic and communications technologies, which allow us to enjoy experiences that were unthinkable up until twenty years ago.

The availability of an enormous amount of information is also pushing the scientific community relating to automation and control science to move towards a data-driven paradigm, which is opposed to class methods based on the knowledge of the mathematical model of the process to be controlled.

This thesis delves into the realm of data-driven control methods in the domains of terrestrial and satellite communication networks, aiming to prove the capability of model–free techniques to optimize network performance, rethink the network selection paradigm, tune dynamically transmission power, and improve signal quality, reliability and availability, allowing for continuous and ubiquitous connectivity.

In order to reach these objectives, this essay presents empirical and simulated evidence demonstrating the effectiveness of data-driven control methods in improving the performance and reliability of both terrestrial and satellite networks. The findings have significant implications for the communication landscape, including (i) improved network performance and efficiency within terrestrial networks in relation with critical applications like autonomous driving and Mobile Augmented Reality, (ii) improved adaptability and dynamic decision-making capabilities, and (iii) signal degradation mitigation and uninterrupted connectivity even under challenging atmospheric conditions in satellite networks.

The thesis is organized in three Parts:

> **Part 1** discusses about the generalities of data–driven control methods relying on Artificial Intelligence and, in particular, Reinforcement Learning. The dissertation starts from the difference between methods based on knowledge of the model and those that rely solely on data coming from sensors, highlighting pros and cons of each one of the two paradigms. Then, the mathematical foundations of the Reinforcement Learning are provided, with the characterization of Markov Decision Processes (the single–agent domain) and Markov Games (the multi–agent scenario). Eventually, the discussion is shifted from discrete spaces to continuous states and actions, introducing the challenging concept of Deep Reinforcement Learning, which exploits a combination of neural networks to build, train, and test in real–time intelligent agents.

Part 2 focuses on the generalities of terrestrial network, with emphasis on the role of the new generations of cellular networks (5G and beyond) and their critical applications, including Virtual and Augmented Reality in the cultural heritage sector. This part of the thesis presents three different control strategies. The first one is a decision framework for the solution of the network selection and traffic steering problems in downlink–only mobile connections. The second one instead considers the uplink plane, proposing a continuous control of transmitting power and image resolution for Mobile Augmented Reality applications. The third and last one poses the attention on another critical application domain of terrestrial networks, i.e., self–driving vehicles. It is shown how it is possible to control vehicle platoons even under the assumption of a complete communication fault.

Part 3 spotlights the sphere of satellite communications, with a debate on the dualism between radio frequency and free space optics, showing how the latter constitute a disruptive technology for high–throughput and secure communication between ground stations and satellite assets. Later on, two Machine Learning–based control laws for site diversity implementation are exhibited, the first one using a single geostationary satellite, and the second one operating with a low Earth orbit satellite constellation.

Eventually, Chapter 11 will draw conclusions on the work carried out, its scientific impact and practical implications, also showing all possible limitations and blind spots. The path will therefore be traced on possible strategies by which these limits can be overcome in the future.

The author of this work hopes that future work and research in applied control science can draw innovative ideas and insights starting from the results exhibited in this doctoral thesis.1

List of Figures

List of Tables

List of Algorithms

List of Abbreviations and Acronyms

A3C Asynchronous Advantage Actor–Critic

AI Artificial Intelligence

AP Access Point

AR Augmented Reality

BS Base Station

CAV Connected Autonomous Vehicle

CbT Iterative Correlation-based Tuning

CDMA Code Division Multiple Access

CNN Convolutional Neural Network

CTCE Centralized Training and Centralized Execution

CTDE Centralized Training and Decentralized Execution

DDPG Deep Deterministic Policy Gradient

DL Deep Learning

DNN Deep Neural Network

DRL Deep Reinforcement Learning

DTDE Decentralized Training and Decentralized Execution

eMBB Enhanced Mobile Broad Band

ESA European Space Agency

FDMA Frequency Division Multiple Access

FSO Free Space Optics

GEO Geostationary Earth Orbit

GPS Global Positioning System

GS Ground Station

HydRON High Throughput Optical Network

IFT Iterative Feedback Tuning

ILC Iterative Learning Control

IoT Internet of Things

IRT Iterative Regression Tuning

KPI Key Performance Indicator

LEO Low Earth Orbit

LQG Linear Quadratic Gaussian

LQR Linear Quadratic Regulator

LSTM Long Short–Term Memory

LTE Long–Term Evolution

MAR Mobile Augmented Reality

MARL Multi Agent Reinforcement Learning

MD Mobile Device

MDP Markov Decision Process

MEC Mobile Edge Computing

MEO Medium Earth Orbit

ML Machine Learning

mMTC Massive Machine–Type Communication

MPC Model Predictive Control

MVR Mobile Virtual Reality

NAF Q–Learning with Normalized Advantage Functions

NASA National Aeronautics and Space Administration

NN Neural Network

OFDMA Orthogonal Frequency Division Multiple Access

OGS Optical Ground Station

PID Proportional–Integral–Derivative

PPO Proximal Policy Optimization

QoE Quality of Experience

QoS Quality of Service

RAT Radio Access Technology

RF Radio Frequency

RL Reinforcement Learning

RNN Recurrent Neural Network

SAC Soft Actor–Critic

SINR Signal–to–Noise Plus Interference Ratio

SL Supervised Learning

SNR Signal–to–Noise Ratio

SPSA Simultaneous Perturbation Stochastic Approximation

TD Temporal Difference

TD3 Twin Delayed Deep Deterministic Policy Gradient

TDMA Time Division Multiple Access

UE User Equipment

UL Unsupervised Learning

UMTS Universal Mobile Telecommunications System

URLLC Ultra Reliable Low Latency Communication

V2I Vehicle to Infrastructure

V2N Vehicle to Network

V2P Vehicle to Pedestrian

V2V Vehicle to Vehicle

V2X Vehicle to Everything

VADUS Virtual Access and Digitization of Unreachable Sites

VR Virtual Reality

VRFT Virtual Reference Feedback Tuning

Part I

Data–Driven Control

Chapter 1

Why Data–Driven Control?

Control comes from the Medieval Latin *contrarotulus*, which is a compound name, given by the union of *contra* (against, opposite to) and *rotulus* (diminutive of *rota*, the Latin word for *wheel*) [1]. This particular word was used to denote a practice carried out by scribes in medieval times, who checked accounts and invoices with a double register.

On the other side, the word *automatic* derives from the ancient Greek adjective αὐτόματος, which literally means *self–moving* and the noun τέχνη (art), thus denoting the specific property of systems capable of carrying out pre–established tasks without any human help [2].

The combination of these two words produce the well–known term *Automatic Control*, which denotes the multidisciplinary field within engineering and mathematics that constitutes the core business and the foundation of this doctoral thesis.

Automatic control is the field of study that aims at governing physical systems without the human manual intervention. Said systems encompass sensors, actuators, and controllers, all of which are designed with the objective of impacting the dynamic behavior of the system. Control systems function by continuously monitoring the output or condition of the system using feedback from sensors. Controllers usually compare this monitored state with a reference signal or setpoint, calculate the appropriate control action based on the observed error, and execute that action through actuators to adjust the system's behavior, trying to minimize the impact of disturbances or uncertainties.

Thanks to the versatility of this scientific field, control in engineering embraces a wide range of applications, from regulating the temperature of an industrial furnace, through governing the steering wheel of an autonomous vehicle, to modulating the transmission power in wireless telecommunications.

1.1 Model–Based Control

The process of designing a control system traditionally involves the preliminary creation of a mathematical model of the system that needs to be controlled. This is usually done via analyzing the law of physics underlying the specific system or process, and translating them into a mathematical language by means of differential equations. The latter shall provide a quantitative understanding of the system's dynamics, which, in a very general shape, can be expressed via a state–space representation:[1]

$$\begin{aligned} \dot{x} &= f(t, x, u, w) \\ y &= h(t, x, u, w, n), \end{aligned} \tag{1.1}$$

where t denotes the time, $x \in \mathbb{R}^n$ is the state space, $u \in \mathbb{R}^m$ is the control signal, $w \in \mathbb{R}^o$ represents disturbances, $y \in \mathbb{R}^p$ is the system output[2], represented as a nonlinear static function, and $n \in \mathbb{R}^p$ is the measurement noise.

Typically, said systems are controlled on the basis of the *feedback* principle, according to which the controller computes its outputs based on the error between a desired reference and the true output. A typical feedback control scheme is depicted in Fig. 1.1. Depending on the system complexity and modeling choices, the nonlinear state–space representation may be reduced to a linear one:

$$\begin{aligned} \dot{x} &= Ax + B_1 u + B_2 w \\ y &= Cx + D_1 u + D_2 w, \end{aligned} \tag{1.2}$$

where $A \in \mathbb{R}^{n \times n}$ is the system dynamical matrix, $B_1 \in \mathbb{R}^{n \times m}$ is the input–to–state matrix, $B_2 \in \mathbb{R}^{n \times o}$ is the disturbance–to–state matrix, $C \in \mathbb{R}^{p \times n}$ is the state–to–output matrix, $D_1 \in \mathbb{R}^{p \times m}$ is the input–to–output matrix and, eventually, $D_2 \in \mathbb{R}^{p \times o}$ is the disturbance–to–output matrix.

Since the famous work by James Clerk Maxwell [4], which is considered to be the first automatic control system based on feedback, a myriad of control techniques based on the model of the system to be controlled have been proposed in the literature. Some of these techniques represent the standard for the control of industrial automation systems and are therefore used successfully every day.

---

[1] Here we are assuming that the system dynamics varies just in time, and not in space. In the latter situation, the system's states or parameters are called *distributed* and its mathematical model shall rely on partial differential equations [3]. A typical example of a system whose dynamics changes also with respect to the space is the movement of a drum or membrane when struck or urged by a force perpendicular to its surface.

[2] In the vast majority of physical system, the output does not depend on the control u.

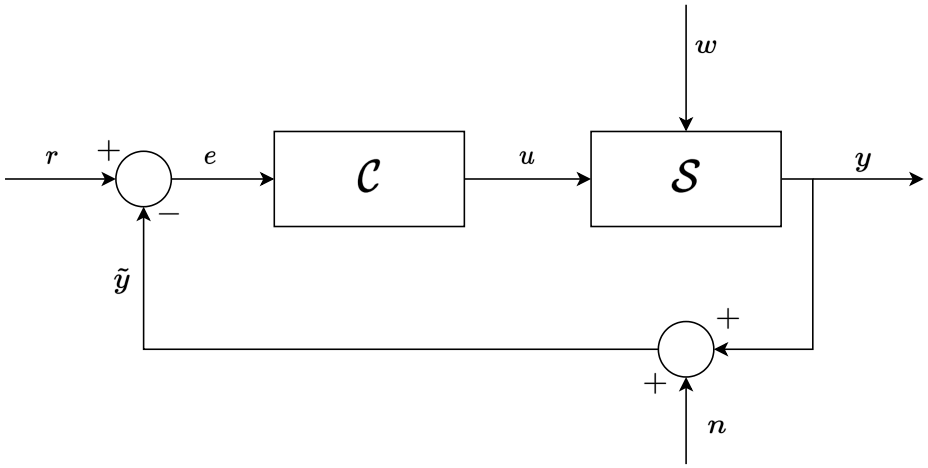


Figure 1.1. Feedback Control Scheme. The block $\mathcal{C}$ denotes the controller, whereas $\mathcal{S}$ indicates the system to be controlled. The variable $e = r - y$ defines the error between the reference signal r and the measured outputs $\tilde{y} = y + n$. All the other quantities are the same as in (1.1).

The most famous model–based control techniques can be summarized as follows:

- Proportional–Integral–Derivative (PID) control is one of the most commonly used techniques in the linear domain [5]. The control inputs is adjusted based on the error (the difference between the desired and actual system output). PID controllers consist of three gains: proportional (P), integral (I), and derivative (D), which shall be tuned to achieve certain desired performance in terms of rise time or overshoot.
- Frequency–based control. It can be used only in the linear domain and assumes the system is represented as a transfer function directly linking the input u and the output y of the system described in (1.2), thus having

$$P_1(s) = C(sI - A)^{-1}B_1 + D_1, \tag{1.3}$$

  where $s \in \mathbb{C}$ is a complex variable. Methods in the frequency domain include the root locus design and the synthesis of corrective networks to modify phase margin and crossing pulsation [6].
- Linear Quadratic Regulator (LQR) is used as well in the linear case to realize control actions minimizing a certain cost function while satisfying given constraints. It is useful in all the situations in which the goal is to save energy or time while having saturation or state constraints. A similar technique, the Linear Quadratic Gaussian (LQG), incorporates both state feedback control and state estimation using Kalman filters [7].

- Model Predictive Control (MPC), which extends concepts typical of optimal control by adding the notion of control horizon and prediction horizon. The former determines the sequence of control inputs to be applied, the latter denotes the period over which the system behavior is forecasted. As in optimal control, MPC can handle state and input constraints and it can be applied as well to nonlinear systems.
- Feedback Linearization. It belongs to the nonlinear control techniques: it deals with manipulating the input so that the system looks linear, at least in the input–output representation. The internal dynamics which cannot be linearized takes the name of zero dynamics [8].
- Backstepping: it is extremely useful for highly nonlinear systems, and it constructs a chain of control laws, each one stabilizing a subset of the entire system dynamics [9].
- Robust control involves a long series of control methods which suppose some (or all) parameters within the mathematical model are unknown or uncertain [10]. Hence, the aim is to stabilize the system around its equilibrium whatever the value of said parameters. One of the most famous technique in the nonlinear domain relies on sliding model control, in which one enforces the system to follow a specific sliding surface insensitive to parameter variations and external disturbances [11].
- Adaptive control. The governor in this case is designed to handle systems with varying dynamics, with parameters adaption based on the observed system behavior.

Although control techniques based on mathematical models are the standard in industry and are continually evolving in academia, they have important limitations.

One of the most relevant difficulties in model–based control is the capability of obtaining accurate mathematical models, capable of capturing the real dynamic behavior of a certain process. This can be easily done with simple mechanical or electrical systems, but real–world systems are often complex and made of interconnection of a myriad of subsystems, which may interact in a stochastic fashion.

It is very likely that parameters related, e.g., to viscous friction in the aerodynamics of an aircraft, rotational inertia of rigid bodies, or biological systems cannot be modeled via a specific stationary constant, thus leading to time–varying or uncertain state–space models, which are really challenging to control. Moreover, the intrinsic physical complexity of a system or interconnection of systems may render their modeling a time consuming task, since models' parameters should be tuned properly to reproduce the behavior of the real system. When someone succeeds in

creating such a complex mathematical model, usually it is embedded into a computer simulation program, thus enabling the so–called Digital Twin paradigm [12], in which one can test control actions in a simulated environment, which is crucial to preserve the security of fragile expensive physical systems like mobile robots or drones.

However, no matter how precise a certain mathematical model is, there may be real–world situations in which the boundary conditions change or the environment in which the system operates is modified either through human intervention or nature–related phenomena. In such cases, control actions which are effective and robust on digital twins may become ineffective when applied to the real system, thus leading to instability of state trajectories.

Eventually, the last critical aspect relies on real–time implementation of control laws based on a mathematical model. Whatever the technique used, feedback control signal usually depends statically or dynamically on the system outputs or upon the state of an asymptotic observer. When the system nonlinearities are too complex, the control law may become too complex, thus leading to latency in sensor measurement and actuator actions. High latencies or delay effectively violate the real-time operation constraint of the system. In this respect, a typical control approach which cannot be applied in real–time for a class of complex nonlinear system is the one relying on the minimization of a cost functional (MPC and optimal control). This because discrete micro–controllers embedded on real–world systems do not have a sufficient computational power to compute within the time limits the optimal control signal based on the state measurements coming from sensors [13].

These challenges underscore the importance of considering alternative control methods, such as data-driven control, in all the scenarios where developing control laws in the traditional model-based way is infeasible or impractical. Data-driven methods can offer more flexibility and adaptability in cases where accurate models are challenging to obtain or, when developed, are too complex to be controlled by model–based controllers.

1.2 From the Model–Based to the Data–Driven Framework

Data-driven control methods, as the name suggests, represent a class of control approaches relying on the use of data to design, optimize, and adapt control strategies to a variety of processes and dynamical systems [14]. These methods have gained significant attention and relevance in the last decade with the advent of Big Data, advanced sensors, and Machine Learning (ML) techniques.

Data-driven control methods offer several advantages over traditional model-based control approaches, as they can be more flexible, adaptable, and suitable for complex systems where accurate models are challenging to develop.

In general, all the techniques developed in literature relying on data–driven methods have as a prerequisite the availability of real–time information about the observable states of a certain physical systems, which can be directly measured through sensing devices. This feature is exactly the same as that found in closed loop control systems in the model-based domain: however, the use made of the signals coming from the sensors is clearly different.

Indeed, data–driven control methods do not rely on explicit mathematical models, and this is why they are also called model–free [15].

It is important to highlight that this claim does not imply that data–driven techniques cannot be implemented when a mathematical model of the system is present. The term *model–free*, indeed, refers to the working principle of the controller, which outputs its control signals without taking into account the model's equation, but relying on intelligent learning–driven algorithms. However, the control actions can be either applied to a real–world process through actuators or they may be first tested on a simulated environment or a digital twin. The former situation implies the complete absence of a mathematical model, since the intelligent agent directly interacts with a real system, whereas the latter scenario envisages the presence of a model which simulates the system behavior, even if such model is not accessible. Hence, in a typical setting, data–driven methods are first tested on a digital twin or simulator, and then deployed in the real world, especially in the case of systems where random actions can lead to serious instability (just think about the automatic control of insulin infusion in type 1 diabetic patients [16]).

Another key feature of model–free controllers is their adaptability to dynamic changes over time, since they adapt their outputs by learning from observation data coming from the process. This property guarantees as well robustness against uncertain or stochastic parameters and unknown disturbances acting at any level in the open–loop control chain.

Eventually, some data–driven control methods allow real-time optimal decision–making, since they are previously pre–trained offline before the deployment on the real–world [17]. Moreover, data–driven controllers can continuously learn and update their control law while performing further real–time training on the process, thus capturing external events or accidents that might be not present during the training phase. This fundamental property overcomes the limitations imposed by model–based controllers, whose control laws are valid only when the modeling assumptions hold.

From the dissertation just concluded, it is clear that the main limitation of data-driven methods is precisely their availability, reliability and variability. When the sensing devices measure the system outputs with a certain non–negligible noise level, data–driven control methods may learn control policies based on noisy data not corresponding to the true behavior of the system. Moreover, another not–so–evident drawback is the one of *overfitting*, namely the situation in which the control strategy becomes too specific to the training data and may not generalize well to new situations [18].

In the last years data-driven control techniques have been successfully applied in various domains, including industrial automation [19], mobile robotics [20], satellite communications [21], smart grids [22], and healthcare [23, 24].

Over the years the scientific community has developed a variety of data-driven control methods which exploits completely different mathematical reasonings and theory in order to design, learn and optimize control strategies. The choice of a certain method depends on the specific application, the quality and quantity of available data, and the complexity of the control problem.

Various perspectives are employed in the literature to categorize data–driven model–free control techniques. For instance, one viewpoint, as outlined in [25], centers around the control system structure. The first category posits that the optimal controller is embedded in the controller structure with one or more unknown parameters, which is derived from experimental knowledge about the process structure. This approach transforms controller design into a direct identification problem for the controller parameters. The second category encompasses controllers designed based on various function approximations or equivalent process descriptions, such as neural networks, fuzzy models, or Taylor approximation. In this case, controller parameters are fine-tuned by minimizing a specified performance criterion using input-output data, including offline and online data.

Another subdivision, the one adopted in this thesis, categorizes data-driven algorithms based on how the controller is synthesized: in one go, or iteratively.

In what follows we recall some of the most famous and used techniques in the data–driven framework.

- Iterative Feedback Tuning (IFT), detailed in [26–28], is a well-recognized iterative data-driven technique that refines controller parameters iteratively by following the gradient direction of an objective function. IFT is applicable when there is an initially parameterized controller with a known finite objective function value.
- Simultaneous Perturbation Stochastic Approximation (SPSA), described in [29, 30], employs gradient-based stochastic approximation algorithms that rely on

estimated gradients of the objective function. SPSA is beneficial for its ability to reduce implementation costs by requiring only two objective function evaluations per iteration.

- Iterative Correlation-based Tuning (CbT) [31] operates within the model reference control framework and focuses on the correlation between the reference input and tracking error. A decorrelation procedure is applied to make the tracking error converge to zero.
- Noniterative CbT is similar to the method described above, but uses a correlation approach to deal with measurement noise and has been demonstrated to outperform other data-driven techniques statistically [32].
- Fuzzy Logic Control. It is a rule-based control approach that can be data-driven. Fuzzy controllers use linguistic rules and fuzzy sets to handle imprecise information and adapt to changing conditions [33].
- Iterative Regression Tuning (IRT) [34] minimizes an objective function dependent on controller tuning parameters and aggregates performance specifications. IRT employs a gradient-based search, either using simulations of the control system behavior or local linear models derived from finite difference approximations and real-world experiments.
- Iterative Learning Control (ILC) states that performance of optimal controller can be improved by using experience gained from previous experiments. ILC can be formulated as iterative parametric optimization, making it applicable to reference input tuning in two-degree-of-freedom control systems [35].
- Unfalsified Control is a data-driven MFC approach that, based on measured input-output data, falsifies controllers that fail to satisfy the performance specifications. The only one which does not falsify the data is implemented [36].
- Virtual Reference Feedback Tuning (VRFT) is a non-iterative technique that minimizes the difference between control system outputs and a reference model [37, 38]. VRFT searches for the global minimum of the objective function optimum, being reduced to an identification problem as far as the controller and not the process is concerned. In the case of restricted complexity controller design, the achieved controller is a good approximation of the restricted complexity global optimal controller. VRFT is a one-shot algorithm, i.e., it can be applied using a single set of input data generated from the process, with no need for additional specific experiments nor iterations.
- Extremum Seeking Control uses a probing signal and demodulation to recover the gradient of the objective function. The stability of this adaptive control

system structure is proven with the averaging method in terms of showing that the system converges to a small neighborhood of the extremum of the objective function [39, 40].

- Machine Learning Algorithms: ML is at the core of many iterative data-driven control systems. Algorithms such as neural networks, support vector machines, decision trees, random forests, and gradient boosting can be used to learn control policies directly from data, adapting to changing system conditions and disturbances [41]. Reinforcement Learning (RL), a subset of ML, is an iterative control method where an intelligent agent learns to interact with a dynamic environment through trial and error. RL agents learn to modify their parameters and control the process based on received rewards.

Among all the data–driven techniques presented above, this work focuses its attention on the last one, namely ML–based control. The next section will present the basics of ML, characterizing the first two out of its three areas.

1.3 Machine Learning

Machine Learning is a subfield of Artificial Intelligence (AI) that focuses on the development of algorithms and statistical models that enable computer systems to learn and improve their performance on a specific task through experience and data, rather than being explicitly programmed [41]. In ML, computers are trained to recognize patterns, make predictions, or make decisions based on data, often with the goal of improving their performance over time. At its core, ML represents a data-driven approach to problem-solving. Rather than relying on explicit programming, ML algorithms learn from data and adapt to make predictions, decisions, or recommendations [41]. The key processes within ML include data collection, data preprocessing, model selection, training, and evaluation. Through this cycle, models are created and refined to extract patterns, make predictions, and provide insights.

The field of ML has witnessed remarkable advancements in the past two decades, transitioning from a scientific curiosity in research labs to a practical technology extensively utilized in various commercial applications. In the realm of AI, ML has become the preferred approach for developing functional software for tasks like computer vision, speech recognition, natural language processing, robot control, and more. Many AI developers now acknowledge that, for numerous applications, it is more effective to train a system by providing examples of desired input-output behavior rather than manually programming it to anticipate responses for all possible inputs.

The impact of ML, however, extends beyond AI, permeating various sectors and industries that deal with data-intensive problems, such as consumer services, complex systems fault diagnosis, and logistics chain management [42]. Moreover, it has had a broad influence on empirical sciences, including fields like biology, cosmology, and social science, where ML methods have been employed to analyze high-throughput experimental data in innovative ways.

The ML machinery exploits wide dataset and data repository which serve as the input of any ML–based technique. These data are used to build function approximation in order to assess a relation between input and output pairs. Said function can be explicitly represented in a parameterized form, or it can be implicit and obtained through a search process or optimization procedure. Regardless of the representation, the key objective is to find parameter values that optimize a given performance metric or Key Performance Indicator (KPI).

A diverse array of ML algorithms has been developed to address various data and problem types. These algorithms search through a large space of potential programs, guided by training experiences, to find the program that optimizes the performance metric. They differ in the way they represent candidate programs (e.g., decision trees, mathematical functions, programming languages) and how they search through this program space (e.g., optimization algorithms, evolutionary search methods).

A central scientific and practical objective in the ML field is to theoretically characterize the capabilities and inherent challenges of specific learning algorithms and learning problems. This involves understanding how accurately an algorithm can learn from a specific type and volume of training data, its resilience to modeling assumptions or training data errors, and whether a given learning problem can be feasibly solved. These characterizations often employ frameworks from statistical decision theory and computational complexity theory.

For this reason, ML as a field resides at the intersection of computer science, statistics, and various disciplines concerned with automatic improvement, inference, and decision-making under uncertainty, drawing insights from fields like psychology, evolutionary biology, adaptive control theory, education, neuroscience, organizational behavior, and economics. While there has been increased interdisciplinary collaboration in recent years, there is still much untapped potential for synergies and diverse formalisms and experimental methods from these disciplines in the study of systems that improve through experience.

Traditionally, ML is commonly categorized into three major types: Supervised Learning (SL), Unsupervised Learning (UL), and Reinforcement Learning (RL).

1.3.1 Supervised Learning

In the SL domain, algorithms learn from labeled data to make predictions. This type of ML is widely used for two types of task:

1. Classification, in which the goal is categorize records or images among a multitude of discrete classes or categories. A typical example is the classification of incoming emails, deciding whether they are *spam* or *not spam*.
2. Regression, whose aim is to creates functions capable of predicting continuous variables, like house prices or market trends.

In the SL setting, training data is represented as a set of (x, y) pairs, and the objective is to generate a prediction

$$y^* = \hat{f}(x^*), \tag{1.4}$$

as a response to a given query x^*. The input data x can take the form of traditional vectors/tensors or more complex entities such as documents, images, DNA sequences, or graphs. Similarly, a wide range of output types y have been investigated. While a significant focus has been on binary classification problems, where y takes one of two values (e.g., *healthy* or *ill*), research has also delved into multiclass classification (where y can take on one of K labels), multilabel classification (where y is assigned several of the K labels simultaneously), ranking problems (where y establishes a partial order within a set), general structured prediction problems (where y is a combinatorial object like a graph, with components that must satisfy a set of constraints), and hybrid classification, in which y is given by a combination of discrete and real-valued components.

One of the key issues in SL pertains to features of the input vector x. In many regression or classification problems, training data are characterized by heterogeneous meaning and structures. As an example, a training set may contain data about longitude, latitude, humidity and atmospheric pressures, thus leading to features having completely different ranges and magnitudes. The scale of individual features can influence a lot the Neural Network (NN) model's performance, since features with larger magnitudes might dominate those with smaller magnitudes during the learning process. This can lead to the model being biased toward features with larger scales, potentially causing the model to perform poorly. Moreover, gradient-based optimization algorithms, such as gradient descent, converge faster when the input features are within a similar range [41]. Features with disparate scales can lead to slow convergence, making it difficult for the algorithm to find the optimal solution in a reasonable time.

The straightforward solution for this issue is the introduction of normalization and standardization techniques which transform the input features of the training and test dataset to ensure that they have a consistent and standardized range.

The most important normalization techniques are:

1. Min–Max Scaling, in which the original data x is changed in the following way

$$\tilde{x} = \frac{x - x_{\min}}{x_{\text{MAX}} - x_{\min}}, \tag{1.5}$$

   so that the normalized feature $\tilde{x} \in [0, 1]$.

2. Standardization, which transforms features to have a zero mean and a standard deviation equal to one:

$$\tilde{x} = \frac{x - \mu}{\sigma}, \tag{1.6}$$

   where μ and σ are the mean and the standard deviation of the original feature, respectively.

The problem of feature normalization will be further analyzed in the following chapters, when dealing with heterogeneity of the observable state space of a dynamical system.

Regarding the mapping $\hat{f}(\cdot)$ that needs to be built, multiple forms have been proposed in literature, including decision trees, decision forests, logistic regression, support vector machines, neural networks, kernel machines, and Bayesian classifiers [43]. Regardless the methodology, all the procedures used for learning $\hat{f}(\cdot)$ from data draw inspiration from optimization theory and numerical analysis, using arguments related to the minimization of a function via gradient descent [41].

A particularly impactful recent advancement in SL pertains to Deep Neural Network (DNN), which consist of multilayer networks composed of threshold units[3], each performing a simple parameterized operation of its inputs [44]. The structure of a typical DNN is depicted in Fig. 1.2.

The subdomain of SL pertaining to DNN is called Deep Learning (DL). DL systems employ gradient-based optimization techniques to adjust parameters throughout such multi-layered networks based on errors defined at their output. Leveraging modern parallel computing hardware, such as graphics processing units initially developed for gaming, it has become feasible to construct DL systems containing billions of parameters, which can be trained on extensive collections of images, videos, records and speech samples accessible on the internet. These large-scale DL systems

[3]Said units are called neurons because they are a simplification of a human neural network, a concept from biology.

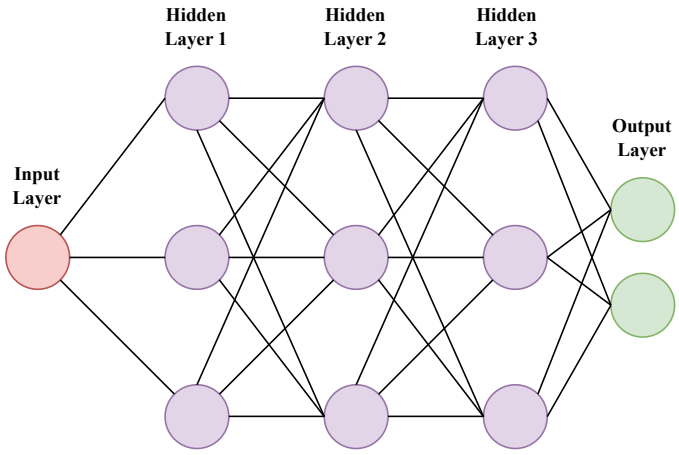


Figure 1.2. A DNN architecture with a single–neuron input layer, three triple–neuron hidden layers and a double–neuron output layer.

have made significant strides in recent years, particularly in the domains of computer vision [45,46], speech recognition [47], and healthcare [23] yielding substantial improvements in performance compared to earlier approaches.

1.3.2 Unsupervised Learning

While a significant portion of the practical achievements in the field of ML has been derived from SL methods, which are instrumental in discovering meaningful representations, efforts have also been directed toward developing DL algorithms capable of unveiling valuable input representations without relying on labeled training data [41]. This broader challenge is known as Unsupervised Learning (UL), representing the second paradigm in the realm of ML research.

UL, in a broad sense, revolves around the analysis of data that lacks explicit labels while making certain assumptions about the underlying structural properties of the data, aiming to find patterns, relationships, or structures within the data. UL involves two main problems:

1. Dimensionality reduction, in which one may assume that data points lie on a low-dimensional manifold and endeavor to explicitly identify and characterize that manifold through data analysis. Methods for dimension reduction, such as principal components analysis, manifold learning, factor analysis, random projections, and autoencoders [48, 49], embody different specific assumptions about the underlying manifold, like whether it is a linear subspace, a smooth nonlinear manifold, or a collection of submanifolds.

2. Clustering, dealing with partitioning observed data and establishing rules for predicting future data, all in the absence of explicit labels that indicate the

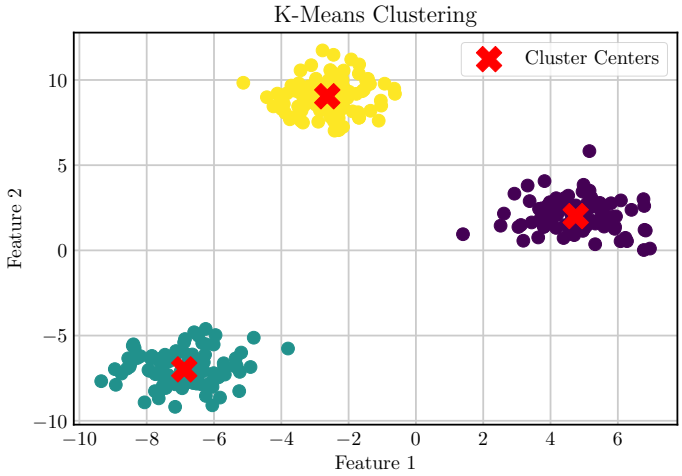


Figure 1.3. Automatic categorization of data points in a two–feature space using three clusters.

desired partition. Several clustering techniques has been developed, each relying on specific assumptions regarding the nature of the *cluster*. The most famous ones are K–Means and Spectral Clustering [50, 51]. In Fig. 1.3 it is possible to see the result of a K–Means algorithm.

Both clustering and dimension reduction place considerable emphasis on the challenge of computational complexity, especially since their aim is to harness the vast datasets available when one forgoes the use of supervised labels.

1.3.3 Reinforcement Learning

Reinforcement Learning is all about learning by interacting with an environment. It constitutes a fundamental technique in autonomous systems control and, in general, AI. In this approach, the information available in the training data falls between that of SL and UL. Instead of having training examples that explicitly specify the correct output for a given input, RL assumes that the training data (or the system's output) provides indications as to whether an action is correct or not, thanks to a feedback reward mechanism. Typical RL problems involve general control-theoretic context, in which the learning task revolves around acquiring a control strategy (referred to as a *policy*) for an agent operating in an unknown dynamic environment. This learned strategy is designed to select actions for any given state, with the objective of maximizing the expected cumulative reward over time [52].

The control methods enlightened in this essay rely entirely on data–driven techniques belonging to the field of RL. In the next chapter the mathematical foundation of RL control will be shown off.

Chapter 2

Markov Decision Processes and Reinforcement Learning

BRIDGES in research between control theory, operations research and computer science have strengthened over the years, with formal mathematical formulations serving as points of intersection with RL. Algorithms belonging to the latter domain often draw upon concepts well-established in the control theory literature, including policy iteration, value iteration, rollouts, and variance reduction [52]. RL methods constitute the most important and successful examples of data–driven control techniques, since they do not rely on any model of the process[1], learning control actions through a learning procedures.

2.1 Mathematical Framework of Markov Decision Processes

The mathematical framework used to model decision–making problems under uncertainty or stochasticity in the RL context relies on Markov Decision Process (MDP). It provides a structured way to model situations where an agent interacts with an environment over a sequence of discrete time steps, making decisions to maximize expected cumulative rewards. In the AI literature, usually the terms *environment* and *agent* substitute the more control–theory related nouns *process* and *controller*, respectively.

[1]There also RL methods exploiting the model of the process, but this work does not consider them.

Formally, a MDP can be defined as a tuple

$$\text{MDP} =< \mathcal{S}, \mathcal{A}, P(\cdot), R(\cdot) >, \tag{2.1}$$

where:

- $\mathcal{S}$ is the finite set of all possible states that the environment can be in. Each state represents a specific observable configuration.
- $\mathcal{A}$ is the finite set of all possible actions an agent can take over the environment.
- $P(s'|s,a) = \Pr(s_{t+1} = s'|s_t = s, a_t = a)$ defines the probability of transitioning[2] from state s to state s' when the agent takes the action a. The function $P(\cdot)$ is conditionally independent of all previous states $s_{t-1}, s_{t-2}, \ldots$ and actions $a_{t-1}, a_{t_2}, \ldots$, thus satisfying the Markov property[3].
- $R(s', s, a)$ is the immediate reward the agent gets from the environment after passing from s to s' due to action a.

MDP are an extension of Markov chains, which are objects representing a sequence of possible events whose probability depends only on the state reached in the previous event. Differently from Markov chains, MDP add actions (allowing choice) and rewards (giving motivation). It follows that, conversely, if only one action exists for each state and all rewards are the same, a MDP collapses to a Markov chain.

The agent behavior in the MDP framework is called *policy*, and can be defined formally as

$$\pi(\cdot) : \mathcal{S} \to \mathcal{A}, \tag{2.2}$$

i.e. a function whose domain lies within the state space $\mathcal{S}$ and whose image coincides with the action set $\mathcal{A}$.

The goal of an agent in a MDP is to find an optimal policy π^* that maximizes the expected cumulative reward, also known as *return*. The latter is defined as the sum of the discounted reward over time:

$$G = R_0 + \gamma R_1 + \gamma^2 R_2 + \cdots = \sum_{t=0}^{\infty} \gamma^t R_t, \tag{2.3}$$

where $\gamma \in [0, 1)$ is the discount factor, representing how much the agent values future rewards compared to immediate rewards. A higher discount factor ($\gamma \approx 1$)

[2]Note that the concept of transition probability function mimics the generic concept of nonlinear dynamics in traditional control systems. $P(\cdot)$ expresses how the environment evolves over time, just like the vector field $f(\cdot)$ in (1.1).

[3]In mathematics, the term *Markov property* refers to the memoryless property of a stochastic dynamical process, i.e. its future evolution is independent of its history [53].

gives more weight to future rewards, whereas $\gamma \approx 0$ is used in problems in which the the immediate system's behavior is more important than the future one.

Hence, from (2.2) and (2.3), the optimal policy can be computed as

$$\pi^* = \text{argmax}_\pi V_\pi(s) = \mathbb{E}\Big[G|s_0 = s, \pi(\cdot)\Big], \tag{2.4}$$

where $V_\pi(s)$ is called state–value function, representing *how good* is for the system to be in a certain state. Defining $V^*(s)$ as the maximum possible value of $V_\pi(s)$, it holds

$$V^*(s) = \max_\pi V^\pi(s). \tag{2.5}$$

Markov Decision Processes provide a solid theoretical foundation for modeling and solving sequential decision-making problems in various engineering domains, including robotics, telecommunications, resource allocation, and finance.

Solving MDP with finite state and action spaces can be accomplished using various techniques, such as dynamic programming. The latter methods can be specifically designed for MDPs with finite state and action spaces, where transition probabilities and reward functions are explicitly provided. This implies that dynamic programming is a framework used to solve MDP problems when an explicit mathematical model of the transition probability $P(\cdot)$ is available.

Standard algorithms belonging to this class requires the storage of two arrays indexed by state: one for the value $V(s)$ and the other for the policy $\pi(s)$. In general, at the end of the procedure, the policy shall contain the optimal solution π^*, while $V_{\pi^*}(s)$ will hold the discounted sum of the expected rewards to be obtained when following policy π^* from state s_0.

There are two main algorithms belonging to the field of dynamic programming:

1. Value iteration [54], in which $\pi(s)$ is not used and embedded iteratively into the computation of the state–value function $V_\pi(s)$, which proceeds as follows:

$$V_{i+1}(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,s',a) + \gamma V_i(s)), \tag{2.6}$$

   with $V_0(s)$ being an initial random guess.

2. Policy iteration [55], in which both $V_\pi(s)$ and $\pi(s)$ are explicitly computed step after step until a stable policy is reached. The first step is then

$$V_\pi(s) = \sum_{s' \in S} P_{\pi(s)}(s,s') \left[R(s,\pi(s),s') + \gamma V_\pi(s')\right], \quad \forall s \in S \tag{2.7}$$

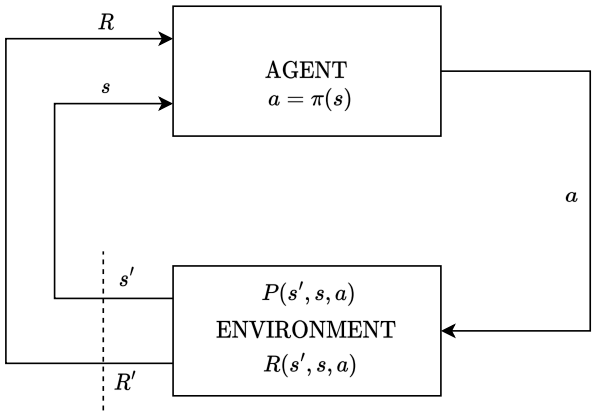


Figure 2.1. The feedback–based scheme of RL.

and the second is

$$\pi_{\text{new}}(s) = \arg\max_{a \in A(s)} \sum_{s' \in S} P(s'|s,a) \left[R(s,a,s') + \gamma V^{\pi}(s') \right]. \tag{2.8}$$

The assumption of perfect knowledge of the environment dynamics constitutes an important limitation, since these methodologies cannot be applied in a data–driven control framework where models are not available.

2.2 Reinforcement Learning

When the transition probability $P(\cdot)$ is not known, it is possible to rely on RL techniques, in which the agent learns the optimal policy through experience. The working principle of any RL algorithm is shown in Fig. 2.1: it is possible to notice that the scheme preserves the *feedback* mechanism of classical control theory, in this case adding a double feedback due to the presence of the reward signal.

All MDPs handled through RL require the estimation of the value function $V_{\pi}(s)$. This approach may provide useful information, but there is a lack of information related to the transition from one state to the other, which depends on the specific action one takes. Hence, the state–value function concept can be extended including the explicit dependency on the action, thus having the so–called action–value function $Q_{\pi}(\cdot)$ for a given policy $\pi(s)$ [52]:

$$Q_{\pi}(s,a) = \mathbb{E}_{\pi}[G_k | s_k = s, a_k = a], \forall s \in \mathcal{S}, \forall a \in \mathcal{A}. \tag{2.9}$$

Action–value functions satisfy recursive relationships through the Bellman Equation, which expresses a link between the action-value function of a state with the action-value function of the next state

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}_\pi[G_k|s_k = s, a_k = \pi(s_k)] \\ &= \mathbb{E}_\pi[R_{k+1} + \gamma G_{k+1}|s_k = s, a_k = a] \\ &= \sum_{s'} P(s'|s,a)(R + \gamma \sum_{a'} \pi(a'|s') Q_\pi(s',a')), \end{aligned} \tag{2.10}$$

where (s', a') is the next state-action couple with respect to (s, a). Hence, solving a MDP through RL means finding the optimal action-value function

$$Q^*(s,a) = \max_\pi Q_\pi(s,a), \tag{2.11}$$

for which it holds the Bellman principle of optimality [52]:

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)(R + \gamma \max_{a'} Q^*(s',a')). \tag{2.12}$$

2.2.1 Monte Carlo Methods

Monte Carlo Methods take the name from the numerical procedures developed by scientists of the Manhattan Project to solve numerically integrals which could not be solved analytically [56]. In these methods, the learning process is done by observing the current state, taking an action, collecting the reward, and repeating it again. This process is called *experience sampling*. The only accessible knowledge, i.e. the reward and the states, helps to estimate the state-value function.

The Monte Carlo techniques can be divided into three main parts:

- Monte Carlo Prediction, used to estimate the state value function of a given policy π, making use of a set of episodes. The latter is a crucial concept in the RL framework, denoting a finite temporal sequence of states, actions, reward and new states, thus characterizing the history of the interaction between the controller and the process. Each episode is made of a set of discrete time steps within a user–defined control horizon. In Monte Carlo Prediction, an empirical mean return, instead of the expected return, is used. This calculation can be done via two different algorithms:

  1. Every-Visit Monte Carlo, where the return is calculated for each state s whenever it is visited in each episode. Then, the mean of the state value $V_\pi(s)$ is calculated by dividing the sum of all the returns by the number of visits.
  2. First-Visit Monte Carlo, in which the return of the state s is considered only when it is visited for the first time within an episode.

- Monte Carlo Exploring Starts trains a policy using only the returns. The idea behind this method is similar to the one of Policy Iteration in the dynamic programming framework; instead of using the state–value function for the improvement, the action–value $Q_\pi(s, a)$ is exploited, and it evaluates the policy at each episode of interaction with the environment. This method has as main drawback the necessity of running for many episodes to converge [52].
- Monte Carlo Control introduces the fundamental concept of stochastic policy, thus selecting either random actions out of the $|\mathcal{A}|$ available with probability ε or the best action according to the state–action function with probability $1 - \varepsilon$. By considering this policy, the exploration of all the actions in all the states is ensured.

2.2.2 Temporal Difference Learning Methods

In the Temporal Difference (TD) framework, like in Monte Carlo, the transition probability is not known. The main difference is that these methods work online to update the estimation of the action–value function. For this reason, TD control is considered to be an upgrade of Monte Carlo RL, thanks to continuous learning, the non-necessity of having an episode and terminating states, and the low variance of the estimation. In addition, TD methods may take into account reward signal delays, since sometimes it cannot be obtained immediately after performing a certain action.

The three main algorithms belonging to TD learning can be summarized as follows:

- TD(0) Learning
- SARSA
- Q–Learning.

TD(0) Learning

This learning method aims to estimate the state value function by using a single-step update, it is an evaluation method for a given policy π. The estimation is done based on the immediate reward and the next state's value estimate according to the given policy π. In the TD(0) algorithm, the update is done right after obtaining the reward and reaching the new state. The update rule at time $t + 1$ is defined as

$$V(s_t) \leftarrow V(s_t) + \alpha \left[R_{t+1} + \gamma V(s_{t+1}) - V(s_t) \right], \tag{2.13}$$

where $\alpha \in \mathbb{R}$ is the learning rate representing the step of the update.

The TD(0) algorithm is described in Alg. 1.

Algorithm 1 TD(0) Learning

Input: Policy π to be evaluated
Output: Updated state–value function $V(s) \quad \forall s \in S$
Initialization: $V(s) \in \mathbb{R} \quad \forall s \in S$ except that $V(terminal) = 0$
for episode $e \leftarrow 1$ **to** ∞ **do**
  Initialize s
  for each time step t in the episode **do**
    $a = \pi(s)$ action given by policy π at s
    Perform action a and observe R' and s'
    $V(s) \leftarrow V(s) + \alpha \left[R' + \gamma V(s') - V(s)\right]$
    $s \leftarrow s'$
  end for
end for

SARSA

The name SARSA comes from the sequence considered in the training phase, which is

$$s \rightarrow a \rightarrow r \rightarrow s \rightarrow a$$

i.e., State-Action-Reward-State-Action. SARSA is an on–policy TD control method, meaning that it finds the next action by following a given policy used also for the update rule, which is:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R(s_t, a_t, s_{t+1}) + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]. \qquad (2.14)$$

The complete algorithm is described in Alg. 2. It is possible to see that the update is done at every transition from a nonterminal state, exploiting the quintuple

$$(s_t, a_t, R_{t+1}, s_{t+1}, a_{t+1}).$$

Q-Learning

As in SARSA, also the Q–Learning algorithm exploits the action–value function $Q(s, a)$. In this case, however, the learning phase follows an off–policy, meaning that the way actions are chosen during training do not change. The name comes from the presence of a specific object, the Q–table, used to store data related to the action–value function $Q(s, a)$. Due to the presence of a terminal state, the learning is carried out in deterministic episodes: in each one of them, the agent continues

Algorithm 2 SARSA

Output: Updated action–value function $Q(s, a)$
Initialization: $Q(s, a) \in \mathbb{R} \forall s \in \mathcal{S}, a \in \mathcal{A}$ except that $Q(terminal, \cdot) = 0$
for episode $e \leftarrow 1$ **to** ∞ **do**
  Initialize s
  $a =$ action given by policy π at s based on the Q-table
  for each time step t in the episode **do**
    Perform action a and observe $r(s')$ and s'
    $a' =$ action given by policy π at s' based on the Q-table
    $Q(s, a) \leftarrow Q(s, a) + \alpha \left[R' + \gamma Q(s', a') - Q(s, a)\right]$
    $s \leftarrow s'$
    $a \leftarrow a'$
  end for
end for

interacting with the environment and changing the Q–values until it reaches the target state from the initial random one. The Q-table can be filled with random initial values, then, the agent detects the current state s and chooses an action a to be performed. After performing the action, the agent observes the reward $R(s, s', a)$ and the new state s' and updates the table by considering the new knowledge. The generic entry $(\bar{s}, \bar{a})$ of the Q–table can be interpreted as the mathematical quantity representing *how good* is performing an action $\bar{a}$ when the environment state is $\bar{s}$. The update rule in the Q–Learning method depends on the temporal difference which makes use of the error and the presence of a learning rate:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R(s_t, a_t, s_{t+1}) + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t)\right]. \quad (2.15)$$

The advantage of using Q-Learning over SARSA is the fast-iteration with the environment, since the policy does not have to be updated online. On the other hand, SARSA is more stable than Q-Learning and is preferred when considering safety constraints in the environment [57]. The complete Q–Learning algorithm is given in Alg. 3.

Q–Learning is a foundational technique in the field of data–driven control since it does not require a model of the system's dynamics, making it suitable for decision–making problems in unknown or complex environments. However, Q–Learning becomes impractical in high–dimensional state and action spaces[4] and can only work with discrete actions and state spaces. This latter feature may require the discretization of continuous spaces, thus leading to information loss.

[4]This issue is known in literature as the *curse of dimensionality*.

Algorithm 3 Q-Learning

Output: Updated Q–table $Q(s, a)$
Initialization: $Q(s, a) \in \mathbb{R} \quad \forall s \in \mathcal{S}, a \in \mathcal{A}$ except that $Q(terminal, \cdot) = 0$
for episode $e \leftarrow 1$ **to** ∞ **do**
  Initialize s
  for each time step t in the episode **do**
    Choose the action a according to off–policy $\pi(s)$
    Perform action a on the environment and observe R' and s'
    $Q(s, a) \leftarrow Q(s, a) + \alpha \left[R' + \gamma \max_{a' \in \mathcal{A}} Q(s', a') - Q(s, a) \right]$
    $s \leftarrow s'$
  end for
end for

For this reason, since in all engineering fields most of control problems involve continuous spaces, there is the need of developing more advanced algorithms capable of learning patterns and policies suitable for complex control tasks.

2.3 Multi–Agent Reinforcement Learning

The notion of MDP is usually applied to single–agent systems, i.e. systems in which a single controller is present. In many practical problems, however, there are multiple agents interacting within the same environment. To properly address these scenarios, the concept of Markov Game is introduced.

Markov Game, also known as Stochastic Game, is a formal framework used to model multi-agent decision-making in a sequential, stochastic environment. Mathematically, a Markov Game is defined as

$$\text{MG} =< \mathcal{I}, \mathcal{S}_i, \mathcal{A}_i, P, R_i >, \tag{2.16}$$

where:

- $\mathcal{I} = \{1, \ldots, N\}$ denotes a finite set of agents, where N is the number of agents. Each one of them is a decision–maker and can take actions over the environment.
- $\mathcal{S}_i$ is the private state space of the i–th agent, representing both local information of the controller status or global information related to the process. The joint state space can be defined as

$$\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_N.$$

- $\mathcal{A}_i$ is the set of actions that agent i can take. The joint space is defined as

$$\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_N.$$

- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the probability of going from one state to another one given a certain set of action. Also in the multi–agent case, the transition probability function satisfies the Markov property.
- $R_i : \mathcal{S}_i \times \mathcal{A}_i \times \mathcal{S}_i \to \mathbb{R}$ is the private reward of each agent i, which provides feedback based on the chosen action and corresponding state.

The challenging problem related to Markov Games is that each agent needs to find its optimal policy $\pi_i : \mathcal{S}_i \to \mathcal{A}_i$ which shall maximise the expected cumulative reward.

Solving a Markov Game involves finding a set of optimal policies for all agents, considering the interactions between them. The solution relies on the notion of Nash equilibrium [58], where no agent has an incentive to unilaterally change its policy. This concept has proven to be fundamental in numerous disciplines, such as macroeconomics, finance and social interactions.

In the context of Markov Games, a Nash equilibrium is defined as follows.

Given a Markov Game with N agents, each agent i selects a policy π_i. Moreover, define as $\pi = (\pi_1, \ldots, \pi_N)$ the joint policy given as the combinations of policies of all agents. A joint policy π^* is a Nash equilibrium if, for each agent i, no unilateral change in policy by agent i can improve its expected reward while keeping the policies of other agents fixed. Mathematically, a joint policy π^* is a Nash equilibrium if, $\forall\, i = 1, \ldots, N$ and $\forall\, \pi_i^{'}$ alternative policy

$$\mathbb{E}\Big[R_i|\pi^*\Big] \geq \mathbb{E}\Big[R_i|\pi_i^{'}, \pi_{-i}^*\Big], \tag{2.17}$$

where π_{-i}^* represents the equilibrium policies for all agents except agent i. In simpler terms, in a Nash equilibrium, each agent's policy choice is optimal given the policies of the other agents. No agent can improve its situation by changing its strategy alone, assuming the strategies of the other agents remain unchanged.

Markov Games can model scenarios where agents may need to cooperate or compete with each other, providing a formal framework for modeling scenarios involving multiple decision-makers, such as collaborative multi–robot tasks or competitive access to shared computing or connectivity resources. For these reasons, Markov Games are widely used and solved in the context of Multi Agent Reinforcement Learning (MARL), in which multiple intelligent agent following the RL basic principles interact together.

A generic MARL problem can be categorized into three scenarios:

1. Cooperative, in which agents are tasked with collaborating to address a common objective, striving to collectively maximize a shared reward. A typical example is a fleet of UAVs trying to accomplish a certain task, like formation control.
2. Competitive, where agents engage in a zero-sum game where only one agent can ultimately prevail. As a result, agents focus on maximizing their individual rewards while simultaneously seeking to minimize the rewards of their peers. Typical examples of competitive scenarios include gaming activities or resource allocation such as car racing, blackjack, and chess.
3. Mixed, with the goal of striking a balance between cooperation and competition. Notable examples include team sports played by humanoid robots.

Moreover, depending on the specifics of the training and execution phase of the specific RL algorithm, cooperative MARL algorithms can be classified into three distinct learning paradigms [59].

2.3.1 Decentralized Training and Centralized Execution

In the Decentralized Training and Decentralized Execution (DTDE) framework, each agent operates with its own policy, which maps its local observations to a unique action distribution represented as $\pi_i : \mathcal{S}_i \rightarrow \mathcal{A}_i$. Importantly, these agents do not share information, and each agent independently learns its policy. This approach can be applied to large–scale multi–agent systems, where a centralized controller cannot be applied. One drawback of this approach is the lack of information sharing, that causes the non-stationarity in the environment, making the learning process more challenging. Despite these limitations, DTDE paradigms has found broad applications in solving tasks such as cooperative navigation and formation control [60]. The typical structure of a DTDE algorithm is shown in Fig. 2.2.

2.3.2 Centralized Training and Centralized Execution

The Centralized Training and Centralized Execution (CTCE) paradigm employs a centralized learner with the objective of acquiring a unified policy for all agents, denoted as $\pi : \mathcal{S} \rightarrow \mathcal{A}$. This collective policy translates distributed observations into a set of action distributions for each individual agent. An essential prerequisite in the CTCE approach is seamless and instantaneous communication among agents.

In cases with a relatively larger number of agents, the CTCE paradigm confronts a challenge known as the curse of dimensionality. This arises because the total

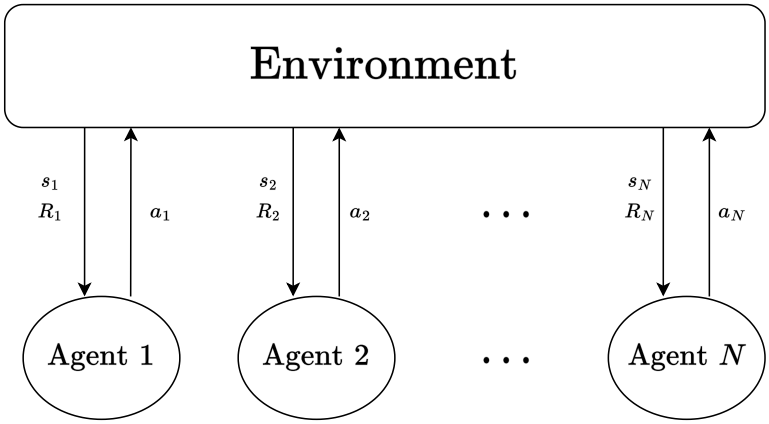


Figure 2.2. DTDE Architecture.

state-action space, considering all agents, can grow exponentially, rendering the quest for an optimal joint policy infeasible. To mitigate the dimensionality issue, a common strategy is to decompose the joint policy into individual agent policies, allowing them to train independently while facilitating information exchange among themselves [61].

However, this approach introduces a new problem referred to as the *lazy agent problem*, in which one agent may have a reduced incentive to learn an effective policy because its actions might hinder another agent from acquiring a superior policy, ultimately resulting in lower collective rewards. In the context of the lazy agent problem, team members exhibit varying levels of performance but still share the same cumulative reward. To address this challenge, researchers have put forth a range of learning and non-learning methods that assign credit to each agent based on their individual contributions [62].

Fig. 2.3 shows the standard CTCE architecture.

2.3.3 Centralized Training and Decentralized Execution

Both the CTCE and DTDE approaches present few drawbacks, which is why a modern MARL paradigm, the Centralized Training and Decentralized Execution (CTDE) combines elements of both to design advanced algorithms. In this hybrid paradigm, each agent possesses its own policy, mapping its local observations to individual action distributions. Notably, a key departure from CTCE is that any supplementary information provided during the training phase is discarded during testing.

In the training phase, agents have the capacity to enhance their learning speed and address non-stationarity in the environment by sharing resources, including computational resources and acquired knowledge. Mutual information allows agents

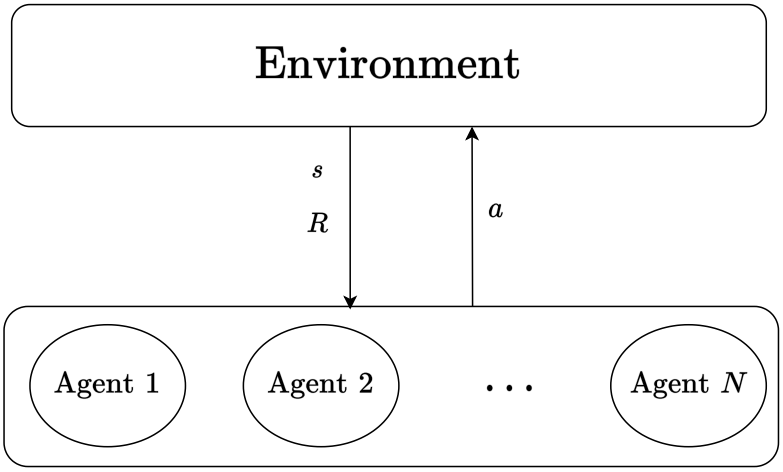


Figure 2.3. CTCE Architecture.

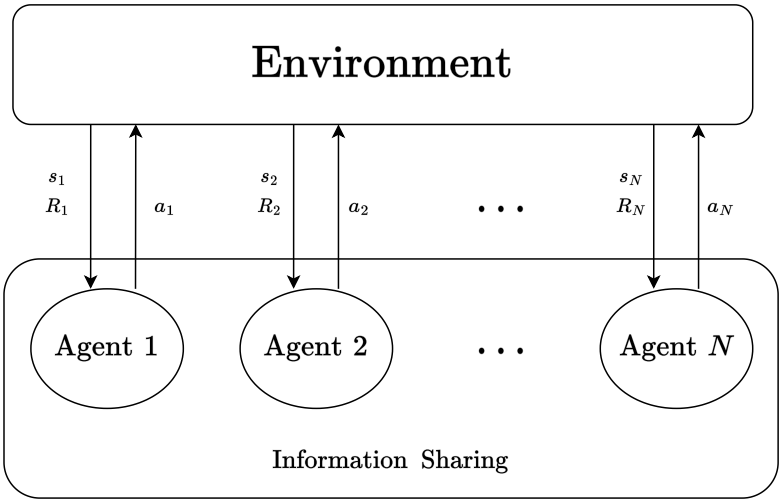


Figure 2.4. CTDE Architecture.

to connect action outcomes with their respective agents, thereby facilitating resource sharing, such as computational power and accumulated knowledge. Later on, during the execution phase, each agent acts independently from the others, but using the local policy learned thanks to sharing information with the others during the training phase [63].

The CTDE general scheme is highlighted in Fig. 2.4.

Chapter 3

From Discrete to Continuous Spaces

The algorithms in the previous chapter work fine only with finite and discrete states and actions. Hence, it is very likely that Q–Learning or SARSA cannot be applied to solve real–world problems in which either spaces are discrete but contains a huge number of possible combinations, or they are continuous, meaning that both system outputs and controls lie in the set of real numbers $\mathbb{R}$.

For this reason, a paradigm shift from the traditional tabular representation is needed, with the aim of shaping in a different way the state–value function $V_\pi(s)$ or the action–value function $Q_\pi(s, a)$.

3.1 Function Approximation Methods

The main idea is to estimate the action–value function as a generic function $\phi(\cdot)$ parameterized by a set of parameters ϑ:

$$Q_\pi(s, a) \approx \phi(s, a|\vartheta). \tag{3.1}$$

This framework lies within the context of function approximation methods, which have gained the attention of the AI and control scientific community since the early 90's [52].

At the beginning of the training procedures, the function parameters ϑ are usually initialized randomly and later on they are adjusted (in a very similar way of SL), trying to reach an optimal policy.

There are several techniques belonging to this class of methods, differentiating one from the other for the shape of the function approximating the action–value $Q_\pi(s, a)$. The most used ones are reported as follows:

- Linear Function Approximation: the function is linear in its features and in the state. Even if it may seem a simplistic approach, it guarantees explainability and interpretability [64] of the agent behavior, making it suitable for many practical RL problems.
- Fourier Approximation: it approximates the value function using a linear combination of sinusoidal functions. They are well-suited in control problems in which the optimal action and the corresponding states repeats over time [65].
- Tile Coding: the state space is partitioned into overlapping tiles, each one associated with a feature vector (usually with linear properties), and the value function is approximated as a weighted sum of these features [66]. This approach is particularly useful when the state space is discrete but may contain a huge number of samples.
- Decision Trees and Random Forests: this approach is very similar to tile coding, since also in this case the state space is partitioned into regions, each one getting a specific value [67].
- Ensemble Methods: they combine multiple models (linear, nonlinear, stochastic) trying to improve approximation accuracy avoiding overfitting. The most known techniques in this field are the bootstrap aggregating (or bagging) and boosting [68].
- Neural Networks: since a generic NN is a nonlinear mapping from the input space x to the output space y (as shown in the section related to SL), they can be used as a matter of fact as nonlinear function approximator of state–value and action–value functions in the RL domain. It has been shown in literature that this latter method outperforms all the previous ones.

The works and research activities related to this PhD thesis refers entirely to function approximation methods carried out through NN. In this framework, SL and RL are linked together in order to build intelligent agent observing states and rewards coming from the environment (feedback principle) and computing actions as the output of a complex nonlinear function of the state, represented as a neural network.

3.2 Deep Q–Learning

One of the greatest scientific successes of the last decade in the field of AI is linked to the development of the Deep Q–Learning, an algorithm that extends the well-known and consolidated Q–Learning allowing to deal with MDP characterized by continuous states and discrete actions.

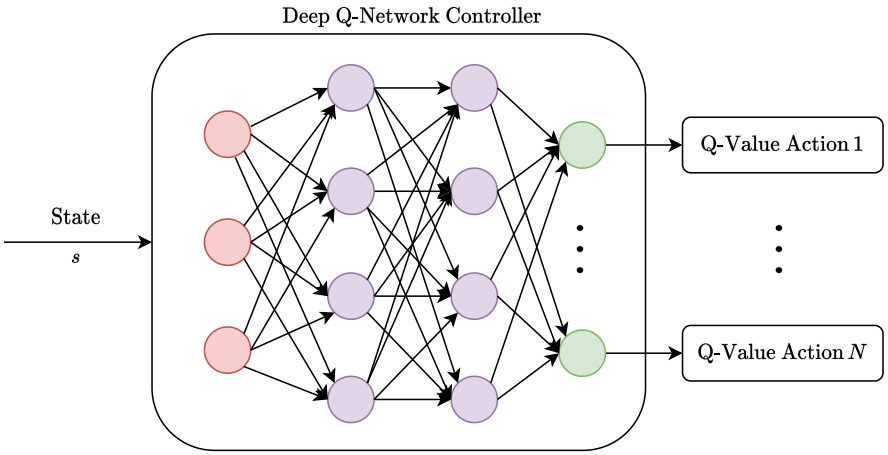


Figure 3.1. Graphical representation of a Deep Q–Network.

This algorithm appeared for the first time in 2013, in a research work which on a first glance may seem to have nothing in common with control theory [69]. The authors present an intelligent algorithm capable of beating the CPU in the classic Atari arcade games, like Arkanoid, Pong and Space Invaders [70]. In particular, this RL algorithm takes as input RGB frames coming from the game, displaying the current situation, and provides as output discrete commands to be executed by the user character within the game. The function approximator used is a Convolutional Neural Network (CNN), a particular type of DNN designed specifically for classification tasks involving RGB images [71]. The research results show that this new algorithm outperforms the other ones used in the domain of RL, being able to surpass even human experts in some of the games.

The work in [69] gave the green light to a new research frontier in the field of data-based control methodologies: Deep Reinforcement Learning (DRL). As the name suggests, DRL refers to the set of techniques exploiting the combination of DNN and RL.

Deep Q–Learning is the first example of using NN for approximating action-value functions. In this framework, the DNN function approximator takes the name of Q–network (see Fig. 3.1). The idea behind is the same as in Q–Learning: the DNN is trained following an off–policy mechanism, with the introduction of the fundamental notion of *experience replay*. The latter is a computer–science technique through which it is possible to store agent's experiences at each time step t, provided as the tuple $e_t = (s_t, a_t, r_t, s_{t+1})$ in a dataset $\mathcal{D} = e_1, \ldots, e_N$.

The Q–Learning update is then applied using some random samples from the dataset $\mathcal{D}$. After performing the experience replay step, the agent selects the action according to a given policy π.

The Q–network is trained by minimizing a sequence of loss functions $L_i(\theta_i)$:

$$L_i(\theta_i) = \mathbb{E}\Big[(y_i - Q(s, a|\theta_i))\Big], \tag{3.2}$$

where θ_i represents weights and biases of the NN, namely the trainable parameters, and y_i is the target action–value function at iteration i, computed as

$$y_i = \mathbb{E}\Big[R + \gamma \max_{a'} Q(s', a'|\theta_{i-1})\Big]. \tag{3.3}$$

It is important to notice that the target values depend on the network weights at the previous iteration θ_{i-1}: this is in contrast with the concept of true label introduced in the SL domain. The targets in that case, indeed, are fixed and provided as the ground truth during the training phase.

Starting from (3.2) it is possible to define the gradient of the loss function

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}\Big[(R + \gamma \max_{a'} Q(s', a'|\theta_{i-1}) - Q(s, a|\theta_i))\nabla_{\theta_i} Q(s, a|\theta_i)\Big]. \tag{3.4}$$

Rather than computing the full expectations in the above gradient, it is often computationally expedient to optimize the loss function by numerical algorithms based on the gradient descent principle. As already stated, Deep Q–Learning is an off–policy algorithm since it learns the optimal policy following a pre–defined behavior distribution which shall ensure an adequate exploration of the state space. The most frequent choice, as in standard Q–Learning, is to pick the ε–greedy strategy, thus choosing a random action with probability ε, and the best action so far with probability $1 - \varepsilon$ [52]. The Deep Q–Learning algorithm is detailed in 4.

Algorithm 4 Deep Q-Learning with Experience Replay

Initialization: Replay memory $\mathcal{D}$ of capacity N and action-value function Q with random weights
for episode $e \leftarrow 1$ **to** M **do**
  Initialize s
  for each time step $t \leftarrow 1$ **to** T **do**
    Select action a_t following ε–greedy strategy
    Perform action a_t and observe R_t and s_{t+1}
    Store the transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
    Sample from $\mathcal{D}$ a random minibatch of transitions (s_j, a_j, r_j, s_{j+1})
    Set $y_j = \begin{cases} r_j & \text{for terminal } s_{j+1} \\ r_j + \gamma \max\limits_{a \in A(s_{j+1})} \hat{Q}(s_{j+1}, a, w) & \text{for non terminal } s_{j+1} \end{cases}$
    Perform a gradient descent step according to (3.4)
  end for
end for

After the breakthrough brought by Deep Q–Learning, other works have shown how it is possible to improve the algorithm's performance by adding other features, like a double NN in a dueling scenario [72]. However, these kind of algorithms have an important drawback: they cannot deal with environment requiring continuous actions. The latter situation is very common in control practice: other approaches and refinements are then required.

3.3 The Policy Gradient Mechanism

Policy gradient methods are a class of RL algorithms that focus on directly learning the policy π, without explicitly modeling or estimating the value function $V(s)$ or $Q(s, a)$. These methods aim to optimize the policy, parameterized as a probability distribution over actions, by adjusting the parameters in order to maximize the expected cumulative reward, which can be seen as a cost function

$$J(\theta) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R_t\Big]. \tag{3.5}$$

The learning step updating the policy parameters is performed using again a gradient–descent methodology, so that the policy is updated in the direction along which the gradient increases.

Policy gradient methods can naturally handle stochastic policies, which can be useful for those MDP which are characterized by a non–deterministic dynamics. However, their most important feature relies on the fact that this family of RL methods can be applied to both continuous and discrete action spaces, thus solving the limitations of the Deep Q–Learning domain.

One of the most simple algorithm implementing the notion of policy gradient is known as REINFORCE, or Monte Carlo Policy Gradient Control [73]. It estimates the gradient of the expected return with respect to the policy parameters using the following formula

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\Big[G_t \nabla_\theta \log \pi_\theta(a_t|s_t)\Big], \tag{3.6}$$

where $G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R(s_k, a_k)$. This gradient estimator encourages actions that lead to higher rewards to have higher probabilities and actions with lower rewards to have lower probabilities. The full algorithm is presented in Alg. 5.

The REINFORCE algorithm does not come without issues. It is affected by a high variance in their gradient estimates, which can lead to slow and unstable learning, since it does not include any policy trust region.

Algorithm 5 REINFORCE: Monte-Carlo Policy Gradient Control

Input: a differentiable policy parameterization
Algorithm parameter: learning rate α
Initialization: Policy parameter θ
repeat
  Generate an episode $s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1}$
  for each time step $t \leftarrow 0$ **to** $T-1$ **do**
    $G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$
    $\theta \leftarrow \theta + \alpha \gamma^t G \nabla_\theta \pi_\theta(a_t|s_t)$
  end for
until forever

3.3.1 Proximal Policy Optimization

An algorithm that takes into account the parameter updates by adding constraints on the size of policy updates and considers other aspects is the Proximal Policy Optimization (PPO) [74].

PPO is a policy gradient method that has a different objective function to enable multiple epochs of minibatch updates. This on-policy method has a simpler implementation with respect to other algorithms that have similar features, like Trust Region Policy Optimizer. PPO makes use of the estimation of the advantages (the difference between the action value and state value function according to π) in such a way the convergence to the optimal policy is faster. In PPO, the objective function $J(\theta)$ to maximize is a variation of the Clipped Surrogate Objective used in the Trust Region Policy Optimization (TRPO) [75]. The maximization problem of the Clipped Surrogate Objective is defined as follows [74]:

$$\begin{aligned} &\max_{\theta} \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right] \\ \text{subject to} \quad &\hat{\mathbb{E}}_t \left[KL \left[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t) \right] \right], \end{aligned} \tag{3.7}$$

where $\hat{A}_t$ represents the estimation of the advantages at time step t calculated as $\hat{A}_t = \hat{Q}_t - \hat{V}_t$, θ_{old} represents the parameter of the policy before the update, $KL[\cdot,\cdot]$ is the Kullback–Leibler divergence operator and $\hat{\mathbb{E}}$ represent the empirical expectation (averaging on the collected values over time steps t).

The function to be maximized in (3.7) is obtained in such a way that its gradient is similar to the gradient in (3.6). The gradient of the objective function can be rewritten as:

$$\hat{\mathbb{E}}_t \left[\sum_a Q_\pi(s_t, a) \nabla_\theta \log \pi_\theta(a|s_t) \right] = \hat{\mathbb{E}}_t \left[\sum_a \left(Q_\pi(s_t, a) - b(s_t) \right) \nabla_\theta \log \pi_\theta(a|s_t) \right], \tag{3.8}$$

where $b(s_t)$ is any function, even a random variable, that does not vary with the action a. The equality is given due to the fact that the added amount

$$\sum_a b(s_t)\nabla_\theta \log \pi_\theta(a|s_t)$$

is equal to zero:

$$\sum_a b(s_t)\nabla_\theta \log \pi_\theta(a|s_t) = b(s_t)\nabla_\theta \sum_a \log \pi_\theta(a|s_t) = b(s_t)\nabla_\theta 1 = 0. \tag{3.9}$$

A possible choice for the function $b(s_t)$ could be the state–value function or its estimation $\hat{V}(s_t)$. Also, in this case, the policy gradient theorem [52] is used to calculate the gradient:

$$\begin{aligned} \nabla_\theta J(\theta) &= \hat{\mathbb{E}}_t \left[\left(Q_{\pi_\theta}(s_t, a) - V_{\pi_\theta}(s_t) \right) \nabla_\theta \log \pi_\theta(a_t|s_t) \right] \\ &= \hat{\mathbb{E}}_t \left[\hat{A}_t \nabla_\theta \log \pi_\theta(a_t|s_t) \right]. \end{aligned} \tag{3.10}$$

At this point, remains only to show that the previous equation differentiates the the function to maximize in (3.7). For this scope, the chain rule is applied to calculate the derivative of the objective function:

$$\nabla_\theta \log f(\theta)|_{\theta_{old}} = \frac{\nabla_\theta f(\theta)|_{\theta_{old}}}{f(\theta_{old})} = \nabla_\theta \left(\frac{f(\theta)}{f(\theta_{old})} \right) \Big|_{\theta_{old}}. \tag{3.11}$$

Thus, the policy gradient, found previously, does actually differentiate, for small policy changes, the objective function to maximize in TRPO. On the other hand, the use of the constraint in this case limits large updates in the policy parameters to avoid instability during the learning. Nevertheless, the given constraint makes the optimization problem very complex, that's why a modification of the PPO is used in such a way the same goal is achieved. Instead of considering a constrained optimization problem, another objective function is used:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right) \right], \tag{3.12}$$

where

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, \tag{3.13}$$

and ϵ is a hyperparameter denoting the amplitude of the clipping range. The $r(\theta)$ is the probability ratio between the new updated policy outputs and the outputs of the previous old version of the policy network. $r(\theta) > 1$ if the action is more likely now than it was in the old version of the policy, conversely $0 < r(\theta) < 1$ if the action is less likely now than it was before the last gradient step.

The multiplication by $\hat{A}_t$ gives an idea about how well or worth the policy is acting with respect to the baseline, which is the state value function. The use of this objective function will continue improving the parameterized policy but the clipping part will limit the updates forcing them to stay within the region $[1-\epsilon,\ 1+\epsilon]$.

Keeping in mind that the advantages could be positive or negative, for better or worse performance of the policy, the min operator will work differently in these two cases. When $\hat{A}_t$ is positive, the objective function, for a certain t, has the maximum value of $\hat{A}_t \cdot (1+\epsilon)$ if $r_t(\theta) > 1+\epsilon$ otherwise the increase is linear. On the other hand, when $\hat{A}_t$ is negative, the maximum value for the objective function in t is $\hat{A}_t \cdot (1-\epsilon)$ if $r_t(\theta) < 1-\epsilon$.

In this way, the step done in the optimization is limited in both cases. When the policy acts better than expected (positive $\hat{A}_t$), the action update is not overdone thanks to the clipping. On the contrary, when the action is worse than expected (negative $\hat{A}_t$), the clipping prevents the overdone update which would reduce largely its probability in the future.

In addition, to combine the strengths of policy gradients and value-based methods, PPO makes use of the Actor–Critic architecture[1]. In this scheme, the policy (actor) is optimized using policy gradients, while a value function (critic) is used to estimate the expected return and to reduce the variance in gradient estimates. The critic network, during the learning, is updated frequently according to the experience collected by the interaction between the agent and the environment. Due to the fact that PPO uses an estimation of the state value function, the final PPO objective function is:

$$L_t^{PPO}(\theta) = L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t \left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t) \right], \quad (3.14)$$

where $c_1, c_2 > 0$, S represents the entropy bonus to ensure sufficient exploration and L_t^{VF} is a squared-error loss $\left(V_\theta(s_t) - \hat{V}_t\right)^2$. Note that the estimation of the state value function, by the critic network, is done by making it share the parameters with the actor network, this is why the squared error of the state value function is put in L_t^{PPO}.

The algorithm of the PPO is shown in 6: the policy $\pi_{\theta_{old}}$ is used to interact with the environment creating episode sequences. For each episode, the advantages are calculated by using the estimated state value function, then after many episodes, the gradient descent is used on the policy by making use of the past experience and

[1]The name Actor–Critic derives from the way of learning typical of children. Indeed, when human beings are very young and not yet rational, they learn to distinguish good policies from the bad ones by listening to adults' hints and suggestions.

the objective function (for better performance it is suggested Adam [74]). A way to calculate the advantages, proposed in [74], is:

$$\hat{A}_t = -V(s_t) + \hat{Q}(s_t, a_t) = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_T), \tag{3.15}$$

where T is the number of timesteps and t specifies the time index in $[0,\ T]$. Notice that all the calculations in PPO exploits a fixed length of the trajectory segments. In 6 it was used N as the number of actors which could be the same as N runs of the interaction with the environment.

Algorithm 6 PPO, Actor-Critic Style

1: **for** iteration=$1, 2, \ldots$ **do**
2: **for** actor=$1, 2, \ldots, N$ **do**
3: Run policy $\pi_{\theta_{old}}$ in environment for T timesteps
4: Compute advantages estimates $\hat{A}_1 \ldots \hat{A}_T$
5: **end for**
6: Optimize L^{PPO} wrt θ with K epochs and minibatch size $M \leq NT$
7: $\theta_{old} \leftarrow \theta$
8: **end for**

3.3.2 Deep Deterministic Policy Gradient

The event finally bridging the gap between the RL and control theory was the publication of the Deep Deterministic Policy Gradient (DDPG) algorithm [76] in 2015. Authors adapt and extend the ideas underlying the success of Deep Q–Learning to the continuous action domain, providing an actor–critic, model–free, off–policy algorithm able to operate over continuous action spaces. In the article it is shown that DDPG is able to robustly solve a lot of typical tasks in control theory, like inverted pendulum stabilization, cartpole swing-up, dexterous manipulation, legged locomotion, and car driving. Moreover, the performance is competitive with those found by a standard control methodology with full access to the system dynamics.

Even in this case, as in PPO, the goal is to learn a policy maximizing the expected return (see (3.5)), exploiting the concept of action–value function. As many others RL algorithms, DDPG make use of the recursive relationship known as Bellman equation, where the target policy can be described in general as a function $\mu : \mathcal{S} \rightarrow \mathcal{A}$, so that

$$Q^{\mu}(s_t, a_t) = \mathbb{E}\Big[R(s_t, a_t) + \gamma Q^{\mu}(s_{t+1}, \mu(s_{t+1}))\Big]. \tag{3.16}$$

As seen in the previous chapter, Q–Learning implements

$$\mu(s) = \text{argmax}_{a' \in \mathcal{A}} Q(s, a'), \tag{3.17}$$

but in general $\mu(\cdot)$ may be any complex nonlinear function.

In the DDPG framework, the $Q(s, a)$ is approximated through NN parameterized by the set of parameters θ^Q, which are optimized by minimizing the loss function

$$L(\theta^Q) = \mathbb{E}\Big[(Q(s_t, a_t|\theta^Q) - y_t)^2\Big], \tag{3.18}$$

where

$$y_t = R(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q). \tag{3.19}$$

To implement the target policy, the actor–critic framework is again exploited, thus defining a parameterized actor function $\mu(s|\theta^\mu)$, which deterministically maps states into actions. The critic is given by the action–value function $Q(s, a|\theta^Q)$ and is updated by performing gradient descent steps on (3.18). It is worth noting that also the target y_t depends on θ^Q (see (3.19)), and thus the gradient of the loss function should involve the differentiation of y_t, but in DDPG this is ignored.

As per the actor, it is updated using the chain rule on the gradient of (3.5):

$$\begin{aligned} \nabla_{\theta^\mu} J &= \mathbb{E}\Big[\nabla_{\theta^m u} Q(s, a|\theta^Q)|_{a=\mu(s|\theta^\mu)}\Big] \\ &= \mathbb{E}\Big[\nabla_a Q(s, a|\theta^Q)|_{a=\mu(s|\theta^\mu)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)\Big]. \end{aligned} \tag{3.20}$$

As it happens in Deep Q–Learning, also DDPG uses the concept of experience replay to learn extracting random samples of experiences $e_i = (s_i, a_i, r_i, s_{i+1})$ from the buffer $\mathcal{D}$.

Since the critic NN $Q(s, a|\theta^Q)$ is also used in calculating the target value (the actor NN), the gradient descent performed on the loss function may lead to divergence. For this reason, in [76] authors modify the concept of target networks already present in [69], applying it to the actor–critic case. This is done through the creation of a copy of the actor and the critic NN, $Q'(s, a|\theta^{Q'})$ and $\mu'(s|\theta^{\mu'})$. The idea is to update their weights slowly during the training phase, assigning to the target weights θ' the learned values of the two real actor–critic weights θ:

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta', \tag{3.21}$$

where $\tau \ll 1$ is a crucial parameter expressing how much one should trust the learned weights. This update rule allows to avoid gradient divergence.

Eventually, as for the exploration within the environment, this is one of the critical aspects which can decree convergence or not. In the continuous domain, it is not possible to follow the standard ε–greedy policy, since it is not possible to sample randomly an action from a finite set. In DDPG the exploration is carried out by defining an exploration policy

$$\mu^{\mathcal{N}}(s_t) = \mu(s_t|\theta_t^{\mu}) + \mathcal{N}, \tag{3.22}$$

where the random variable $\mathcal{N}$ denotes a noise distribution from a given process. One of the possible choices is to pick the Ornstein–Uhlenbeck noise [77], related to the Brownian motion in fluids.

When working with physical systems, it is common that the observation have dissimilar physical units and magnitudes[2]. This discrepancy can pose challenges for the network's effective learning and complicate the task of finding hyperparameters that can generalize across environments with varying state value scales. This issue is directly correlated to the train data normalization problem in SL, discussed on Chapter 1.

One way to tackle this issue is to manually adjust the features so that they have similar ranges across different environments and units. Authors in [76] address this problem by adopting a recent DL technique known as batch normalization, introduced in [78]. This method normalizes each dimension within a minibatch, ensuring that they have a mean and variance equal to one. Furthermore, it maintains a running average of the mean and variance, which is used for normalization during testing, such as during exploration or evaluation in our context. In DNN, this technique is employed to minimize covariance shifts during training by ensuring that each layer receives standardized input. Hence, in DDPG batch normalization is applied to the state observation seen by the agent, which can be effectively learn across a wide range of tasks involving various unit types without the need for manual adjustments to ensure that the units fell within a specific range.

The full DDPG algorithm is shown in Alg. 7.

Part of this thesis will focus its attention on the application of the DDPG control algorithm to multi–agent systems in telecommunication networks: it will be shown how it is possible to train agents in competitive and collaborative scenarios, realizing control task like reference tracking and energy minimization.

---

[2]Think about a pendulum on a cart. The pendulum angle ϕ varies periodically in the specific interval $[0,\ 2\pi]$, while the cart horizontal position x may assume any real value.

Algorithm 7 Deep Deterministic Policy Gradient

Initialize randomly the networks $Q(s, a|\theta^Q)$ and $\mu(s|\theta^\mu)$ with weights θ_0^Q and θ_0^μ
Initialize target networks $Q'(\cdot)$ and $\mu'(\cdot)$ with weights $\theta_0^{Q'} \leftarrow \theta_0^Q$ and $\theta_0^{\mu'} \leftarrow \theta_0^\mu$
Initialize the replay buffer $\mathcal{D}$
for episode=$1, 2, \ldots, M$ **do**
  Set a random process $\mathcal{N}$ for action exploration
  Observe the first state s_1
  for t=$1, 2, \ldots, T$ **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$
    Apply a_t on the environment and observe r_t and s_{t+1}
    Store transition $e_t = (s_t, a_t, r_t, s_{t+1})$ in $\mathcal{D}$
    Sample a random minibatch $e_i = (s_i, a_i, r_i, s_{i+1})$ from $\mathcal{D}$
    Set the target $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic minimizing the loss $L(\theta^Q) = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the gradient rule in (3.20)
    Update the target networks weight following (3.21)
  end for
end for

3.4 Classification of Reinforcement Learning Algorithms

Apart from the algorithms presented so far, there are other four algorithms worth mentioning. Each one of them allows to handle continuous state spaces and continuous actions, like PPO and DDPG. These techniques are summarized in what follows.

- Soft Actor–Critic (SAC) is a DRL algorithm that falls under the category of off–policy methods. As DDPG, it is designed for continuous action spaces. SAC combines elements of actor-critic architecture with entropy regularization to encourage exploration [79]. The entropy regularization helps balance the trade-off between exploration and exploitation in a more principled way. As all the other DRL techniques, the DNN in SAC is trained using samples from a replay buffer.
- Asynchronous Advantage Actor–Critic (A3C) is a distributed, on–policy reinforcement learning algorithm that leverages multiple actor-learner agents to train a single shared NN policy and value function simultaneously. It is designed for both discrete and continuous action spaces and exploits the notion of advantage function to assess the quality of actions taken and improve the policy. It has been particularly successful in solving complex tasks in environments with high-dimensional observations [80–82].

- Q–Learning with Normalized Advantage Functions (NAF) is an off–policy DRL algorithm designed for continuous action spaces. It reuses the concept of advantage function from A3C, approximating both state–value and advantage function using NN, providing a more accurate estimation of the value of taking specific actions in continuous action spaces [83]. As a matter of fact, this algorithm is a direct extension of Deep Q–Learning to continuous actions.
- Twin Delayed Deep Deterministic Policy Gradient (TD3) is an off-policy reinforcement learning algorithm designed for continuous action spaces. It is an extension of the Deep Deterministic Policy Gradients (DDPG) algorithm and introduces several modifications to enhance its stability and performance, like the creation of a smooth target value estimate to mitigate overestimation bias [84]. Moreover, it maintains two Q–networks to further stabilize training.

Tab. 3.1 summarizes the algorithms described in this thesis, categorizing them based on the type of policy used (on–policy VS off–policy), state and action spaces (discrete VS continuous) and the type of value function.

Table 3.1. Reinforcement Learning algorithms classification

Algorithm	**Policy Type**	**State Space**	**Action Space**	**Value Function**
Monte Carlo	Either	Discrete	Discrete	Sample-means
Q-learning	Off-policy	Discrete	Discrete	Q-value
SARSA	On-policy	Discrete	Discrete	Q-value
DQN	Off-policy	Continuous	Discrete	Q-value
REINFORCE	On-policy	Continuous	Either	Advantage
TRPO	On-policy	Either	Continuous	Advantage
PPO	On-policy	Either	Continuous	Advantage
DDPG	Off-policy	Continuous	Continuous	Q-value
A3C	On-policy	Continuous	Continuous	Advantage
NAF	Off-policy	Continuous	Continuous	Advantage
TD3	Off-policy	Continuous	Continuous	Q-value
SAC	Off-policy	Continuous	Continuous	Advantage

Part II

Terrestrial Networks

Chapter 4

Enhancing Cultural Heritage with 5G–Powered Augmented Reality

TERRESTRIAL communication networks are interconnected systems that rely on land–based electrical components to transmit data and information between a transmitting device and a receiving device, both located on the Earth's surface. Hence, in terrestrial communications wired and wireless signals cross and remain into the Earth's atmosphere.

Terrestrial communication networks are implemented on various transmission mediums, which constitute the physical pathways through which data and information are transmitted from one point to another. Terrestrial means of transmissions can be divided into wired and wireless ones:

1. **Wired**. Data transmission is realized through a direct cabled link between the transmitter and the receiver. All wired terrestrial communications are implemented using either copper or fiber optics cables.

    - Copper cables have been the baseline of terrestrial communication for decades. There are two primary types of copper wires: (i) twisted-pair cables, consisting of copper wire pairs twisted together and traditionally used for voice communication and digital subscriber line (DSL) internet connections, and (ii) coaxial cables, comprising a central conductor surrounded by insulation and a metallic shield, typically used for broadband services [85].
    - Fiber optic cables are one of the most advanced and widely used wired technologies. They consist of thin strands of glass or plastic fibers that

transmit data in the form of binary code (1s and 0s) as pulses of light. Fiber optics offer tremendous advantages with respect to copper cables, including high bandwidth, low signal attenuation, immunity to electromagnetic interference, and the capability for long-distance transmission [85]. They provide the highest data transfer speeds among wired technologies, which render fiber optics the ideal communication technology to handle demanding applications requiring high data–rates, like live High–Definition video–streaming and cloud computing.

In general, wired communications are characterized by their high reliability, due to the absence of signal degradation or interference, security, and high data transfer rates with low latency. These aspects have as main drawbacks the maintenance cost, users' mobility impossibility and the absence of a wide coverage areas.

2. **Wireless**. Data transfer is conceived without cables, exploiting instead electromagnetic waves. The electromagnetic radiation used for wireless transmission is commonly called *radio wave* and covers the frequency range from 3 KHz to 300 GHz [86]. The electronic device that allows to transmit radio signals is called a *radio transmitter*; if it is only capable of receiving it is called a *radio receiver*; if it is capable of both receiving and transmitting it is called a *transceiver*.

   In order to cover the distance between the transmitting radio and the receiving radio, it is necessary to use an antenna: a transducing device capable of transforming an electrical quantity into electromagnetic signals. The length and shape of the transmitting and receiving antennas are proportional to the wavelength of the frequency used. In order to transmit information from a transmitter to a receiver, it is necessary to define a frequency within the radio spectrum and a modulation scheme.

   Wireless communications realized through radio waves is used in copious applications including terrestrial television, cellular networks – relying on cell towers or base stations that provide coverage to mobile devices like tablets or smartphones – and Wi–Fi networks, short-range communication technologies often used for local area networks (LANs) and in–home or in–office networks [86].

   Wireless technologies have several advantages over the wired counterpart, including full mobility of users (think about listening to the radio while in a moving car), flexibility due to easy reconfiguration, and possibility to access remote areas and scalability, since a single wireless signal can serve a huge

number of users, like it happens in radio and television broadcasts [86]. It is straightforward that wireless communication have drawbacks in all aspects where the wired counterpart is strongly robust, like interference and security.

The second part of this thesis will focus its attention on wireless cellular networks, which allow users to make phone calls, send messages and access the internet while on the move.

4.1 Cellular Networks

A cellular telecommunications network (also called cellular network or mobile network) is a network that allows telecommunications in all points of a territory divided into small areas, called *cells*, for cellular mobile radio telephony, each served by a different Base Station (BS).

4.1.1 Base Stations

A BS is a radio transceiving system and it represents the basic cellular telephony infrastructure used in radio links of cellular mobile networks at the radio interface of the cellular system. Each BS belongs to a given Radio Access Technology (RAT), term through which one defines the physical connection method for a radio–based communication network. A generic BS does not act as a repeater, as in the case of radio links, but generates it and transmits it over the air, or receives it working at the physical level and datalink level of a network architecture [86]. In this essay, the term Access Point (AP) will be used as a synonym of BS[1].

In cellular telephony, from a topological point of view, radio BSs are logical switching or relay nodes in the radio interface of the cellular radio system, while from a transmission point of view they reroute the service request towards the same covering radio cell perform logical and physical functions of regenerative repeaters. A typical functionality, in addition to establishing a connection with the user terminal during communication, is also to spread a broadcast signal on the respective coverage cell to the various user terminals present which informs of the availability of the service and from which to obtain the known levels of field. Conversely, from the terminal it receives information about its presence in the cell useful for the various roaming functions.

Typically, radio BS are composed of transceiver antennas placed at a certain height on support pylons which are in turn placed in raised locations with respect to

[1]From a technical point of view, usually APs refer to indoor devices like routers and switches, and BSs denote outdoor devices like roof antennas and similar.

the coverage area of the radio cell to avoid disturbances and fixed radio propagation obstacles such as reliefs and vegetation, thus maximizing the coverage area, useful signal strength, signal-to-noise ratio and carrier/interference ratio. Thus, in urban environments the antennas are typically placed on the roofs of buildings (public or private) with the operator paying the owners of the building a rental fee for the concession of the installation in compliance with the regulations imposed on electromagnetic pollution, while in semi-urban and rural environments are located on small elevations and hills clear of vegetation.

4.1.2 Mobile Devices

BS provide connectivity services to users through systems called in general Mobile Devices (MDs) or User Equipments (UEs). The latter can be defined as user devices capable of transmitting and receiving data thanks to a wireless cellular network [86]. This functionality is realized through a special equipment mounted in each UE, called wireless network card.

A UE, in addition to the transceiver functions, in order to realize a successful communication with a BS must also have the following capabilities:

- Be able to synchronize and lock both to the frequency of the cell to which it belongs and temporally with the time-slot or frame dedicated to the user within the cell band during the radio connection. Typically at a logical level this procedure is implemented after measuring the power levels of the signal sent by the various base stations of neighboring cells and choosing the one with the highest power to maximize the signal/noise ratio or therefore the quality of the transmission.
- Periodically report its presence to the radio base station of the cell to which it belongs through a identification code (of the user, of the mobile phone, of the SIM card) to allow roaming, i.e. to be traced within the same network of an operator or by networks mobile phones from other operators. Typically this functionality is achieved directly when connecting to the radio cell which will therefore keep the information on all connected terminals in memory. This user information is then dynamically stored in a database available to the entire network.
- Adapt the power level emitted during a transmission according to the actual distance from the base station of the respective coverage cell, thus limiting the interference contribution on the neighboring co-channel cells and improving the efficiency of energy consumption or according to the real conditions of radio propagation present. This functionality is made possible by the constant

measurement of the Signal–to–Noise Ratio (SNR) with the base station. It follows that the power consumption (sum of transmission contribution and pre–processing contribution) of a MD during a transmission depends on the distance from the base station within the coverage cell, and is greater in transmission than in reception, where only the energy required for processing is needed.

- Practice handover, i.e. the change of communication channel within the same cell or between different cells when the terminal moves to the area of competence of another cell (cell switching) without interrupting communication. This functionality also involves the constant measurement of the power level of the signal received from radio base stations of neighboring cells and the connection to the destination cell when a certain pre–established power threshold is exceeded compared to that of the signal of the originating cell. This is then followed by synchronization over time and the reporting of identification for roaming. Some cellular telephone systems allow connection to other cells neighboring the one of residence even when the traffic in this cell is too high to be supported, thus guaranteeing greater service availability.
- Carry out the usual source coding and channel coding in transmission and the respective reverse coding (decoding) in reception if the communication is digital (as in all modern cellular networks). Furthermore, in any case, the UE will also have to provide for the encryption of the data in transmission and the respective deciphering in reception to guarantee the privacy or confidentiality of the communication on the radio medium.

Apart from the above mentioned features, which are present in any UE, the latter can be classified according to the ways in which the connection with the base network station is created:

1. **Multi–Mode Terminal**. Modern mobile radio terminals also have the ability to connect to the various mobile radio communication systems available in a territory, thanks to automatic switching procedures from one system to another and to multiple transceiver devices, i.e. therefore having multiple forms of connectivity available depending on the estimated quality of transmission in the various systems detected and/or costs. These features are in turn made possible by the interoperability between existing wireless technologies thanks to appropriate handover procedures from one system to another which attempt as much as possible to keep the same browsing session alive, while varying the quality specifications of transmission service by passing from one system to another. A MD whose network controller card have interfaces to multiple

RATs, but that can connect to one and only one BS at a time is referred to as multi–mode device. Everyday commercial devices like smartphones and tablets belong to this category. As a example, they can connect either to a 4G–LTE BS or to a Wi–Fi AP, but they cannot exploit both communications simultaneously.

2. **Multi–Homed Terminal**. These are UEs which can establish connections with multiple BSs at a time (also belonging to different RATs). This features is enabled when the network adapter within the MD comprises two or more antennas.

4.1.3 Radio Access Technologies

As already stated, a RAT is a standard physical method allowing connectivity between an UE and an AP. Focusing on cellular networks, RAT refers to the various established standards since the early 80s'. From the first-generation (1G) analog systems to the cutting-edge 5G networks, each generation has introduced technical features that have shaped the way we communicate and access data on our mobile devices. Cellular RATs can be divided according the their generation (G), and their evolution is summarized as follows:

1G. It stands for *First Generation*. In the 1980s, 1G signaled the beginning of cellular communication. These early devices were mostly analog and had limited voice communication capability [87]. Frequency Division Multiple Access (FDMA) was the principal multiple access approach, in which each user's calls were assigned a unique frequency. Because the bandwidth allocated to 1G networks was rather narrow, data transfer capabilities were limited, hence 1G was not used for data services bu just for voice communication.

2G. It was a significant advancement in mobile technology. It introduced digital cellular networks, which improved call quality and enabled limited data services. The introduction of digital signals in 2G networks improved voice quality and increased call capacity. As multiple access techniques, Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA) were utilized, allowing many users to share the same frequency spectrum [87]. The ability to send and receive text messages was enabled, and it quickly became a popular feature. While data capacity was restricted, 2G networks enabled services such as mobile web browsing and email, and roaming capabilities allowed users to move across regions and nations while maintaining connectivity.

3G. Mobile networks resulted in significant advances in data speed as well as the introduction of mobile data services. 3G networks provided much higher data rates, allowing for speedier internet access and multimedia services. Universal Mobile Telecommunications System (UMTS) [88] was a well-known 3G technology that featured CDMA-based multiple access and higher data speed, in the context of video calls, mobile TV and video streaming. For efficient data transmission, 3G networks incorporated packet-switched technology.

4G. Networks implementing this technology constituted a paradigm breakthrough in mobile technology, allowing for greater data speeds and ushering in the era of mobile internet and multimedia services. Long–Term Evolution (LTE) is the main 4G technology, providing much faster data speeds than 3G [88]. Orthogonal Frequency Division Multiple Access (OFDMA) was the primary multiple access technique, enabling for more efficient spectrum utilization and increased network capacity. 4G networks reduced latency, allowing real-time applications such as video calls and online gaming to run more smoothly.

5G. Said networks are at the forefront of mobile technology, providing unmatched data rates, minimal latency, and compatibility for upcoming applications. 5G offers much faster data rates, using mmWave frequencies to provide extremely high data speeds reaching multiple gigabits per second.

A technical comparison of the five technologies is carried out in Tab. 4.1.

It appears now clear that a wireless telecommunication link is possible when the three main ingredients presented above are present: (i) a RAT, (ii) an AP and (iii) a MD.

Table 4.1. Characterization of cellular standards from 1G to 5G

Gen.	Year	Multiple Access	Data Rate	Key Features	Applications
1G	1980s	Analog	Low	Voice calls, limited roaming	Basic voice communication
	1990s	Digital			Improved voice quality
2G	1990s	TDMA (GSM)	9.6-14.4 kbps	SMS, data services	Text messaging, early data services
	1990s	CDMA (IS-95)			Improved call capacity, digital transmission
3G	2000s	UMTS (WCDMA)	384 kbps - 2 Mbps	Video calls	Enhanced data services
	2000s	CDMA2000			Faster data, mobile internet
	2000s	TD-SCDMA			Advanced network architecture
	2000s	WiMAX			High-speed wireless broadband
4G	2009	LTE (OFDMA)	100 Mbps - 1 Gbps	Low latency	High-speed mobile internet
	2010s	WiMAX 2			Improved data rates
	2010s	LTE-Advanced			Enhanced network capacity
5G	2019	NR (New Radio)	1 Gbps - 20 Gbps	Low latency, mmWave	Ultra-high data rates
	2019	NR (Sub-6 GHz)			Enhanced mobile broadband
	2019	NR (mmWave)			Low-latency IoT
	2019	NR (mMTC)			Massive IoT connectivity
	2019	NR (URLLC)			Ultra-reliable low-latency communication
	Ongoing	Network Slicing			Customized networks
	Ongoing	Beamforming			Improved coverage and signal quality

4.2 The 5G Technology

Compared to the third and fourth generations of cellular networks that favored the spread of the Internet on mobile devices, 5G represents a certain rather clear discontinuity. There are three evolutionary dimensions that have made this generational transition unique:

- **Enhanced Mobile Broad Band (eMBB)**. It focuses on greater network access speeds up to 10 Gbps. This features enable the possibility to enjoy several application services (not manageable by 3G and 4G networks due to bandwidth limitations), like streaming 4K and 8K videos and games without buffering. Moreover, while eMBB primarily targets enhanced broadband for users, it indirectly benefits IoT applications by providing a robust network foundation for connected devices.
- **Massive Machine–Type Communication (mMTC)**, which addresses the exponential growth of Internet of Things (IoT). Through mMTC, 5G handles the massive number of sensors, smart devices, and industrial applications that are expected to join the IoT ecosystem, reducing their energy consumption and thus extending their battery life. This is particularly useful in a variety of scenarios such as smart cities, agriculture, environmental monitoring, and industrial automation. The 5G standards assesses that a generic 5G BS can handle up to 1 million devices per km^2.
- **Ultra Reliable Low Latency Communication (URLLC)**, intended for use in applications requiring extremely low latency ($\approx 1 - 2$ ms) and high reliability. This use case is critical for critical domain applications such as: (i) autonomous vehicles, improving safety and allowing real–time communication between vehicles and between vehicles and infrastructure [59]; (ii) healthcare, enabling remote surgery and telemedicine [89]; and (iii) industrial automation for human–robot real–time cooperation and predictive maintenance [90].

5G introduces a higher complexity both at infrastructure and protocol level. This complexity of 5G, while on the one hand increases the infrastructural challenges relating to coverage, frequency and connection, on the other opens up applications and business opportunities that are barely imaginable with 4G standards.

In this part of the discussion we focus on the main architectural innovations relating to 5G.

- **High Data–Rates and Low Latency**. The new mobile radio network guarantees greater calculation and data transmission speed as well as lower latency and response times. The increase in the available frequency spectrum

(in Italy it ranges from 700 MHz to 26 GHz), combined with the introduction of dynamic antennas, capable of multiplying the capacity of the system, allows 5G to achieve greater efficiency of the allocated spectrum.

- **Virtualization and Slicing**. 5G can be implemented in specialized logical networks. Network slicing allows you to create multiple virtual networks on the same physical infrastructure by exploiting virtualization techniques of both transmission and calculation resources. This means that a 5G network guarantees the full functioning of multiple applications in parallel, which can be managed by new players.
- **Computing and Mobile Edge Computing**. The fifth generation enables large-scale edge computing with high computational capabilities and low latency in accessing computing resources. This property transforms the 5G network into a truly programmable platform for user applications. The physical positioning of the servers that provide this most interesting computing capacity is towards the periphery, precisely at the edge.
- **Coverage dedicated to high frequencies**. For many B2B 5G applications, high traffic requires *ad hoc* coverage capable of meeting the performance requirements of these applications. In this sense, millimeter wave technologies are of fundamental importance for defining high-capacity coverage such as those in high-traffic urban areas or in indoor contexts.

The 5G technology was launched a few years ago and there are still few users who have MDs capable of supporting this radio technology, just as there are still few areas in which BSs capable of providing connectivity services have been installed. As an example, Fig. 4.1 show a heat map of 5G (purple) vs 4G connectivity (orange, red) in Italy[2]. It can be noted that 5G is present along the highways and in the main cities, while it is completely absent in most rural areas, which are in any case covered by the 4G RAT.

4.3 Virtual and Augmented Reality

5G has the potential to unlock a wide range of applications across various industries, transforming how we interact with technology and enabling innovative solutions. Two significant areas where 5G is expected to have a profound impact are Virtual Reality (VR) and Augmented Reality (AR).

[2]The map refers to October 21, 2023 and the reference network operator is TIM.

Figure 4.1. Heat map of 5G coverage in Italy. Image source: [91]

4.3.1 Virtual Reality History

Virtual Reality is a technology that immerses human users in a computer–generated fully virtual world, effectively replacing their real-world surroundings with a simulated one. The ultimate VR system would make it nearly impossible for users to differentiate the virtual environment from reality. This concept, initially introduced in a science fiction work called *Pygmalion's Spectacles*[3] in the 1930s, envisioned a set of goggles that could engage a user's five senses to offer an experience of fictional realms [92].

In the 1960s, Heilig developed the Sensorama, the first prototype of VR system. It was a multi-modal theater cabinet capable of displaying stereoscopic 3D images, stereo sound, aromas, winds, and vibrations during film presentations [93]. Then, in 1968, Ivan Sutherland presented the first functional see-through head-mounted

[3]Pygmalion is a character from Greek mythology. He was a sculptor and the king of Cyprus, most famous for creating a beautiful statue of a woman, which he carved out of ivory or marble. The myth says he invested so much time and effort into his sculpture that he fell in love with it, so much so Pygmalion prayed to the goddess Aphrodite to bring the statue to life. Aphrodite, moved by his love and dedication, granted his wish: the statue was miraculously transformed into a living woman. The modern notion of VR actually arises from the myth of Pygmalion, who falls in love with a fictitious woman, immersing himself entirely in a parallel world completely detached from the reality of things.

display (HMD) with head-tracking capabilities [94]. This groundbreaking system allowed users to observe computer-generated 3D wireframe objects seamlessly integrated into their real environment, marking the official birth of VR/AR.

Notably, one of the earliest large-scale networked VR systems was SIMNET, a military simulation created for DARPA in 1983 [95]. In the late 1980s, the Naval Postgraduate School introduced NPSNET, a battlefield simulation system capable of accommodating hundreds of users simultaneously [96].

In 1987, Jaron Lanier coined the term 'virtual reality' and founded a company called VPL Research. One of their early products, the EyePhone, considered one of the first commercial HMDs, found its way into numerous research laboratories, contributing to the initial surge of interest in virtual reality in the early 1990s.

Today's booming VR industry is largely attributed to the emergence of cost-effective smartphone-based head-mounted displays (HMDs). One of the pioneering systems was the FOV2GO papercraft HMD, developed at the University of Southern California in 2012. It was from the same research group that the Oculus Rift was launched the same year. According to ABI Research's most updated forecasts [97], the global VR market is valued at approximately $16.7 billion in 2023.

Currently, standard VR systems generate some realistic images, sounds, and other sensations that simulate a user's physical presence in a virtual environment using either VR headsets or multi-projected environments. A person who uses VR equipment can look around the virtual world, move around in it, and interact with virtual features or items. VR headsets with a head-mounted display and a small screen in front of the eyes are commonly used to create the effect, but it can also be achieved through specially designed rooms with multiple large screens.

VR applications include entertainment (immersive gaming) and education (medical, military, and automotive training).

4.3.2 Augmented Reality History

Augmented Reality is a technology that overlays computer-generated information onto the real world, effectively enhancing our surroundings to make tasks easier. Traditionally, the objectives of AR have revolved around task-driven enhancements, presenting the most relevant information at precisely the right time and location. On the contrary, another version of this system seamlessly integrates virtual content within the real world or modifies existing objects in a manner imperceptible to the user. The term *Augmented Reality* was coined by Caudell and Mizell in 1992 for the HMD-based system they developed for wire bundle assembly at Boeing [98].

In 1999, the marker-based open-source tracking library, ARToolKit, was released, further catalyzing the development and dissemination of augmented real-

ity [99]. Later on, in the late 2000s, AR gained public recognition through the entertainment industry with games like *The Eye of Judgment* in 2007, considered the first consumer AR game, and the Nintendo 3DS in 2011. To date, the AR game *Pokémon Go* alone has attracted over 200 million players and generates more than $2 million in daily revenue worldwide. According to Fortune Business Insights [100], the global AR market is projected to grow from $62.75 billion in 2023 to $1,109.71 billion by 2030.

4.3.3 Challenges of AR/VR

From a technical perspective, VR and AR systems share substantial similarities. Both necessitate sensing (input) and display (output) subsystems, as well as a scene management subsystem. Sensing subsystems must track user position (and motion) and accept various command inputs. Display subsystems render a 3D scene based on the user's position and provide sensory information like sound. The hardware and software architecture for both systems are quite alike, if not identical. Minimizing end-to-end latency, from motion to photon, is crucial for both VR and AR, as latency negatively affects user comfort in VR and visual quality in AR, among other adverse effects. Accurate position tracking (or registration in augmented reality) is essential for both systems, but it is often more challenging to achieve for AR due to the mobile nature of its applications. In general, display hardware for AR is more complex to develop than for VR, primarily because of the intricate optics involved in see-through displays and the interaction between virtual content and the real environment. AR systems typically involve drawing supplemental material, making the rendering requirements often higher than in VR, where the entire scene must be drawn every frame, often at higher frame rates.

Notably, VR/AR challenges are related to:

- Latency: achieving low latency is a crucial challenge in AR/VR systems. This is due to the fact that said systems require high computational capabilities in order to process terabytes of data either to reconstruct or to navigate and render models in real–time. Sensor delay is usually imperceptible in modern devices like HoloLens and VR headsets, since they are wired to a graphical workstation like PS5 or similar. However, latency introduced in the wireless network communication domain can affect user experience, especially in gaming and haptic applications [101].
- Bandwidth: bandwidth limitations become significant in applications requiring the transmission of large amounts of data in a small fraction of time, such as 3D reconstruction, telepresence and remote surgery.

- Quality of Service (QoS): maintaining consistent QoS is essential for applications like video conferencing and multiplayer gaming. Frequent disconnects or degraded image quality can deter users from adopting AR and VR technologies.
- Availability: for a widespread adoption, AR and VR systems must be highly available, similar to the ubiquitous access to mobile phones made possible by 3G and beyond networks. Nowadays this is not true, since AR and VR can be implemented only on specific hardware not accessible to everyone.
- Security and Safety: as AR and VR extend beyond in-home use, ensuring security and safety becomes a priority. Protecting private information, setting application privileges, and preventing physical dangers (e.g., distractions) are critical considerations.
- Social Adoption: like any technology, AR and VR adoption will take time and must consider societal rules, safety mechanisms, and devices that intelligently regulate user interactions. Smart networks can also contribute to the safe and practical use of virtual systems.

The challenges and issues presented above can be handled with great performance in wired AR and VR systems, in which users' headset is directly connected to a graphical workstation equipped with a powerful video card. However, this framework does not allow for user movement: people enjoying AR and VR applications are typically forced to remain within a specific room and can move just by few meters. Actually, this limitation may be irrelevant in VR systems, since user can move within the virtual environment even with a joystick, but it becomes crucial in the AR framework, especially when reconstruction tasks are required. If users are not able to freely move in a given area, they cannot capture with the video camera some real images on which to mount virtual objects.

As stated in the previous section, the 5G infrastructure represents a significant advancement over 4G–LTE, offering speeds up to 10 Gbps per device and reducing latency to less than 1 ms. This RAT is then a valid candidate to benefit applications demanding ultra high bandwidth and ultra low latency, like VR and AR.

The exploitation of the 5G technology constitutes the enabler for the introduction of massive and distributed Mobile Augmented Reality (MAR) and Mobile Virtual Reality (MVR) applications, i.e., network system in which the AR technology is implemented in MD capable of connecting to a 5G BS, thus avoiding the traditional wired setting and allowing users to freely move. This new and fascinating technology can revolutionize the way we interact with objects and other people, and has direct implications in various application fields, including:

- **Gaming and Entertainment**: improving network connectivity through 5G will allow real-time personalized streaming and will enhance player localization, interactions, and the sharing of AR content even when users are not at home. Moreover, games involving haptic interfaces will become more advanced, since low-latency haptic devices and controllers will enable realistic tactile feedback with latencies less than 1 ms [101].
- **Vision Augmentation**: beyond adding virtual elements to the real world, vision augmentation seeks to improve or modify human vision using AR technology. This can include altering focus, integrating optical elements, and providing x–ray or predictive vision interfaces [101]. 5G will facilitate instant access to offloaded processing for more effective augmentations and digital assistance, even when the user is moving at high speed by car or train.
- **Cultural Heritage**: MAR through 5G can combine digital information with real-world experiences through MDs, providing tourists and cultural enthusiasts with innovative ways to engage with archaeological sites, museums, and other heritage locations. MAR applications can offer interactive and immersive experiences for visitors. The latter, while freely moving into an archaeological park, can point their UE camera at a historical site, thus reconstructing historic buildings as they appeared at the time they were built, or modified as if they were placed in a dystopian and futuristic context [102]. This activity allows for the creation of virtual storytelling experiences that bring history and culture to life. In this way, users can witness historical events, meet important figures, or explore ancient civilizations through interactive narratives, making learning more engaging and memorable.

4.4 The VADUS Project

The integration of 5G and MAR in the cultural heritage sector is the ambitious goal of one of the research projects the author of this thesis has worked on and contributed to in his whole PhD period.

The project is called Virtual Access and Digitization of Unreachable Sites (VADUS) – grant agreement No 4000132720 – (`https://business.esa.int/projects/vadus`), and has received funding from the European Space Agency (ESA) in relation with the call ARTES 20 by ESA ITT AO/1-10065/19/NL/AF *Applications integrating space asset(s) and 5G networks in L'Aquila, the Abruzzo region, Roma Capitale and Municipality of Torino (L'ART)*.

As already stated in the previous section, AR/VR technologies are powerful tools for developing new services to enhance tourists' experiences when visiting

archaeological sites and monuments. However, currently available services have some drawbacks: users are constrained at a static position and personalization is very limited, whereas the spectacularization of the experience overwhelms the rigor of the scientific basis of the virtually reconstructed or recreated artifacts.

VADUS aims to overcome these constraints by introducing new solutions and digital contents for a fully immersive experience, free of mobility and duration constraints, by providing high-quality 3D models enriched with multimedia, informative, and scientific content. As the name suggests, VADUS refers mainly to indoor environments which are not accessible to the public: this can occur either because the rooms are at risk of collapse (catacombs or similar) or because the mosaics, graffiti, paintings and frescoes present within these environments are particularly fragile to atmospheric and environmental agents.

Three different case studies and pilot sites have been identified in the context of the VADUS project:

1. The **House of Diana**, located in the archaeological park of Ostia Antica, near Rome, Italy.
2. The **Pietro Micca Museum**, located in the city center of Turin, northern Italy.
3. The **House of the Griffins** and the **Isiac Hall**, both located in the Palatine Forum, archaeological park of Coliseum, Rome, Italy.

All these three sites are characterized by indoor environments that nowadays tourists cannot visit.

Regarding the project technical concepts, they comprise elements belonging to very different scientific sectors:

- Mapping of real environments with laser scanning technologies (RGB–ITR and LIF hardware – see Fig. 4.2) to create high-definition 3D models of inaccessible sites, and to analyze the composition of frescos, walls, and artifacts.
- Digitization of the obtained 3D models as a chain of image frames and spatial videos, typical of a VR application. As it can be seen from the example in Fig. 4.3, the 3D images are faithful to the original environment and are characterized by high quality (Full HD 1080p resolution).
- Storage of VR models in a server (back–end) based on the cloud computing protocol, reachable from the user's MD (front–end) via a terrestrial cellular network (for example 4G LTE or 5G).
- Development of a VR application for smartphones, through which users can move around the virtual environment, effectively visiting environments that

(a) *Inaccessible room in the House of Diana.*

(b) *ENEA operators next to a LIF machine.*

Figure 4.2. Images taken during the measurement campaign at the House of Diana. Courtesy of ENEA.

(a) *Original picture.*

(b) *3D reconstructed model.*

Figure 4.3. An example of 3D reconstruction starting from a photo depicting wall frescoes. Photo credits: [103].

are not otherwise accessible. All the graphical computations required to move within the virtual spatial environment are performed remotely by the cloud server, and not locally in the UE.

- Definition of optimal dynamic strategies for choosing the BSs to connect to, and for allocating traffic and computational resources, also taking into account service personalization. The latter allows each user (tourist or professional) to choose from a list of services and select the ones that best match his/her expectations, such as zoom and details selection.

The next chapter will focus on the last topic, which poses an important technical issue about the interdependencies between a VR application and the terrestrial network infrastructure. It will be shown how it is possible to exploit data–driven control methods as a service for bridging the gap between VR/AR and the 5G technology.

Chapter 5

The Multi-RAT Network Selection Problem

In everyday life it is quite trivial to choose the way we connect to terrestrial networks with our MDs. Since they are implemented as multi–mode terminals, it is possible to select just one AP at a time. In this way, usually, users connect to a private Wi–Fi network when they are at home, whereas they connect to mobile networks through their telephone operator when they are outside home, or maybe to the free Wi–Fi network in a city square or airport. However, this standard approach in terrestrial communication is usually not enough to guarantee the bandwidth and throughput requirements imposed by AR and VR applications, especially because it is common that a certain BS is overloaded due to requests from other users[1].

This chapter tackles the technological limitations imposed by the multi–mode framework, considering the exploitation of multi–homed UEs, which can activate more than one connection at a time. The control problem, referred to as multi-RAT network selection problem, consists in determining the best way of splitting users' requests among different communication channels while satisfying the challenging QoS requirements. The problem is tackled through an innovative user–centric AI–based multi–agent traffic steering control framework. The proposed control architecture, developed both for single- and multi-homed devices, is able to dynamically satisfy users' requests by simultaneously exploiting multiple telecommunication channels, even belonging to different RATs.

---

[1]This is a frequent situations in scenarios like concerts or sport events, where there are thousands of people asking connectivity to the very same BS. This happens because each UE connects to the antenna guaranteeing the best Signal–to–Noise Plus Interference Ratio (SINR). In this way, the BS is saturated and MDs are not capable of satisfying not even the mildest services, such as sending a text message.

It will be shown that the multi–agent formulation of the control problem, together with a fast training phase, guarantees the scalability of the proposed traffic steering algorithm. The performance of the proposed solution have been evaluated both in crowded and not crowded cultural sites with respect to several Key Performance Indicators, and computer simulations prove that the proposed approach outperforms other widely adopted connectivity protocols in terms of guaranteed QoS and traffic load distribution.

All material in this chapter refers to [104].

5.1 State of the art

5.1.1 Motivations

As discussed widely in the previous chapter, AR/VR services have become powerful tools in many application fields including industry [105–107], education [108–111], health [112–114] and cultural heritage [115–117]. In the last years, the latter application domain experienced a slowdown mainly due to technological constraints and human factors [118]. AR and VR services, indeed, are challenging applications stressing and often exceeding the capacity of telecommunication networks.

So far, VR and AR technologies have been provided to end-users by means of smartphones, tablets or headsets characterized by limited on-board processors and storage capacities or wired to consoles (such as Personal Computers or PlayStation) which, in this case, shall have wide graphical processing capabilities. This means that, up to now, due to bandwidth and latency limitations, VR and AR can be exploited only when a user is near to a graphic computation source [101]. As a matter of fact, providing a fully virtual tour via a VR/AR visor requires specific and powerful GPUs, that usually cannot be mounted on a headset for users' comfort reasons. In this respect, it has been proven that the 4G-LTE technology does not guarantee lag-free and dropout-free 3D viewing experiences [119].

On the other hand, as specified in the 3GPP 5G standard, 5G communication technologies can arrange up to 10 Gbps data-rate per device which are from 10 to 100 times faster than 4G [120]. Furthermore, the 5G infrastructure can provide a latency less than one millisecond over the radio path and the mid-haul (e.g., Multiaccess Edge Computing - MEC) and back-haul components. These are some of the features rendering 5G the true technological enabler of VR and AR applications [121]. Another key technology bridging 5G and VR/AR services is represented by paradigms such as the edge and cloud computing protocols [122]. Indeed, under said protocols, the 5G infrastructure embeds hardware resources for performance monitoring, network optimization and processing. More in detail, cloud computing

provides groups of high-performance servers allowing end-users' devices to perform heavy computational tasks. Edge computing, instead, allows to perform said heavy tasks to high-performance resources located at the border of the network thus reducing latency. Another relevant peculiarity of 5G technologies for the provisioning of VR and AR applications is represented by the possibility of seamlessly integrating multiple RATs. In this case, the telecommunication network is referred to as heterogeneous network and the algorithms allowing to steer UEs traffic over multiple RATs are referred to as multi-connectivity algorithms. In other words, user devices, which in this respect are usually called multi-homed devices, can exploit the resources of multiple APs, even belonging to different RATs [123, 124]. This means that, in this scenario, a given UE is able to access the (heterogeneous) telecommunication network through multiple points.

Before reviewing the literature, it is worth highlighting the different problems that arise in the considered context from the telecommunication point of view:

- User Assignment (UA): the problem consists in understanding the best AP or set of APs a given user should be connected to.
- Resource Allocation (RA): it consists in determining the best way of allocating wireless resources (e.g., transmitting power, channels, ...) in presence of bandwidth limitations.
- Development of multi-RAT algorithms: it refers to the definition of control algorithms able to steer traffic considering different typologies of RATs.
- Development of multi-connectivity algorithms: it deals with the design of traffic steering algorithms in which a given UE can be simultaneously served by multiple APs belonging to different RATs.
- Task offloading: it refers to the definition of control algorithms allowing to exploit computing resources at the edge of the network to minimize latency and to optimize users' experiences.

In this respect, this chapter is devoted to the design of multi-connectivity algorithms to fairly distribute users' requests over the available APs which may belong to different RATs. Hence, all the above-mentioned problems will be addressed.

5.1.2 Related works

Previous works (e.g., [122,125]) have shown that edge computing frameworks, more than the cloud-based ones, allow to satisfy strong requirements in terms of time delay and video quality constraints. It has been proved that placing graphical resources at the edge in standard 2D mobile gaming applications improves by 20% the

users' perceived QoE [126]. In the Extended Reality (XR) domain, edge computing allows to handle archaeology-based AR/VR applications which are characterized by 360° 3D models with huge dimensions in terms of Gigabytes. This is because such 3D models can be stored on remote edge/cloud servers which, in turn, allow to adopt less powerful wearable devices in terms of computational power (translating in cheaper devices with longer battery life). In this respect, several works addressed the traffic offloading problem for a full exploitation of edge computing resources [127–131]. Furthermore, due to their ultra-low latency, such contents can be elaborated in the GPUs of edge servers and then accessed in real-time by end-users guaranteeing acceptable levels of the perceived QoS, referred to as Quality of Experience (QoE).

For a full exploitation of multiple RATs it is necessary to solve the so-called multi-RAT assignment problem consisting in the selection of the most appropriate RATs able to satisfy given QoS or QoE constraints. The problem, which was already present and studied for older generation networks [132], like 3G UMTS, has assumed more and more importance over the years, thanks to the widespread diffusion of 4G-LTE and 5G-NR wireless base stations. There are many criteria that can be used to perform said selection, such as the avoidance of APs' congestions, the reduction of the latency experienced by users and so on.

Several model-based and data driven solutions have been proposed in the literature for solving the multi-RAT assignment problem. As an example of model-based techniques, in [133] the authors solve the multi-RAT assignment problem by means of a dynamic game-theoretic approach. Similarly, in [134], the authors present an algorithm based on Wardrop's equilibria for adversarial routing. In this framework each UE is considered as a player of a game demanding for network resources. Each player (i.e., each UE) aims at minimizing a selfish objective function which depends on the other players' actions. In [135], it is presented an approach based on MPC, which uses a model of the telecommunication network; the authors defined a global objective function to be minimized aiming at reducing the overall connection energy and usage costs. Under specific conditions, these approaches are able to provide optimal solutions to the network selection problem. However, they are prone to scalability issues when the dimension of the problem significantly increases.

On the other hand, ML–based approaches are particularly suited for solving the multi-RAT network assignment problem, as they allow to handle highly stochastic dynamical processes such as those represented by heterogeneous networks. It is indeed almost impossible to estimate in advance how many users will request the service, their positions and velocities, as well as disturbance that may affect in a negative way the signal power between a generic UE and the APs to which it is

connected. ML approaches exploit data observed from the environment and do not estimate the system model, as it happens in MPC or in Optimal Control problems.

In the last years, RL and DRL techniques have been successfully applied to multi-RAT systems. The data-driven techniques proposed in the literature are either centralized or distributed. In the first framework, there is a unique controller that steers the traffic for all users and access points, whereas in the distributed case each user (or access point) has its own local control unit to make decisions and act. In the latter case, each user competes for getting the requested data-rate from the available base-stations. As an example of this kind of techniques, in [136] the authors solve the QoS management problem in heterogeneous communication networks relying on a hierarchical control architecture based on RL. In [137], the network selection problem is tackled by means of an approach based on Markov Games and friend-or-foe RL. In [138] the authors adopted DRL techniques for creating a multi-connectivity system referred to as DeepRAT. In [139], the authors combine Genetic Algorithms (GAs) and Artificial Neural Networks (ANNs) to derive optimal strategies to solve the multi-RAT network assignment problem, considering just multi–mode terminals.

To address scalability issues, several works proposed decentralized and multi-agent control frameworks. As an example, in [140], the authors define a multi-agent DRL framework and use a Dueling Double Deep Q-Network (D3QN) to learn nearly optimal policies. A multi-agent DRL framework has also been proposed in [141] to address the energy-efficient task offloading problem. In this case, the authors adopted the DDPG algorithm to learn the optimal policy. In [142], the authors propose a multi-agent Q-Learning algorithm to tackle the resource assignment problem in Multi-User Multiple-Input Multiple-Output (MU-MIMO) systems. In [143], the authors proposed a distributed multi-agent RL algorithm to solve the multi-RAT access problem. The proposed user-centric solution further reduces the complexity of the algorithm by means of Nash Q-Learning allowing to reduce the dimension of the strategy space in the learning process. In [144], a distributed multi-agent RL control framework has been proposed for power control in heterogeneous wireless networks.

The development of multi-RAT systems (and related control algorithms) represents a fruitful research topic. However, some issues have not been tackled yet. To the best of the authors' knowledge, in the literature there are no studies addressing the integration of multi-RAT assignment algorithms considering VR/AR applications running on multi-homed terminals [145].

The research gaps highlighted in this section are summarized in Table 5.1: the features used to characterize the reviewed articles are:

Table 5.1. Research gaps in the literature

Reference	F1	F2	F3	F4	F5	F6	F7
[116]	✗	✗	✗	✓	✗	✗	✓
[123]	✓	✗	✗	✗	✗	✓	✗
[125]	✗	✗	✗	✓	✗	✓	✗
[126]	✗	✓	✓	✗	✗	✗	✓
[127]	✗	✗	✗	✓	✓	✓	✗
[128]	✗	✓	✓	✗	✓	✓	✗
[129]	✗	✓	✗	✗	✓	✗	✗
[130]	✗	✓	✓	✗	✗	✗	✓
[131]	✗	✓	✓	✗	✗	✓	✗
[132]	✗	✗	✗	✗	✗	✗	✓
[133]	✗	✗	✗	✗	✗	✗	✓
[134]	✓	✓	✗	✗	✗	✗	✗
[135]	✓	✓	✗	✗	✓	✗	✗
[136]	✓	✓	✓	✗	✗	✗	✓
[137]	✗	✗	✗	✗	✗	✗	✓
[138]	✗	✓	✓	✗	✓	✗	✗
[140]	✗	✓	✓	✗	✓	✗	✗
[141]	✗	✓	✗	✗	✗	✗	✗
[139]	✗	✗	✓	✗	✓	✗	✗
[142]	✗	✓	✓	✗	✗	✗	✗
[143]	✗	✗	✓	✗	✗	✗	✓
[144]	✓	✓	✓	✗	✓	✗	✓
[145]	✗	✓	✓	✗	✗	✗	✗

- F1: consideration of multi–connectivity algorithms by means of which each UE is considered to be a multi–homed terminal.
- F2: adoption of a scalable control logic with respect to the UEs' number.
- F3: model-free nature of the adopted control framework.
- F4: consideration of AR/VR streaming services.
- F5: development of energy-aware control algorithms.
- F6: consideration of moving UEs.
- F7: user-centric nature of the proposed solution.

5.1.3 Contributions

The main features of the proposed control framework are as follows:

- the ability to simultaneously exploit multiple RATs for matching challenging QoS constraints;

- model-free, user-centric multi-agent formulation of the network selection problem allowing to tackle scalability issues;
- adaptive bitrate assignment to reduce the transmission power required for streaming services;
- particularization of the considered control problem in the context of AR and VR services for cultural heritage applications.

The proposed control framework can be used in different ways. From the one hand, cultural sites' operators (or city managers) can understand how to dimension the telecommunication infrastructure to support AR and VR services based on forecast operational scenarios. On the other hand, the proposed solution can be deployed to optimize the actual fruition of heavy streaming services. Being the considered connection downlink only (i.e., users do not send data to servers through APs), the task offloading problem does not apply to the issue tackled hereby.

In what follows, the control problem discussed above will be stated and modeled formally, with the description of the proposed MARL formulation. Eventually, extensive simulations will validate the effectiveness of the proposed approach with respect to system's KPIs.

5.2 System Model and Problem Formulation

The considered multi-RAT network selection problem, depicted in Figure 5.1, consists in understanding how to assign connectivity requests of a set users to a given set of APs. As already mentioned, this work considers multi–homed user terminals and heterogeneous wireless networks, in which the APs belong to different RATs. More in detail, let N be the number of UEs requiring connectivity services from a set of M APs. The APs coverage area and the set of users define the so-called *connectivity environment* object of the network selection problem.

As depicted in Figure 5.2, at each discrete time instant k, the i-th user requires a given data-rate $w_i(k)$ (expressed in bit/s) for a specific service. User requests, in terms of data-rate, have been modeled as square-wave signals with random duty period and different amplitude per each user. High values represent heavier services, like the ones required for mobile AR/VR. UEs are characterized in terms of their buffers $q_i(k)$ capturing the amount of traffic requested by the i-th user but not satisfied. The data-rate assigned by the j-th AP to the i-th UE is denoted with $u_j^i(k)$ and represents the control variable in the considered control problem. The load of the j-th AP is denoted with $l_j(k)$ and is expressed as the percentage of allocated Physical Resource Bloks (PRBs).

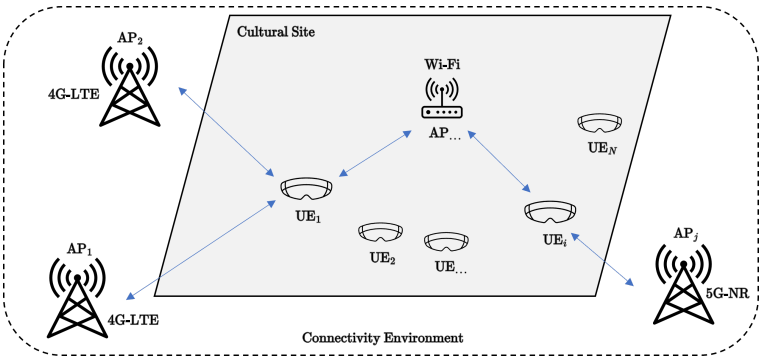


Figure 5.1. Multi-RAT connectivity scenario for cultural sites.

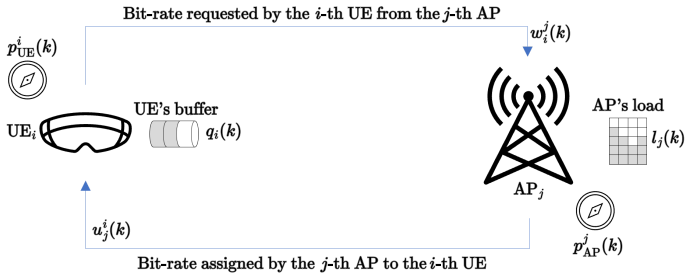


Figure 5.2. Interaction between the i-th UE and the j-th AP.

The buffer dynamics of the i-th UE can be described by

$$q_i(k+1) = q_i(k) + T_s \left(w_i(k) - \sum_{j=1}^{M} u_i^j(k) \right), \forall i = 1, \ldots, N, \tag{5.1}$$

where T_s is the sampling time.

The considered communication type is downlink only, meaning that all the users require streaming services and not server uploading services. Users are assumed to be able to freely move in a 2D environment, according to the law

$$p^i_{\text{UE}}(k+1) = p^i_{\text{UE}}(k) + T_s v^i_{\text{UE}}(k), \forall i = 1, \ldots, N, \tag{5.2}$$

where $p^i_{\text{UE}}(\cdot) = [x^i_{\text{UE}} \quad y^i_{\text{UE}}]^\top$ and $v^i_{\text{UE}}(\cdot) = [v^i_{\text{UE},x} \quad v^i_{\text{UE},y}]^\top$ are the position and velocity of the i-th UE, respectively. Furthermore, their initial position p^i_{UE} is randomly chosen at the beginning of each training episode, thus allowing to learn the optimal policy in any scenario. The position p^j_{AP} of APs is considered fixed, but application scenarios in which connectivity is provided by moving UAVs may be easily addressed by considering the following law

$$p^j_{\text{AP}}(k+1) = p^j_{\text{AP}}(k) + T_s v^j_{\text{AP}}(k), \forall j = 1, \ldots, M, \tag{5.3}$$

Table 5.2. Multi–connectivity nomenclature

Symbol	Description
$\mathcal{A}_i$	Action space of the i-th UE
d_i^j	Distance between the i-th UE and the j-th AP
h	Number of quantized levels used for UEs' queues
i	Index used to refer to UEs
j	Index used to refer to APs
k	Generic discrete time instant
$l_j(k)$	Load of the j-th AP at time k
M	Number of APs
N	Number of UEs
$p^i_{\text{AP}}(k)$	Position at time k of the j-th AP
$p^i_{\text{UE}}(k)$	Position at time k of the i-th UE
$q_i(k)$	Queue of the i-th UE at time k
$\mathcal{S}_i$	State space of the i-th UE
T_s	Sampling time
$u^i_j(k)$	Bitrate assigned at time k by the j-th AP to the i-th UE
$v^i_{\text{AP}}(k)$	Velocity at time k of the j-th AP
$v^i_{\text{UE}}(k)$	Velocity at time k of the i-th UE
$w_i(k)$	Bitrate requested at time k by the i-th UE
$w^j_i(k)$	Bitrate requested at time k by the i-th UE to the j-th AP
z	Number of quantized levels used for APs' loads

where $p^j_{\text{AP}}(\cdot) = [x^j_{\text{AP}} \quad y^j_{\text{AP}}]^\top$ and $v^j_{\text{AP}}(\cdot) = [v^j_{\text{AP},x} \quad v^j_{\text{AP},y}]^\top$ are the position and velocity of the j-th AP, respectively.

The power of the signal between UEs and APs is computed using the Signal to Noise plus interference Ratio (SINR) [146] and a Free-Space Path Loss (FSPL) model [147].

5.3 Multi-Agent Reinforcement Learning Formulation

The user-centric network selection problem can be modeled exploiting the MDP and the MARL framework, both presented in the first part of this essay.

In the hereby proposed multi–agent formulation, the state $s_i \in \mathcal{S}_i$ of the i-th UE is the $(M+1)$-dimensional vector defined as

$$s_i = [\tilde{q_i} \quad \tilde{l_1} \quad ... \quad \tilde{l_M}], \tag{5.4}$$

where $\tilde{q_i}$ and $\tilde{l_j}$ (with $j = 1, \ldots, M$) are discrete levels used to capture the UEs' queue levels and the APs' loads, respectively. Said discrete levels are defined as follows:

$$\tilde{q_i} = \begin{cases} 0 & \text{if } q_i < \bar{q}_0 \\ 1 & \text{if } \bar{q}_0 \leq q_i < \bar{q}_1 \\ 2 & \text{otherwise} \end{cases} \quad \forall i = 1, \ldots, N, \tag{5.5}$$

$$\tilde{l_j} = \begin{cases} 0 & \text{if } l_j < \bar{l}_0 \\ 1 & \text{if } \bar{l}_0 \leq l_j < \bar{l}_1 \\ 2 & \text{if } \bar{l}_1 \leq l_j < \bar{l}_2 \\ 3 & \text{otherwise} \end{cases} \quad \forall j = 1, \ldots, M. \tag{5.6}$$

Let h and z be the number of the quantized levels chosen for the UEs' queues and APs' loads, respectively. The cardinality of $\mathcal{S}_i$ is then

$$|\mathcal{S}_i| = hz^M, \quad \forall i = 1, \ldots, N. \tag{5.7}$$

In the considered network selection problem, UEs are multi-homed terminals which can simultaneously connect to one or more APs at any given time. Hence, the i-th UE control action $a_i \in \mathcal{A}_i$ can be defined as the M-dimensional vector

$$a_i = [p_1 \quad \cdots \quad p_M], \tag{5.8}$$

where the Boolean entries $p_j \in \{0; 1\}$ specify if the i-th UE requests connectivity services to the j-th AP (i.e., if $p_j = 1$) or not (i.e., $p_j = 0$). It follows the the dimension of the i-th UE action space is

$$|\mathcal{A}_i| = 2^M, \quad \forall i = 1, \ldots, N. \tag{5.9}$$

The reward function $R_i(\cdot)$ per each user i is defined as follows:

$$R_i(\cdot) = \begin{cases} +1 & \text{if } |a_i| > 0 \wedge (q_i > 0 \text{ or } w_i > 0) \\ -1 & \text{otherwise} \end{cases}, \tag{5.10}$$

where $|a_i|$ is the L^1-norm of a_i capturing the number of APs the i-th UE is trying to connect to. Note that, with this modeling choice, the agents receive a positive reward when they request connectivity services (i.e., when $|a_i| > 0$) and they are requesting traffic (i.e., $w_i > 0$) and/or their queue is not empty (i.e., if $q_i > 0$).

Every time that a user requests connection to more than one AP, the request w_i is split among the base stations taking into account the relative distance between

the UE and AP. In particular, the formula for the generic w_i^j can be easily obtained as follows

$$w_i^j = w_i \frac{1 - \frac{d_i^j}{\sum_j d_i^j}}{|a_i| - 1}, \tag{5.11}$$

where d_i^j is the Euclidean distance between the i-th UE and the j-th AP. This modeling choice allows to reduce the energy required for the streaming services, since the UEs do not waste a high amount of power for sending traffic to remote APs.

Let $\hat{u}_i$ be the i–th user data–traffic request. This control signal follows a discrete–time PID control rule, which aims at discharging the user's queue [148]:

$$\hat{u}_i(k) = K_P e(k) + K_I \sum_{h=0}^{k} e(h) + K_D[e(k) - e(k-1)], \tag{5.12}$$

where $e(k) = q(k) - q^{\text{des}}(k) = q(k)$ is the error and K_P, K_I and K_D are the PID gains. It is worth noting that each user is free to tune its own PID gains, implementing mild or urgent control actions.

Moreover, let $\tilde{u}_i^j$ be the maximum amount of data–rate that the j–th AP can afford to the i–th user. Then, the real traffic data received by the user is given by

$$u_i^j = \min\{\hat{u}_i^j, \tilde{u}_i^j\}. \tag{5.13}$$

If $u_j^i = w_i^j$ at each time step, then the i-th user data-rate request is perfectly satisfied by the j-th access point. Each episode (or communication round) terminates after T steps, i.e., the amount of time after which no UE requests communication services.

Note that, thanks to the modeling choice reported in (5.5)–(5.6), the dimensions of the agents' state spaces $\mathcal{S}_i$ and action spaces $\mathcal{A}_i$ are limited and, more specifically, they do not grow exponentially with the number of UEs. This, in turn, means that it is not mandatory to rely on DRL techniques to approximate the action-value functions. Furthermore, as already mentioned, under specific conditions, the Q-Learning algorithm is guaranteed to stochastically converge to the optimal policy. This is not the case with DRL approaches for which convergence to the optimal solution can be proved only from an empirical point of view.

For these reasons, a variant of the Q–Learning algorithm 3 has been chosen, adapting it to the multi–agent scenario. The proposed Multi–Agent Q–Learning algorithm is presented in Algorithm 8.

Algorithm 8 Multi-Agent User-Association Q-Learning

Inputs: learning rate $\alpha \in [0, 1)$; discount rate $\gamma \in [0; 1]$; small $\varepsilon > 0$
Output: $Q_i(s, a)\, \forall i$
Define the number of training episodes E
Initialize $Q_i(s, a), \quad \forall s, \quad \forall a, \quad \forall i$
for ep $= 1, \ldots, E$ **do**
  reset $s_i, \quad \forall i = 1, \ldots, N$
  for $t = 1, \ldots, T$ **do**
    for each user $i = 1, \ldots, N$ **do**
      get observation s_i
      choose action a_i following ε-greedy policy
      based on Equation (5.11), compute $w_i^j(k), \forall j$
      perform action a_i considering the computed w_i^j and observe the next state s_i' and the obtained reward r_i'
      perform Q-Learning update rule over $Q_i(s, a)$
      $s_i \leftarrow s_i'$
    end for
  end for
  update α
  update ε
end for

Table 5.3. APs Features

AP	μ	Frequency	Bandwidth	Power	Max Data–Rate	Position
AP_1	1	800 MHz	20 MHz	20 W	1000 Mbit s^{-1}	$[200 \quad 800]^\top$ m
AP_2	1	1700 MHz	40 MHz	20 W	1000 Mbit s^{-1}	$[500 \quad 100]^\top$ m
AP_3	1	1900 MHz	40 MHz	20 W	1000 Mbit s^{-1}	$[800 \quad 800]^\top$ m

5.4 Simulations and Results

5.4.1 Simulations' environment, parameters and KPIs

The simulations were carried out on a computer equipped with an Intel Core i5-10210U quad-core CPU @ 1.60 GHz and 16 GB of RAM and exploited the Network Simulator Environment described in [149]. In such environment, each user is able to activate and deactivate connections with the available APs. As detailed in the next sub-sections, the proposed control framework has been tested in two different scenarios: in the first one, a non–crowded area with few tourists has been considered, whereas the second one presents a more crowded area. In both scenarios user can have at their disposal $M = 3$ BSs implementing the 5G-NR technology, whose technical features (like numerology μ, frequency and transmission power) are reported in Table 5.3.

The thresholds' values $\bar{q}_0, \bar{q}_1$ and $\bar{l}_0, \bar{l}_1, \bar{l}_2$ (see (5.5)–(5.6)) used to define discrete levels to define the values of UEs' queues q_i and APs' loads l_j, respectively, have been defined as follows:

$$\begin{aligned} q_\tau &= [0.4 \quad 0.7]^\top \\ l_\tau &= [0.2 \quad 0.5 \quad 0.8]^\top. \end{aligned} \tag{5.14}$$

The Q-Learning training parameters have been chosen in the following way. The number of training episodes has been set to $E = 20000$; the initial learning rate α_0 is equal to 1 and evolves according to the following decay law

$$\alpha(e) = \frac{1}{0.05e + 1}, \tag{5.15}$$

where e is the generic episode; the initial ε-greedy policy parameter is equal to 1 and decays following the law:

$$\varepsilon(e) = \exp\left(\frac{-e}{0.2E}\right). \tag{5.16}$$

Eventually, the discount rate γ is set to 0.9 and the initial action-value function $Q_i(s, a)$, is the zero matrix $\forall i$.

Furthermore, a loss function per each user has been defined as

$$e_i(k) = (Q_i^{\text{old}}(s, a) - Q_i^{\text{new}}(s, a))^2, \tag{5.17}$$

in order to monitor how much time does the training process take to maximize the reward. At the beginning of each episode the UEs' initial positions in the environment are chosen randomly, in such a way as to allow for the widest generalization possible. Users are assumed to freely move in a squared area with sides 1 km.

The proposed Multi-RAT approach has been compared with two other connectivity protocols:

1. Max–SINR. It is the standard approach for commercial user devices [150]. With this protocol the device is multi–mode and it can establish connection with just one AP at a time. The latter is chosen as the one guaranteeing the highest SINR.

2. Single–RAT Q–Learning. In this case, an intelligent agent is trained for solving the network selection problem, enabling only one connection at a time.

The obtained results have been evaluated also through the definition of five KPIs, defined as follows:

- KPI 1: percentage of time instants in which users experience queues (the lower, the better).
- KPI 2: percentage of users who never experience queues (the higher, the better).
- KPI 3: percentage of users who experience saturated queues (the lower, the better).
- KPI 4: percentage of time instants in which there are saturated queues (the lower, the better).
- KPI 5: base stations' load distribution fairness, computed as the average standard deviation of APs levels (the lower, the better). Supposing to have M APs, each one with load $l_j(k)$ at time k, and control horizon K_f, the load balancing metric is computed as

$$\text{KPI } 5 = \frac{\sum_{k=0}^{K_f-1} \sigma_{\text{AP}}(k)}{K_f}, \tag{5.18}$$

where

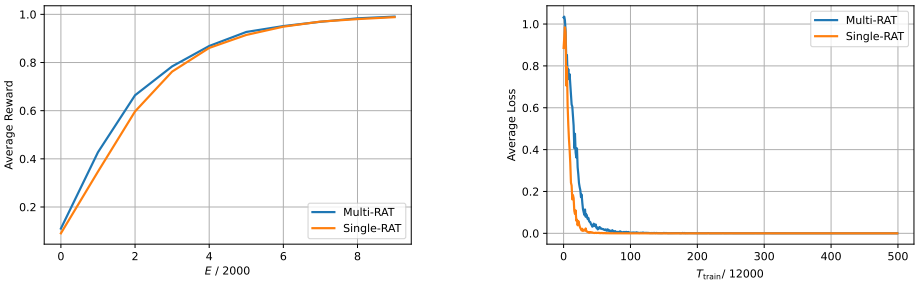
$$\sigma_{\text{AP}}(k) = \sqrt{\frac{\sum_{j=1}^{M} (l_j(k) - l_\mu(k))^2}{M}}, \tag{5.19}$$

with $l_\mu(k)$ being the average AP load at time k.

The remainder of this section is organized as follows. Section 5.4.2 focuses on the algorithm's training phase and shows how the proposed algorithm converges. In Section 5.4.3, the learned policy is deployed in a simulation environment characterized by few users and is aimed at comparing the performances of different algorithms when telecommunication resources are not scarce. Eventually, Section 5.4.4 tests the same learned policy in a crowded scenario in which telecommunication resources are not enough to match all users' connectivity requests.

5.4.2 Training phase

To deal with scalability issues, the training of the Single- and Multi-RAT intelligent agents has been performed considering a single agent (i.e., a single user). To take into account the multi-agent nature of the environment in which the trained agents will be deployed (in other words, to consider the mutual impact between users), several environment's parameters have been randomized in different steps of the training algorithm. More in detail, at the beginning of each training episode, a



(a) *Averaged reward function.* **(b)** *Averaged loss function.*

Figure 5.3. Averaged reward and loss during the training phase.

random position is assigned to the agent to be trained and, in addition, during the episode, it moves with a random speed.

Furthermore, during the episode, before the agent selects an action (i.e., before it decides to which APs the connectivity requests should be send), the APs' loads are also randomized, in order to simulate the impact of others external users on the telecommunication network. The learned policy is then assigned to the N agents (i.e., to the N UEs). Each episode lasts 30 seconds, with sampling time $T_s = 0.1\,\mathrm{s}$, thus leading to $T_{\mathrm{train}} = 6 \times 10^6$ training steps.

This training strategy is justified by the findings of other works such as [151] and [152]. With respect to these works, the considered application domain does not suffer from the challenges outlined by the authors posed by adopting homogeneous policies in multi-agent environments. As pointed out by the authors, adding small noises in the action and/or state space would allow to overcome such issue. In this work, it has been assumed that all users have the same requirements in terms of QoS constraints. Future works may consider different QoS requirements for different classes of users. In this case, an homogeneous policy shall be learned for each one of said users' classes. Using this approach, it is possible to realize a fast and agile training phase, which lasts only around 17 s both for the Multi–RAT and the Single–RAT agent.

Fig. 5.3 displays the reward averaged over 2000 episodes and the loss averaged over 12000 training steps, respectively. Both agents are able to learn a policy which maximizes the reward within the chosen number of episodes. It is worth noticing that since both loss functions converge after few steps, it is possible to further improve the training phase, e.g., by reducing the number of episodes or letting ϵ converge more rapidly to zero.

5.4.3 Non–overcrowded Scenario

In the first simulation, the performances of the proposed algorithms are evaluated in not overcrowded scenario envisaging $N = 9$ users. The simulation emulates a virtual tour lasting 30 min with sampling time equal to the one used in the training phase. The computational time to perform the test is around 1.22 s per each user, meaning that the network control paradigm could be implemented with negligible delay also in applications that require $T_s < 1$ min.

Figure 5.4 and Table 5.4 report the loads on the three APs and the values of the considered KPIs, respectively. As expected, in a scenario in which telecommunication resources are not scarce, all three algorithms are able to match users requests. However, from the values of KPI 5 (capturing the APs' load distribution fairness), it can be observed that the Multi-RAT Q-Learning algorithm is able to distribute loads more fairly (by one order of magnitude) than the other algorithms. The Single-RAT Q-Learning algorithm, on the other hand, has performances sligthly worse than the Max-SINR algorithm. This could be solved by tuning the rewards obtained by the Single-RAT intelligent agent. In general terms, it can be said that in absence of scarcity of resources, advanced user association techniques do not provide significant improvements besides a better usage of network resources (which, from the network operators' point of view, may represent a relevant aspect).

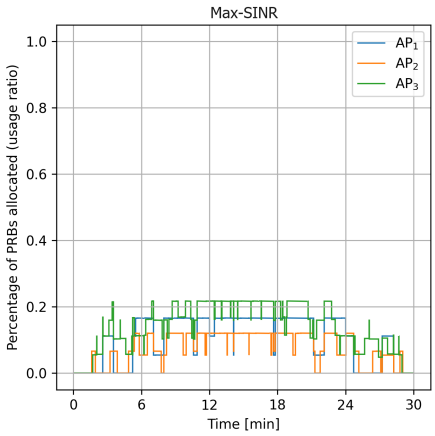
Table 5.4. Non–overcrowded scenario simulation results

Connectivity Protocol	KPI 1	KPI 2	KPI 3	KPI 4	KPI 5
Multi–RAT	0%	100%	0%	0%	0.001
Single–RAT	0%	100%	0%	0%	0.07
Max–SINR	0%	100%	0%	0%	0.03

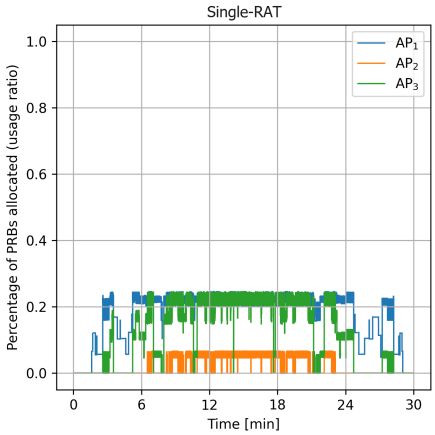
5.4.4 Crowded Scenario

In the second simulation, the performances of the proposed algorithms are evaluated in an overcrowded scenario with $N = 45$ users. Since the amplitude of user requests and the environment are the same as in the first simulation, in this case the scenario emulates a narrow crowded zone (like a real archaeological park) with many users requesting high quality VR/AR streaming services. As per computational times, the same considerations carried out in the first simulation apply.

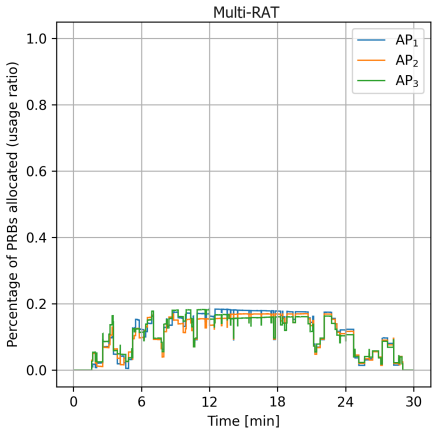
The loads on the three APs are shown in Figure 5.5. It can be noted that the Max–SINR approach causes high occupancy levels on AP_2 and AP_3, with an inefficient usage of the first AP. On the other hand, the Multi–RAT approach allows to achieve better load balancing on the APs with respect to both the other



(a)



(b)



(c)

Figure 5.4. Loads on the three BSs, non–overcrowded scenario.

Table 5.5. Crowded scenario simulation results

Connectivity Protocol	KPI 1	KPI 2	KPI 3	KPI 4	KPI 5
Multi–RAT	1.48%	74.67%	0%	0%	0.001
Single–RAT	26%	0%	88%	1.28%	0.05
Max–SINR	25.33%	42%	54.67%	4.9%	0.2

approaches. In particular, with respect to the Single–RAT algorithm, with the Multi-RAT algorithm the maximum burden on the APs lasts less.

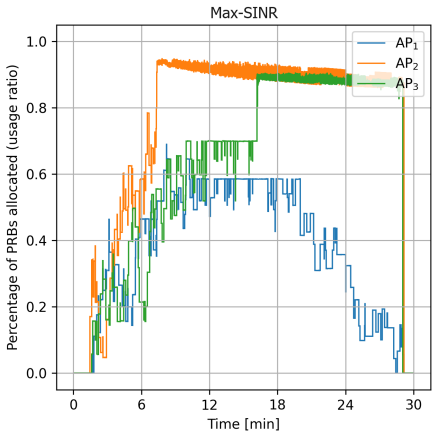
Table 5.5 shows the values of the five considered KPIs. By analysing such data, it is clear that the Multi-RAT Q-Learning algorithm outperforms the others techniques with respect to all the considered KPIs. In particular, it can be noted that such algorithm allows to avoid saturated queues for all the duration of the visit. For what concerns the Single-RAT Q-Learning algorithm, it can be seen that it performs better with respect to the Max-SINR algorithm in terms of load fairness (KPI 5: 0.05 vs 0.2) and in terms of percentage of time instants in which there are saturated queues (KPI 4: 1.28% vs 4.9%). This latter aspect is directly related to lags experienced by users during virtual tours.

Concerning the first three KPIs, the Max-SINR algorithm seems to perform better than the Single-RAT algorithm. Indeed, the Max-SINR algorithm shows better performances with respect to the percentages of time instants in which users experience queues (KPI 1: 25.33% vs 26%) and of users who experience saturated queues (KPI 3: 54.67% vs 88%). Furthermore, the Max-SINR algorithm outperforms the Single-RAT algorithm in terms of the number of users who never experience queues (KPI 2: 42% vs 0%).

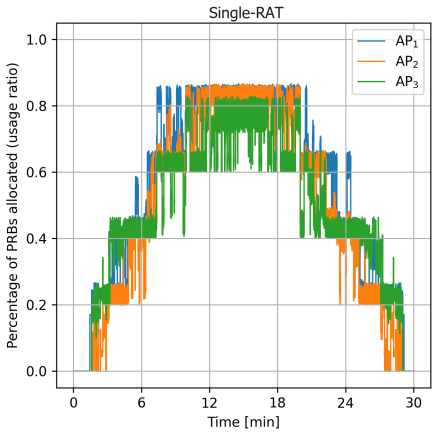
From this analysis, it emerges that the Single-RAT tries to distribute the consequences of a lack of resources among all the users, whereas with the Max-SINR algorithm the number of users who never experience lags is higher. More specifically, the Single-RAT algorithm minimizes the number of time instants in which there are saturated queues at the price of having more users experiencing queues. Hence, with respect to the Max-SINR algorithm, the Single-RAT is more fair not only in the AP load distribution (see KPI 5) but also regarding the resources assigned to the users.

5.5 Discussion and Future Works

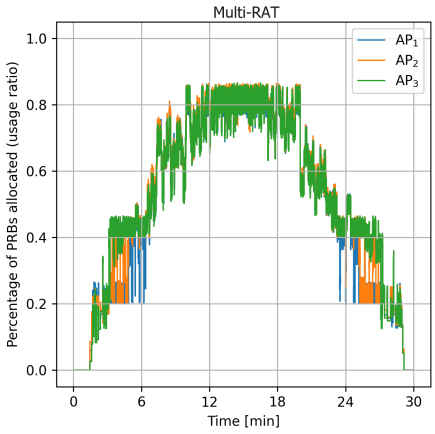
In the framework of applications for the enhancement of cultural heritage, this work tackled the problem of guaranteeing smooth AR and VR tours in urban and remote cultural sites. The availability of such services allows to provide virtual access to



(a)



(b)



(c)

Figure 5.5. Loads on the three BSs, crowded scenario.

areas not open to the public and to guarantee the accessibility to cultural sites to people with mobility difficulties. This work was aimed at supporting user-centric virtual tours removing constraints regarding the computational power of the smart devices used for visualizing AR and VR contents and the need of using wired devices while guaranteeing users freedom of movement and choice of contents.

The proposed solution is based on MARL and exploits a Q-Learning algorithm to learn the optimal policies. Said solution has been developed both for single- and for multi-homed devices. The latter leverages on an important feature of the 5G allowing to simultaneously exploiting multiple RATs. The adopted training strategy proved to be very fast (less than 20s) allowing to converge to the maximum obtainable reward. The performances of the Multi-RAT and Single-RAT Q-Learning algorithms have been compared with the ones of the Max-SINR algorithm, which is a widely adopted connectivity protocol. While the performances of the Multi-RAT algorithm always outperformed the other two algorithms, the Single-RAT and the Max-SINR algorithms showed behaviors requiring a discussion. Indeed, simulations show that the Single-RAT algorithm allows to minimize the global service discomfort whereas the Max-SINR algorithm favors those users with better SINR values. The proposed Q-Learning strategies allow to achieve more fair performances both from the network operators point of view (in terms of APs' loads distribution) and from the users' point of view (distributing drops in performances among all the users).

Future research in this area could expand on the proposed solution in the following directions:

- Algorithmic Enhancement: to reduce the risk of overwhelming APs, one can customize the allocation of data based on the unique characteristics of each AP. This means limiting data requests to less advanced RATs. Additionally, incorporating DNNs to approximate the action-value function in Q-Learning can enhance scalability, but caution is needed since it may not guarantee optimal resource allocation, and fine-tuning of hyper-parameters can affect result reproducibility.
- Modeling Enhancement: considering scenarios where UEs can send data to APs and integrating computing resources for traffic offloading can lead to more efficient control algorithms tailored to specific use cases.

Chapter 6

Power and Resolution Control in Mobile Augmented Reality Applications

The previous chapter focused its attention to the issue of network selection considering just the downlink phase of the communication, i.e. the phase in which the MD gets the data from a cloud or edge server via a set of heterogeneous BSs which shall provide connectivity services.

However, in applications like MAR, it is crucial to consider performance also during the uplink phase, i.e. when the UE sends data to a server via the BS. Since most of the devices cannot perform directly heavy graphical computations like 3D reconstruction or virtual object placement, they shall rely on a remote machine for processing images and videos. This process is called *offloading*.

This task can be performed using three paradigms, which differentiate themselves with respect to the location where data processing occurs:

1. **Cloud Computing**. It entails storing, managing, and processing data at remote data centers, usually located far away from the final users (in the order of hundreds of kilometers). It offers on-demand access to a variety of computing resources accessible through an internet connection. This approach is highly scalable, allowing businesses to easily expand their resources as needed without purchasing and deploying complex hardware locally. Furthermore, cloud providers typically provide high levels of dependability and uptime. The main disadvantage is that it may introduce latency, which can be problematic for real-time applications. Furthermore, because cloud services rely on an internet connection, they can be disrupted if there are network issues or network congestions [153].

2. **Edge Computing**. It is the practice of processing data closer to the source or *edge* of the network, usually on local devices or gateway servers. This method is intended to reduce latency and improve real-time processing capabilities. Because data is processed closer to its source, edge computing significantly reduces latency [153]: this is especially important for applications that require real-time or near-real-time responses, like MAR or autonomous guidance. Moreover, this paradigm reduces the amount of data to be sent over the internet to a cloud server, thus saving energy and costs. However, edge computing solutions can be limited in scalability because edge devices are typically less powerful and have fewer resources than cloud servers.

3. **Fog Computing**. Fog computing is an extension of edge computing in which computing resources are deployed at an intermediate layer between edge devices and the cloud. It aims to address some of edge computing's scalability and resource constraints, offering a scalable solution by introducing intermediary nodes that can aggregate and process data from edge devices before sending it to the cloud (or avoiding to do so) [153]. While not as low-latency as edge computing, fog computing is still faster than pure cloud computing. Moreover, Data processing can be distributed across fog nodes, allowing for the execution of more resource-intensive tasks, and exploiting, as an example, the framework of Federated Learning. However, implementing and managing a fog computing infrastructure can be complicated and time-consuming, as fog node deployment can be costly in terms of hardware and maintenance.

This chapter will focus its attention on the so–called Mobile Edge Computing (MEC) paradigm, in which a MD requires computation resources to an edge resource which is geographically located near to the device itself or attached to the BS the MD connects to.

While traditional offloading strategies rely on static optimization or heuristics, here a multi–input data–driven dynamic control of uplink power and image compression rate will be carried out, introducing a Policy Broadcasting DRL approach, based on the DDPG algorithm. The proposed solution is aimed at matching the challenging Quality of Service constraints, in terms of maximum round–trip latency and minimum resolution accuracy, while minimizing the energy consumption. Simulations will show the effectiveness and scalability of the proposed approach for real–time applications.

All material in this chapter refers to [154].

6.1 Motivations and Related Works

The rapid evolution of wireless communication technologies has driven a paradigm shift in the design and deployment of wireless networks. One of the key developments has been the emergence of Heterogeneous Networks (HN) characterized by (i) the integration of wireless communication systems belonging to different RATs and (ii) the coexistence of various network elements, such as macrocells, microcells, picocells, and femtocells [155]. The pivotal enabler of the HN deployment is 5G, the fifth generation of wireless communication technology, which is able to address the challenges posed by the increasing demand for higher data rates, low latency, massive device connectivity, and diverse user application requirements [156]. The evolution of wireless communication technologies is accompanied by the widespread diffusion of mobile devices for daily use, such as tablets and smartphones. Regarding the latter, the number of mobile network subscriptions worldwide reached almost 6.4 billion in 2022, and is forecast to exceed 7.7 billion by 2028 [157].

Over the years, these digital devices made available to users have become increasingly sophisticated and are equipped with powerful processors (CPUs), very high resolution screens, video cameras and sensors of different types. This allows smartphones and tablets to run very complex applications and functions aimed at teleworking, virtualization and entertainment [158]. In the latter domain, one of the key emerging technology is MAR, which has become an integral part of various industries, ranging from e–commerce and gaming to education and healthcare [159].

In general, AR refers to the integration of computer-generated sensory information, such as visuals, sounds, and haptic feedback, into the real-world environment through user devices. This technology extends human perception by overlaying virtual objects onto our physical surroundings, creating an immersive and interactive user experience [160]. However, in the MAR context, the inherent constraints posed by limited computational resources and battery capacity within MDs make the accomplishment of object analytics while adhering to stringent low-latency requirements a challenging endeavor. Since usually MDs like smartphones or headsets cannot mount built–in GPUs, they need to perform computation offloading towards network cloud servers, which arrange object detection or creation of tasks, sending back the augmented frames to MDs [161]. This operation usually is characterized by high latency, not allowing to enjoy real–time AR applications.

To address this challenge, the concept of MEC, standardized by the European Telecommunications Standards Institute (ETSI) [162], emerges as a promising solution. The MEC offers computational resources to MDs at the network edge, positioned physically closer to MDs than conventional cloud servers. This allows to reduce the network communication latency, enabling real–time MAR application.

Recent research endeavors have focused on efficient computation offloading to MEC servers, taking into account latency, bandwidth, and computational resource limitations as Key Performance Indicators (KPIs) [127, 163, 164]. These works have been enriched considering also accuracy and resolution control [165], and MD's cache management [166] which constitute the distinct attributes of object analytics within the realm of MAR. While the above papers adopt a static nonlinear optimization, several other works consider the problem of uplink power management from a control systems perspective. As an example, in [167], the authors demonstrate the internal stability of standard power allocation dynamic procedures. More recent works (e.g., [168–173]), instead, have focused the attention on data–driven control methodologies, namely RL [174]. A multi–agent Q-Learning formulation has been described in [175] where the authors consider a multitude of users requesting streaming services at the same time. In [176], the authors exploit DRL to deal not only with power control, but also with user association in a HN environment.

However, none of the mentioned works have considered, jointly, the problem of dynamic uplink power allocation and image resolution control for MAR applications in a MEC scenario.

This research advances the state of the art introducing the following innovations:

- simultaneous data-driven control of uplink power allocation and image compression by means of DRL, exploiting users' spatial information and implementing continuous control actions;
- definition of a Policy Broadcasting approach, through which the training phase of the RL procedure is limited to one agent only, whereas the execution is broadcast to multiple agents;
- introduction of a novel reward function leveraging on the trade-off between image accuracy and energy consumption, while guaranteeing operational constraints.

In the next sections, the Policy Broadcasting learning/execution approach will be detailed, and the MAR wireless network system will be modeled as a dynamical system. Building upon this, the simulations and results will be presented, highlighting the advantages of the proposed control architecture. Eventually, a brief final discussion will highlight the practical implications of the study's insights, outlining potential avenues for future researches.

6.2 Policy Broadcasting Reinforcement Learning

Building upon a standard MDP formulation, as in the previous work, here a Policy Broadcasting Reinforcement Learning (PBRL) approach is introduced. A unique policy is learned during the training phase (see Algorithm 9) and then it is broadcast to multiple agents, which perform in parallel their actions during the execution phase, sharing the same environment (see Algorithm 10). This algorithmic choice allows to deal with one of the typical drawbacks of RL algorithms in multi-agent scenarios which is represented by the required training times. When the agents share an homogeneous nature, it is possible to save computational time by training only a single agent instead of dedicating a local training for each of the agents.

Algorithm 9 PBRL Single Agent Learning Phase

```
Initialize policy π randomly
for episode ← 1 to end do
    Initialize state s
    while s is not terminal do
        Take action a according to exploration strategy
        Observe reward r and new state s′
        Update internal state and policy π
    end while
end for
```

Algorithm 10 PBRL Multi Agent Execution Phase

```
Initialize vector state s = [s_1, ..., s_N]
while s is not terminal do
    for agent i ← 1 to end do
        Observe state s_i
        Select action a_i using policy π
        Receive reward r_i and new state s_{i,′}
        Update internal state
    end for
end while
```

The training phase relies on the Deep Deterministic Policy Gradient (DDPG) algorithm 7, which, as seen in the first part of this thesis, can tackle problems with continuous action and state spaces.

6.3 MAR System Modeling

Let us consider a single–cell MEC system, consisting of an edge server wired (e.g., by means of optical fiber) to a 5G Base Station (BS), whose fixed 2D position is

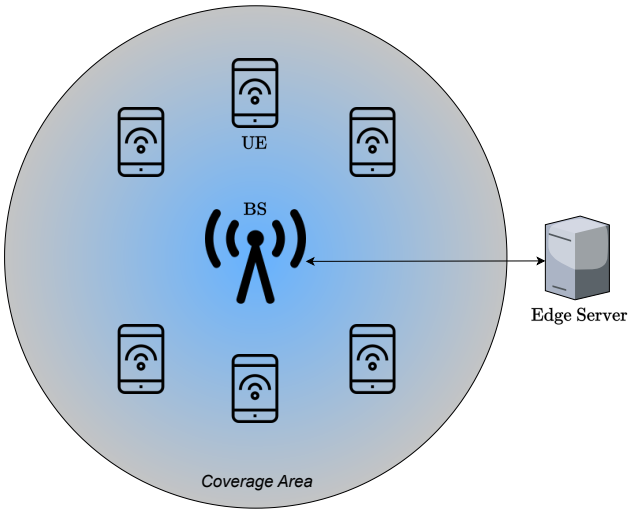


Figure 6.1. MAR system scenario.

$\boldsymbol{x}^{\mathrm{BS}} = \begin{bmatrix} x_1^{\mathrm{BS}} & x_2^{\mathrm{BS}} \end{bmatrix}^T$. The server hosts a virtual machine with guaranteed GPU capabilities for object detection and image augmentation purposes. The MDs are single–antenna devices, sparsely scattered within the coverage area of the BS and each one of them occupies a non–overlapping bandwidth W of the BS, running an active session of a MAR application. The latter aim is to capture images with the smartphone camera, send them to the edge server, thus receiving back the new images with modified and augmented information. The MDs have the same technological characteristics and cannot communicate one with each other. The system scenario is depicted in Fig. 6.1.

Consider now a single MD, and suppose its owner is allowed to move only within the coverage area of the BS with speed $\boldsymbol{v} = \begin{bmatrix} v_1 & v_2 \end{bmatrix}^T$ while running its session, thus moving according to the first order differential equation

$$\dot{\boldsymbol{x}} = \boldsymbol{v}, \tag{6.1}$$

where $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$. Moreover, suppose that the user can estimate, at any given time, its position $\boldsymbol{x}$ through a GPS–like navigation system.

Let p be the transmission power of the MD, then the Signal to Noise ratio (SNR) can be defined as:

$$\mathrm{SNR} = \frac{pG_{\mathrm{MD}}G_{\mathrm{BS}}h}{WN}, \tag{6.2}$$

where G_{MD} and G_{BS} are respectively the transmitter and receiver gains, N is the noise power spectral density, and h is the attenuation of radio energy, modeled following the free-space path loss propagation model as:

$$h = \left(\frac{\lambda}{4\pi d}\right)^2, \tag{6.3}$$

where λ is the signal wavelength, and $d = ||\boldsymbol{x} - \boldsymbol{x}^{\mathrm{BS}}||$ is the Euclidean distance between the mobile device and the base station.

As a result, remembering the Shannon–Hartley theorem, the uplink transmission rate cannot exceed the amount

$$R(p) = W \log_2(1 + \mathrm{SNR}). \tag{6.4}$$

All the images captured by a MD undergo a compression process before being sent to the edge server which ultimately processes them.

We assume that the MD has compression capacity V, and captures K-pixel raw images which are compressed to s pixels, each one containing σ bits of information.

The accuracy of the image processing task can be computed as follows:

$$A(s) = 1 - 1.578 \exp(-6.5 \times 10^{-3} s^{1/2}). \tag{6.5}$$

Assuming that the server is capable of providing a minimal computation speed of f (TFLOPS), the processing workload is given by:

$$\psi(s) = 7 \times 10^{-10} s^{3/2} + 0.083. \tag{6.6}$$

The latency occurring for a round trip path can be divided into four components, defined as follows:

- $L_{\mathrm{IC}} = \sigma K / V$ is the latency related to image compression at the i-th MD level.
- $L_{\mathrm{T}} = \sigma s / R(p)$ is the latency due to the 5G connection between the MD and the BS.
- $L_{\mathrm{ES}}^{\mathrm{BS}}$ is the latency between the BS and the edge server. This quantity can be neglected compared to the other terms.
- $L_{\mathrm{ES}} = \psi / f$ is the latency related to processing workload at the edge server level.

Hence, the total cumulative latency is

$$L(p, s) = L_{\mathrm{IC}} + L_{\mathrm{T}} + L_{\mathrm{ES}}. \tag{6.7}$$

Eventually, energy consumption arguments at MD level are considered. Let $E(p, s)$ be the total energy consumed by the MD. It is made by two elements:

- $E_{\mathrm{IC}} = \varepsilon\sigma(K - s)$ is the energy spent for image compression, with ε being the energy consumption for compressing 1 bit of data.
- $E_{\mathrm{T}} = pL_{\mathrm{T}}$ is the energy related to data transmission.

In what follows the mathematical formulation presented above is mapped on a MDP, in order to exploit the DRL control framework.

The state of the agent is given by

$$\mathcal{S} = \langle d \rangle, \tag{6.8}$$

i.e., it corresponds to the relative distance from the BS. This modeling choice, together with the assumption of the existence of non–overlapping bandwidths, allows to apply the PBRL approach, thus training only a single agent, and then deploying its policy to multiple MDs.

The action space

$$\mathcal{A} = \langle p, s \rangle \tag{6.9}$$

is composed of the transmitting power $p \in [p_{\min}, p_{\max}]$, and the image resolution after compression $s \in [s_{\min}, s_{\max}]$.

The goal of the agent is to solve the power allocation and image processing problems in such a way that QoS and QoE KPIs are satisfied. To this end, let μ and δ be the maximum sustainable latency and the minimum feasible accuracy, respectively. Hence, it is possible to define the following two usage constraints:

$$\begin{aligned} L(p, s) &\leq \mu \\ A(s) &\geq \delta. \end{aligned} \tag{6.10}$$

In addition, for energy saving purposes, it is desirable to govern image compression and data transmission spending the least possible amount of energy. As a result, the reward is defined as follows:

$$r = -\big(E(p, s) + \alpha Q(s)\big) - \mathrm{sgn}\big(L(p, s) - \mu\big) + \mathrm{sgn}\big(A(s) - \delta\big), \tag{6.11}$$

where $Q(s) = 1 - A(s)$ is the accuracy loss, α is a weight factor balancing energy minimization and accuracy loss, and $\mathrm{sgn}(\cdot)$ is the sign function.

6.4 Simulations and Results

In order to validate the effectiveness of our proposed approach, we consider a scenario of 10 MDs moving inside a square having a side $l = 300$ [m], with the BS located at the center of the square. This scenario is typical of archaeological parks

Table 6.1. System Parameters' Numerical Values

Parameter	Value	Unit
f	0.5	TFLOPS
G_{BS}	15	-
G_{MD}	3	-
λ	5×10^{-3}	m
K	4×10^{5}	pixel
N	3.98×10^{-16}	W
$p_{\max}$	3.5	W
$p_{\min}$	0	W
$s_{\max}$	4×10^{5}	pixel
$s_{\min}$	0	pixel
σ	24	bit/pixel
V	1×10^{8}	bit/s
W	1×10^{6}	Hz

that offer AR services: each tourist is equipped with a MD and is free to move within a limited area.

The numerical parameters of the MAR system are summarized in Table 6.1.

As for the users' QoS and QoE, the constraints parameters have been set such as $\delta = 0.85$ [%], $\mu = 0.85$ [s]. The cost parameter, fixed as $\alpha = 4$, weights more the accuracy than the energy consumption.

A single agent is trained for $E_{\text{ep}} = 200$ episodes, each one lasting $T = 30$ [s], and adopting an integration of the system dynamics (6.1) using the Runge–Kutta 4th order method with time step $dt = 0.1$ [s].

At the beginning of each episode the agent starts from a random distance d_0 from the BS and he moves with random velocity $\boldsymbol{v} \in [0 \ 1.11]^{2,T}$ [km/h], to mimic a typical human walking behavior. The episode ends if $A(s)$ and/or $L(p, s)$ become unfeasible or if the time limit T is reached.

During the evaluation procedure, the learned policy resulting from the training phase is deployed to all MDs, which start at different distances from the BS and move with different random velocities $\boldsymbol{v}_i(t)$. The system dynamics evolve for the same amount of steps per episode as in the training phase.

The DDPG hyperparameters have been carefully selected as follows: the actor learning rate $\beta_{\text{a}} = 1 \times 10^{-3}$, the critic learning rate $\beta_{\text{c}} = 2 \times 10^{-3}$, the discount factor $\gamma = 0.9$, and the memory capacity is selected as 1000 transitions. Both the actor and critic share the same neural architecture composed of three layers, the first two made of 512 and 128 neurons with ReLU as the activation function [177]. The last layer of the critic is made of a ReLU single neuron, whereas the last layer of the actor network has the $\tanh(\cdot)$ as activation function for the actor.

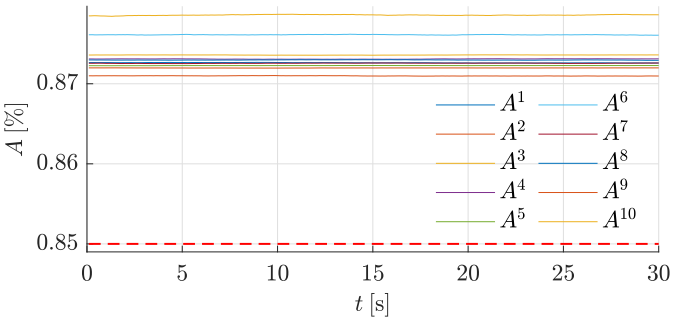


Figure 6.2. Accuracy $A_i(s)$ evaluation for each MD $i = 1, \ldots, 10$.

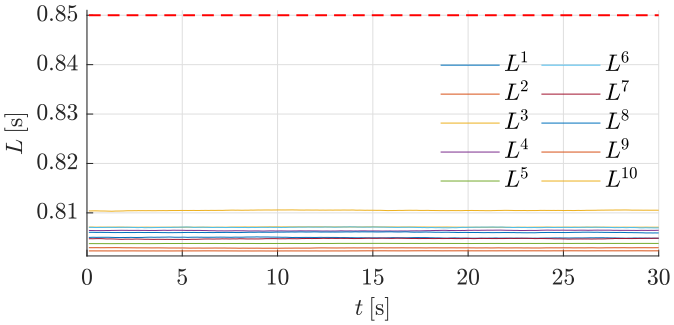


Figure 6.3. Latency $L_i(s)$ evaluation for each MD $i = 1, \ldots, 10$.

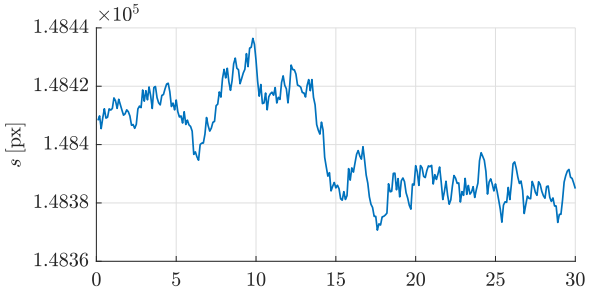
Training and evaluation are carried out in almost 1 hour using Tensorflow and Keras on an Intel Xeon platform with 13 GB of RAM and Nvidia Tesla T4.

Figures 6.2-6.3 show the effectiveness of the proposed approach in terms of meeting the accuracy $A(s)$ and latency $L(p, s)$ requirements respectively. Although each MD moves with a random velocity, each agent is able to choose the correct action pair $\langle p_i(t), s_i(t) \rangle$ which guarantees continuity of service at every instant.

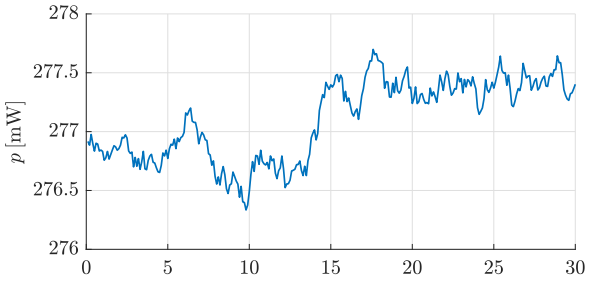
The dotted red line in both figures shows the KPI-correspondent value to be satisfied. In particular, the highest accuracy value is achieved by the tenth MD, $A_{10,\max} = 0.8787$ [%], while the second agent achieves the lowest latency.

Figure 6.4 shows the behavior of the second MD (the others are similar), which, starting from a distance of 152.69 [m] from the BS, dynamically adjust its transmission power $p_2(t)$ and image resolution $s_2(t)$ to ensure QoS to its user. It is worth noting that the MD increases its uplink power and decreases the image quality when the user moves away from the BS, as expected.

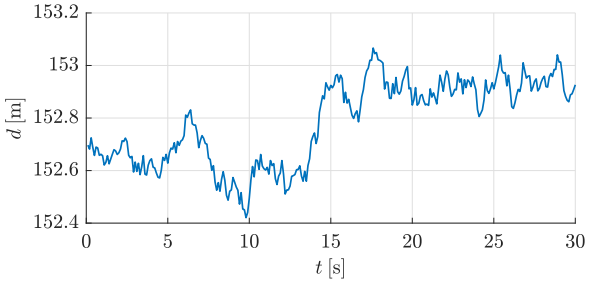
Overall, each MD has a different initial distance from the BS, and its owner moves with different dynamically changing random speed, influencing his position, as shown in Figure 6.4 for the second agent. The mean values (computed over the evaluation steps) for distance $\bar{d}_i$, transmission power $\bar{p}_i$ and image resolution $\bar{s}_i$ are reported in Table 6.2 for all MDs, $i = 1, \ldots, 10$.



(a) *Image compression* $s^2(t)$.



(b) *Transmission power* $p^2(t)$.



(c) *Distance from the BS* $d_2(t)$.

Figure 6.4. Time evolution of relative distance, power, and compression rate (second agent).

Table 6.2. Distance, transmission power and image resolution mean values for all MDs

Index	$\bar{d}_i$	$\bar{p}_i$	$\bar{s}_i$
1	129.05	205.40	1.50×10^5
2	152.78	277.10	1.48×10^5
3	118.04	174.69	1.51×10^5
4	124.03	190.35	1.50×10^5
5	133.35	219.02	1.49×10^5
6	96.94	131.79	1.53×10^5
7	129.74	207.56	1.50×10^5
8	125.37	194.27	1.50×10^5
9	141.66	246.69	1.49×10^5
10	87.08	114.09	1.56×10^5

Table 6.3. Average energy consumption $\bar{E}_i(p, s)$ for all the MDs (Joule)

Index	$\bar{E}_i(p, s)$
1	0.21
2	0.25
3	0.20
4	0.21
5	0.22
6	0.18
7	0.22
8	0.21
9	0.23
10	0.17

The same is done for the energy consumption, whose mean values $\bar{E}_i(p, s)$ for each MD are reported in Table 6.3.

To summarize, in this chapter a novel RL-based paradigm to cope with the computation offloading QoS–constrained problem for a MAR application in the MEC scenario has been proposed. Leveraging on the PBRL control approach, the knowledge of MDs' distance from the BS turns out to be sufficient to satisfy latency and accuracy requirements in scenarios where all MDs move within the BS's coverage area with random velocity. The introduction of an energy consumption term within the reward function extends MDs' battery life, thus guaranteeing continuity of service.

Future works may involve the introduction of more complex connectivity protocols in the context of multi Radio Access Technology networks, thus mixing the image resolution and power control with the MD–BS association problem.

Chapter 7

Resilient Systems Against Telecommunication Failures

The previous chapters covered decision-making and control problems applied to the field of mobile augmented reality. Now, another application with stringent constraints in terms of latency and safety is considered: autonomous driving.

7.1 Autonomous Driving and Connected Automated Vehicles

Over the last decades progressive urbanization and the increase in private cars has posed critical transport and mobility challenges, especially in densely populated cities. According to the WHO, traffic accidents are the leading cause of death for children and young adults between 5 and 29 years, and approximately 1.3 million people die each year by road traffic crashes [178]. Most of the latter are caused by human drivers' errors, which are due to long reaction times, non–cooperativeness or to the irrationality in taking actions while driving their cars.

Since the 80s', the automotive industry has been shifting towards higher levels of automation, thus introducing the notion of autonomous driving.

Autonomous driving, also known as self-driving or driverless driving, refers to a vehicle's ability to operate and navigate without the intervention of a human. To perceive their surroundings, make driving decisions, and control the vehicle, autonomous vehicles are outfitted with advanced sensors, cameras, radar, Lidar, and AI software. The ultimate goal of self-driving cars is to provide a safe and efficient mode of transportation with little or no human intervention.

The Society of Automotive Engineers (SAE) has established six levels of driving automation [179]:

1. Level zero signifies a complete absence of automation, and it was the state of the art during the whole previous century up to the 90's.
2. Level one denotes basic driver assistance systems, like adaptive cruise control, anti-lock braking systems, and stability control. Most of the commercial cars that we use daily fall into this category.
3. Level two represents partial automation, which includes advanced assistance systems such as emergency braking and collision avoidance. Thanks to advancements in vehicle control knowledge and industry experience, level two automation has become a feasible technology.
4. Level three introduces conditional automation, allowing the driver to focus on tasks other than driving during regular operation. Nevertheless, the driver must promptly respond to an emergency alert from the vehicle and be prepared to take control. Level three automated driving systems are constrained to specific operational design domains, such as highways.
5. Level four can only operate within limited roads where specialized infrastructure or detailed maps are available. If these areas are exited, the vehicle must automatically park itself, concluding the trip.
6. Level five requires no human attention and characterize the full driving automation. These cars can function on any road network and in all weather conditions. As of now, no production vehicle is equipped with level four or level five driving automation: Toyota Research Institute has indicated that no one in the industry is anywhere close to achieving level five automation [180].

Implementing level four and above driving automation in urban road networks is a formidable and unresolved challenge. Environmental variables, from unpredictable weather conditions to the intricacies of human behavior, make the problem highly stochastic and unpredictable.

In fact, in the last years several fatal accidents [181, 182] occurred with autonomous vehicles implementing the fourth automation level, thus raising an important ethical and judicial question.

One of the key characteristic of autonomous driving is the fact that each vehicle is connected to the internet exploiting a terrestrial network. This means that self–driving vehicles can communicate either between themselves or with other entities in any given environment. These vehicles are called Connected Autonomous Vehicles (CAVs). This paradigm introduces several benefits in terms of improved safety, predictive traffic management, and enhanced infotainment.

In scientific literature, two types of connectivity for self-driving vehicles are distinguished:

1. **Vehicle to Vehicle (V2V)** communication allows vehicles to communicate directly with other vehicles in the vicinity. It is an essential component of CAVs and plays an important role in improving road safety. This paradigm is critical because vehicles can share information about their speed, position, and heading to detect and avoid potential collisions, allowing them to maintain safe following distances by adjusting their speed in real-time [183]. Furthermore, autonomous vehicles can be alerted to the presence of emergency vehicles or traffic conditions in order to optimize fuel consumption.
2. **Vehicle to Everything (V2X)** realizes communication between vehicles and other elements of the transportation ecosystem, such as infrastructure, cloud or edge servers, pedestrians, and cyclists [184]. Said paradigm is in turn divided into three large areas:
    - **Vehicle to Infrastructure (V2I)**. It allows vehicles to exchange data with roadside infrastructure like traffic lights and road signs [185]. This allows for better traffic management and routing.
    - **Vehicle to Pedestrian (V2P)**. It occurs when vehicles and pedestrians interact. This can include providing pedestrians with warning signals and ensuring their safety in urban areas.
    - **Vehicle to Network (V2N)**. It extends vehicle network connectivity, allowing them to access real-time data from the cloud such as traffic conditions, weather updates, and road closures.

7.2 State of the art

It has already been shown how controlled CAVs can overcome human drivers' limits and mitigate the frequent negative effects due to man driving a non–autonomous car [186, 187].

The collective behavior of multiple (CAVs) is governed by their mutual awareness of individual states, such as inter-vehicle distance and speed. This awareness is achieved through inter-vehicle sensing and communication. The information obtained from these processes is a crucial input for each local controller, greatly influencing the overall collective behavior. A set of CAVs moving together on the road can be modeled using different information flow topologies (IFTs), through which it is possible to model the set of CAVs as a multi–agent system described mathematically using a directed graph [188].

A wide number of studies have tackled control issues of multi–agent CAVs. Standard approaches involve linear consensus control [189], distributed robust control [190], sliding–mode control [191], and model predictive control [192, 193]. The

objective of the mentioned works is to synchronize the speeds of all vehicles within the same group (internal stability), while also maintaining desired spacing between adjacent vehicles (string stability). This approach aims to enhance traffic capacity, improve overall traffic safety, and reduce fuel consumption.

During the last years, researchers began to use machine learning in transportation research [194] and, in particular, for the management of CAVs [195]. In addition to the use of supervised learning models for vision and perception purposes [196], several works examined the potential of Reinforcement Learning (RL) control for CAVs platoons [59, 197]. In particular, the Multi–Agent Reinforcement Learning (MARL) domain [198] appears to be suitable to tackle the distributed control of CAVs: in this scenario, multiple agents take decisions and perform actions over a shared environment to maximize their long–term return. One of the first works adopting MARL for CAVs control demonstrated the effectiveness of a centralized controller in dampening traffic oscillations and reducing the electric vehicle energy consumptions [199]. Authors in [200] have proposed a communication proximal policy optimization algorithm to reduce the fuel consumption, and a similar approach has been used in [201] in a mixed scenario where CAVs need to interact with human–driven vehicles (HDVs). Other recent articles used Deep Reinforcement Learning (DRL), focusing on the minimization of crash risk [202], on formation control under communication failures [203], on human passengers' comfort over CAVs [204], and, eventually, on safety improvement at intersections [205].

None of the above mentioned articles deals with the implementation of traffic rules for safe autonomous driving with complete telecommunication fault, i.e. when the vehicles are not connected to a wireless network and, hence, cannot exchange information one with each other. The original contributions in this chapter are:

- the implementation of a non-cooperative control framework for autonomous vehicle platooning relying on inter-vehicle sensing only;
- evaluation of traffic rules adherence in safe distance assessment;
- modeling and generation of traffic waves compelling HDVs behavior.

In the next sections, a fully automated hybrid vehicle platoon will be formally modeled as a multi–agent dynamical system, and the consequent control approach will be detailed. The validity of the proposed solution will be proven through various simulations, and future work directions building up on limitations and blind spots will be itemized.

The proposed control methodology and relative simulations and results refer to [206].

Figure 7.1. AVs platooning system scenario. Each AV is equipped with a range sensor to estimate the space gap from the vehicle ahead.

7.3 Mathematical Model

The system scenario is depicted in Fig. 7.1: it consists of a set of N AVs following a HDV, which is considered to be the leader of the platoon. Each vehicle cannot exchange information with the other ones, but it can rely on a distance sensor mounted on the front bumper, which allows to measure how far the next vehicle is.

Dynamical models of a single AV are usually split into three main components [207, 208]:

1. longitudinal dynamics;
2. bounce and pitch dynamics;
3. lateral, yaw and roll dynamics.

This work focuses on the longitudinal dynamics, which is parallel to the ground and oriented along the direction of motion. Said dynamics is inherently nonlinear, and in literature it is usually linearized for tractable issues. The most used models are the single integrator [209] whose control input is the vehicle longitudinal velocity, the double–integrator [210, 211] in which the input is the vehicle acceleration, and the third–order model [212] that takes into consideration the powertrain internal dynamics.

In this work we use a fully nonlinear second–order model to describe the longitudinal dynamics of the AV, resulting from considering all the forces acting on the vehicles modeled as a point-mass.

Consider a single AV. Let $x = [x_1 \;\; x_2]^T = [p \;\; v]^T \in \mathbb{R}^2$ be the state of the dynamical system, corresponding to its position and velocity. The resulting dynamics is described via the following equations:

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \dfrac{1}{m}\Big(F_{\mathrm{T}} - F_{\mathrm{AV}} - F_{\mathrm{G}} - F_{\mathrm{DRAG}}\Big) \end{cases}, \tag{7.1}$$

where F_{T} is the longitudinal thrust, $F_{\mathrm{AV}} = \frac{\mu_v}{R_w} mg\cos\alpha$ the rolling friction, $F_{\mathrm{G}} = mg\sin\alpha$ the gravity acting on a slope, and $F_{\mathrm{DRAG}} = \frac{1}{2}\rho_{\mathrm{air}} C_d A_v (v_w + x_2)^2$ is the aerodynamic drag. In particular, g is the gravitational acceleration, α is the slope of the surface, ρ_{air} the air density, v_w the speed of the wind, μ_v the rolling friction coefficient, R_w the wheel radius, m the mass, C_d the drag coefficient and A_v the cross-sectional area of the vehicle.

Assuming that the control input is the acceleration $u = \dfrac{F_{\mathrm{T}}}{m} \in \mathbb{R}$, (7.1) can be written as

$$\dot{x} = f(x) + Bu, \tag{7.2}$$

in terms of the vector fields

$$f(x) = \begin{bmatrix} x_2 \\ -\dfrac{\mu_v}{R_w} g\cos(\alpha) - g\sin(\alpha) - \\ +\dfrac{1}{2m}\rho_{\mathrm{air}} C_d A_v (v_w + x_2)^2 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{7.3}$$

The longitudinal dynamics of the HDV leader is modeled as a double integrator, with its input u^l, corresponding to its acceleration, as a noisy sinusoidal wave, mimicking the typical human behavior in traffic-waves conditions. Let $x^l = [p^l \; v^l]^T$ be the state of the leader, composed of its position and velocity respectively, then the dynamical equations are as follows:

$$\begin{cases} \dot{x}_1^l &= x_2^l \\ \dot{x}_2^l &= u^l \end{cases}, \tag{7.4}$$

where $u^l = (A + \hat{A})\sin\big((\omega + \hat{\omega})t\big)$, with A, ω its amplitude and pulse, and $\hat{A} \in \mathcal{N}(\mu_{\hat{A}}, \sigma_{\hat{A}})$, $\hat{\omega} \in \mathcal{N}(\mu_{\hat{\omega}}, \sigma_{\hat{\omega}})$ their corresponding noises.

While the sinusoidal function resembles the ordinary "start and stop" traffic scenario, the employment of the additive noises to both its amplitude and pulse try to model human interventions, such as sudden braking or full throttle events.

Building up on the mathematical framework of Markov Games, the mathematical model presented so far can be translated as follows.

The state space of the i-th AV is given by

$$\mathcal{S}^i = \langle v^i(t), d_{i-1}^i(t) \rangle, \tag{7.5}$$

where $v^i(t) \in [v_{\min}, v_{\max}]$ represents its sensed velocity, and $d_{i-1}^i(t) \in [d_{\min}, d_{\max}]$ the measurement of its distance from the vehicle in front.

Note that each AV uses its own local information to solve its corresponding MDP, hence there are as many actors as the number of vehicles, and each one of

them contributes equally to the environment update. Therefore, the dimension of each MDP does not vary with the number of agents, improving the scalability of the proposed approach.

The action space of each agent corresponds to the vehicle acceleration input, which is a saturated one:

$$\mathcal{A}^i = \langle u^i(t) \rangle, \qquad u^i(t) \in [u_{\min}, u_{\max}]. \tag{7.6}$$

Given the platooning problem, our proposed approach is to steer the velocity $v^i(t)$ of the i-th agent towards a desired value $v_d^i(t)$, which is computed as a function of its distance from the vehicle in front $d_{i-1}^i(t)$, so that the latter becomes a safe distance. In other words, in accordance to traffic regulations, we adjust $v^i(t)$ to make sure that $d_{i-1}^i(t)$ is an adequate stopping distance.

By approximating the i-th agent with a point-mass, $v_d^i(t)$ results from solving the following second–order equation:

$$d_{i-1}^i(t) = \frac{v_d^i(t)^2}{|u_{\min}|} + v_d^i(t)\, t_r, \tag{7.7}$$

where t_r is the AV reaction time.

Hence, the immediate reward $r^i(t)$ is shaped as follows:

$$r^i(t) = \begin{cases} -100, & \text{if} \quad d_{i-1}^i(t) < d_{\min} \\ -\left(v^i(t) - v_d^i(t)\right)^2, & \text{if} \quad d_{\min} < d_{i-1}^i(t) < d_{\max} \\ -\left(v^i(t) - v_{\max}\right)^2, & \text{otherwise} \end{cases}. \tag{7.8}$$

Due the intrinsic nature of the control problem, the chosen data–driven method to control in a distributed fashion the non–coperative AVs' platoon relies on a multi–agent DDPG, with a DTDE approach (see Fig. 2.2 in Chapter 2).

7.4 Simulations and Results

In order to validate the robustness of our proposed approach, we consider a platoon of five AVs following a HDV.

Note that each AV differs from the others in terms of its mass m^i, length l^i, wheel radius R_w^i, and drag area $A_d^i = C_d^i A_v^i$. Table 7.1 details their values, in accordance with [213] and manufacturer data.

The road over which the vehicles are traveling is supposed to have no slope ($\alpha = 0\,[^\circ]$), in Standard Ambient and Temperature Pressure (SATP) ($\rho = 1.225\,[\mathrm{kg/m^3}]$), with tarmac in good condition ($\mu_v = 0.015$), in presence of head wind ($v_w =$

Table 7.1. AVs' mechanical and aerodynamical characteristics

Nr.	Name	l [m]	m [kg]	R_w [m]	A_d [m^2]
1	Mercedes Benz A180d	4.41	1353	0.48	0.33
2	Opel Calibra	4.5	1190	0.31	0.5
3	Hummer H3	4.74	2654	0.41	1.56
4	BMW i8	4.69	1560	0.35	0.54
5	Tesla Model S P85	4.97	2307	0.36	0.56

1.78 [m/s]) and constant gravity ($g = 9.81$ [m/s^2]). Moreover, it is assumed that vehicles move on a highway road with maximum speed $v_{\max} = 36.1$ [m/s], without the possibility to reverse ($v_{\min} = 0$ [m/s]).

All the AVs are equipped with a radar sensor with maximum range $d_{\max} = 150$[m], whose measurement is corrupted via a Gaussian noise signal $n(t) \in \mathcal{N}(0, 0.14)$. In addition, we consider a safe minimum distance between vehicles $d_{\min} = 3$ [m].

The action of each agent lies between $u_{\min} = -4$ [m/s^2] and $u_{\max} = 6$ [m/s^2], which results to be common values for standard vehicles, while the Gaussian noise signal $\hat{A}$, with $\mu_{\hat{A}} = 0.3375$ and $\sigma_{\hat{A}} = 0.58$, corrupts the amplitude of the leader's acceleration, and $\hat{\omega}$, with $\mu_{\hat{\omega}} = 0.05$ and $\sigma_{\hat{\omega}} = 0.22$, its pulse.

Since all RL algorithms only deal with discrete-time dynamics, the equations in (7.1) have been discretized via the Runge-Kutta 4th order method, with time step $dt = 0.1$ [s], which corresponds as well to the reaction time t_r before applying the braking action.

Each agent is trained for $E = 300$ episodes, each one lasting $T = 30$ [s], for a total of 300 steps (T/dt).

Starting from its initial state $x_0^i = [p_0^i \;\; v_0^i]^T$ at $t = 0$ [s], the i-th agent has to adjust its speed $v^i(t)$ according to its measured distance $d_{i-1}^i(t)$ from the vehicle in front. Having multiple AVs, at each time step t, they apply simultaneously the acceleration actions on the environment, observe the corresponding rewards and next states, and subsequently are trained in parallel. The episode ends if an agent gets too close to the vehicle in front, namely $d_{i-1}^i(t) < d_{\min}$, or if the time limit T is reached.

The hyper parameters of DDPG are carefully selected as follows: the actor learning rate $\alpha = 0.001$, the critic learning rate $\beta = 0.002$, the discount factor $\gamma = 0.9$, and the memory capacity is selected as 1000 transitions. Both the actor and critic share the same neural architecture composed of three layers, the first two of 512 and 128 neurons with ReLU [177] activation function, and the last one of one neuron only, with tanh as activation function for the actor.

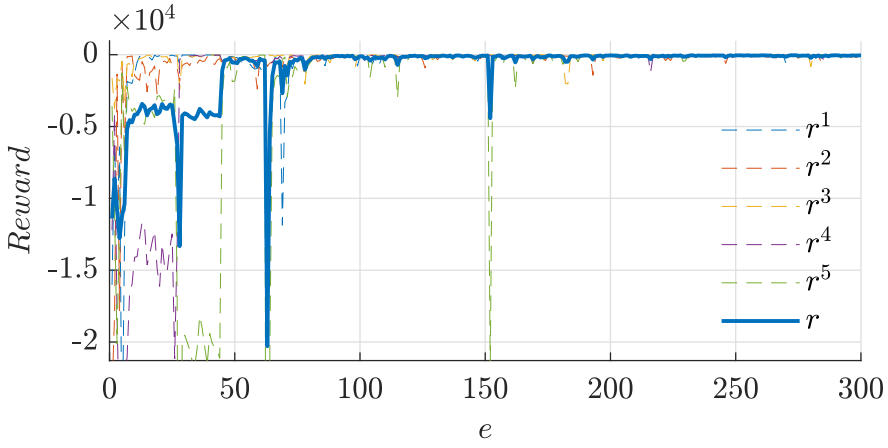


Figure 7.2. Mean reward r (solid line) and individual reward r^i, $i = 1, \dots, 5$ (dotted lines).

At the beginning of each training procedure ($t = 0$ [s]), the nominal initial state $x_0^i = [p_0^i \ v_0^i]^T$ of the i-th agent is perturbed via an additive Gaussian noise signal in such a way the position is corrupted by $\hat{p}_0^i \in \mathcal{N}(0, 1)$, and the velocity by $\hat{v}_0^i \in \mathcal{N}(0, 0.55)$.

Simulations are conducted using Tensorflow and Keras on an Intel i9 9900k platform with 128 GB of RAM and a Nvidia GTX 3090, which allows the training procedure to be completed in 33.54 [min] for each agent.

Figure 7.2 shows the results achieved by each agent at the end of each training episode $e = 1, \dots, E$, namely the changes in reward as the session progresses; the dotted lines corresponds to the reward r^i of each agent $i = 1, \dots, 5$, whereas the blue solid line is the mean reward r.

While the first four agents achieve convergence, each maximizing its own reward, after an exploration phase which lasts roughly 100 episodes, the last agent seems to require a more episodes, as it is shown by the negative peaks in its reward r^5 at episodes 110, 153, and 160, causing a drop in the cumulative mean reward r. This could be explained considering that its initial distance from the vehicle in front and velocity is smaller than the ones of the others, as shown in Figure 7.3 detailing the evolution of platoon's position over time during the evaluation phase.

Vehicles start from the initial state, namely

$$\begin{aligned} x_1(0) &= [310,\ 220,\ 140,\ 70,\ 5]^T \\ x_2(0) &= [28.0,\ 20.8,\ 22.7,\ 25.0,\ 19.3]^T \\ x_1^l(0) &= 400 \text{ [m]} \\ x_2^l(0) &= 25.0 \text{ [m/s]}, \end{aligned} \tag{7.9}$$

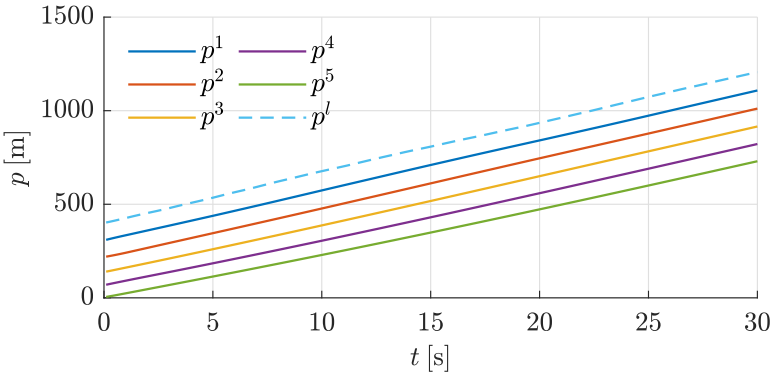


Figure 7.3. Platoon's position evolution over time during evaluation phase.

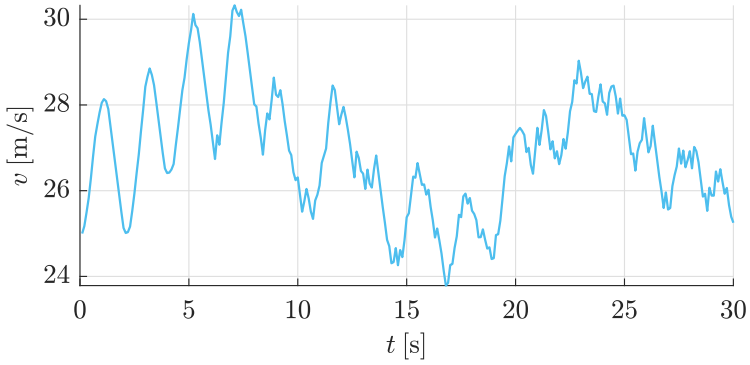


Figure 7.4. Leader velocity $v^l(t)$ over time.

and their dynamics is propagated for the same amount of steps as in the training phase.

As time goes by, each agent successfully keeps a safe distance from the vehicle in front by adjusting its own velocity, proving to have successfully learned the dynamics of the agent in front, as well as its own. In particular, given the velocity of the leader, which is shown in Figure 7.4, the first agent is able to follow its desired speed profile with a Mean Absolute Error (MAE) of $e_r^1 = 0.1405$, as shown in Figure 7.5. AVs' velocity errors are detailed in Table 7.2.

Overall the agents keep a mean distance of 88.17 [m] from the vehicle in front, meaning that they are capable of adjusting their control action to reach their goal (see Figure 7.6).

Figure 7.7 demonstrates the ability of the proposed control law in tracking the velocity reference, and, in the meantime, steering all the velocities towards a common value.

To summarize, this chapter presented a fully scalable non-cooperative multi-agent platooning strategy for maintaining safe distance between vehicles. The pro-

Table 7.2. Agents' Velocity MAE

Nr.	MAE
1	0.1405
2	0.1567
3	0.0359
4	0.2217
5	0.0553

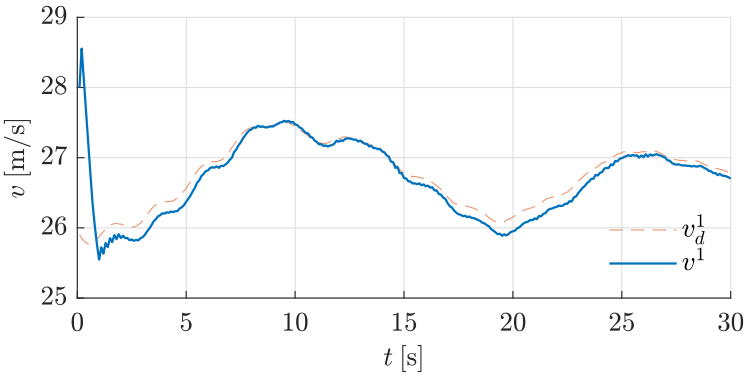


Figure 7.5. Comparison of desired and actual velocity of the first agent over time.

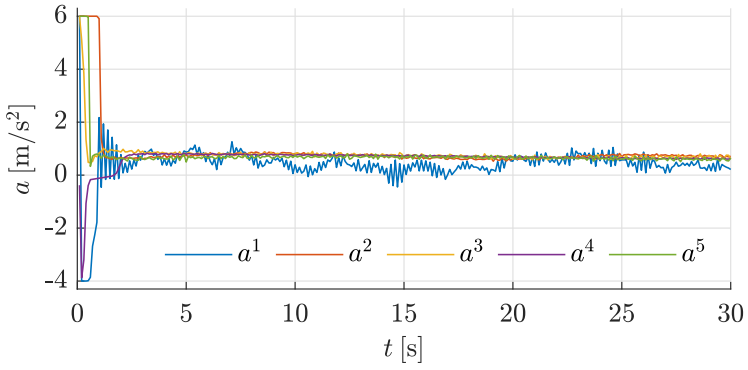


Figure 7.6. Platoon's acceleration evolution over time.

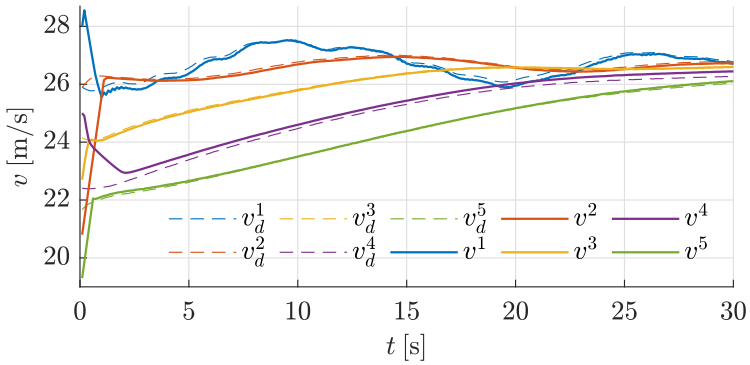


Figure 7.7. Platoon's velocity evolution over time.

posed method is based on information derived from agents' local sensing, specifically the measurement of their inter-distance, and does not require inter-vehicle communications. Simulations show that using a velocity signal as a reference for each agent, each of which is modeled by its own longitudinal dynamics, allows for safe vehicle inter-distance enforcement as well as steering the platoon's velocity towards a common value while adhering to traffic rules.

The effectiveness and robustness of the proposed approach are demonstrated by the evaluation of agents with different mechanical and aerodynamic properties, as well as a novel model for HDV-issued traffic waves. Future work could include the consideration of the longitudinal dynamics in the presence of road offset (bumps or ditches – $\alpha \neq 0$), energy optimization arguments, vehicle lateral dynamics, or traffic scenario enrichment including road obstacles like pedestrians, cyclists or trucks.

Part III

Satellite Networks

Chapter 8

From Radio Frequency to Free Space Optical Communications

Satellite communication is a form of wireless communication that involves the transmission of signals between Earth-based stations and artificial satellites orbiting the Earth. Hence, differently from terrestrial communication networks, the data transmission takes place outside the Earth's atmosphere, with transmitter and receiver being located at high distance one with each other. Typical applications of this kind of technology includes satellite TV broadcasting, internet access, positioning and tracking, weather and Earth surface monitoring, military operations, and scientific research.

A satellite communication system is realized through three main components: space segment, ground segment and the transmission medium.

8.1 Space Segment

The space segment consists of artificial satellites orbiting in space, which act as relay stations for transmitting signals between ground stations. The space segment may include three different types of satellites, according to their orbits' height:

- **Geostationary Earth Orbit (GEO) Satellites**. They are positioned at an altitude of approximately 35,786 km above the equator [214]. Their main distinctive feature is that they maintain a fixed position relative to the Earth's surface, orbiting at the same rotational speed as the Earth. This results in a stationary coverage area which does not change in time. For this reason, GEO satellites offer a wide, continuous coverage area, making them ideal for broadcasting and providing consistent connectivity over specific regions. The majority of satellite TV services exploit this kind of satellite. As an example,

the GEO satellite HotBird 13° East is the one used for TV services all around Europe [215]. However, the long distance between GEO satellites and Ground Stations (GSs) results in higher signal latency, which may not be suitable for time-sensitive applications.

- **Medium Earth Orbit (MEO) Satellites**. Their orbits height ranges from around 2,000 to 35,786 km above the Earth's surface [214]. These satellites provide relatively broad coverage areas, making them suitable for global and regional communication services. The Global Positioning System (GPS) and the Galileo Navigation System are examples of MEO satellite constellations.

- **Low Earth Orbit (LEO) Satellites**. They are positioned at altitudes ranging from approximately 180 to 2,000 km above the Earth's surface [214]. This property results in lower signal latency and shorter signal travel times compared to higher orbits. The latter are completed in roughly 90 to 120 minutes, providing frequent coverage of different regions across the world. Due to their low altitude, at any given time LEO satellites offer a poor coverage area, and this is why they are usually deployed in large constellations. The most famous ones are the Iridium and the Starlink. LEO fleets are well–suited for mobile satellite services, such as satellite phones and broadband internet access in remote areas, also for real–time applications.

A summarized comparison of the three main types of space segment is reported in Tab. 8.1.

Table 8.1. Comparison of LEO, MEO, and GEO Satellite Systems

Feature	**LEO**	**MEO**	**GEO**
Altitude	180 - 2,000 km	2,000 - 35,786 km	35,786 km
Orbit Time	90 - 120 minutes	2 - 24 hours	24 hours
Coverage Area	Small footprint	Medium footprint	Wide footprint
Latency	Low	Moderate	High
Number of Satellites	Large constellation	Medium constellation	Individual satellites
Signal Strength	Weak	Moderate	Strong
Coverage Frequency	Frequent passes	Intermittent passes	Continuous coverage
Applications	Global internet, remote sensing	GPS, regional communication	Television broadcasting, fixed services
Examples	Starlink, Iridium	GPS, Galileo	Hotbird 13° East

8.2 Ground Segment

The ground segment encompasses the Earth-based stations that transmit and receive signals to and from the satellites. These stations are equipped with specialized satellite communication equipment, such as antennas and transceivers, to establish communication links, thus ensuring seamless connectivity between remote locations and global communication networks [216]. GSs are responsible for controlling satellite operations. They manage satellite movements, transponder allocation, and payload configuration. Furthermore, they continuously monitor the health and status of satellites, contributing to their operational longevity.

Data received by GSs is then distributed to various end-users, ranging from individual consumers to government agencies, research institutions, and commercial enterprises. This data may include telecommunications traffic, internet services, TV broadcasts, weather data, and more. Moreover, the ground segment is crucial for ensuring the security and integrity of satellite communications. It manages encryption, authentication, and data protection measures to safeguard sensitive information transmitted over the network.

Each GS is equipped with hardware components capable of traducing the receiving signals into useful data information. Other components of the ground segments include control rooms intended for satellite orbits over–watch, and modules fro precise time synchronization and satellite tracking [216]. The ground segment of satellite communications has seen significant technological advancements over the years, such as software–defined GSs and automatic reconfiguration via AI. These innovations have contributed to enhanced efficiency, increased data throughput, and the expansion of satellite communication services.

8.3 Transmission Medium

The transmission medium denotes the technology through which the wireless communication is realized. There two main types of mediums: radio–frequency and optics.

8.3.1 Radio Frequency Satellite Communications

Traditionally, satellite communications are implemented using Radio Frequency (RF), a method that uses electromagnetic radio waves to exchange information between Earth-based stations and artificial satellites orbiting the Earth. This mode of communication uses radio frequencies in the electromagnetic spectrum to transmit and receive signals through space, enabling a wide range of applications such

as telecommunication, broadcasting, data exchange, and global connectivity. RF–based satellite communications can provide global coverage over large Earth's zones, enabling communication in remote and geographically isolated areas where terrestrial infrastructure is limited or absent. This is particularly important for disaster response, remote research, and global connectivity [214]. Moreover, satellite communication is known for its reliability. It is less susceptible to local outages or infrastructure failures, serving as a critical lifeline for emergency services, military operations, and other vital communication needs.

RF satellites are also widely employed for broadcasting, facilitating the simultaneous transmission of content to large audiences (up to 1 million of users for a single GEO satellite). This makes them a cost-effective medium for media companies to distribute TV, radio, and multimedia content. Moreover, when handled through LEO fleets, RF-based satellites are capable of transmitting substantial volumes of data rapidly, rendering them suitable for applications that demand high bandwidth, such as broadband internet, video conferencing, and data transfer.

This transmission mean suffers from two main drawbacks. The first is related to limited spectrum, which is finite and shared among various systems, including other satellite networks and terrestrial communication systems. As the demand for satellite services increases, spectrum congestion can lead to interference issues, reducing the QoS and potentially causing signal degradation [214]. The second limitation relies on security concerns, since RF communication can be vulnerable to interception and jamming. Unauthorized access to or interference with satellite signals can compromise the confidentiality and integrity of data transmitted over satellite networks [217].

8.3.2 Free Space Optical Communications

In the telecommunications domain, Free Space Optics (FSO) indicates all those wireless communications which, instead of making use of radio carriers in the form of a radio communication, make use of electromagnetic carriers belonging to the range of optical or infrared frequencies or wavelengths, aimed at transporting information between a transmitter, called optical satellite, and a receiver, called Optical Ground Station (OGS) [218].

FSO is not a new concept in engineering. Throughout history, optical communications have taken on various forms and have been utilized for millennia. The first optical communication system was invented in ancient Greece by Polybius[1], who

[1]Polybius was a Greek statesman and military commander born around 200 BC and died around 118 BC. He made great contributions to the development of military decision-making strategies,

devised an alphabetic signaling system using torches, showing how it was possible to communicate crucial information without sending human messengers.

Hundreds of years later, optical communication has been used in the photophone system invented in 1880 by Alexander Graham Bell and Charles Sumner Tainter. This technology was referred to as optical telegraphy in the subsequent years, and it was mostly used for military purposes during World War I and World War II.

The event which revolutionized the actual implementation of FSO also outside the Earth's atmosphere was the invention of lasers in the 60s'. The latter still represent the most widespread technological means for the physical creation of both terrestrial and satellite optical communication systems.

The first successful laser–communication space–to–ground link was carried out by Japan in 1995 between the JAXA's ETS-VI GEO satellite and the National Institute of Information and Communications Technology (NICT)'s OGS in Tokyo achieving a data–rate of 1 Mbps [220]. Few years later, the ESA satellite Artemis achieved the world's first laser intersatellite link in space in November 2001, providing an optical data transmission link with the CNES Earth observation satellite SPOT 4. The data–rate was 50 Mbps over a distance of 40,000 km [221].

From there, continuous advancements and improvements were achieved by the scientific community, until April 28, 2023, when National Aeronautics and Space Administration (NASA) and its partners achieved another significant milestone in the future of space communications: 200 Gbps throughput on a space-to-ground optical link between an orbiting satellite and Earth, the highest data rate ever achieved by optical communications technology [222].

A FSO transceiver schematic illustration is depicted in Fig. 8.1.

FSO offers several advantages over traditional radio frequency (RF) communication systems when used in space-based applications. In particular, the main benefit of FSO communication relies on data transfer rates. Indeed, a FSO system based on lasers can achieve much higher data transfer rates with respect to RF communications. This is due to the fact that laser light has a much shorter wavelength than RF waves: the wavelength of laser light falls within the optical spectrum, typically in the range of 400 to 700 nanometers (nm), while RF waves can have much longer wavelengths, ranging from millimeters to meters [224]. The shorter wavelength of laser light allows for higher frequency modulation, which means that data can be

---

together with the development of sophisticated communications systems based on the use of torches and live firebrands. However, he is best known for his work *The Histories*, a historical account of the Mediterranean world from 264 BC to 146 BC, covering the period of the Punic Wars and the rise of the Roman Republic as a dominant power. His work is considered one of the foundational texts in the study of ancient history and politics [219].

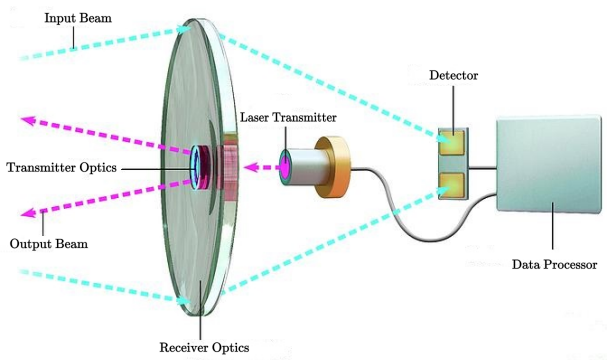


Figure 8.1. FSO transceiver representation. The optical part of the device is the receiver optics (centre left). The input beams (cyan) are focused by the optics onto the detector (centre right), which then passes the signal to the data processor (far right). Outgoing data is passed from the processor to a laser transmitter (centre), which sends the data as output beams (pink) via the transmitter optics (centre left). Background image taken from [223].

encoded onto the carrier signal at much higher frequencies. This enables a more significant number of data bits to be transmitted per unit of time.

Moreover, laser communication systems can exploit a larger portion of the electromagnetic spectrum, including multiple wavelength channels, to transmit data simultaneously. This multiplexing capability increases the total data capacity of the communication link.

Other advantages of FSO systems over the RF counterpart rely on (i) the smaller divergence of the laser beam, which enables a higher concentration of optical power [225], (ii) lower interference thanks to a point-to-point communication with a direct line of sight [226], (iii) lower latency over longer distances [227], and (iv) more robust security due to the inherent difficulty to intercept FSO signals without being located precisely in the path of the beam [228].

However, FSO communication systems come also with drawbacks and limitations related to atmospheric turbulence. The latter can significantly impact the performance of FSO system, causing most of the time (i) scintillation of the received optical signal [229], (ii) beam wander [230], and (iii) beam divergence [231].

In general, the effects of adverse weather conditions on FSO systems become more pronounced as the distance between the transmitter and the receiver increases. This is why, in the satellite telecommunications domain, a FSO link between an OGS and a satellite is typically operated by means of LEO satellites.

Hence, the main problem in the FSO domain relies on defining control strategies to mitigate the laser signal degradation effects brought by thick clouds, fog, rain, and other similar inclement weather conditions.

8.4 The HyDEMO Project

As shown in the previous sections, laser–based satellite communication technology holds the potential to extend terrestrial network functionalities to satellite networks, addressing the digital connectivity challenges across a range of applications. These applications encompass (among the others) virtual private networks, edge computing, advanced 5G/6G services, internet connections to and from space, and communication with airborne assets. These applications go beyond the current capabilities of satellites. The High Throughput Optical Network (HydRON) project, launched by the ESA aims to develop these new technologies in space for European and Canadian industries [232].

HydRON is part of ESA's Advanced Research in Telecommunications Systems (ARTES) 4.0 Strategic Programme Line, specifically under the *Optical & Quantum Communications–ScyLight* program. This vision introduces an Optical Transport Network concept that combines extremely high throughput Optical Ground Space and Optical Inter-Satellite Links, with in-orbit routing capabilities that seamlessly integrate into existing terrestrial networks.

The primary objective of the HydRON Demonstration System is to advance the development and validation of HydRON technology integrated into terrestrial networks, with a capacity of terabits per second. The demonstration system will encompass the end-to-end network, including critical key technologies and a minimum viable service. It will showcase networking capabilities with seamless interoperability with high-capacity terrestrial networks and will present an operational concept reflecting a scalable HydRON framework. The latter will include two space-based laser communication payloads in LEO and GEO satellites, interconnected with each other, along with several OGS and terrestrial fiber optic networks.

Contents in the next two chapters have been inspired by the research activities the author of this thesis has been carried out in the context of HydRON.

Chapter 9

Intelligent Ground Station Selection in GEO Optical Communication Systems

One of the most frequent problems of satellite communications is bad weather conditions. In such a situation, any communication going from the satellite to the ground may suffer significant interference [233]. Since geostationary satellites cover very widespread geographical areas, it is possible to exploit the different weather conditions of each zone covered by the satellite. In order to limit bad weather effects, two or more ground stations (GS) receiving the same satellite signal may be linked, so that if the signal suffers some attenuation in an area, the other ground stations, located in areas where the weather is favourable, may compensate said attenuation. The communication loss is mitigated by continuously forwarding the signal to the OGS(s) under untoward weather conditions, at least until the latter improve. This technique is called site diversity [234].

9.1 The Site Diversity Technique

Site diversity refers to the practice of using multiple geographically diverse ground stations or receiving sites to improve the reliability and availability of communication links in the context of telecommunications and satellite communications. This strategy is crucial for systems that require high availability and minimal downtime, such as satellite communication systems, and is particularly important for critical applications such as military and emergency services communications.

Site diversity is frequently used in satellite communication as part of a larger strategy to maximize link availability and minimize signal degradation. It is par-

ticularly important for systems requiring continuous, high-reliability connectivity, such as military, emergency response, and critical infrastructure communications. When combined with advanced satellite tracking and switching technologies, site diversity can provide seamless and reliable communication.

When the communication channel is optical, we refer to optical site diversity and optical ground stations (OGS). In this setting, the communication is switched from the satellite to one of the other OGSs for the needed amount of time. In order to design in an accurate way the site diversity technique, a statistical analysis is needed to determine the probability of rain events in a given area. In literature, this assessment has been done either through direct measurement campaigns [235, 236] or by exploiting statistical models. Among them, the most used ones are the log-normal model for the single-site distribution of rainfall intensity [237] and the Monte Carlo simulation for the prediction of join statistics of rain attenuation in multiple locations [238].

Together with the choice of a statistical model for rain prediction, some metrics are needed to define the performance of the site diversity technique against the nominal downlink/uplink optical communication between the OGS affected by rainy conditions and the satellite itself. The site diversity performance is usually measured as a function of several parameters, including baseline orientation, communication link frequency, path elevation angle and site separation [239].

In multi-station site diversity scenarios [238], it is possible to establish a link between the weak OGS and one of the others: this makes it necessary to choose the station from which to broadcast. Of course, the choice of which OGSs to point at any time instant is driven by a series of KPIs and follows an optimal routing/resource allocation logic [240, 241]: the most important one is the link availability (i.e., the probability that both the optical links are not working should be minimized), but other design drivers for the multi-station site diversity algorithms include the energy consumption for the movement/re-pointing of LCTs, which impacts on the total power budget for the on-board payload, the topology of the OGSs network (i.e., specific OGSs network topologies may prevent the possibility to re-route the user traffic from one OGS to the others, thus limiting the subset for choosing the second OGS to be the second LCT pointed towards), user plane latency and jitter (i.e., specific application may require stringent latency requirements and/or minimal jitter; this may prevent some OGSs to be selected as backup optical link) and on-board switching capabilities (i.e., the impossibility for the switching matrix on board of the satellite to switch traffic from one LCT to the other in case of handover). Moreover, the installation of redundant OGSs represents a waste of investment for the network operator. Hence, recently, several works focused on the minimisation

of the number of required OGSs to guarantee a given system performance (e.g., availability).

In [242] a two-parts joint optimisation method is proposed for ground stations' positioning and backbone network improvement, whereas authors in [243] use genetic algorithms to minimise two objective functions in high-frequency satellite networks. A more rigorous mathematical formalisation of the optimisation problem is given in [244], where authors exploit a branch and bound algorithm. A different optimisation approach relies on the hypothesis that OGSs have been already positioned and the problem focuses on how to choose the set of OGSs to connect to in order to maximise the availability. In [245] authors calculate the correlated and uncorrelated availability for OGS networks in the scenario of space-to-ground optical communication links with GEO satellites. An efficient optimisation algorithm is presented, in order to choose the best OGS starting from five years of cloud data. It is shown how many OGSs deployed in a very wide area can guarantee a network availability near to 100%. A complementary optimization approach is proposed in [246], with the selection of the minimum number of ground stations satisfying the monthly availability requirements of the total network, so minimizing service and maintenance costs. Eventually, in the scenario presented in [247], the optimisation process consists in selecting the best ground station among several candidates, trying to provide a reliable connectivity through large-scale site diversity. Results show that the optimal choice mostly depends on the altitude and the zenith angle of the set of ground stations.

Unlike the aforementioned existing works in literature, this article makes use of a deep learning-based weather forecasting algorithm [248] to define a predictive handover strategy. The latter's main features are:

- proactive decisions based on weather forecasts. This approach allows to consistently reduce the time window in which the system is unavailable for transmitting user traffic;
- exploitation of limited on-board computing capabilities, with the installation of a ready-to-go deep learning models;
- customization of artificial intelligence models with respect to each OGS area, to improve prediction accuracy and, consequently, the availability of the handover service;
- implementation of an automatic and dynamic switching between the LCTs mounted on the satellite.

Hence, the proposed control framework can be used in a fully online fashion to decide in advance the backup rotation and switching operations between the LCTs.

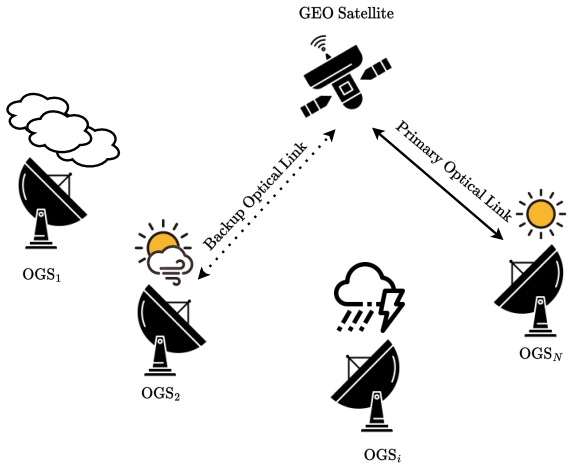


Figure 9.1. Communication between a set of OGS and a GEO satellite.

The next section will detail the main features of the system scenario and the related control problem.

The contents of this chapter entirely rely on the work carried out in [249].

9.2 Problem Modeling

In optical satellite telecommunication systems, the visibility between Optical Ground Stations (OGSs) and on-board Laser Communication Terminals (LCTs) is fundamental to successfully transmit data. This means that, as the weather conditions change, a proper site-diversity technique has to be applied to guarantee service availability. Indeed, only favorable weather conditions can guarantee the necessary SNR for such type of communications, while clouds may completely block the laser signal.

Here we consider a set of N terrestrial OGSs, located far away one from each other, and a single geostationary satellite equipped with 2 LCTs (see Fig. 9.1). One of the LCTs is active and transmits/receives user data to/from the OGS it is pointing to. We refer to this first LCT as the *primary* one. The other LCT, called *backup*, instead, points towards a different OGS and it is not used for the transmission of user traffic, even if it is ready to handover the primary LCT in case it is needed. Moreover, the backup LCT has the capability of dynamically changing its pointing direction by means of rotations of the terminal itself, a feature that will constitute the core of the developed control strategy.

The OGSs on the ground are supposed to be interconnected (so realizing a traditional site diversity scheme) and to be able to handle handover procedures in case of switching between primary and secondary LCTs. Each OGS zone could

be characterized by the most diverse weather conditions: from sunny to overcast cloudy, from hazy to foggy and from rainy to snowy. The proper transmission of user traffic through the primary LCT is not possible in case of inclement weather conditions (rain, thunderstorm, snow, fog, and others), whereas, on the contrary, the communication between the GEO satellite and the primary LCT–pointing OGS is considered to be viable under mild weather conditions, such as sunny or partly cloudy skies.

9.3 Proposed Deep Learning Control Strategy

The first phase of the strategy hereby presented relies on the exploitation of a set of area-tailored Deep Learning models for weather forecasting. The AI machinery, which is a supervised one, takes as input historical numerical weather data characterizing each OGS geographical zone. Each data is labeled with the actual weather condition at that specific time, with a temporal resolution T_R that must be compliant with the time constants characterizing the LCTs' rotation and switching operations. The goal is to predict future weather conditions by looking at the meteorological data in the previous hours or days, trying to find a pattern between the available features (atmospheric pressure, temperature, etc.) and the weather condition.

To achieve this aim, the most promising neural network structure is the Recurrent Neural Network (RNN).

9.3.1 Recurrent Neural Networks

An RNN is a type of artificial neural network designed for processing time sequences of data. Unlike traditional feedforward neural networks, which have a fixed architecture, RNNs are equipped with loops or recurrent connections that allow them to store memory about previous inputs [250]. This memory enables RNNs to process sequences of data, such as time series, natural language, or any other data with a temporal or sequential structure.

The long-short term memory (LSTM) network was first suggested in [251] to address the well-known problem of vanishing gradient that characterises RNNs. The LSTM structure is therefore ideally adapted to handle time-series data, such as the one we are addressing in our work. By specifying a certain time window $\mathcal{T}_p$ of length T_W, the AI model tries to predict weather at time $k+1$ by looking at the actual weather encountered in the T_W previous time instants, i.e.:

$$\mathcal{T}_p = \Big\{k, k-1, \ldots, k-T_W-1\Big\}. \tag{9.1}$$

At their core, LSTMs are comprised of memory cells that enable them to store and manipulate information over extended sequences. These memory cells have three crucial components:

1. Cell State: it is like a conveyor belt that runs through the entire LSTM network. It can transport information across time steps without much modification. The cell state can be updated, allowing it to capture relevant information and discard irrelevant details.
2. Hidden State: also known as the output state, carries information from previous time steps to the current one. It acts as a working memory that helps LSTMs remember past information that is crucial for making predictions or decisions.
3. Gates: LSTMs employ three types of gates to control the flow of information:
    - Forget Gate: this gate decides what information from the cell state should be discarded or kept. It takes as input the previous hidden state and the current input and outputs a value between 0 and 1 for each component of the cell state, where 0 means *forget* and 1 means *keep*.
    - Input Gate: this gate determines what new information should be added to the cell state. It computes a candidate cell state and decides which parts of it should be added to the current cell state.
    - Output Gate: the output gate controls what information should be output as the hidden state. It takes the current cell state and the input, and it generates the new hidden state.

In this work, each LSTM network is trained on local OGS meteorological data, because if not so it would be difficult for a single predictor to generalize across the various climates of the OGSs' geographic regions, which can actually be located at very different latitudes.

9.3.2 Control Logic

The second control phase, which is the online one, begins as soon as all the models have been trained: they are deployed on board the satellite, ready to make inference on future weather data. The inference at time k estimates the probability that the considered OGS will be under bad weather conditions at time $k+1$. Consequently, the OGS will be monotonically increasingly sorted with respect to the computed probabilities.

Let OGS_p^k and OGS_b^k be the OGSs towards which the GEO satellite points the primary LCT and backup LCT, respectively, at time k. Moreover, let OGS_*^k be the

OGS with the lowest predicted probability for having bad weather (not considering the two OGSs mentioned before) and let p_p^{k+1}, p_b^{k+1} and p_*^{k+1} the predicted rain probability for the primary LCT OGS, the backup LCT OGS and the lowest probability, respectively. The control strategy exploits two tunable parameters, namely τ_1 and τ_2, in order to perform backup rotations and primary-backup switchings. The proposed control algorithm is summarized in 11 and the correspondent functional architecture is depicted in Fig. 9.2.

Algorithm 11 Control Strategy for Site Diversity

Inputs: Actual weather data at each time k

Initialize OGS_b^0 and OGS_p^0
for each time instant k **do**
  predict weather conditions $p_i^{k+1},\ \forall i = 1, \ldots, N$
  if $p_b^{k+1} \geq \max\{\tau_1, p_*^{k+1}\}$ **then**
    rotate backup LCT from OGS_b^k to OGS_*^k
    $p_b^{k+1} \leftarrow p_*^{k+1}$
    $\text{OGS}_b^{k+1} \leftarrow \text{OGS}_*^k$
  else
    $\text{OGS}_b^{k+1} \leftarrow \text{OGS}_b^k$
  end if
  if $p_p^{k+1} \geq \max\{\tau_2, p_b^{k+1}\}$ **then**
    $\text{OGS}_p^{k+1} \leftarrow \text{OGS}_b^{k+1}$
    $\text{OGS}_b^{k+1} \leftarrow \text{OGS}_p^k$
  else
    $\text{OGS}_p^{k+1} \leftarrow \text{OGS}_p^k$
  end if
end for

9.4 Simulations and Results

9.4.1 Simulation setup

In order to demonstrate the capabilities of the hereby designed control algorithm, the LSTM deep neural network was trained on a publicly available weather dataset covering approximately 5 years of weather data (from October 1, 2012 to November 30, 2017), with temporal resolution $T_R = 1\,\text{h}$, for several cities in the USA and Canada [252]. The available features for training are the following:

- humidity;
- atmospheric pressure;
- wind direction;

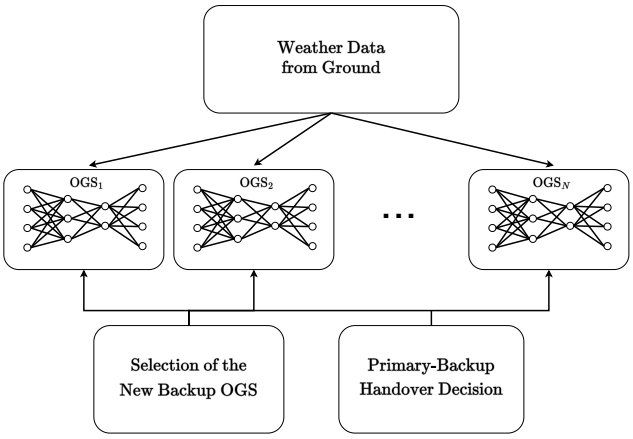


Figure 9.2. Sketch of the Proposed Control Architecture.

- temperature;
- wind speed;
- month of the year;
- weather conditions within the time window prediction $\mathcal{T}_p$.

The missing data for each feature was filled in by taking up the numerical value of the feature of the previous entry: this approach makes it possible not to break the hourly time sequence of the meteorological data.

As per the weather, the dataset contains a very detailed description of the weather conditions. The latter have been mapped into binary labels for training the model: label 0 has been assigned to clear sky, few/scattered clouds and haze, which correspond to mild weather conditions allowing satellite–OGS communication, whereas the label 1 indicates inclement weather.

For the training phase, the data from October 1, 2012 up to December 20, 2016 have been selected. The model accuracy has been evaluated by splitting the remaining part of the dataset with respect to the four seasons, in accordance with the 2016 and 2017 astronomical tables [253].

The chosen LSTM model architecture is depicted in Fig. 9.3 and the selected hyperparameters are the following:

- number of epochs $E = 5$;
- Adam optimizer with constant learning rate $\eta = 0.001$;
- time window length $T_W = 24$;
- dropout rate $\zeta = 0.2$.

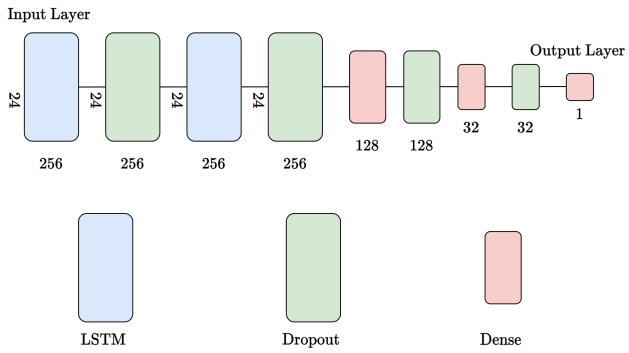


Figure 9.3. LSTM Neural Network Architecture.

The LSTM model performance has been compared to the accuracies of other three state-of-the-art machine learning models:

- support vector machine (SVM) with linear kernel;
- standard feedforward neural network (NN), having as architecture the same as the one occurring downstream the two LSTM layers of the LSTM deep learning model;
- a single standard feedforward neural network (SNN) trained on the data of all locations. In this case, the satellite would use just one model instead of having one of them for each OGS location.

As per the evaluation of the control strategy, the following metrics/KPIs have been defined:

- link availability L_A, defined as

$$L_A = \frac{n_s}{n_t}, \tag{9.2}$$

  where n_s is the number of times the primary LCT is able to handle user traffic and n_t is the whole time period length used in the simulation;
- outage probability $O_P = 1 - L_A$;
- number of rotations R made by the backup LCT;
- number of switching S between primary and backup;
- number of outages due to wrong predictions n_W, i.e., the number of times the primary LCT is pointed towards an OGS that is under adverse weather conditions and for which the AI model has predicted instead a favorable weather condition.

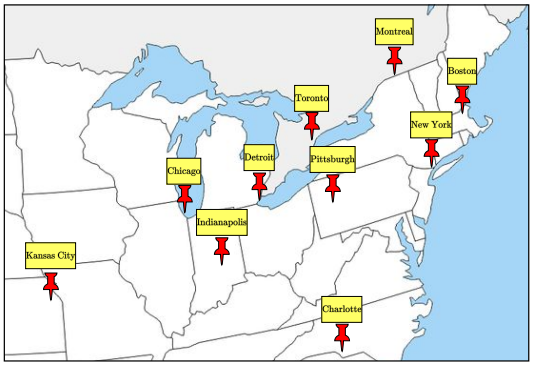


Figure 9.4. Map of the ten locations in the north-east side of the American continent.

The experiment was run with parameters $\tau_1 = 0.1$ and $\tau_2 = 0.2$ in Algorithm 11.

The dataset has been used to perform two groups of simulations, characterized in the following subsections.

The goal of the simulations of our proposed control system is to show that it does not deteriorate the system availability, i.e., by do not selecting an OGS with unfavorable weather conditions when there is at least one between the remaining ones that has good weather conditions.

9.4.2 10–Cities Simulation

A set of 10 cities in the north-east of North America continent has been selected as candidate for locating OGSs, as depicted in Fig. 9.4. The number of cities and their relative distances were chosen as to replicate the EU-99 topology studied in the early phases of the HydRON project [254].

The training phase was performed on a computer equipped with a NVIDIA GeForce RTX 3050 and 16GB RAM, using Python3.8 and the Tensorflow/Keras libraries [255]. The training computational time when using the LSTM approach is around 13 s for each city. Similar times are required by the other approaches. This property suggests that either the training phase may be performed online, for instance when an update of the models with respect to new weather data is required.

The LSTM model performance against all seasons, in terms of test accuracy, have been reported in Tab. 9.1.

We show the comparison between the four AI architectures considering just the test accuracies obtained in the spring test set, since equivalent conclusions can be drawn for the other three seasons. In Fig. 9.9 it is possible to notice that the LSTM–based model outperforms the other three ones in every considered locations, and, as expected, the SNN model has worse performance with respect to the NN model, trained in a tailored way per each OGS location.

Table 9.1. LSTM accuracies per season

City	Winter	Spring	Summer	Autumn
New York	0.984	0.986	0.982	0.989
Montréal	0.988	0.981	0.963	0.974
Boston	0.993	0.989	0.987	0.989
Chicago	0.973	0.978	0.952	0.971
Charlotte	0.976	0.976	0.972	0.975
Pittsburgh	0.991	0.984	0.979	0.988
Detroit	0.997	1.000	0.996	0.998
Kansas City	0.992	0.990	0.983	0.990
Toronto	0.999	0.999	0.985	0.988
Indianapolis	0.992	0.986	0.981	0.992

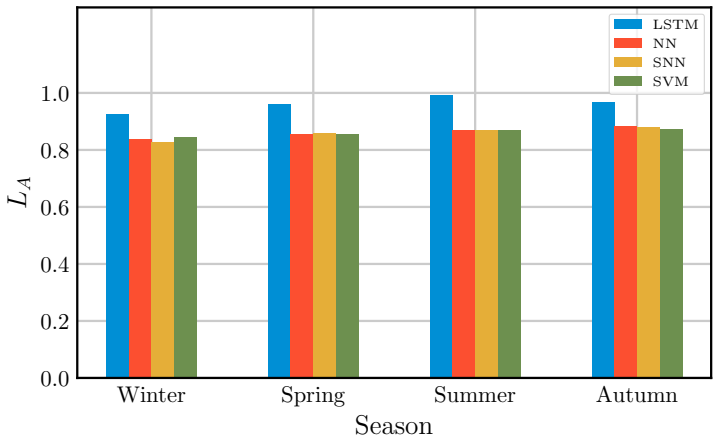


Figure 9.5. Link availability – L_A.

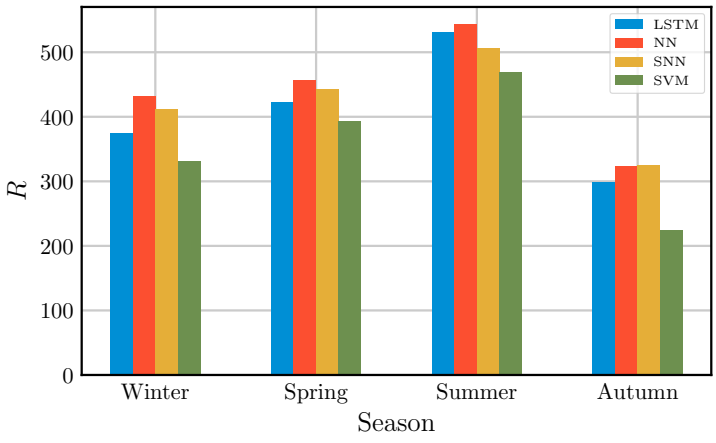


Figure 9.6. Number of rotations – R.

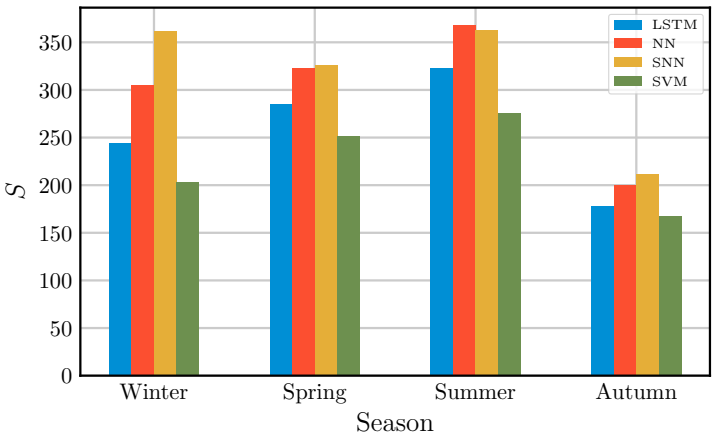


Figure 9.7. Number of switchings – S.

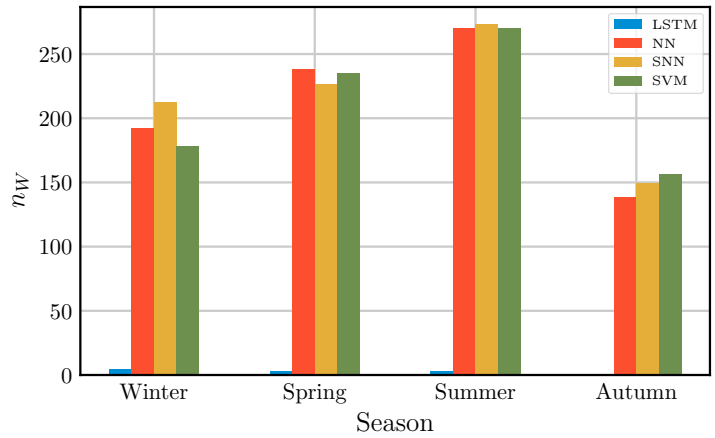


Figure 9.8. Number of outages due to wrong predictions – n_W.

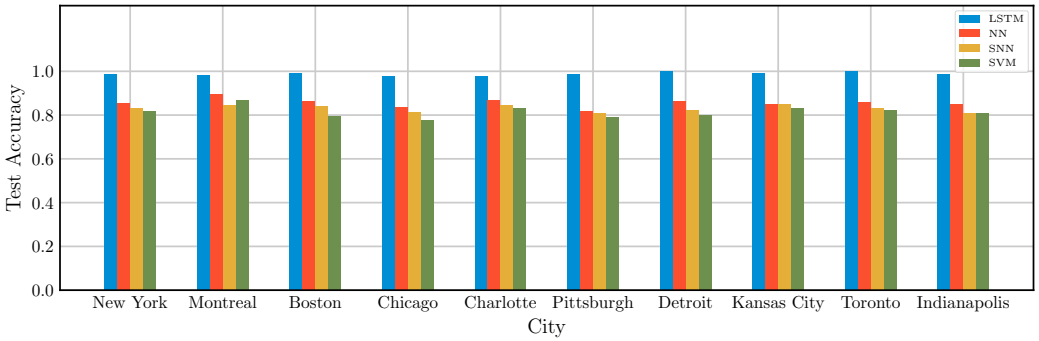


Figure 9.9. Comparison between test accuracy for the four considered model in the spring season.

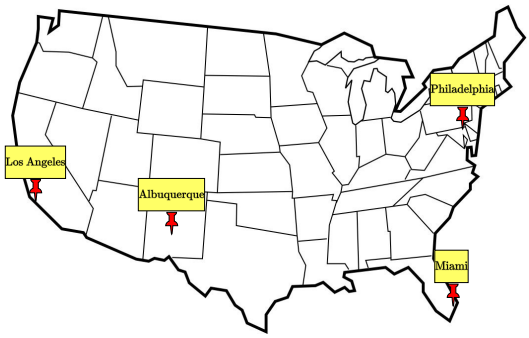


Figure 9.10. Map of the four USA locations tackled in the second simulation.

The results with respect to the above-mentioned KPIs, obtained with the four AI models architectures, are revealed in Fig. 9.5-9.8. It is possible to notice that when using the LSTM model, the GEO satellite performs more rotations and switchings, but achieves the highest score with respect to L_A and n_W. In particular, with the LSTM model, almost no outages are due to a misprediction of the weather conditions of the next timestep, leading to the maximum possible availability for the chosen set of OGSs.

9.4.3 4–Cities Simulation

In the second simulation we addressed 4 USA locations (see Fig. 9.10), different from the previous ones. For such 4-cities topology a dedicated availability study has been provided by the OT4NGsat project [256]. The results obtained with the LSTM approach are evaluated during the 2017 spring season with respect to the outage probability KPI, namely O_P. Fig. 9.11 represents a graph with semilog scale on the y–axis in which we first plot the outage using only one OGS located in Los Angeles, secondly the one obtained adding Albuquerque and so on, up to the fourth OGS, located in Philadelphia. From the figure it is possible to see that the final result, when using all the four locations, is $O_P = 0.03$. The latter matches perfectly with the theoretical availability study provided by OT4NGsat, which foresees $O_P = 0.033$ when using 4 OGSs.

9.5 Discussion and Future Works

This work tackled the problem of site diversity in optical satellite communication, in order to maintain high-availability even in case of unfavorable weather conditions at some OGSs. In this work we considered a scenario composed by a GEO satellite equipped by 2 LCTs, one of which is active and the other one is used as backup. The goal of this paper is to proactively select the best OGSs to be pointed at

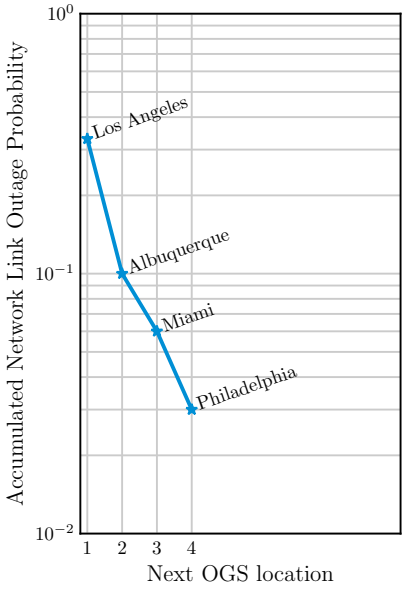


Figure 9.11. Outages when using one, two, three and four OGSs, respectively.

any point in time and to switch between active and backup in case bad weather is envisaged in the next timestep. The proposed algorithm for site diversity is based on Deep Learning weather forecasting for selecting the best OGSs to point to and for deciding the handover between active and backup LCT. The training phase is conducted offline, whereas the inference models of each OGS area can be mounted onboard the satellite, allowing a hard real–time implementation with hourly time resolution.

The proposed algorithm has been simulated and tested against historical weather data in two different simulative scenarios. From the results it is possible to notice that the proposed algorithm is able to succeed in reducing the number of outage events, so guaranteeing a wide time window link availability.

Future works may consider to model the energy requirements for the rotational movement of the LCTs, so as not to exceed a total on-board energy budget. Moreover, a similar approach could be investigated for MEO/LEO satellites and for multiple (and even inter-satellite) links.

Chapter 10

Data Path Control for LEO Satellite–Driven Communications

In today's ever-connected world, the demand for high-speed, reliable, and secure data transmission has never been greater. The proliferation of data-intensive applications, such as streaming video [257], cloud computing [258], and the Internet of Things (IoT) [259], continues to place unprecedented strain on traditional communication networks. To meet these growing demands, the development of innovative communication technologies has become imperative. Among them, FSO has emerged as a promising solution to address these challenges, as broadly discussed in the previous chapter.

10.1 The Synergy Between FSO and LEO Satellite

LEO satellites play a pivotal role in the advancement of FSO communication. Situated at altitudes between 180 and 2,000 km above the Earth's surface [260], LEO satellites have relatively short orbital periods, typically completing one orbit around the Earth in 90-120 minutes [261]: this frequent orbiting allows for better coverage and faster data transmission.

Due to their relatively low altitude, atmospheric effects, such as signal attenuation due to rain fade and atmospheric turbulence, have a reduced impact on a FSO system compared to GEO satellites. This results in more reliable and consistent FSO communication links, characterized by low latency, high data throughput and improved signal strength, making them a preferred choice for real-time, high-data-rate FSO applications [262].

Moreover, this type of satellite can be deployed in large constellations [263, 264], which provide continuous global coverage and improve the overall reliability of the FSO communication. This aspect is crucial for applications that require uninterrupted connectivity, such as satellite-based internet services.

LEO satellites are commonly used in the space industry for scientific research and Earth observation purposes and for military operations, as well as for communication, navigation, and remote sensing applications [265].

On the other hand, LEO satellites present two main disadvantages compared with the GEO ones. The first one deals with their shorter lifespan, requiring periodic orbital adjustments to maintain their position in the orbit, or replacements of units within the fleet [266, 267]. The second one is related with visibility issues: since LEO satellites' rotation speed is much higher than the Earth's rotational speed, FSO terrestrial signals have to be handed over to another satellite within the fleet. A satellite handover is performed when the serving satellite is below a minimum elevation angle relative to the corresponding OGS: this may have a significant impact on the communication quality, because of communication loss during the handover process [268].

Despite these downsides, LEO satellites represent the ideal technology for optical satellite communications, problems related to laser signal attenuation in the presence of adverse atmospheric conditions remain. To address these technological difficulties, various methodologies have been proposed by the scientific community.

10.2 Related Works

Researchers and engineers have created a variety of strategies and technologies to solve the problems of power attenuation in FSO systems due to atmospheric fading. A standard procedure rely on adaptive optics [269, 270]. Systems implementing this technology correct for turbulence-induced distortions by changing the geometry of optical components like mirrors or deformable lenses based on real-time observations of air turbulence. This technique aids in optical beam stabilisation and minimises scintillation effects.

Other techniques rely on filtering and error correction, in which proper filters and modulation methods try to filter out noise coming from the interference of fog or clouds [271]. Since in many situations it is not possible to filter out the noise, it is possible to employ broader laser beams to reduce the effects of beam spreading brought on by turbulence [231]. This strategy, nevertheless, could result in slower data transfer rates [272]. Eventually, another common standard approach is to implement redundant FSO lines, equipping satellites or OGSs with more than one

laser communication terminal (LCT) [249], or FSO/RF hybrid systems [273], in order to enhance the communication system reliability.

The aforementioned fading mitigation techniques intervene at the hardware level on the individual receiver or transmitter, but do not take into consideration any changes to the architecture or topology of the communication system.

In order to limit bad weather effects, it is possible to intervene at architecture level linking in a wired fashion two or more OGSs within a same network. In this way, if the signal suffers some degradation in an area, the other OGSs, located in areas where the weather is favourable, may compensate said attenuation. The communication loss is mitigated by continuously forwarding the signal to the OGS(s) under untoward weather conditions, at least until the latter improve. This technique is called site diversity [234].

The site diversity has proven to be a disruptive approach for the reliability of FSO communications, since it (i) enables geographical diversity to reduce the likelihood of simultaneous signal degradation at all sites [234], (ii) realises spatial separation to ensure that the OGSs locations are subject to different weather patterns and atmospheric conditions [274], and (iii) involves using multiple antennas at each site, pointing in different directions or at different elevation angles. This configuration allows the system to quickly switch between antennas to find the clearest signal path, thus improving the link availability and reducing the number of outages or dead times [275].

Although the site diversity technique adds complexity and high cost to the infrastructure, the benefits in terms of improved reliability and availability often justify its implementation, particularly for mission-critical applications.

Said technique employs sophisticated control and switching mechanisms to monitor the quality of signals received at different sites in real-time. When one site experiences signal degradation, the system automatically switches to an alternate site with better signal quality. These switching mechanisms were initially manual and human-driven, while nowadays are usually based on statistical analysis of weather forecasts [235–238], with the switching system being controlled by intelligent algorithms.

The choice of which OGS to point at or to transmit from is driven by a series of Key Performance Indicators (KPIs) and follows an optimal routing/resource allocation logic [240, 241]. The most important KPI for any satellite communication system is the link availability, but other design drivers for the multi-station site diversity algorithms may include (i) the energy consumption for the movement/repointing of LCTs, which impacts on the total power budget for the on-board payload, (ii) the topology of the OGSs network (i.e., specific OGSs network topologies

may prevent the possibility to re-route the user traffic from one OGS to the others), (iii) user plane latency and jitter, and (iv) on-board switching capabilities.

Since the installation of redundant OGSs may represent a waste of investment for the network operator, several works focused on the minimisation of the number of required OGSs to guarantee a minimum given system performance [242–244]. A different optimisation approach relies on the hypothesis that OGSs have been already positioned and the problem focuses on how to choose the set of OGSs to connect to in order to maximise the availability. In [245] authors calculate the correlated and uncorrelated availability for OGS networks in the scenario of space-to-ground optical communication links with GEO satellites. An efficient optimisation algorithm is presented, in order to choose the best OGS starting from five years of cloud data. It is shown how many OGSs deployed in a very wide area can guarantee a network availability near to 100%. A complementary optimisation approach is proposed in [246], with the selection of the minimum number of ground stations satisfying the monthly availability requirements of the total network, so minimising service and maintenance costs. Eventually, in the scenario presented in [247], the optimisation process consists in selecting the best ground station among several candidates, trying to provide a reliable connectivity through large-scale site diversity. Results show that the optimal choice mostly depends on the altitude and the zenith angle of the set of ground stations.

Recent advancements in artificial intelligence (AI), particularly in the field of Reinforcement Learning (RL), have opened up new possibilities for optimising satellite communication strategies. Authors in [249] propose an AI-based predictive handover strategy for optical communications between a GEO satellite equipped with two LCTs and an OGS network, making use of machine learning–based weather forecasts. Other works making use of AI and RL focused on the resource allocation and traffic splitting for RF satellite communications [276–279], shifting attention from the OGS network level to the one of the LEO constellation.

However, none of the above-mentioned works have tackled the issue of defining an intelligent handover and path planning procedure for a FSO–based point-to-point communication system between terrestrial OGS networks and a LEO fleet.

This chapter will combine the LSTM–based weather forecast technique presented in 9.3.1 with a centralized Q–Learning controller, in order to tackle the problem of minimizing the outage probability. The main innovations of the present work are:

- multi–hop data routing between two OGS networks that cannot communicate directly, but only passing through a LEO satellite fleet;
- weather predictions over the OGSs areas via Supervised Learning exploiting historical hourly weather data;

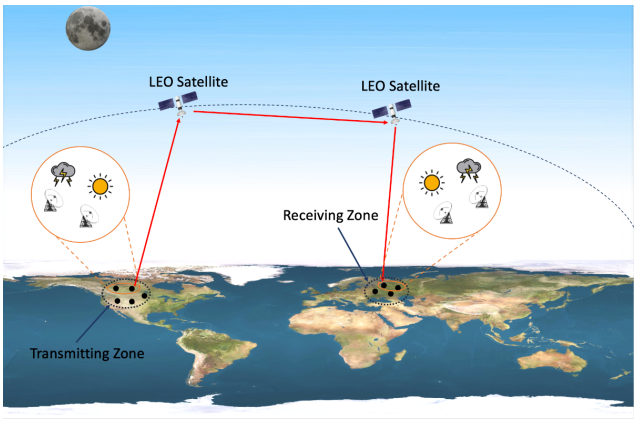


Figure 10.1. System Scenario.

- a centralised control law realised through an intelligent agent exploiting the RL framework with an intrinsic optimisation of the link availability.

The subsequent section within this chapter will detail the mathematical modeling of LEO satellites' and Earth's dynamical motion, with a control approach based on the combination of Long Short–Term Memory (LSTM) networks and centralized Q–Learning. Extensive simulations on three case studies will show the effectiveness of the proposed approach with respect to other benchmark solutions.

The contents of this chapter entirely rely on the scientific article issued in [280].

10.3 Mathematical Modeling of a Multi–Hop LEO–Driven Data Transfer

Let us consider a FSO–like communication system made by two OGS networks, one transmitting data from N_{tr} OGSs and the other one acting as receiver with N_{re} stations. Each of the two zones can be subject to different atmospheric conditions, going from sunny to cloudy to stormy, which affect the data delivery from the transmitting to the receiving zone. The two sets of OGSs cannot communicate using terrestrial wired or wireless technologies, but they must rely on a LEO constellation composed of N_{sat} satellites. The communication is a point–to–point one realised through laser beams. The system scenario is depicted in Fig. 10.1.

In what follows, a detailed mathematical modeling of the overall communication system is presented, including the formulation of the orbiting LEO satellites equations of motion, the ground–to–satellite and inter–satellite visibility assessment, and the MDP characterization.

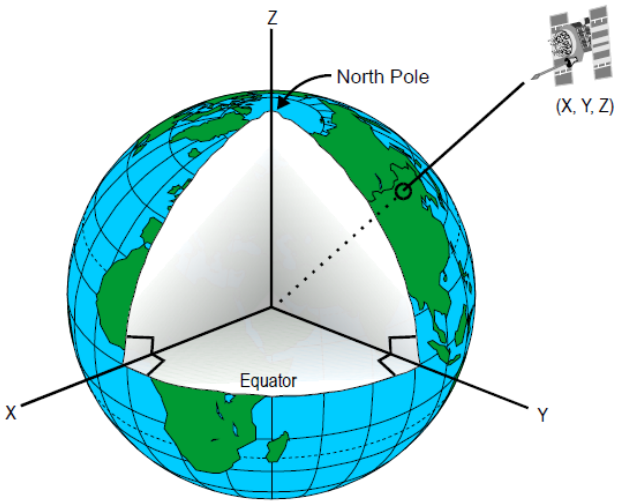


Figure 10.2. Earth Centered Inertial reference frame.

10.3.1 Satellite equations of motion

Low Earth Orbit satellites are considered one of the best options for satellite communication due to their short orbital period, which provides wide coverage and an high service availability.

In order to define a LEO constellation, the orbit of each satellite must be characterized. In this work we suppose that each satellite follows exactly its initial orbit with deviating from it with time[1]. Given an inertial frame of reference and an arbitrary epoch (a specified point in time), exactly six parameters are necessary to unambiguously define an arbitrary and unperturbed orbit. These are the semi-major axis a, the eccentricity e, the inclination i, the argument of perigee ω, the longitude of the ascending node Ω, also denoted as the Right Ascension of the Ascending Node (RAAN) for geocentric orbits, and the true anomaly f [281], [282].

The orbital parameters can be used to compute, at every epoch, the position and velocity of the satellite around that orbit. To describe the motion of satellites, it is usually used a coordinate frame which is inertial and fixed with respect to the stars, namely the Earth Centered Inertial (ECI) reference frame [283]. In particular the x–y plane coincides with the equatorial plane of Earth. The x-axis is permanently fixed in a direction relative to the celestial sphere, which does not rotate as Earth does. The z-axis lies at a 90° angle to the equatorial plane and extends through the North Pole (see Fig. 10.2).

[1]In the real world, satellites progressively abandon their initial orbit due to the presence of external variable forces like solar radiation pressure. Hence, they have a certain lifetime after which they shall be replaced.

Let us define as $R_x(\phi)$, $R_y(\eta)$ and $R_z(\psi)$ the standard rotation matrices:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{10.1}$$

$$R_y(\eta) = \begin{bmatrix} \cos\eta & 0 & -\sin\eta \\ 0 & 1 & 0 \\ \sin\eta & 0 & \cos\eta \end{bmatrix} \tag{10.2}$$

$$R_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{10.3}$$

Algorithm 12 shows how to pass from the orbital parameters to the satellite position and velocity in the ECI coordinates.

Algorithm 12 Orbital Parameters to ECI coordinates

Inputs: $a, e, \Omega, i, \omega, f$
Parameters: $\mu = 3.986004418 \times 10^{14}[m^3/s^2]$
Outputs: r, v
$p = a(1 - e^2)$ {semilatus rectum}
$cf = \cos f, sf = \sin f$
$r = p/(1 + e(cf))$ {Safe Division}
$v = \sqrt{mi/p}$ {Safe sqrt and safe division}
Define a rotation matrix based on angles and axes
$\text{ang} = \begin{bmatrix} \omega & i & \Omega \end{bmatrix}^T$
$\text{axes} = \begin{bmatrix} 3 & 1 & 3 \end{bmatrix}^T$
$M = \text{ang2mat}(\text{ang}, \text{axes})$
Compute position and velocity in ECI
Transpose M
$r \leftarrow rM \begin{bmatrix} cf & sf & 0 \end{bmatrix}^T$
$v \leftarrow vM \begin{bmatrix} -sf & e + cf & 0 \end{bmatrix}^T$

At this point it is possible to define the satellite equations of motion as a second-order differential equation which is dependent on the satellite position vector r:

$$\ddot{r} = -\mu \frac{r}{\|r\|^3}, \tag{10.4}$$

where $\|r\|$ is the euclidean norm of the position vector and $\mu = 3.986004418 \times 10^{14}$ $[\text{m}^3/\text{s}^2]$ is the geocentric gravitational constant.

10.3.2 Visibility Analysis

In order for the satellite to exchange information with an OGS or with another satellite there is a condition which needs to be analysed, the visibility. The latter is a very important concept since it can determine if a certain information exchange can happen or not and how good is the communication channel in terms of noises. Visibility can be of two kind: (i) geometric visibility, which is related to the fact that the relative position vector between one satellite and the other does not have to intersect the Earth, and (ii) electronic visibility, which deals with analysing the elevation angle and Carrier to Noise ratio (C/N0). The angle of elevation is the angle between the horizontal line and the line of sight which is usually above the horizontal line. The C/N0 expresses how high is the noise component with respect to the information carrier: the lower the ratio is, the more the noise is prevalent and vice-versa.

Since this work does not focus on the quality of the communication link, some assumptions have been made to simplify the analysis:

1. The information exchanged between the satellite and the OGS and between one satellite and another one is always good with a negligible amount of noise.
2. The satellite is visible by the OGS if the elevation angle is greater than a certain threshold, in order to exclude the case of interference of buildings in the vicinity of the OGS.
3. The satellite is visible with respect to another one if the geometric visibility condition is satisfied.

In the following subsections the implementation of the visibility algorithms related to assumptions 2 and 3 will be detailed.

Ground Station to Satellite Visibility

As already explained, a satellite is considered visible from an OGS if the elevation angle is above a certain threshold. The elevation angle is computed with respect to the horizontal plane of the OGS, so the East-North-Up (ENU) coordinate frame has been considered, which is the reference frame of the ground station's antenna. This implies a change of coordinates of the satellite position and velocity vectors from the ECI reference frame to the ENU frame. This transformation can be performed by applying two rotations to the original coordinates: the first one to pass from the ECI to the Earth Centered Earth Fixed (ECEF) coordinates, the second one to pass from the ECEF to ENU coordinates.

Since the ECEF reference frame is non-inertial and is rotating along with the Earth, a new dynamic equation must be introduced to take this rotation into account. Defining θ as the angle of rotation of the Earth, the latter's rotational dynamics can be easily written as:

$$\dot{\theta} = \omega_E, \tag{10.5}$$

where $\omega_E = 2\pi/86400 \approx 7.29 \times 10^{-5}$ [rad/s] is the angular velocity of the Earth. In Algorithm 13 and Algorithm 14 the steps to compute the two rotations are detailed. In the following, the notation x_{RF} with RF $\in$ {ECI, ECEF, ENU} denotes the reference frame of the generic vector x, while the notation $x_{\text{RF},c}$ with $c \in \{x, y, z\}$ denoting the three components of the generic vector x expressed in the RF coordinates.

Algorithm 13 ECI to ECEF coordinates transformation

Inputs: $r_{\text{ECI}}, v_{\text{ECI}}, \theta$
Parameters: ω_E
Outputs: $r_{\text{ECEF}}, v_{\text{ECEF}}$
$R = R_z(\theta)$
$r_{\text{ECEF}} = R r_{\text{ECI}}$
$a = v_{\text{ECI},x} + \omega_E r_{\text{ECI},y}$
$b = v_{\text{ECI},y} - \omega_E r_{\text{ECI},x}$
$c = v_{\text{ECI},z}$
$\tilde{v} = \begin{bmatrix} a & b & c \end{bmatrix}^T$
$v_{\text{ECEF}} = R\tilde{v}$

Algorithm 14 ECEF to ENU coordinates transformation

Inputs: $r_{\text{ECEF}}, \phi, \nu$ {ϕ, ν: lat and long of the GS}
Outputs: r_{ENU}
$$R = \begin{bmatrix} -\sin\nu & \cos\nu & 0 \\ -\cos\nu\sin\phi & -\sin\nu\sin\phi & \cos\phi \\ \cos\nu\cos\phi & \sin\nu\cos\phi & \sin\phi \end{bmatrix}$$
$r_{\text{ENU}} = R r_{\text{ECEF}}$

As a last step, from the ENU coordinates it is possible to compute the Azimuth (A), Elevation (E) and Range (ρ) of the satellite with respect to the OGS's antenna. For our case only the elevation angle will be used in the visibility analysis. Algorithm 15 details the mathematical steps to compute these three parameters.

Algorithm 15 ENU to Azimuth, Elevation, Range parameters

Inputs: r_{ENU}
Outputs: A, E, ρ
$\rho = \|r_{\text{ENU}}\|$
$\sigma = r_{\text{ENU}}/\rho$
$E = \arcsin \sigma_z$ [rad]
$A = \arctan(\sigma_x, \sigma_y)$ [rad]

Satellite to Satellite Visibility

Due to the short field of view of the LEO satellites, in order to exchange information between two sites far away from each other, a constellation of satellites is needed. This implies the creation of a communication link between two satellites of the same constellation in order to reach the remote site efficiently. The concept of visibility applies also in this case. To simplify the analysis only the geometric visibility is considered. Algorithm 16 details the procedure for the geometric visibility check.

Algorithm 16 Geometric visibility check between satellite A and B

Inputs: r_A, r_B
Parameters: $R_{\text{Earth}} = 6378136.3$ [m]
Outputs: isSatVis [bool]
Initialize output
isSatVis = False
norm = $\|r_A\|$
if $r_A == r_B$ (the same point in space) **then**
  isSatVis = True
  return isSatVis
else
  $r_C = r_A - r_B$ (relative position vector)
  min dist = Minimum distance between r_C and the centre of the Earth
  if min dist $\geq R_{\text{Earth}}$ **then**
    isSatVis = True
  end if
  return isSatVis
end if

10.3.3 Markov Decision Process Formulation

The system dynamics described above has been translated to a MDP in order to exploit the RL framework.

The state space is

$$\mathcal{S} =< t >, \quad t = 0, \ldots, T, \tag{10.6}$$

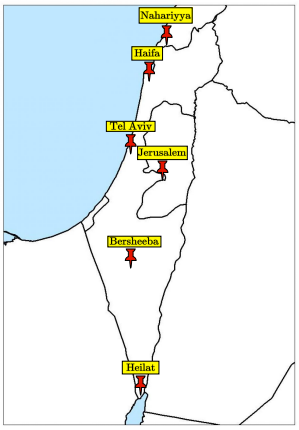


Figure 10.3. Map of the receiving OGSs in Israel.

where t is the generic time step and T is the final step within the transmission window period.

The action space is

$$\mathcal{A} =< \mathrm{OGS_T}, \mathrm{SAT}_1, \mathrm{SAT}_2, \mathrm{OGS_R} >, \tag{10.7}$$

where $\mathrm{OGS_T}$ is the index of the transmitting OGS, SAT_1 is the index of the first satellite receving data from the transmitter, SAT_2 is the index of the second satellite receiving data from the first one, and $\mathrm{OGS_R}$ is the index of the receiver.

Eventually, the reward function models the success rate of the end-to-end handover and is defined in the following way:

$$R = \begin{cases} +1, & \text{if transmission is successful} \\ -1, & \text{otherwise.} \end{cases} \tag{10.8}$$

10.4 Simulations and Results

In order to simulate and validate our control approach, two geographical areas from two different continents have been considered, namely:

1. the east coast of United States and Canada, with $N_{\mathrm{tr}} = 10$ transmitting OGSs located in the main cities, as in Fig. 9.4;
2. the territory of Israel, with $N_{\mathrm{re}} = 6$ receiving OGSs, shown in Fig. 10.3.

To perform the weather forecast for all the OGS zones, the very same strategy, LSTM structure and numerical hyparameters described in 9.4.1 have been chosen.

The LSTM model performance on unseen data (from December 21, 2016 to November 30, 2017) against all seasons and per each city, in terms of test accuracy,

Table 10.1. LSTM accuracies per season (both transmitting and receiving sites)

City	Winter	Spring	Summer	Autumn
New York	0.984	0.986	0.982	0.989
Montréal	0.988	0.981	0.963	0.974
Boston	0.993	0.989	0.987	0.989
Chicago	0.973	0.978	0.952	0.971
Charlotte	0.976	0.976	0.972	0.975
Pittsburgh	0.991	0.984	0.979	0.988
Detroit	0.997	1.000	0.996	0.998
Kansas City	0.992	0.990	0.983	0.990
Toronto	0.999	0.999	0.985	0.988
Indianapolis	0.992	0.986	0.981	0.992
Beersheba	0.944	0.956	0.966	0.998
Tel Aviv District	0.977	0.955	0.966	0.991
Eilat	0.932	0.923	0.968	0.999
Haifa	0.944	0.982	0.987	0.999
Nahariyya	0.989	0.985	0.945	0.997
Jerusalem	0.991	0.981	0.959	0.998

have been reported in Tab. 10.1. It is evident that the neural network model is able to predict correctly almost all the weather conditions within the test set, both in the transmitting zone (east coast of North America) and in the receiving zone (Israel), thus representing a powerful tool to estimate in advance the precipitation or thick clouds probability over the zone in which the OGS is located.

The RL–based controller hyperparameters for the training phase have been selected as follows:

- $\gamma = 0.9$;
- $\varepsilon_0 = 1$ with episodic decay law with respect to the generic episode η:

$$\varepsilon(\eta) = \mathrm{e}^{-\frac{\eta}{\beta N_{\mathrm{ep}}}},$$

  with $\beta = 0.2$ being the decay rate and N_{ep} the number of episodes;
- $\alpha_0 = 1$ with episodic decay law with respect to the generic episode η:

$$\alpha(\eta) = \mathrm{e}^{-\frac{\eta}{1000}}.$$

As for the evaluation phase, the controller performance has been figured out over a transmission period of $T = 2$ days.

The AI-based control law has been evaluated in terms of link availability, defined as follows:

$$L_A = \frac{N_S}{N_T}, \tag{10.9}$$

where N_S is the number of times a successful data transmission is achieved, and N_T is the total number of transmissions attempted.

Results with respect to the above–defined KPI have been compared with other benchmark routing approaches in the FSO domain, listed as follows:

- B1. Both transmitting and receiving OGSs and both LEO satellites are chosen randomly.
- B2. Transmitting and receiving OGS are chosen with a reactive approach based on the current weather condition, and the satellites are chosen with the min range technique, following the reasoning and modelling provided in [284].
- B3. Transmitting and receiving OGS are chosen with a reactive approach based on the current weather condition, and the satellites are chosen as those with the maximum elevation angle.
- B4. Transmitting and receiving OGS are chosen with the LSTM–based weather forecasts, and the satellites are chosen with the min range technique.
- B5. Transmitting and receiving OGS are chosen with the LSTM–based weather forecasts, and the satellites are chosen as those with the maximum elevation.

The proposed control approach has been tested over three different case studies, in which the communication between the two OGSs networks is realised with different LEO constellations:

- Case study 1. $N_{\text{sat}} = 15$ satellites from the Iridium constellation.
- Case study 2. $N_{\text{sat}} = 15$ satellites from the Starlink constellation.
- Case study 3. $N_{\text{sat}} = 30$ satellites given by the combination of satellites from case study 1 and case study 2.

The satellite orbital parameters and generic data have been gathered via Two–Line Elements (TLEs) files from [285]. A Two–Line Element file is a data format encoding a list of orbital elements of an Earth-orbiting object for a given point in time.

The propagation of the satellites motion over time is performed by using the 4-th order Runge-Kutta algorithm as integrator with fixed time step $dt = 1$ minute.

All the simulations have been carried out using Tensorflow framework on Python3.10 on a machine equipped with an Intel Core i5-10210U CPU and 16GB RAM.

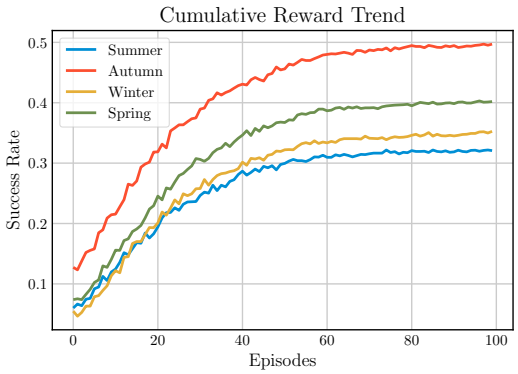


Figure 10.4. Iridium case study: season–related reward trend of the RL controller

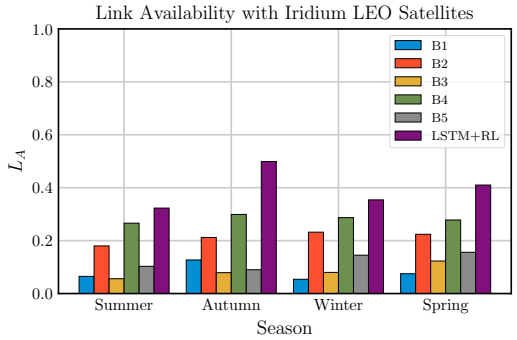


Figure 10.5. Iridium case study: season–related link availability comparison

10.4.1 Iridium Constellation

In this case study the number of episodes for training the RL controller has been set as $N_{\text{ep}} = 100$. The season–related cumulative reward trend over the training episodes is shown in Fig. 10.4. In all the four cases, the reward converges to a steady–state value in terms of data transmission success rate, which is higher in the autumn season due to the presence of a higher number of hours with favourable weather conditions both at transmitting and receiving zone.

The comparison of the performance of the proposed approach with respect to the benchmark solutions is shown in Fig. 10.5. It is worth noting that the RL controller together with a LSTM–based weather prediction achieves higher link availability with respect to the other standard techniques for FSO communication.

10.4.2 Starlink Constellation

In this case study the number of episodes for training the RL controller has been set as $N_{\text{ep}} = 100$. The cumulative reward trend per season is shown in Fig. 10.6: it converges to a steady–state value in terms of data transmission success rate.

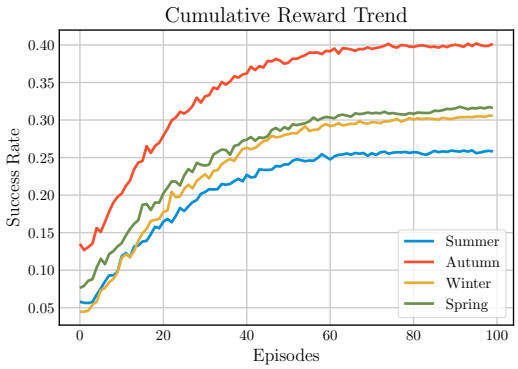


Figure 10.6. Starlink case study: season–related reward trend of the RL controller.

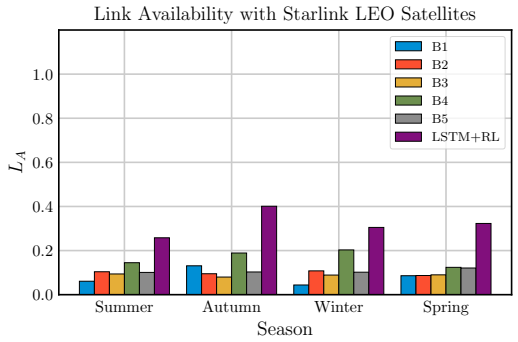


Figure 10.7. Starlink case study: season–related link availability comparison.

Fig. 10.7 depicts the comparison of the performance of the RL controller with respect to the benchmark solutions. The proposed control strategy achieves the best performances in terms of link availability. However, it shall be noticed that the overall performances are worse with respect to those achieved by means of the Iridium constellation. As an example, in the autumn season the RL controller guarantees $L_A = 0.401$ using Starlink satellites and $L_A = 0.499$ with Iridium satellites: similar results hold for the other seasons. This is due to the fact that the Starlink constellation orbits have been designed to cover mainly the North–American continent, thus guaranteeing poor coverage within the Israel territory.

10.4.3 Mixed Constellation

In the last case study the number of episodes for training the RL controller has been increased to $N_{\text{ep}} = 300$, in order to allow a broader exploration due to the availability of double the amount of LEO satellites with respect to the previous case studies.

Fig. 10.8 and 10.9 show training and evaluation performances against all seasons. As expected, the increased number of satellites guarantees higher cumulative reward

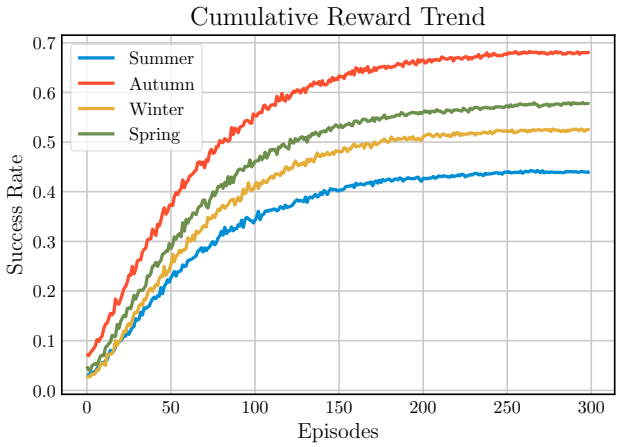


Figure 10.8. Mixed case study: season–related reward trend of the RL controller.

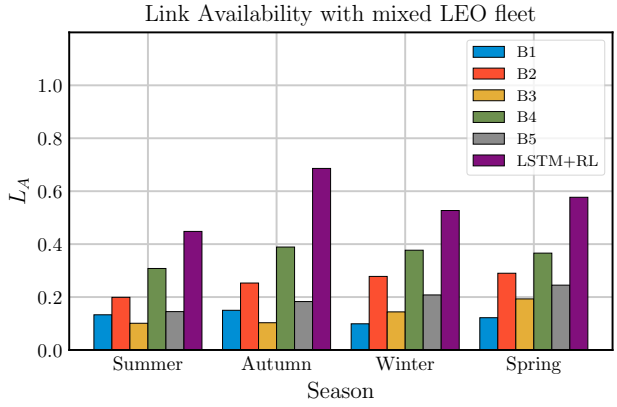


Figure 10.9. Mixed case study: season–related link availability comparison.

trend and, hence, link availability for the FSO transmission. This is due to the fact that increasing the number of satellites leads to a wider coverage over the Earth surface: this allows to establish a successful communication with guaranteed inter–satellite visibility for longer periods. Also in this case, the performances of the proposed control algorithm are better than the benchmark ones.

10.5 Future Works

In this work a mixed AI and RL approach for FSO point-to-point communication has been proposed. This technique exploits weather prediction algorithms to improve the quality of the communication link, as well as a dynamical data–driven optimisation for maximising the link–availability in a data transmission scenario between two terrestrial OGS networks communicating through LEO satellites. The proposed decision and control approach has been compared with several benchmark

solutions, achieving better performances in all seasons over the three analysed case studies in which different LEO constellations have been exploited.

However, some limitations hold. The developed RL–based algorithm does not take care about the frequent rotation of the LCT due to the OGS–satellite and inter–satellite dynamical switching, and no physical considerations on signal attenuation and beam spreading due to atmospheric condition and relative distance have been made.

Future works could focus on the problems defined above, proposing control algorithms that take signal attenuation into consideration, introducing link budget and beam spreading modelling, also with strategies aimed at saving energy on the various LEO devices.

Chapter 11

Conclusions

In an era marked by the breakthrough of Big Data and an ever-growing demand for seamless and reliable network connectivity, the application of data-driven control methods to both terrestrial and satellite networks represents a promising avenue to optimize performance, adapt to dynamic conditions, and deliver on the lofty expectations of modern communication systems. This thesis has embarked on a journey to explore, understand, and harness the potential of data-driven control methods in the context of telecommunication networks. With an array of methodologies, analyses, and case studies, this research has sought to advance our understanding of how data-driven approaches can reshape the landscape of communication.

The comprehensive investigation into terrestrial networks has unveiled the remarkable potential of data–driven control methodologies in

- providing intelligent user association and traffic steering solutions for multi–homed user terminals which can establish multiple connections into the heterogeneous world of cellular networks. Said automatic control laws may enable the full mobile fruition of high demanding applications, like Virtual and Augmented Reality;
- reducing user equipment energy consumption and costs by dynamically regulating uplink power and image compression rate in Mobile Augmented Reality applications, simultaneously satisfying rigorous constraints on latency and image or video accuracy;
- keeping safety and robustness in the autonomous driving scenario under a complete telecommunication fault which does not allow vehicles to communicate with each other or with any other infrastructure. In this domain, it has been shown how it is possible to implement a full distributed DRL control to keep a safe distance between vehicles in a platoon.

From these results, it seems clear that the adaptability of RL and DRL methods to real-time monitoring and decision-making provides terrestrial networks with a dynamic edge, enabling them to adapt to shifting patterns and evolving user demands.

In the realm of satellite networks, data-driven control methods have been instrumental in addressing unique challenges, particularly those related to signal quality and availability. This essay has demonstrated that

- it is possible to design a predictive control strategy for implementing site diversity in GEO–driven satellite communications, exploiting historical weather data to decide in advance to which Optical Ground Station the Laser Communication Terminal shall point;
- the combination of predictive weather forecasts with centralized RL control allows to design an optimized data path between two ground networks communicating through a LEO fleet, guaranteeing link availability according to the satellites orbital motion.

All these approaches have been validated through empirical evidence and simulations, underscoring their efficacy in reducing signal degradation and supporting uninterrupted communication. The satellite sector may now benefit from these findings, shifting towards data–driven approaches on the promise of uninterrupted connectivity, even in the face of atmospheric turbulence.

While this PhD thesis has advanced the general understanding of data-driven control methods in terrestrial and satellite networks, it is imperative to acknowledge that this is just the beginning of a transformative journey. There are several exciting avenues for future exploration:

- Intermodal Integration. The integration of terrestrial and satellite networks presents an intriguing area for future research. Data-driven control methods can facilitate seamless handovers and load balancing between these networks, optimizing coverage and performance, potentially allowing users to simultaneously exploit both terrestrial and satellite–like Access Points.
- Security and Privacy. As data-driven control methods proliferate, this thesis recalls the need for heightened attention to cybersecurity and data privacy. Ensuring the robustness of these networks against threats and preserving user privacy are pivotal.
- Energy Efficiency. In an increasingly eco-conscious world, the ecological transition imposes the development of data-driven control methods to optimize energy consumption within communication networks and reduce air pollu-

tion. This will imply the design of data–driven control techniques capable of balancing network performance with environmental sustainability.

In conclusion, this PhD thesis has embarked on a mission to harness the power of data-driven control methods for the benefit of terrestrial and satellite networks. The findings and insights derived from this research serve as a proof to the potential benefits of these methodologies. Automated terrestrial networks may now have a more efficient means to meet the burgeoning demands of an interconnected world, whereas AI–driven satellite optical networks are better equipped to transcend the atmospheric challenges that have long hampered their reliability. As we move forward, the convergence of these methodologies is poised to reshape the landscape of communication, making the dream of seamless and ubiquitous connectivity a reality.

As the journey continues, the spirit of inquiry, innovation, sustainability and multi–disciplinary collaboration must persist towards the creation of a seamless, immersive and ubiquitous digital world.

Appendix A

Implementation of a Reinforcement Learning Agent in Python

The appendix is devoted to a quick tutorial about the implementation of a Reinforcement Learning agent in the programming language which has been used to obtain all the simulations in this thesis: *Python*.

Python[1] is a high-level, versatile, and interpreted programming language world-wide known for its simplicity and readability. Python has steadily gained popularity over the years and is now one of the most widely used programming languages in the scientific community. It is known for its clean and easy-to-read syntax, which makes it an excellent choice for beginners and experienced programmers alike. Its first version (the 1.0) was released in January 1994 and currently (by October 2023) the newest version is the 3.12.0.

It is widely used in various domains, including web development, data science, machine learning (NumPy, pandas, and TensorFlow), and automation. Thanks to the simplicity in realizing complex DNNs and gradient–descent algorithms, Python represents undoubtedly the best choice to implement and test DRL algorithms.

The following sections show how it is possible to implement a DDPG controller in Python, applied to an academic nonlinear dynamical system, the pendulum.

---

[1]The name *Python* was inspired by the British comedy group Monty Python, which the Python founder Guido van Rossum loved. The Python world often includes references to Monty Python in various places, such as class and module names, and documentation.

A.1 The Environment

The motion of a pendulum oscillating in a two–dimensional plane can be expressed through the following dynamics:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{g}{l}\sin(x_1) - bx_2 + u, \end{aligned} \tag{A.1}$$

where $x_1 = \theta$ represents the pendulum angular position, $x_2 = \omega$ represents its angular velocity, g is the gravity acceleration on the Earth, l is the rod length, b is the viscous friction coefficient, and u is the control input in the form of an angular acceleration.

The Python code implementing the environment ready for the interaction with a RL–based controller is attached in what follows.

```
#Library for vector computations
import numpy as np

class Pendulum:
    #Initialization
    def __init__(self, t0=0, dt=0.1, tf=30):
        #Gravity acceleration on Earth
        self.g = 9.81
        #Length of the rod
        self.l = 1
        #Friction
        self.b = 0.4
        #Random initial state
        self.state = np.random.rand(2)
        self.dt = dt
        self.t0 = t0
        self.tf = tf
        self.steps = 0
        #Maximum number of steps within an episode
        self.MAX_steps = int((self.tf - self.t0) / self.dt)
        #Control saturation
        self.u_max = 2
        self.u_min = -2
        self.num_states = 2
        self.num_actions = 1

    #Right-hand part of the nonlinear dynamics
    def f(self, x, u):
        #States
        x_1 = x[0]
        x_2 = x[1]
        #Nonlinear dynamics
        dx_1 = x_2
        dx_2 = -self.g/self.l * np.sin(x_1) - self.b * x_2 + u
        return np.array([dx_1, dx_2])

```

```
37      #Single integration step
38      def rk4_step(self, u):
39          x = self.state
40          h = self.dt
41          #Calculate one RK4 step
42          k1 = self.f(x,u)
43          k2 = self.f(x + 0.5 * k1* h,u)
44          k3 = self.f(x + 0.5 * k2 * h,u)
45          k4 = self.f(x +       k3 * h,u)
46          #Compute the new state
47          new_x = x + h / 6.0 * ( k1 + 2 * k2 + 2 * k3 + k4)
48          self.state = new_x
49
50      #Compute reward
51      def computeReward(self):
52          return -(self.state[0])**2
53
54      #Apply control action on the environment
55      def step(self, action):
56          #Apply saturation on the actuator command
57          u = np.clip(action, self.u_min, self.u_max)
58
59          #Integrate the nonlinear dynamics with RK4
60          self.rk4_step(u)
61
62          #Compute reward
63          reward = self.computeReward()
64
65          #Increment the number of steps
66          self.steps += 1
67          terminated = False
68          truncated = False
69
70          #The run ends if the integration reaches
71          #the maximum steps
72          if self.steps == self.MAX_steps:
73              terminated = True
74          #Additional info, if any
75          info = {}
76          return self.state, reward, terminated,
77                       truncated, info
78
79      #Reset the environment after the episode ends
80      def reset(self):
81          self.steps = 0
82          #Random initial state
83          self.state = np.random.rand(2)
84          return self.state
```

It can be seen that the environment is given by a Python class which includes the following essential functions:

- `__init__` is the initializer, with info on the system parameters, initial state (random), the timespan (initial time, final time, and sampling time), and control saturation;

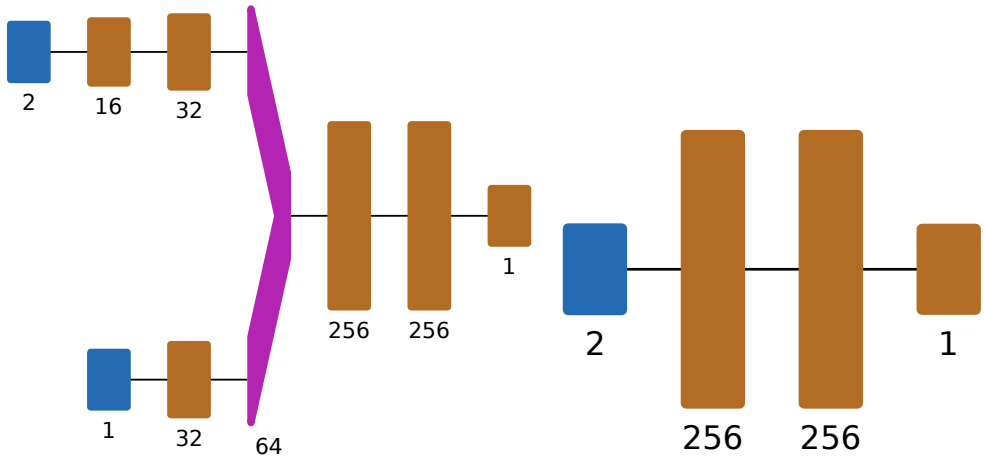
- `step` is the function realizing the application of the control action on the environment;
- `rk4_step` performs the integration using the Runge–Kutta 4th order method;
- `reset` re–initializes the environment when an episode begins, so that the steps return to zero and the initial state is again a random one.

A.2 The Agent

The agent is implemented according to the RL algorithm chosen. In this case, the choice fell on the DDPG algorithm.

The first thing to do is to define the actor and critic DNNs. They are standard NN with some hidden layers. In Python, they are implemented using Tensorflow and Keras libraries. Their graphical representation is depicted in Fig. A.1.

```
def get_actor():
    # Initialize weights between -3e-3 and 3-e3
    last_init = tf.random_uniform_initializer(minval=-0.003, maxval
    =0.003)

    inputs = layers.Input(shape=(num_states,))
    out = layers.Dense(256, activation="relu")(inputs)
    out = layers.Dense(256, activation="relu")(out)
    outputs = layers.Dense(1, activation="tanh", kernel_initializer=
    last_init)(out)

    #The output is now between -1 and 1
    model = tf.keras.Model(inputs, outputs)
    return model

def get_critic():
    # State as input
    state_input = layers.Input(shape=(num_states))
    state_out = layers.Dense(16, activation="relu")(state_input)
    state_out = layers.Dense(32, activation="relu")(state_out)

    # Action as input
    action_input = layers.Input(shape=(num_actions))
    action_out = layers.Dense(32, activation="relu")(action_input)

    # Both are passed through seperate layer before concatenating
    concat = layers.Concatenate()([state_out, action_out])

    out = layers.Dense(256, activation="relu")(concat)
    out = layers.Dense(256, activation="tanh")(out)
    outputs = layers.Dense(1)(out)

    # Outputs single value for give state-action
    model = tf.keras.Model([state_input, action_input], outputs)
    return model
```



(a) *Critic Network. In blue the two inputs (states in the upper side, actions in the lower site), in brown the dense layers and and the output (the Q–value), and in purple the concatenation.*

(b) *Actor Network. In blue the input layer (the state), in brown the dense layers and the output layer (the action).*

Figure A.1. Example of actor and critic networks.

The agent policy is given using a Gaussian noisy exploration with exponential annealing standard deviation. Note how, after the application of the random variable, there is a check over the validity of the action. Indeed, since the activation function of the last layer of the actor network is the $\tanh \cdot$, the output of the neural network is always between -1 and 1, and it should remain so even after adding the noise.

```
def get_noise_policy(episode):

    beta = 0.2
    sigma = np.exp(episode / (beta*total_episodes) )
    noise = sigma * np.random.rand()

    return noise

def policy(state, noise):

    # Actions are the output of the actor having as input
    # a specific state
    sampled_actions = tf.squeeze(actor_model(state))

    # Adding noise to action
    sampled_actions = sampled_actions.numpy() + noise

    # Make sure action is within bounds
    legal_action = np.clip(sampled_actions, -1, 1)

    return [np.squeeze(legal_action)]
```

As for the experience replay mechanism, the buffer $\mathcal{D}$ has been implemented as a Python class, with fixed capacity and batch size for training.

```
class Buffer:
    def __init__(self, capacity=500000, batch_size=256):

        # Maximum number of experiences
        self.buffer_capacity = capacity

        # Training experience
        self.batch_size = batch_size

        # Its tells us num of times record() was called.
        self.buffer_counter = 0

        self.state_buffer = np.zeros((self.buffer_capacity,num_states
    ))
        self.action_buffer = np.zeros((self.buffer_capacity,
    num_actions))
        self.reward_buffer = np.zeros((self.buffer_capacity, 1))
        self.next_state_buffer = np.zeros((self.buffer_capacity,
    num_states))

    def store(self, obs_tuple):

        # Takes (s,a,r,s') obervation tuple as input
        # and stores it in the buffer.
        # If the capacity is exceeded, the process begin
        # again replacing the first istance in the buffer

        index = self.buffer_counter % self.buffer_capacity

        self.state_buffer[index] = obs_tuple[0]
        self.action_buffer[index] = obs_tuple[1]
        self.reward_buffer[index] = obs_tuple[2]
        self.next_state_buffer[index] = obs_tuple[3]

        self.buffer_counter += 1
    @tf.function
    def update(
        self, state_batch, action_batch, reward_batch,
    next_state_batch,
    ):

        with tf.GradientTape() as tape:
            target_actions = target_actor(next_state_batch, training=
    True)
            y = reward_batch + gamma * target_critic(
                [next_state_batch, target_actions], training=True
            )
            critic_value = critic_model([state_batch, action_batch],
    training=True)
            critic_loss = tf.math.reduce_mean(tf.math.square(y -
    critic_value))

        critic_grad = tape.gradient(critic_loss, critic_model.
```

```
    trainable_variables)
47          critic_optimizer.apply_gradients(
48              zip(critic_grad, critic_model.trainable_variables)
49          )
50
51          with tf.GradientTape() as tape:
52              actions = actor_model(state_batch, training=True)
53              critic_value = critic_model([state_batch, actions],
    training=True)
54              actor_loss = -tf.math.reduce_mean(critic_value)
55
56          actor_grad = tape.gradient(actor_loss, actor_model.
    trainable_variables)
57          actor_optimizer.apply_gradients(
58              zip(actor_grad, actor_model.trainable_variables)
59          )
60
61      # We compute the loss and update parameters
62      def learn(self):
63          # Get sampling range
64          record_range = min(self.buffer_counter, self.buffer_capacity)
65
66          # Randomly sample indices
67          batch_indices = np.random.choice(record_range, self.
    batch_size)
68
69          # Convert to tensors
70          state_batch = tf.convert_to_tensor(self.state_buffer[
    batch_indices])
71          action_batch = tf.convert_to_tensor(self.action_buffer[
    batch_indices])
72          reward_batch = tf.convert_to_tensor(self.reward_buffer[
    batch_indices])
73          reward_batch = tf.cast(reward_batch, dtype=tf.float32)
74          next_state_batch = tf.convert_to_tensor(self.
    next_state_buffer[batch_indices])
75          self.update(state_batch, action_batch, reward_batch,
    next_state_batch)
```

The last needed function is the one responsible for the update of the target networks, according to the rate τ.

```
1 @tf.function
2 def update_target(target_weights, weights, tau):
3     for (a, b) in zip(target_weights, weights):
4         a.assign(b * tau + a * (1 - tau))
```

A.3 Training Phase

As a first step, the RL agent needs to be trained for a certain number of episodes. Hence, all the needed variables are initialized and the training loop is implemented in the following way.

```
#Initialize the environment
env = Pendulum()

num_states = env.num_states
num_actions = env.num_actions

actor_model = get_actor()
critic_model = get_critic()

target_actor = get_actor()
target_critic = get_critic()

# Making the weights equal initially
target_actor.set_weights(actor_model.get_weights())
target_critic.set_weights(critic_model.get_weights())

# Learning rate for actor-critic models
critic_lr = 0.002
actor_lr = 0.001

# Define optimization algorithm based on gradient descent
critic_optimizer = tf.keras.optimizers.Adam(critic_lr)
actor_optimizer = tf.keras.optimizers.Adam(actor_lr)


total_episodes = 150

# Discount factor for future rewards
gamma = 0.99
# Target networks update rate
tau = 0.005


buffer = Buffer(50000, 256)

#%% DDPG TRAINING


# To store reward history of each episode
ep_history = []

for ep in range(total_episodes):

    prev_state = env.reset()
    episodic_reward = 0

    noise = get_noise_policy(ep)

    while True:

        tf_prev_state = tf.expand_dims(tf.convert_to_tensor(
    prev_state), 0)

        action = policy(tf_prev_state, noise)

        action = action[0]

```

```
        # Get state and reward from the environment
        state, reward, ter, tru, info = env.step(action)


        # Store transition in the buffer
        buffer.store((prev_state, action, reward, state))

        #Increase the cumulative reward
        episodic_reward += reward

        # Perform a training step
        buffer.learn()

        # Update target networks
        update_target(target_actor.variables, actor_model.variables, tau)
        update_target(target_critic.variables, critic_model.variables, tau)

        # Check if the episode is terminated
        done = ter or tru
        if done:
            break

        prev_state = state

    ep_history.append(episodic_reward)
    print("Episode * {} Cumulative Reward * {}".format(ep, episodic_reward))
```

After the training process has terminated its execution, one may save the weights and biases of the four nNNs, in such a way to not waste again time. Indeed, the training process may require long execution time, usually from ten minutes up to entire days.

```
# Save the weights
actor_model.save_weights("pendulum_actor.h5")
critic_model.save_weights("pendulum_critic.h5")
target_actor.save_weights("pendulum_target_actor.h5")
target_critic.save_weights("pendulum_target_critic.h5")
```

A.4 Evaluation Phase

In the second phase, the agent is deployed into the environment with the updated weights, in order to evaluate its performance (e.g., the tracking of a reference signal). This is done loading the pre–saved weights and running the agent for just one episode over the environment, following its policy without noise.

```
actor = get_actor()  # Create an empty model
critic = get_critic()  # Create an empty model
target_actor = get_actor()  # Create an empty model
```

```
target_critic = get_critic()  # Create an empty model

actor.load_weights("pendulum_actor.h5")
critic.load_weights("pendulum_critic.h5")
target_actor.load_weights("pendulum_target_actor.h5")
target_critic.load_weights("pendulum_target_critic.h5")
#%%

state = env.reset()

state = env.reset()
tot_rew = 0

#Trajectories
theta = []
omega = []
control = []
theta.append(state[0])
omega.append(state[1])

done = False
while not done:

    tf_prev_state = tf.expand_dims(tf.convert_to_tensor(state), 0)

    # In the evaluation phase the agent is deterministic
    action = policy(tf_prev_state, 0.0)

    u = action[0]
    control.append(u)

    next_state, reward, ter, tru, info = env.step(u)

    tot_rew += reward
    done = ter or tru
    state = next_state

    theta.append(state[0])
    omega.append(state[1])
```

The user–defined variables `control`, `theta`, and `omega` are useful in order to evaluate the system trajectories and the applied control effort.

Bibliography

[1] "Online etymology dictionary", `https://www.etymonline.com/word/control`. Accessed: 2023-07-12.

[2] S. Monaco, C. Califano, P. Di Giamberardino, and M. Mattioni, *Teoria dei Sistemi. Lineari stazionari a dimensione finita.* Società Editrice Esculapio, 2021.

[3] M. Renardy and R. C. Rogers, *An introduction to partial differential equations*, vol. 13. Springer Science & Business Media, 2006.

[4] J. C. Maxwell, "I. on governors", *Proceedings of the Royal Society of London*, no. 16, pp. 270–283, 1868.

[5] K. J. Åström and T. Hägglund, "PID control", *IEEE Control Systems Magazine*, vol. 1066, 2006.

[6] A. Isidori, *Sistemi di controllo.* Società Editrice SIDEREA, 1992.

[7] C. Bruni and G. Di Pillo, *Metodi Variazionali per il controllo ottimo.* Aracne, 2007.

[8] A. Isidori, *Nonlinear control systems: an introduction.* Springer, 1985.

[9] T. Madani and A. Benallegue, "Backstepping control for a quadrotor helicopter", in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3255–3260, IEEE, 2006.

[10] C. I. Byrnes, F. Delli Priscoli, and A. Isidori, *Output regulation of nonlinear systems.* Springer, 1997.

[11] S. V. Drakunov and V. I. Utkin, "Sliding mode control in dynamic systems", *International Journal of Control*, vol. 55, no. 4, pp. 1029–1037, 1992.

[12] F. Tao, B. Xiao, Q. Qi, J. Cheng, and P. Ji, "Digital twin modeling", *Journal of Manufacturing Systems*, vol. 64, pp. 372–389, 2022.

[13] B. Wittenmark, J. Nilsson, and M. Torngren, "Timing problems in real-time control systems", in *Proceedings of 1995 American Control Conference-ACC'95*, vol. 3, pp. 2000–2004, IEEE, 1995.

[14] R.-E. Precup, R.-C. Roman, and A. Safaei, *Data-driven model-free controllers.* CRC Press, 2021.

[15] S. Formentin, K. van Heusden, and A. Karimi, "Model-based and data-driven model-reference control: A comparative analysis", in *2013 European Control Conference (ECC)*, pp. 1410–1415, 2013.

[16] **Andrea Wrona**, D. Menegatti, and F. Baldisseri, "Data-driven control of insulin injection for type 1 diabetic patients", Submitted to ECC 2024.

[17] W. Tang and P. Daoutidis, "Data-driven control: Overview and perspectives", in *2022 American Control Conference (ACC)*, pp. 1048–1064, IEEE, 2022.

[18] X. Ying, "An overview of overfitting and its solutions", in *Journal of physics: Conference series*, vol. 1168, p. 022022, IOP Publishing, 2019.

[19] S. Yin, H. Gao, and O. Kaynak, "Data-driven control and process monitoring for industrial applications—part i", *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6356–6359, 2014.

[20] Z. Lai, Z. Jia, and M. Chen, "Autonomous learning and navigation of mobile robots based on deep reinforcement learning", *Journal of Physics: Conference Series*, vol. 2171, no. 1, p. 012024, 2022.

[21] K. Thanapalan and S. Veres, "Agent based controller for satellite formation flying", in *2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 385–389, IEEE, 2005.

[22] O. F. Ruiz-Martinez, J. C. Mayo-Maldonado, G. Escobar, B. A. Frias-Araya, J. E. Valdez-Resendiz, J. C. Rosas-Caro, and P. Rapisarda, "Data-driven control of lvdc network converters: Active load stabilization", *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2182–2194, 2019.

[23] **Andrea Wrona**, F. Baldisseri, D. Menegatti, *et al.*, "Deep neural network regression to assist non-invasive diagnosis of portal hypertension", *Healthcare*, vol. 11, no. 18, p. 2603, 2023.

[24] **Andrea Wrona**, M. M. H. Atanasious, and F. Delli Priscoli, "Deep reinforcement learning–based automatic augmentation for gastrointestinal disease classification", Submitted to ACM Transactions on Computing for Healthcare.

[25] Z. Hou and S. Jin, *Model free adaptive control: theory and applications*. CRC press, 2013.

[26] H. Hjalmarsson, S. Gunnarsson, and M. Gevers, "A convergent iterative restricted complexity control design scheme", in *Proceedings of 1994 33rd IEEE conference on decision and control*, vol. 2, pp. 1735–1740, IEEE, 1994.

[27] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, "Iterative feedback tuning: theory and applications", *IEEE control systems magazine*, vol. 18, no. 4, pp. 26–41, 1998.

[28] H. Hjalmarsson, "Iterative feedback tuning—an overview", *International journal of adaptive control and signal processing*, vol. 16, no. 5, pp. 373–395, 2002.

[29] J. C. Spall and J. A. Cristion, "Model-free control of nonlinear stochastic systems with discrete-time measurements", *IEEE transactions on automatic control*, vol. 43, no. 9, pp. 1198–1210, 1998.

[30] I.-J. Wang and J. C. Spall, "Stochastic optimisation with inequality constraints using simultaneous perturbations and penalty functions", *International Journal of Control*, vol. 81, no. 8, pp. 1232–1238, 2008.

[31] L. Mišković, A. Karimi, and D. Bonvin, "Iterative controller tuning by minimization of a generalized decorrelation criterion", *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 1137–1142, 2003.

[32] K. Van Heusden, A. Karimi, and D. Bonvin, "Data-driven model reference control with asymptotically guaranteed stability", *International Journal of Adaptive Control and Signal Processing*, vol. 25, no. 4, pp. 331–351, 2011.

[33] W. Kickert and E. Mamdani, "Analysis of a fuzzy logic controller", in *Readings in Fuzzy Sets for Intelligent Systems*, pp. 290–297, Elsevier, 1993.

[34] K. Halmevaara and H. Hyötyniemi, "Data-based parameter optimization of dynamic simulation models", in *The 47th Conference on Simulation and Modelling-SIMS 2006, Helsinki, Finland, 28-29 September 2006*, pp. 68–73, Finnish Society of Automation, 2006.

[35] R. Chi, Z. Hou, B. Huang, and S. Jin, "A unified data-driven design framework of optimality-based generalized iterative learning control", *Computers & Chemical Engineering*, vol. 77, pp. 10–23, 2015.

[36] M. G. Safonov and T.-C. Tsao, "The unfalsified control concept and learning", in *Proceedings of 1994 33rd IEEE conference on decision and control*, vol. 3, pp. 2819–2824, IEEE, 1994.

[37] M. C. Campi, A. Lecchini, and S. M. Savaresi, "Virtual reference feedback tuning: a direct method for the design of feedback controllers", *Automatica*, vol. 38, no. 8, pp. 1337–1346, 2002.

[38] M. C. Campi, A. Lecchini, and S. M. Savaresi, "An application of the virtual reference feedback tuning method to a benchmark problem", *European Journal of Control*, vol. 9, no. 1, pp. 66–76, 2003.

[39] M. Krstić, "Performance improvement and limitations in extremum seeking control", *Systems & Control Letters*, vol. 39, no. 5, pp. 313–326, 2000.

[40] M. Krstić and H.-H. Wang, "Stability of extremum seeking feedback for general nonlinear dynamic systems", *Automatica*, vol. 36, no. 4, pp. 595–601, 2000.

[41] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects", *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[42] L. Li, "Predicting the investment risk in supply chain management using bpnn and machine learning", *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[43] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2. Springer, 2009.

[44] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial", *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[45] J. C. S. J. Junior, S. R. Musse, and C. R. Jung, "Crowd analysis using computer vision techniques", *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 66–77, 2010.

[46] A. Giuseppi, L. P. L. Porto, **Andrea Wrona**, and D. Menegatti, "Landslide susceptibility prediction from satellite data through an intelligent system based on deep learning", in *2023 31st Mediterranean Conference on Control and Automation (MED)*, pp. 513–520, IEEE, 2023.

[47] D. Hakkani-Tür, G. Riccardi, and A. Gorin, "Active learning for automatic speech recognition", in *2002 IEEE international conference on acoustics, speech, and signal processing*, vol. 4, pp. IV–3904, IEEE, 2002.

[48] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, R. Kaluri, D. S. Rajput, G. Srivastava, and T. Baker, "Analysis of dimensionality reduction techniques on big data", *Ieee Access*, vol. 8, pp. 54776–54788, 2020.

[49] M. A. Carreira-Perpinán, "The elastic embedding algorithm for dimensionality reduction.", in *ICML*, vol. 10, pp. 167–174, 2010.

[50] T. M. Ghazal, "Performances of k-means clustering algorithm with different distance metrics", *Intelligent Automation & Soft Computing*, vol. 30, no. 2, pp. 735–742, 2021.

[51] Y. Wang, Y. Jiang, Y. Wu, and Z.-H. Zhou, "Spectral clustering on multiple manifolds", *IEEE Transactions on Neural Networks*, vol. 22, no. 7, pp. 1149–1161, 2011.

[52] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[53] A. A. Markov, "The theory of algorithms", *Trudy Matematicheskogo Instituta Imeni VA Steklova*, vol. 42, pp. 3–375, 1954.

[54] R. Bellman, "A markovian decision process", *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[55] R. A. Howard, *Dynamic Programming and Markov Processes*. John Wiley, 1960.

[56] A. Sood, R. A. Forster, B. J. Archer, and R. C. Little, "Neutronics calculation advances at los alamos: Manhattan project to monte carlo", *Nuclear Technology*, vol. 207, no. sup1, pp. S100–S133, 2021.

[57] Z. Jin, M. Ma, S. Zhang, Y. Hu, Y. Zhang, and C. Sun, "Secure state estimation of cyber-physical system under cyber attacks: Q-learning vs. sarsa", *Electronics*, vol. 11, no. 19, p. 3161, 2022.

[58] J. Nash, "Non-cooperative games", *Annals of mathematics*, pp. 286–295, 1951.

[59] P. Yadav, A. Mishra, and S. Kim, "A comprehensive survey on multi-agent reinforcement learning for connected and automated vehicles", *Sensors*, vol. 23, no. 10, p. 4710, 2023.

[60] D. Strouse, M. Kleiman-Weiner, J. Tenenbaum, M. Botvinick, and D. J. Schwab, "Learning to share and hide intentions using information regularization", *Advances in neural information processing systems*, vol. 31, 2018.

[61] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning", in *Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 16*, pp. 66–83, Springer, 2017.

[62] L. Feng, Y. Xie, B. Liu, and S. Wang, "Multi-level credit assignment for cooperative multi-agent reinforcement learning", *Applied Sciences*, vol. 12, no. 14, p. 6938, 2022.

[63] G. Chen, "A new framework for multi-agent reinforcement learning–centralized training and exploration with decentralized execution via policy distillation", *arXiv preprint arXiv:1910.09152*, 2019.

[64] W. Guo, "Explainable artificial intelligence for 6g: Improving trust between human and machine", *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, 2020.

[65] G. Konidaris, S. Osentoski, and P. Thomas, "Value function approximation in reinforcement learning using the fourier basis", in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 380–385, 2011.

[66] S. Whiteson and S. Whiteson, "Adaptive tile coding", *Adaptive Representations for Reinforcement Learning*, pp. 65–76, 2010.

[67] L. Auret and C. Aldrich, "Interpretation of nonlinear relationships between process variables by use of random forests", *Minerals Engineering*, vol. 35, pp. 27–42, 2012.

[68] D. C. Selvaraj, S. Hegde, N. Amati, F. Deflorio, and C. F. Chiasserini, "An ml-aided reinforcement learning approach for challenging vehicle maneuvers", *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1686–1698, 2022.

[69] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning", *arXiv preprint arXiv:1312.5602*, 2013.

[70] "Atari games", `https://atari.com/collections/games`. Accessed: 2023-09-25.

[71] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition", *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[72] M. Sewak and M. Sewak, "Deep q network (dqn), double dqn, and dueling dqn: A step towards general artificial intelligence", *Deep Reinforcement Learning: Frontiers of Artificial Intelligence*, pp. 95–108, 2019.

[73] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine learning*, vol. 8, pp. 229–256, 1992.

[74] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", *arXiv preprint arXiv:1707.06347*, 2017.

[75] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization", in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[76] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning", *arXiv preprint arXiv:1509.02971*, 2015.

[77] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the brownian motion", *Physical review*, vol. 36, no. 5, p. 823, 1930.

[78] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", in *International conference on machine learning*, pp. 448–456, pmlr, 2015.

[79] X. Zhou, X. Zhang, H. Zhao, J. Xiong, and J. Wei, "Constrained soft actor-critic for energy-aware trajectory design in uav-aided iot networks", *IEEE Wireless Communications Letters*, vol. 11, no. 7, pp. 1414–1418, 2022.

[80] M. Babaeizadeh, I. Frosio, S. Tyree, J. Clemons, and J. Kautz, "Reinforcement learning through asynchronous advantage actor-critic on a gpu", *arXiv preprint arXiv:1611.06256*, 2016.

[81] S. Kuutti, R. Bowden, H. Joshi, R. De Temple, and S. Fallah, "End-to-end reinforcement learning for autonomous longitudinal control using advantage actor critic with temporal context", in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2456–2462, IEEE, 2019.

[82] H. Tang, A. Wang, F. Xue, J. Yang, and Y. Cao, "A novel hierarchical soft actor-critic algorithm for multi-logistics robots task allocation", *Ieee Access*, vol. 9, pp. 42568–42582, 2021.

[83] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep q-learning with model-based acceleration", in *International conference on machine learning*, pp. 2829–2838, PMLR, 2016.

[84] N. A. Mosali, S. S. Shamsudin, O. Alfandi, R. Omar, and N. Al-Fadhali, "Twin delayed deep deterministic policy gradient-based target tracking for unmanned aerial vehicle with achievement rewarding and multistage training", *IEEE Access*, vol. 10, pp. 23545–23559, 2022.

[85] S. Babani, A. Bature, M. Faruk, and N. Dankadai, "Comparative study between fiber optic and copper in communication link", *Int. J. Tech. Res. Appl*, vol. 2, no. 2, pp. 59–63, 2014.

[86] L. Ahlin, J. Zander, and S. Ben Slimane, *Principles of wireless communications*. Studentlitteratur, 2006.

[87] P. Sharma, "Evolution of mobile wireless communication networks-1g to 5g as well as future prospective of next generation communication network", *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 8, pp. 47–53, 2013.

[88] T. Ojanpera and R. Prasad, "An overview of air interface multiple access for imt-2000/umts", *IEEE communications Magazine*, vol. 36, no. 9, pp. 82–86, 1998.

[89] K. Pandav, A. G. Te, N. Tomer, S. S. Nair, and A. K. Tewari, "Leveraging 5g technology for robotic surgery and cancer care", *Cancer Reports*, vol. 5, no. 8, p. e1595, 2022.

[90] Č. Stefanović, "Industry 4.0 from 5g perspective: Use-cases, requirements, challenges and approaches", in *2018 11th CMI International Conference: Prospects and Challenges Towards Developing a Digital Economy within the EU*, pp. 44–48, IEEE, 2018.

[91] "3g / 4g / 5g mappa di copertura, italy", `https://www.nperf.com/it/map/IT/-/230.TIM-Mobile/signal/?ll=41.549433941803954&lg=12.570000000000016&zoom=5`. Accessed: 2023-10-21.

[92] S. G. Weinbaum, *Pygmalion's Spectacles*. Simon & Schuster, 1935.

[93] M. Heilig, "Sensorama simulator, us patent no. 3050870", *US Patent and Trademark Office. https://patents. google. com/patent/US3050870A/en*, 1962.

[94] I. E. Sutherland, "A head-mounted three dimensional display", in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, pp. 757–764, 1968.

[95] J. A. Thorpe, J. Shiflett, G. Bloedorn, M. Hayes, and D. Miller, "The simnet network and protocols", tech. rep., Technical Report 7102, BBN systems and technologies corporation, 1989.

[96] M. J. Zyda, R. B. McGhee, C. M. McConkle, A. H. Nelson, and R. S. Ross, "A real-time, three-dimensional moving platform visualization tool", *Computers & Graphics*, vol. 14, no. 2, pp. 321–333, 1990.

[97] E. Abruzzese, "Making sense of the virtual reality market in 2023", `https://www.abiresearch.com/blogs/2023/07/13/virtual-reality-vr-market-2023-update/`. Accessed: 2023-08-05.

[98] P. C. Thomas and W. David, "Augmented reality: An application of heads-up display technology to manual manufacturing processes", in *Hawaii international conference on system sciences*, vol. 2, ACM SIGCHI Bulletin, 1992.

[99] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system", in *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pp. 85–94, IEEE, 1999.

[100] "Augmented reality market size, share & covid 19 impact analysis", `https://www.fortunebusinessinsights.com/augmented-reality-ar-market-102553`. Accessed: 2023-08-05.

[101] J. Orlosky, K. Kiyokawa, and H. Takemura, "Virtual and augmented reality on the 5g highway", *Journal of Information Processing*, vol. 25, pp. 133–141, 2017.

[102] B. Fanini, A. Pagano, E. Pietroni, D. Ferdani, E. Demetrescu, and A. Palombini, "Augmented reality for cultural heritage", in *Springer Handbook of Augmented Reality*, pp. 391–411, Springer, 2023.

[103] "Vadus", `https://business.esa.int/projects/vadus`. Accessed: 2023-10-17.

[104] **Andrea Wrona**, A. Tortorelli, and F. Liberati, "A multi-agent q-learning control framework to support augmented and virtual reality streaming services in heterogeneous wireless networks", Submitted to IEEE Transactions on Mobile Computing.

[105] V. Reljic, I. Milenkovic, S. Dudic, J. Sulc, and B. Bajci, "Augmented reality applications in industry 4.0 environment", *Applied Sciences*, vol. 11, no. 12, 2021.

[106] K. Lavingia and S. Tanwar, "Augmented reality and industry 4.0", in *A Roadmap to Industry 4.0: Smart Production, Sharp Business and Sustainable Development*, pp. 143–155, Springer, 2020.

[107] P. Fraga-Lamas, T. M. Fernández-Caramés, O. Blanco-Novoa, and M. A. Vilar-Montesinos, "A review on industrial augmented reality systems for the industry 4.0 shipyard", *IEEE Access*, vol. 6, pp. 13358–13375, 2018.

[108] G. Kiryakova, N. Angelova, and L. Yordanova, "The potential of augmented reality to transform education into smart education", *TEM Journal*, vol. 7, no. 3, pp. 556–565, 2018.

[109] M. Sirakaya and D. A. Sirakaya, "Augmented reality in stem education: a systematic review", *Interactive Learning Environments*, vol. 30, no. 8, pp. 1556–1569, 2020.

[110] D. Sahin and R. M. Yilmaz, "The effect of augmented reality technology on middle school students' achievements and attitudes towards science education", *Computers & Education*, vol. 144, 2020.

[111] H. Kose and N. Guner-Yildiz, "Augmented reality (ar) as a learning material in special needs education", *Education and Information Technologies*, vol. 26, pp. 1921–1936, 2021.

[112] Y. Fu, V. Sundstedt, and C. Fagerstrom, "A survey of possibilities and challenges with ar/vr/mr and gamification usage in healthcare", *Proceedings of the 14th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2021)*, vol. 5, pp. 733–740, 2021.

[113] T. Zhang, C. Shi, T. Zhao, Z. Ye, P. Walker, N. Saxena, Y. Wang, and Y. Chen, "Personalized health monitoring via vital sign measurements leveraging motion sensors on ar/vr headsets", *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, pp. 529—530, 2022.

[114] W. S. Khor, B. Baker, K. Amin, A. Chan, K. Patel, and J. Wong, "Augmented and virtual reality in surgery—the digital surgical environment: applications, limitations and legal pitfalls", *Annals of Tanslational Medicine*, vol. 4, no. 23, 2016.

[115] M. C. Tom Diek and T. H. Jung, "Value of augmented reality at cultural heritage sites: A stakeholder approach", *Journal of Destination Marketing & Management*, vol. 6, no. 2, pp. 110–117, 2017.

[116] G. Bozzelli, A. Raia, S. Ricciardi, M. De Nino, N. Barile, M. Perrella, M. Tramontano, A. Pagano, and A. Palombini, "An integrated vr/ar framework for user-centric interactive experience of cultural heritage: The arkaevision project", *Digital Applications in Archaeology and Cultural Heritage*, vol. 15, 2019.

[117] A. Marto, A. Gonçalves, M. Melo, and M. Bessa, "A survey of multisensory vr and ar applications for cultural heritage", *Computers & Graphics*, vol. 102, pp. 426–440, 2022.

[118] M. K. Bekele, R. Pierdicca, E. Frontoni, E. S. Malinvergni, and J. Gain, "A survey of augmented, virtual, and mixed reality for cultural heritage", *ACM Journal on Computing and Cultural Heritage*, vol. 11, no. 2, pp. 426–440, 2018.

[119] A. Baratè, G. Haus, L. A. Ludovico, E. Pagani, and N. Scarabottolo, "5g technology for augmented and virtual reality in education", in *Proceedings of the International Conference on Education and New Developments*, vol. 2019, pp. 512–516, 2019.

[120] A. Ghosh, A. Maeder, M. Baker, and D. Chandramouli, "5g evolution: A view on 5g cellular technology beyond 3gpp release 15", *IEEE access*, vol. 7, pp. 127639–127651, 2019.

[121] 3GPP, "QoE parameters and metrics relevant to the Virtual Reality (VR) user experience", Technical Report (TR) 26.929, 3rd Generation Partnership Project (3GPP), 06 2019. Version 16.0.0.

[122] M. Erol-Kantarci and S. Sukhmani, "Caching and computing at the edge for mobile augmented reality and virtual reality (ar/vr) in 5g", *Ad Hoc Networks*, pp. 169–177, 2018.

[123] O. Galinina, A. Pyattaev, S. Andreev, M. Dohler, and Y. Koucheryavy, "5g multi-rat lte-wifi ultra-dense small cells: Performance dynamics, architecture, and trends", *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 6, pp. 1224–1240, 2015.

[124] S. Chandrashekar, A. Maeder, C. Sartori, T. Höhne, B. Vejlgaard, and D. Chandramouli, "5g multi-rat multi-connectivity architecture", in *2016 IEEE International Conference on Communications Workshops (ICC)*, pp. 180–186, IEEE, 2016.

[125] S. Maheshwari, D. Raychaudhuri, I. Seskar, and F. Bronzino, "Scalability and performance evaluation of edge cloud systems for latency constrained applications", in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pp. 286–299, IEEE, 2018.

[126] X. Zhang, H. Chen, Y. Zhao, Z. Ma, Y. Xu, H. Huang, H. Yin, and D. O. Wu, "Improving cloud gaming experience through mobile edge computing", *IEEE Wireless Communications*, vol. 26, no. 4, pp. 178–183, 2019.

[127] A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications", *IEEE Wireless Communications Letters*, vol. 6, no. 3, pp. 398–401, 2017.

[128] Y. Dai, D. Xu, S. Maharjan, G. Qiao, and Y. Zhang, "Artificial intelligence empowered edge computing and caching for internet of vehicles", *IEEE Wireless Communications*, vol. 26, no. 3, pp. 12–18, 2019.

[129] J. Li, M. Dai, and Z. Su, "Energy-aware task offloading in the internet of things", *IEEE Wireless Communications*, vol. 27, no. 5, pp. 112–117, 2020.

[130] X. He, H. Lu, H. Huang, Y. Mao, K. Wang, and S. Guo, "Qoe-based cooperative task offloading with deep reinforcement learning in mobile edge networks", *IEEE Wireless Communications*, vol. 27, no. 3, pp. 111–117, 2020.

[131] H. Guo, J. Liu, J. Ren, and Y. Zhang, "Intelligent task offloading in vehicular edge computing networks", *IEEE Wireless Communications*, vol. 27, no. 4, pp. 126–132, 2020.

[132] Q. Song and A. Jamalipour, "Network selection in an integrated wireless lan and umts environment using mathematical modeling and computing techniques", *IEEE wireless communications*, vol. 12, no. 3, pp. 42–48, 2005.

[133] M. G. Anany, M. M. Elmesalawy, and E. S. El Din, "A matching game solution for optimal rat selection in 5g multi-rat hetnets", in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1022–1028, IEEE, 2019.

[134] F. Delli Priscoli, E. De Santis, A. Giuseppi, and A. Pietrabissa, "Capacity-constrained wardrop equilibria and application to multi-connectivity in 5g networks", *Journal of the Franklin Institute*, vol. 358, no. 17, pp. 9364–9384, 2021.

[135] A. Ornatelli, A. Tortorelli, and A. Giuseppi, "Iterative mpc for energy management and load balancing in 5g heterogeneous networks", in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 0467–0471, IEEE, 2020.

[136] A. Ornatelli, A. Tortorelli, A. Giuseppi, and F. Delli Priscoli, "Hierarchical rl for load balancing and qos management in multi-access networks", in *2021 29th Mediterranean Conference on Control and Automation (MED)*, pp. 886–891, IEEE, 2021.

[137] A. Giuseppi, E. De Santis, F. Delli Priscoli, S. H. Won, T. Choi, and A. Pietrabissa, "Network selection in 5g networks based on markov games and friend-or-foe reinforcement learning", in *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 1–5, IEEE, 2020.

[138] A. Alwarafy, B. S. Ciftler, M. Abdallah, and M. Hamdi, "Deeprat: A drl-based framework for multi-rat assignment and power allocation in hetnets", in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, IEEE, 2021.

[139] R. M. Sandoval, S. Canovas-Carrasco, A.-J. Garcia-Sanchez, and J. Garcia-Haro, "Smart usage of multiple rat in iot-oriented 5g networks: A reinforcement learning approach", in *2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K)*, pp. 1–8, IEEE, 2018.

[140] N. Zhao, Y.-C. Liang, D. Niyato, Y. Pei, M. Wu, and Y. Jiang, "Deep reinforcement learning for user association and resource allocation in heterogeneous cellular networks", *IEEE Transactions on Wireless Communications*, vol. 18, no. 11, pp. 5141–5152, 2019.

[141] X. Chen and G. Liu, "Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks", *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10843–10856, 2021.

[142] G. Bu and J. Jiang, "Reinforcement learning-based user scheduling and resource allocation for massive mu-mimo system", in *2019 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 641–646, IEEE, 2019.

[143] M. Yan, G. Feng, J. Zhou, and S. Qin, "Smart multi-rat access based on multiagent reinforcement learning", *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4539–4551, 2018.

[144] A. Ornatelli, A. Tortorelli, and F. Liberati, "A distributed reinforcement learning approach for power control in wireless networks", in *2021 IEEE World AI IoT Congress (AIIoT)*, pp. 0275–0281, IEEE, 2021.

[145] L. Wang and G.-S. G. Kuo, "Mathematical modeling for network selection in heterogeneous wireless networks—a tutorial", *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 271–292, 2012.

[146] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks", *IEEE journal on selected areas in communications*, vol. 27, no. 7, pp. 1029–1046, 2009.

[147] S. Kurt and B. Tavli, "Path-loss modeling for wireless sensor networks: A review of models and comparative evaluations.", *IEEE Antennas and Propagation Magazine*, vol. 59, no. 1, pp. 18–37, 2017.

[148] F. Yanfie, R. Fengyuan, and L. Chuang, "Design a pid controller for active queue management", in *Proceedings of the Eighth IEEE Symposium on Computers and Communications. ISCC 2003*, pp. 985–990, IEEE, 2003.

[149] A. Giuseppi, S. M. Shahid, E. De Santis, S. H. Won, S. Kwon, and T. Choi, "Design and simulation of the multi-rat load-balancing algorithms for 5g-allstar systems", in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 594–596, IEEE, 2020.

[150] C. Wilson and V. Veeravalli, "A convergent version of the max sinr algorithm for the mimo interference channel", *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2952–2961, 2013.

[151] T. Kent, A. Richards, and A. Johnson, "Single-agent policies for the multi-agent persistent surveillance problem via artificial heterogeneity", in *Multi-Agent Systems and Agreement Technologies* (N. Bassiliades, G. Chalkiadakis, and D. de Jonge, eds.), (Cham), pp. 243–260, Springer International Publishing, 2020.

[152] T. Kent, A. Richards, and A. Johnson, "Homogeneous agent behaviours for the multi-agent simultaneous searching and routing problem", *Drones*, vol. 6, no. 2, p. 51, 2022.

[153] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, "Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research", *Internet of Things*, vol. 12, p. 100273, 2020.

[154] **Andrea Wrona**, D. Menegatti, and A. Tortorelli, "Deep reinforcement learning–based image resolution and uplink power control for mobile augmented reality applications", Submitted to both L-CSS and ACC 2024.

[155] Y. Xu, G. Gui, H. Gacanin, and F. Adachi, "A survey on resource allocation for 5g heterogeneous networks: Current research, future trends, and challenges", *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 668–695, 2021.

[156] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey", *IEEE communications surveys & tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[157] Statista, "Number of smartphone mobile network subscriptions worldwide from 2016 to 2022, with forecasts from 2023 to 2028", `https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/`. Accessed: 2023-08-08.

[158] S. Melumad and M. T. Pham, "The smartphone as a pacifying technology", *Journal of Consumer Research*, vol. 47, no. 2, pp. 237–255, 2020.

[159] Y. Chen, Q. Wang, H. Chen, X. Song, H. Tang, and M. Tian, "An overview of augmented reality technology", *Journal of Physics: Conference Series*, vol. 1237, no. 2, p. 022082, 2019.

[160] J. Carmigniani, B. Furht, M. Anisetti, P. Ceravolo, E. Damiani, and M. Ivkovic, "Augmented reality technologies, systems and applications", *Multimedia tools and applications*, vol. 51, pp. 341–377, 2011.

[161] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems", *Mobile networks and Applications*, vol. 18, pp. 129–140, 2013.

[162] F. Giust, X. Costa-Perez, and A. Reznik, "Multi-access edge computing: An overview of etsi mec isg", *IEEE 5G Tech Focus*, vol. 1, no. 4, p. 4, 2017.

[163] Q. Liu, S. Huang, J. Opadere, and T. Han, "An edge network orchestrator for mobile augmented reality", in *IEEE INFOCOM 2018-IEEE conference on computer communications*, pp. 756–764, IEEE, 2018.

[164] D. Xu, Q. Li, and H. Zhu, "Energy-saving computation offloading by joint data compression and resource allocation for mobile-edge computing", *IEEE Communications Letters*, vol. 23, no. 4, pp. 704–707, 2019.

[165] J. Ahn, J. Lee, S. Yoon, and J. K. Choi, "A novel resolution and power control scheme for energy-efficient mobile augmented reality applications in mobile edge computing", *IEEE Wireless Communications Letters*, vol. 9, no. 6, pp. 750–754, 2019.

[166] Y.-J. Seo, J. Lee, J. Hwang, D. Niyato, H.-S. Park, and J. K. Choi, "A novel joint mobile cache and power management scheme for energy-efficient mobile augmented reality service in mobile edge computing", *IEEE Wireless Communications Letters*, vol. 10, no. 5, pp. 1061–1065, 2021.

[167] F. Gunnarsson and F. Gustafsson, "Power control in wireless communications networks-from a control theory perspective", *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 183–194, 2002.

[168] A. Anzaldo and A. G. Andrade, "Deep reinforcement learning for power control in multi-tasks wireless cellular networks", in *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pp. 61–65, IEEE, 2022.

[169] T. Zhao, L. He, X. Huang, and F. Li, "Drl-based secure video offloading in mec-enabled iot networks", *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18710–18724, 2022.

[170] Y. S. Nasir and D. Guo, "Deep reinforcement learning for joint spectrum and power allocation in cellular networks", in *2021 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2021.

[171] Z. El Jamous, K. Davaslioglu, and Y. E. Sagduyu, "Deep reinforcement learning for power control in next-generation wifi network systems", in *MILCOM 2022-2022 IEEE Military Communications Conference (MILCOM)*, pp. 547–552, IEEE, 2022.

[172] F. H. C. Neto, D. C. Araújo, M. P. Mota, T. F. Maciel, and A. L. de Almeida, "Uplink power control framework based on reinforcement learning for 5g networks", *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 5734–5748, 2021.

[173] T. Zhang and S. Mao, "Smart power control for quality-driven multi-user video transmissions: A deep reinforcement learning approach", *IEEE Access*, vol. 8, pp. 611–622, 2019.

[174] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey", *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[175] S. Dzulkifly, L. Giupponi, F. Said, and M. Dohler, "Decentralized q-learning for uplink power control", in *2015 IEEE 20th International Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD)*, pp. 54–58, IEEE, 2015.

[176] H. Ding, F. Zhao, J. Tian, D. Li, and H. Zhang, "A deep reinforcement learning for user association and power control in heterogeneous networks", *Ad Hoc Networks*, vol. 102, p. 102069, 2020.

[177] A. F. Agarap, "Deep learning using rectified linear units (relu)", *arXiv preprint arXiv:1803.08375*, 2018.

[178] W. H. Organization, "Road traffic injuries", `https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries`, 2022. Accessed: August 1, 2023.

[179] SAE, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles", standard, SAE, Apr. 2021.

[180] E. Ackerman, "Toyota's gill pratt on self-driving cars and the reality of full autonomy", *IEEE Spectrum*, vol. 23, 2017.

[181] D. Lavrinc, "This is how bad self-driving cars suck in the rain", `https://jalopnik.com/this-is-how-bad-self-driving-cars-suck-in-the-rain-1666268433`. Accessed: 2023-07-14.

[182] A. Davies, "Google's self-driving car caused its first crash", `https://www.wired.com/2016/02/googles-self-driving-car-may-caused-first-crash/`. Accessed: 2023-07-14.

[183] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, J. Wang, *et al.*, "Vehicle-to-vehicle communications: readiness of v2v technology for application", tech. rep., United States. National Highway Traffic Safety Administration, 2014.

[184] M. H. C. Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Şahin, and A. Kousaridas, "A tutorial on 5g nr v2x communications", *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1972–2026, 2021.

[185] V. Milanés, J. Villagrá, J. Godoy, J. Simó, J. Pérez, and E. Onieva, "An intelligent v2i-based traffic management system", *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 49–58, 2012.

[186] W. J. Schakel, B. Van Arem, and B. D. Netten, "Effects of cooperative adaptive cruise control on traffic flow stability", in *13th International IEEE Conference on Intelligent Transportation Systems*, pp. 759–764, IEEE, 2010.

[187] I. H. Zohdy and H. A. Rakha, "Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept", *Journal of Intelligent Transportation Systems*, vol. 20, no. 1, pp. 17–32, 2016.

[188] S. E. Li, Y. Zheng, K. Li, Y. Wu, J. K. Hedrick, F. Gao, and H. Zhang, "Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities", *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 3, pp. 46–58, 2017.

[189] E. Shaw and J. K. Hedrick, "String stability analysis for heterogeneous vehicle strings", in *2007 American control conference*, pp. 3118–3125, IEEE, 2007.

[190] F. Gao, S. E. Li, Y. Zheng, and D. Kum, "Robust control of heterogeneous vehicular platoon with uncertain dynamics and communication delay", *IET Intelligent Transport Systems*, vol. 10, no. 7, pp. 503–513, 2016.

[191] D. Swaroop and J. K. Hedrick, "String stability of interconnected systems", *IEEE transactions on automatic control*, vol. 41, no. 3, pp. 349–357, 1996.

[192] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain", *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1236–1248, 2012.

[193] J.-Q. Wang, S. E. Li, Y. Zheng, and X.-Y. Lu, "Longitudinal collision mitigation via coordinated braking of multiple vehicles using model predictive control", *Integrated Computer-Aided Engineering*, vol. 22, no. 2, pp. 171–185, 2015.

[194] S. V. Balkus, H. Wang, B. D. Cornet, C. Mahabal, H. Ngo, and H. Fang, "A survey of collaborative machine learning using 5g vehicular communications", *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1280–1303, 2022.

[195] A. Khodayari, A. Ghaffari, R. Kazemi, and R. Braunstingl, "A modified car-following model based on a neural network model of the human driver effects", *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 42, no. 6, pp. 1440–1449, 2012.

[196] D. Kavitha and S. Ravikumar, "Designing an iot based autonomous vehicle meant for detecting speed bumps and lanes on roads", *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 7417–7426, 2021.

[197] J. Dinneweth, A. Boubezoul, R. Mandiau, and S. Espié, "Multi-agent reinforcement learning for autonomous vehicles: A survey", *Autonomous Intelligent Systems*, vol. 2, no. 1, p. 27, 2022.

[198] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms", *Handbook of reinforcement learning and control*, pp. 321–384, 2021.

[199] X. Qu, Y. Yu, M. Zhou, C.-T. Lin, and X. Wang, "Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: a reinforcement learning based approach", *Applied Energy*, vol. 257, p. 114030, 2020.

[200] M. Li, Z. Cao, and Z. Li, "A reinforcement learning-based vehicle platoon control strategy for reducing energy consumption in traffic oscillations", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5309–5322, 2021.

[201] H. Shi, D. Chen, N. Zheng, X. Wang, Y. Zhou, and B. Ran, "A deep reinforcement learning based distributed control strategy for connected automated vehicles in mixed traffic platoon", *Transportation Research Part C: Emerging Technologies*, vol. 148, p. 104019, 2023.

[202] M. Li, Z. Li, C. Xu, and T. Liu, "Deep reinforcement learning-based vehicle driving strategy to reduce crash risks in traffic oscillations", *Transportation research record*, vol. 2674, no. 10, pp. 42–54, 2020.

[203] H. Shi, Y. Zhou, X. Wang, S. Fu, S. Gong, and B. Ran, "A deep reinforcement learning-based distributed connected automated vehicle control under communication failure", *Computer-Aided Civil and Infrastructure Engineering*, vol. 37, no. 15, pp. 2033–2051, 2022.

[204] L. Jiang, Y. Xie, N. G. Evans, X. Wen, T. Li, and D. Chen, "Reinforcement learning based cooperative longitudinal control for reducing traffic oscillations and improving platoon stability", *Transportation Research Part C: Emerging Technologies*, vol. 141, p. 103744, 2022.

[205] Y. Gong, M. Abdel-Aty, J. Yuan, and Q. Cai, "Multi-objective reinforcement learning approach for improving safety at intersections with adaptive traffic signal control", *Accident Analysis & Prevention*, vol. 144, p. 105655, 2020.

[206] **Andrea Wrona**, D. Menegatti, and A. Giuseppi, "Deep reinforcement learning platooning control of non-cooperative autonomous vehicles in a mixed traffic environment", Submitted to both L-CSS and ACC 2024.

[207] D. Schramm, M. Hiller, and R. Bardini, "Vehicle dynamics", *Modeling and Simulation. Berlin, Heidelberg*, vol. 151, 2014.

[208] T. Ersal, I. Kolmanovsky, N. Masoud, N. Ozay, J. Scruggs, R. Vasudevan, and G. Orosz, "Connected and automated road vehicles: state of the art and future challenges", *Vehicle system dynamics*, vol. 58, no. 5, pp. 672–704, 2020.

[209] F. Lin, M. Fardad, and M. R. Jovanovic, "Optimal control of vehicular formations with nearest neighbor interactions", *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2203–2218, 2011.

[210] S. Darbha and P. Pagilla, "Limitations of employing undirected information flow graphs for the maintenance of rigid formations for heterogeneous vehicles", *International journal of engineering science*, vol. 48, no. 11, pp. 1164–1178, 2010.

[211] C.-Y. Liang and H. Peng, "Optimal adaptive cruise control with guaranteed string stability", *Vehicle system dynamics*, vol. 32, no. 4-5, pp. 313–330, 1999.

[212] L. Xiao and F. Gao, "Practical string stability of platoon of adaptive cruise control vehicles", *IEEE Transactions on intelligent transportation systems*, vol. 12, no. 4, pp. 1184–1194, 2011.

[213] T. M. Company, "Coefficient of drag for selected vehicles", `http://www.mayfco.com/tbls.htm`, 2001. Accessed: August 2, 2023.

[214] G. Maral, M. Bousquet, and Z. Sun, *Satellite communications systems: systems, techniques and technology*. John Wiley & Sons, 2020.

[215] "13° east satellites", `https://www.eutelsat.com/en/satellites/eutelsat-13-east-hotbird.html`. Accessed: 2023-06-22.

[216] B. Elbert, *The satellite communication ground segment and earth station handbook*. Artech House, 2014.

[217] M. G. Arslan and F. Alagoz, "Security issues and performance study of key management techniques over satellite links", in *2006 11th international workshop on computer-aided modeling, analysis and design of communication links and networks*, pp. 122–128, IEEE, 2006.

[218] H. Kaushal, V. Jain, and S. Kar, *Free space optical communication*. Springer, 2017.

[219] B. C. McGing, *Polybius' Histories*. Oxford University Press, 2010.

[220] K. Araki, Y. Arimoto, M. Shikatani, M. Toyoda, M. Toyoshima, T. Takahashi, S. Kanda, and K. Shiratama, "Performance evaluation of laser communication equipment onboard the ets-vi satellite", in *Free-Space Laser Communication Technologies VIII*, vol. 2699, pp. 52–59, SPIE, 1996.

[221] "A world first : Data transmission between european satellites using laser light", `https://www.esa.int/Applications/Connectivity_and_Secure_Communications/A_world_first_Data_transmission_between_European_satellites_using_laser_light`. Accessed: 2023-09-02.

[222] F. Tavares, "Nasa, partners achieve fastest space-to-ground laser comms link", `https://www.nasa.gov/centers-and-facilities/ames/nasa-partners-achieve-fastest-space-to-ground-laser-comms-link/`. Accessed: 2023-09-02.

[223] M. J. Jensen. https://fineartamerica.com/featured/free-space-optical-transceiver-mikkel-juul-jensenscience-photo-library.html. Accessed: 2023-10-21.

[224] D. W. Ball, "The electromagnetic spectrum: a history", *Spectroscopy*, vol. 22, no. 3, p. 14, 2007.

[225] D. Killinger, "Free space optics for laser communication through the air", *Optics and photonics news*, vol. 13, no. 10, pp. 36–42, 2002.

[226] H. Burchardt, N. Serafimovski, D. Tsonev, S. Videv, and H. Haas, "Vlc: Beyond point-to-point communication", *IEEE Communications Magazine*, vol. 52, no. 7, pp. 98–105, 2014.

[227] A. Biswas, M. Srinivasan, S. Piazzolla, and D. Hoppe, "Deep space optical communications", in *Free-Space Laser Communication and Atmospheric Propagation XXX*, vol. 10524, pp. 242–252, SPIE, 2018.

[228] H. Hauschildt, C. Elia, H. L. Moeller, and D. Schmitt, "Scylight—esa's secure and laser communication technology framework for satcom", in *2017 IEEE International Conference on Space Optical Systems and Applications (ICSOS)*, pp. 250–254, IEEE, 2017.

[229] L. C. Andrews, R. L. Phillips, C. Y. Hopen, and M. Al-Habash, "Theory of optical scintillation", *JOSA A*, vol. 16, no. 6, pp. 1417–1429, 1999.

[230] G. Berman, A. Chumak, and V. Gorshkov, "Beam wandering in the atmosphere: The effect of partial coherence", *Physical Review E*, vol. 76, no. 5, p. 056606, 2007.

[231] Z. Wang, J. Zhang, and H. Gao, "Impacts of laser beam divergence on lidar multiple scattering polarization returns from water clouds", *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 268, p. 107618, 2021.

[232] "Esa awards two contracts for laser-communication projects related to hydron", https://connectivity.esa.int/news/esa-awards-two-contracts-lasercommunication-projects-related-hydron. Accessed: 2023-02-05.

[233] F. Giannetti and R. Reggiannini, "Opportunistic rain rate estimation from measurements of satellite downlink attenuation: A survey", *Sensors*, vol. 21, no. 17, p. 5872, 2021.

[234] C. Sinka and J. Bitó, "Site diversity against rain fading in lmds systems", *IEEE microwave and wireless components letters*, vol. 13, no. 8, pp. 317–319, 2003.

[235] J. P. Baptista and P. Davies, "Reference book on attenuation measurement and prediction", in *2nd Workshop OPEX*, OPEX, 1994.

[236] J. Goldhirsh, B. H. Musiani, A. W. Dissanayake, and K.-T. Lin, "Three-site space-diversity experiment at 20 ghz using acts in the eastern united states", *Proceedings of the IEEE*, vol. 85, no. 6, pp. 970–980, 1997.

[237] S. Lin, H. Bergmann, and M. Pursley, "Rain attenuation on earth-satellite paths—summary of 10-year experiments and studies", *Bell System Technical Journal*, vol. 59, no. 2, pp. 183–228, 1980.

[238] M. Luglio, R. Mancini, C. Riva, A. Paraboni, and F. Barbaliscia, "Large-scale site diversity for satellite communication networks", *International journal of satellite communications*, vol. 20, no. 4, pp. 251–260, 2002.

[239] C. Nagaraja and I. E. Otung, "Statistical prediction of site diversity gain on earth-space paths based on radar measurements in the uk", *IEEE Transactions On Antennas And Propagation*, vol. 60, no. 1, pp. 247–256, 2011.

[240] C. Bruni, F. Delli Priscoli, G. Koch, A. Pietrabissa, and L. Pimpinella, "Network decomposition and multi-path routing optimal control", *Transactions on Emerging Telecommunications Technologies*, vol. 24, pp. 154–165, May 2012.

[241] A. Pietrabissa, L. Ricciardi Celsi, F. Cimorelli, V. Suraci, F. Delli Priscoli, A. D. Giorgio, A. Giuseppi, and S. Monaco, "Lyapunov-based design of a distributed wardrop load-balancing algorithm with application to software-defined networking", *IEEE Transactions on Control Systems Technology*, vol. 27, pp. 1924–1936, Sept. 2019.

[242] S. Poulenard, M. Crosnier, and A. Rissons, "Ground segment design for broadband geostationary satellite with optical feeder link", *Journal of Optical Communications and Networking*, vol. 7, no. 4, pp. 325–336, 2015.

[243] T. Rossi, M. De Sanctis, F. Maggio, M. Ruggieri, C. Hibberd, and C. Togni, "Smart gateway diversity optimization for ehf satellite networks", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, pp. 130–141, 2019.

[244] C. N. Efrem and A. D. Panagopoulos, "Globally optimal selection of ground stations in satellite systems with site diversity", *IEEE Wireless Communications Letters*, vol. 9, no. 7, pp. 1101–1104, 2020.

[245] C. Fuchs and F. Moll, "Ground station network optimization for space-to-ground optical communication links", *Journal of Optical Communications and Networking*, vol. 7, no. 12, pp. 1148–1159, 2015.

[246] N. K. Lyras, C. N. Efrem, C. I. Kourogiorgas, and A. D. Panagopoulos, "Optimum monthly based selection of ground stations for optical satellite networks", *IEEE Communications Letters*, vol. 22, no. 6, pp. 1192–1195, 2018.

[247] E. Erdogan, I. Altunbas, G. K. Kurt, M. Bellemare, G. Lamontagne, and H. Yanikomeroglu, "Site diversity in downlink optical satellite networks through ground station selection", *IEEE Access*, vol. 9, pp. 31179–31190, 2021.

[248] M. A. Zaytar and C. El Amrani, "Sequence to sequence weather forecasting with long short-term memory recurrent neural networks", *International Journal of Computer Applications*, vol. 143, no. 11, pp. 7–11, 2016.

[249] **Andrea Wrona**, E. De Santis, F. Delli Priscoli, and F. G. Lavacca, "An intelligent ground station selection algorithm in satellite optical communications via deep learning", in *2023 31st Mediterranean Conference on Control and Automation (MED)*, pp. 493–499, IEEE, 2023.

[250] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures", *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[251] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[252] D. Beniaguev, "Historical hourly weather data 2012-2017", `https://www.kaggle.com/datasets/selfishgene/historical-hourly-weather-data`, Dec 2017. Accessed June 28, 2023.

[253] GMT, "Astronomical tables for equinoxes and solstices", `https://greenwichmeantime.com/longest-day/equinox-solstice-2010-2020/`. Accessed July 6, 2023.

[254] ESA, "High throughput optical network (hydron) (scylight sl.021)",

[255] F. Chollet *et al.*, "Keras", https://github.com/fchollet/keras, 2015. Accessed: 2023-01-29.

[256] ESA, "Ot4ngsat",

[257] A. Yaqoob, T. Bi, and G.-M. Muntean, "A survey on adaptive 360 video streaming: Solutions, challenges and opportunities", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2801–2838, 2020.

[258] M. N. Sadiku, S. M. Musa, and O. D. Momoh, "Cloud computing: opportunities and challenges", *IEEE potentials*, vol. 33, no. 1, pp. 34–36, 2014.

[259] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey", *Information systems frontiers*, vol. 17, pp. 243–259, 2015.

[260] P. Chitre and F. Yegenoglu, "Next-generation satellite networks: architectures and implementations", *IEEE Communications Magazine*, vol. 37, no. 3, pp. 30–36, 1999.

[261] X. Mao, D. Arnold, V. Girardin, A. Villiger, and A. Jäggi, "Dynamic gps-based leo orbit determination with 1 cm precision using the bernese gnss software", *Advances in space research*, vol. 67, no. 2, pp. 788–805, 2021.

[262] C. B. Lim, A. Montmerle-Bonnefois, C. Petit, J.-F. Sauvage, S. Meimon, P. Perrault, F. Mendez, B. Fleury, J. Montri, J.-M. Conan, *et al.*, "Single-mode fiber coupling with adaptive optics for free-space optical communication under strong scintillation", in *2019 IEEE International Conference on Space Optical Systems and Applications (ICSOS)*, pp. 1–6, IEEE, 2019.

[263] C. Zhang, J. Jin, L. Kuang, and J. Yan, "Leo constellation design methodology for observing multi-targets", *Astrodynamics*, vol. 2, pp. 121–131, 2018.

[264] A. Tantucci, **Andrea Wrona**, and A. Pietrabissa, "Precise orbit determination on leo satellite using pseudorange and pseudorange-rate measurements", in *2023 31st Mediterranean Conference on Control and Automation (MED)*, pp. 341–347, IEEE, 2023.

[265] Y. Su, Y. Liu, Y. Zhou, J. Yuan, H. Cao, and J. Shi, "Broadband leo satellite communications: Architectures and key technologies", *IEEE Wireless Communications*, vol. 26, no. 2, pp. 55–61, 2019.

[266] P. Yue, J. An, J. Zhang, J. Ye, G. Pan, S. Wang, P. Xiao, and L. Hanzo, "Low earth orbit satellite security and reliability: Issues, solutions, and the road ahead", *IEEE Communications Surveys & Tutorials*, 2023.

[267] P. K. Chowdhury, M. Atiquzzaman, and W. Ivancic, "Handover schemes in satellite networks: State-of-the-art and future research directions", *IEEE Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 2–14, 2006.

[268] S. He, T. Wang, and S. Wang, "Load-aware satellite handover strategy based on multi-agent reinforcement learning", in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pp. 1–6, IEEE, 2020.

[269] M. Kasper, E. Fedrigo, D. P. Looze, H. Bonnet, L. Ivanescu, and S. Oberti, "Fast calibration of high-order adaptive optics systems", *JOSA A*, vol. 21, no. 6, pp. 1004–1008, 2004.

[270] R. K. Tyson and B. W. Frazier, *Principles of adaptive optics*. CRC press, 2022.

[271] G. Vosselman and H.-G. Maas, "Adjustment and filtering of raw laser altimetry data", in *Proceedings of OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Detailed Digital Terrain Models, Stockholm, Sweden*, OEEPE, 2001.

[272] S. W. Jolly, O. Gobert, and F. Quéré, "Spatio-temporal characterization of ultrashort laser beams: a tutorial", *Journal of Optics*, vol. 22, no. 10, p. 103501, 2020.

[273] B. Rödiger, D. Ginthör, J. P. Labrador, J. Ramirez, C. Schmidt, and C. Fuchs, "Demonstration of an fso/rf hybrid-communication system on aeronautical and space applications", in *Laser Communication and Propagation through the Atmosphere and Oceans IX*, vol. 11506, p. 1150603, SPIE, 2020.

[274] T. Nakatani, Y. Maekawa, Y. Shibagaki, and K. Hatsuda, "Relationship between rain front motion and site diversity in ku-band satellite links", in *25th AIAA International Communications Satellite Systems Conference (organized by APSCC)*, p. 3173, APSCC, 2007.

[275] P. Petropoulou, E. T. Michailidis, A. D. Panagopoulos, and A. G. Kanatas, "Radio propagation channel measurements for multi-antenna satellite communication systems: A survey", *IEEE Antennas and Propagation Magazine*, vol. 56, no. 6, pp. 102–122, 2014.

[276] X. Hu, Y. Zhang, X. Liao, Z. Liu, W. Wang, and F. M. Ghannouchi, "Dynamic beam hopping method based on multi-objective deep reinforcement learning for next generation satellite broadband systems", *IEEE Transactions on Broadcasting*, vol. 66, no. 3, pp. 630–646, 2020.

[277] X. Hu, S. Liu, Y. Wang, L. Xu, Y. Zhang, C. Wang, and W. Wang, "Deep reinforcement learning-based beam hopping algorithm in multibeam satellite systems", *IET Communications*, vol. 13, no. 16, pp. 2485–2491, 2019.

[278] S. Liu, X. Hu, and W. Wang, "Deep reinforcement learning based dynamic channel allocation algorithm in multibeam satellite systems", *IEEE Access*, vol. 6, pp. 15733–15742, 2018.

[279] X. Hu, S. Liu, R. Chen, W. Wang, and C. Wang, "A deep reinforcement learning-based framework for dynamic resource allocation in multibeam satellite systems", *IEEE Communications Letters*, vol. 22, no. 8, pp. 1612–1615, 2018.

[280] **Andrea Wrona** and A. Tantucci, "Artificial intelligence–based data path control in leo satellites–driven optical communications", *Artificial Intelligence*, vol. 19, p. 1, 2023.

[281] G. Seeber, *Satellite geodesy*. Walter de gruyter, 2003.

[282] O. M. E. Gill and O. Montenbruck, *Satellite orbits*. Springer, 2013.

[283] N. Ashby, "The sagnac effect in the global positioning system", in *Relativity in rotating frames: relativistic physics in rotating reference frames*, pp. 11–28, Springer, 2004.

[284] H. Henniger and O. Wilfert, "An introduction to free-space optical communications", *Radioengineering*, vol. 19, no. 2, 2010.

[285] CelesTrak, "Norad gp element sets current data", `https://celestrak.org/NORAD/elements/`. Accessed: 2023-09-09.