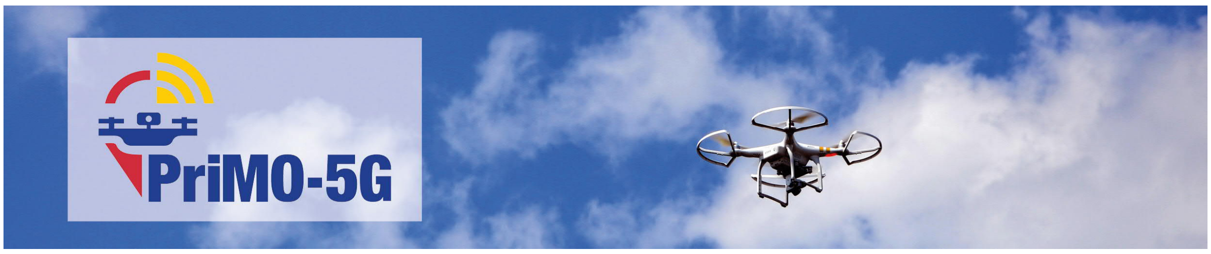


Project Title	Virtual Presence in Moving Objects through 5G
Project Acronym	PriMO-5G
Grant Agreement No	815191
Instrument	Research and Innovation Action
Topic	One of the main topics of PriMO-5G project explicitly addresses AI and machine learning based MEC and 5G network performance improvements.
Start Date of Project	01.07.2018
Duration of Project	36 Months
Project Website	https://primo-5g.eu/

Work Package	WP2, 5G Core network
Lead Author (Org)	Joongheon Kim (KU)
Contributing Author(s) (Org)	Edward Mutafungwa (AALTO), Jose Costa-Requena (CMC), András Zahemszky (EAB), HyungJoon Jeon (EUCAST), Joongheon Kim (KU), Soohyun Park (KU), Yeongeun Kang (KU),
Due Date	30.10.2020, M28
Date	13.11.2020
Version	13 (submitted)

Dissemination Level

X	PU: Public
	PP: Restricted to other programme participants (including the Commission)
	RE: Restricted to a group specified by the consortium (including the Commission)
	CO: Confidential, only for members of the consortium (including the Commission)

The work described in this document has been conducted within the project PriMO-5G. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 815191. The project is also supported by the Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-00170, Virtual Presence in Moving Objects through 5G). The dissemination of results herein reflects only the author's view, and the European Commission, IITP and MSIT are not responsible for any use that may be made of the information it contains.

Disclaimer

PriMO-5G has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 815191. The project is also supported by the Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-00170, Virtual Presence in Moving Objects through 5G). The dissemination of results herein reflects only the author's view, and the European Commission, IITP and MSIT are not responsible for any use that may be made of the information it contains.

Table of Contents

Table of Contents

List of Acronyms

Acronym	Definition
5G-PPP	5G Public Private Partnership
5GS	5G System
AAA	the authentication, authorization. and accounting
AI	artificial intelligence
AIA	Aerospace Industries Association
ANN	Artificial Neural Network
APIs	Application Program Interfaces
BS	base station
CAPEX	capital expenditure
CEPT	European Conference of Postal and Telecommunications Administrations
C-ITS	Cooperative Intelligent Transportation System
CoMP	Cooperative multipoint
COTS	Commercial-Off-The-Shelf
CRAN	cloud radio access network
CRIU	Checkpoint/Restore in Userspace
DCSP	Data Centre Service Provider
DoS	Denial of Service
DPP	drift-plus-penalty
E2E	end-to-end
ECC	Electronic Communications Committee
eMBB	enhanced Mobile BroadBand
ETSI	European Telecommunications Standards Institute
FCC	Federal Communications Commission
FD-MIMO	full dimension MIMO
GCSs	ground control stations
gNB	Next Generation NodeB
GPUs	graphical processing units
GUEs	ground users

Acronym	Definition
HAPs	High-altitude platforms
HHE	HTTP Header Enrichment
IaaS	infrastructure as-a-service
IAP	IP Announcement Point
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
KPIs	key performance indicators
LAPs	Low-altitude platforms
LOS	line-of-sight
LR	Location Register
LXD	Linux Container Daemon
MAC	media access control
MANO	Management and Orchestration
MCC	Mobile Cloud Computing
MDP	Markov Decision Process
MEC	multi-access edge computing
MNOs	Mobile Network Operators
NFV	Network Functions Virtualisation
NNs	Neural Networks
NOP	Network Operator
NSaaS	Network Slice as a Service
NWDAF	Network Data Analytics Function
OPEX	Operating Expenditure
OVS	Open vSwitch
PaaS	Platform as a Service
PPPs	Poisson Point Processes
PSNR	Peak-signal-to-noise ratio
QoS	Quality of Service
RL	Reinforcement Learning

Acronym	Definition
RMa	rural-macro
RNIS	radio network information service
RRC	radio resource control
RRM	radio resource management
RTMP	Real-Time Messaging Protocol
SC	Service Customer
SDN	Software-Defined Networking
SLAs	service level agreements
SMEs	Small and Medium-sized Enterprises
SMF	Session Management Function
SMO	Service Mobility Orchestrator
SNR	signal to noise ratio
SP	Service Provider
SSID	service set identifier
SSIM	structural similarity
SVC	scalable video coding
TCP	Transmission Control Protocol
TR	technical report
TS	technical specification
UAV	unmanned aerial vehicles
UMa	urban-macro
UMi	urban-micro
UPF	User Plane Function
URLLC	ultra reliable low latency communication
V2X	Vehicle-to-Everything
VISP	Virtualisation Infrastructure Service Provider
VMs	virtual machines

Executive Summary

The title of Deliverable D2.3 is the *AI-MEC System Design Specification and Integration Report*. This deliverable is the third report from *WP2 5G Core network*, which aims to investigate artificial intelligence (AI) and machine learning techniques for mobile edge computing. Furthermore, this document describes the system requirements and parameters for the system.

Therefore, this document describes 5G and mobile edge computing network architectures, at first. After that potential AI methods for 5G/MEC networks are presented such as artificial neural network, reinforcement learning, super-resolution, deep reinforcement learning, and Lyapunov optimization framework. Based on these foundations, various use cases are discussed. Then, the AI-MEC system design specification and integration are discussed.

1 Introduction

1.1 Purpose and Scope

The EU-KR PriMO-5G project brings together partners from several European countries from and a number of partners from South Korea, who together are addressing objectives of the 'EUK-02-2018:5G' call in the area "a) Focus on mmWave and super broadband services". Specifically, the PriMO-5G aims to demonstrate an end-to-end 5G system providing immersive video services for moving objects. This is achieved by both local and cross-continental testbeds that integrate radio access and core networks developed by different project partners to showcase end-to-end operations of envisaged use cases, particularly those related to firefighting.

This deliverable titled D2.3 *AI-MEC system design specification and integration report* is the third deliverable report from *WP2 5G Core Network*. The deliverable aims to investigate artificial intelligence (AI) and machine learning techniques for multi-access edge computing (MEC), as well as, description of system requirements and parameter considerations for demanding use cases.

1.2 PriMO-5G Use Case

The main PriMO-5G use case is about smart firefighting using mobile unmanned aerial vehicles (UAV). There are two main firefighting scenarios identified that will be investigated in the project specifically:

- Fires in rural areas with mobile infrastructure only partially available but less spectrum competition
- Fires in urban areas with mobile infrastructure available but competition in spectrum usage

Figure 1-1 shows a basic illustration of the network setup and involved entities.

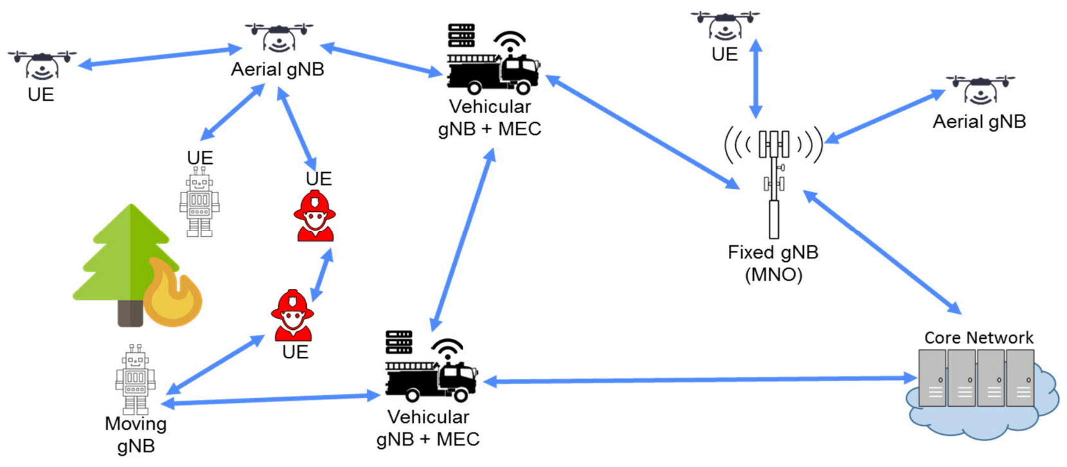


Figure 1-1 : Illustration of drone-assisted smart robot firefighting (from D1.1)

1.3 Structure of the document

In this deliverable report, Section 2 describes 5G and mobile edge computing network architectures in. Thereafter, in Section 3 potential AI methods for 5G/MEC networks are presented such as artificial neural network, reinforcement learning, super-resolution, deep reinforcement learning, and Lyapunov optimization framework. Based on these foundations, various use cases are discussed in Section 4. Application relocation is also discussed Section 4, which is applicable to edge architectures and is complementing Optimal Routing, which is included in PriMO-5G architecture. Section 5 then outlines the AI-MEC system design specification and integration, and finally conclusions are provided in Section 6.

1.4 Relationship to other project outcomes

This deliverable is inspired by the challenges derived from the use cases specified PriMO-5G WP1. Specifically, deliverable *D1.1 PriMO-5G use case scenarios* includes use cases that considered AI-based performance improvements. Furthermore, D2.1 (Initial design of MEC and Network Slice Manger) and D2.2 (PriMO-5G core interoperability report) are all discussing about 5G core networks and MEC networks. Lastly, WP4 is for AI-based communications. Therefore, the corresponding research on the use of Artificial Intelligence (AI) and machine learning improved networking are strongly related to this D2.3.

2 5G and MEC Network Architecture

2.1 Introduction

5G provides massive connectivity for everything from human-held smart devices to sensors and machines and it has the ability to support critical machine communications with instant action and ultra-high reliability. First 5G specifications are available with 3GPP Rel.15 where the focus was primarily to serve mobile operator needs in terms of extreme mobile broadband services. A second release (3GPP Rel.16) was completed in July 2020 that includes several features to support vertical in terms of enablers for Industrial IoT and URLLC. However, this is only the initial step, further enhancements and optimizations are still needed to design a 5G System that meets the challenging requirements from the vertical industries. This deliverable report also summarizes findings from 5G Public Private Partnership (5G-PPP) Phase 2 and Phase 3 projects with the aim of further identifying potential impacts to future standards releases [8].

2.2 Overall Architecture

5G networks have been targeted to meet the requirements of a highly mobile and fully connected society. The coexistence of human-centric and machine type applications will define very diverse functional and performance requirements that 5G networks will have to support. Within the 5G System (5GS), end-to-end (E2E) network slicing, service-based architecture, Software-Defined Networking (SDN), and Network Functions Virtualisation (NFV) are seen as the fundamental pillars to support the heterogeneous key performance indicators (KPIs) of the new use cases in a cost-efficient way. The 5GS gives mobile network operators the unique opportunities to offer new services to consumers, enterprises, verticals, and third-party tenants by addressing their respective requirements. To this end, 5G Infrastructure Public Private Partnership (5G PPP) Phase I/II collaborative research projects as well as standardisation bodies have specified and developed the main elements of the 5G architecture [8].

2.2.1 Roles in 5G Ecosystem

The 5G ecosystem should enable manufacturers, solution integrators, network, and service providers, and Small and Medium-sized Enterprises (SMEs) to efficiently compete and cooperate, e.g., by means of virtualisation, standardised interfaces and protocols, or open APIs. SMEs will be able to provide technological solutions which will be compatible with the overall system, e.g., new hardware components in the infrastructure or software components in the Management and Organization layers. Manufacturers and solution integrators can offer rapid deployment enabled by virtualisation and standardised interfaces to increase the level of innovation. Mobile Network Operators (MNOs) and infrastructure providers will create tailored slices with specific functionalities and Over-The-Top applications and services to address requirements of vertical industries.

This section mainly focusses on single-domain service provisioning, but do not elaborate on, e.g., cross-operator scenarios. Additionally, the 3GPP roles are defined from the point of view of an operator. 5G PPP Phase I/II collaborative research projects have extended these roles to allow various possible customer-provider relationships between verticals, operators, and another stakeholder, as in Figure 2-1:

- **Service Customer (SC):** uses services that are offered by a Service Provider (SP). In the context of 5G, vertical industries are considered as one of the major SCs.
- **Service Provider (SP):** comprises three sub-roles, depending on the service offered to the SC: Communication Service Provider offering traditional telecom services, Digital Service Provider offering digital services such as enhanced mobile broadband and IoT to various vertical industries, or Network Slice as a Service (NSaaS) Provider offering a network slice along with the services that it may support and configure. SPs design, build and operate services using aggregated network services.

- **Network Operator (NOP):** in charge of orchestrating resources, potentially from multiple virtualised infrastructure providers (VISP). The NOP uses aggregated virtualised infrastructure services to design, build, and operate network services that are offered to SPs.
- **Virtualisation Infrastructure Service Provider (VISP):** Provides virtualised infrastructure services and designs, builds, and operates virtualisation infrastructure(s). The infrastructure comprises networking (e.g., for mobile transport) and computing resources (e.g., from computing platforms).
- **Data Centre Service Provider (DCSP):** Provides data centre services and designs, builds, and operates its data centres. A DCSP differs from a VISP by offering "raw" resources (i.e., host servers) in rather centralised locations and simple services for consumption of these raw resources. A VISP rather offers access to a variety of resources by aggregating multiple technology domains and making them accessible through a single API.

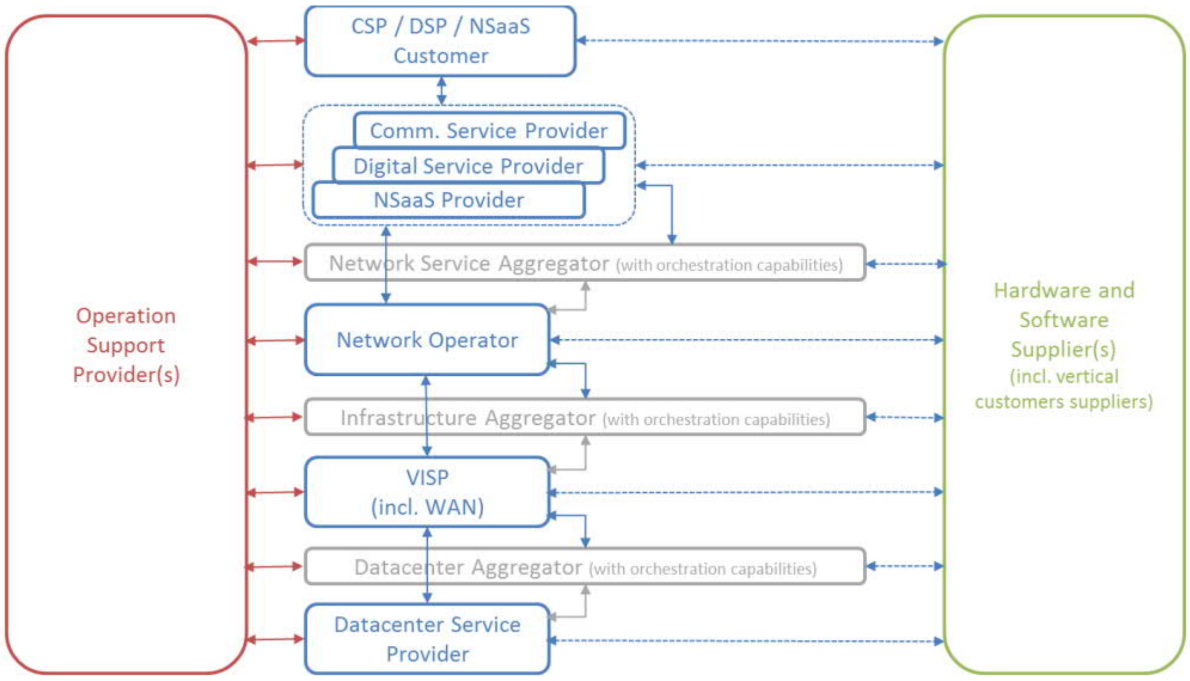

Operation Support Provider(s)
CSP / DSP / NSaaS Customer
Comm. Service Provider
Digital Service Provider
NSaaS Provider
Network Service Aggregator (with orchestration capabilities)
Network Operator
Infrastructure Aggregator (with orchestration capabilities)
VISP (incl. WAN)
Datacenter Aggregator (with orchestration capabilities)
Datacenter Service Provider
Hardware and Software Supplier(s) (incl. vertical customers suppliers)

Figure 2-1 : Roles in 5G Ecosystem

2.2.2 *Overall System Architecture Enhanced by 5G*

The prospects of network slicing, i.e., executing multiple logical mobile network instances on a shared infrastructure, require a continuous reconciliation of customer-centric service level agreements (SLAs) with infrastructure-level network performance capabilities. Service customers, e.g., from the vertical industries, request the creation of (tele)communication services by providing "customer-facing" on-demand service requirement descriptions to Service Providers. In the past, operators executed such mapping in a manual manner on a limited number of service/slice types (mainly mobile broadband, voice, and SMS). With an increased number of such customer requests, an E2E framework for Service Creations and Service Operations will therefore have to exhibit a significantly increased level of automation for the lifecycle management of network slice instances.

On the Service Level, lifecycle management automation must be realized by closed-loop Service Assurance, Service Fulfilment, and Service Orchestration functions (cf. Figure 2-2) covering all lifecycle phases: preparation phase, instantiation, configuration and activation phase, run-time phase, and decommissioning phase. Two fundamental technological enablers include softwarisation, e.g., virtualisation of network functions, as well as software-defined, programmable network functions and infrastructure resources. E2E Service Operations functions interact with functions for Management of Domain Resources and Functions. Example domains include RAN, Core & Transport Network, as well as NFV and MEC. Besides orchestration, closed-loop procedures for resource fulfilment, resource assurance, and network intelligence comprise building blocks within each management domain. On a more fine-grained temporal and spatial level, domain-specific controllers, incl. SDN controllers, can be programmed to efficiently execute policies and rules on the Resources and Functional Level.

Finally, a common platform, where data can be accessed by system entities from all levels, uses scalable data exposure governance and access control mechanisms to provides services for data acquisition, processing, abstraction, and distribution. This includes data related to subscribers, to the network and underlying resources, to network slice and service instances, and, if required by the vertical customer, to applications.

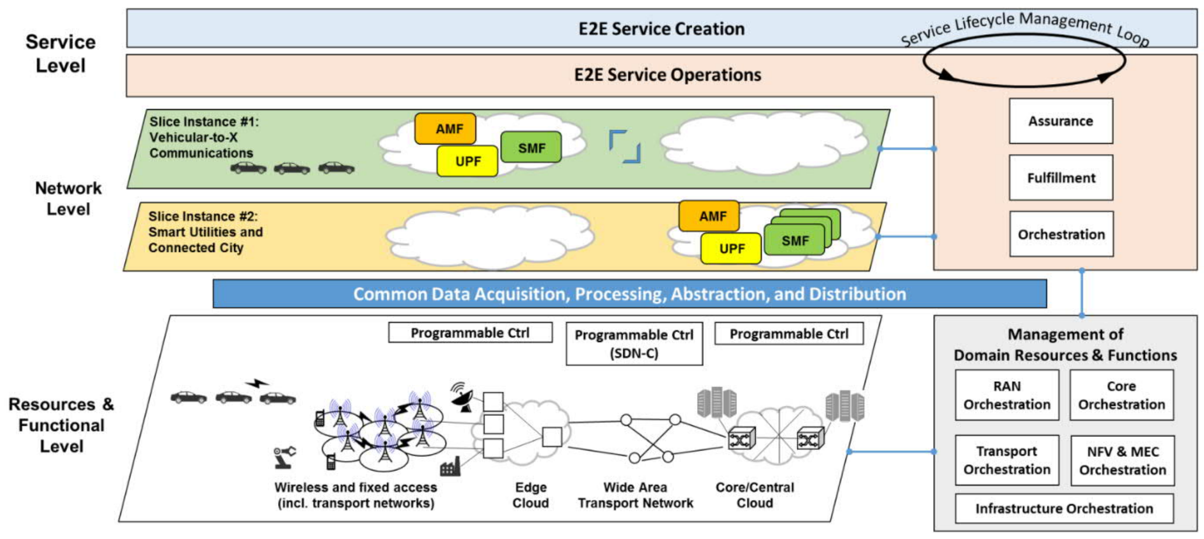

Service Level
E2E Service Creation
Service Lifecycle Management Loop
E2E Service Operations
Network Level
Slice Instance #1: Vehicular-to-X Communications
AMF
SMF
UPF
Assurance
Fulfillment
Orchestration
Slice Instance #2: Smart Utilities and Connected City
Common Data Acquisition, Processing, Abstraction, and Distribution
Resources & Functional Level
Programmable Ctrl
Programmable Ctrl (SDN-C)
Programmable Ctrl
Management of Domain Resources & Functions
RAN Orchestration
Core Orchestration
Transport Orchestration
NFV & MEC Orchestration
Infrastructure Orchestration
Wireless and fixed access (incl. transport networks)
Edge Cloud
Wide Area Transport Network
Core/Central Cloud

Figure 2-2: Overall Architecture Enhanced by 5G

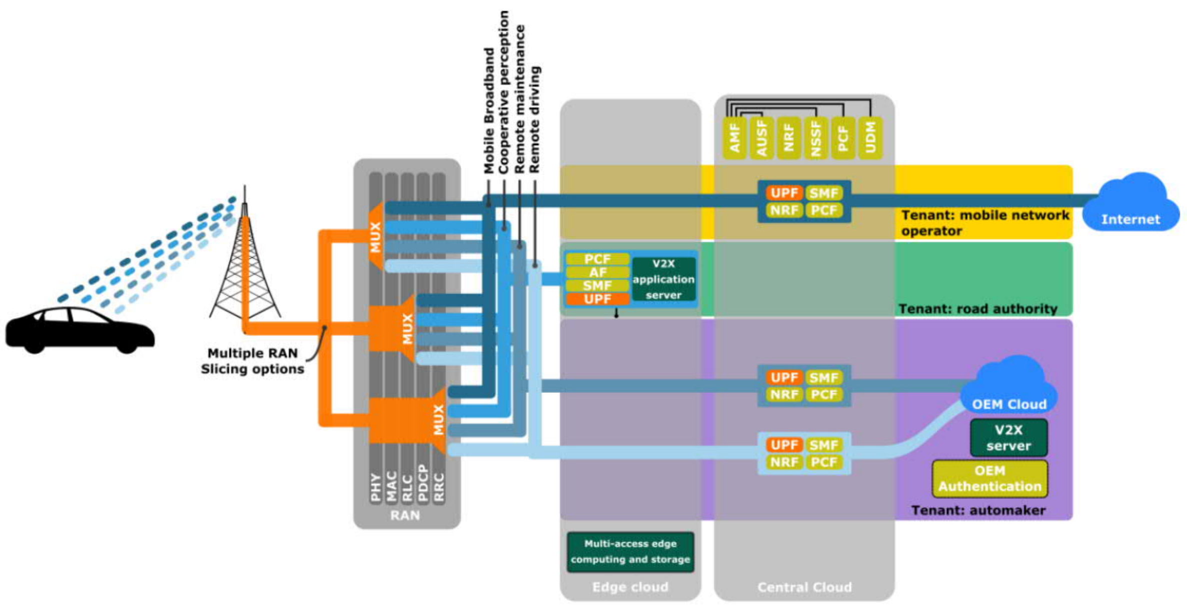
The proposed architecture realizes a recursive structure. A recursive structure in the 5G context can be defined as a design, rule, or procedure that can be applied repeatedly. In a network service context, this recursive structure applies to a specific part of either a network service or the deployment platform. It is defined as the ability to build a service out of existing services, including another instance of the very same service. As with a recursive service definition, a recursive structure in the 5G architecture can be instantiated and linked repeatedly. It improves scalability, as the same service category can be deployed many times, at different places at the same time. Delegating parts of the service to multiple instances of the same software block is a natural way to handle more complex and larger workloads or service graphs.

In the context of virtualised infrastructure, such recursive structure allows a slice instance operating on top of the infrastructure resources provided by another slice instance. For example, each tenant can own and deploy its own Management and Orchestration (MANO) system. To support the recursion, a set of homogeneous APIs are needed for providing a layer of abstraction for the management of each slice and controlling the underlying virtual resources which is transparent to the level of the hierarchy where the tenant is operating. Different tenants request the provisioning of slices through these APIs. By means of a template, blueprint, or SLA, each tenant specifies not only the slice characteristics (topology, QoS, etc.) but also some extended attributes such as the level of resiliency, management, and control desired.

2.2.3 *Extensions to Vehicular Communications*

Vehicular communications simultaneously involve multiple use cases, traffic types, and communication paths. In fact, in addition to transmissions routed through the core network towards remote servers, links between vehicular UEs in proximity may involve the PC5 link for direct Vehicle-to-Vehicle communications, whereas local breakout can be applied for network assisted links routed through the edge network. In this latter case, the base station or roadside unit can locally relay the messages to the UEs in proximity, and/or route them to UEs attached to neighbouring base stations via short routing paths passing through the edge data centre.

Automotive applications include a wide set of services, offered by different providers, each imposing a specific set of requirements. For this reason, vehicular communications rely on the network slicing feature, where the lifecycle management of each slice is tailored to support the related service, cf. Figure 2-3 This collection is composed by slices belonging to the standard types already defined by 3GPP, notably eMBB and URLLC, thus exploiting the flexibility provided by the standardised slice types.



Control Plane	AMF	Core Access and Mobility Management Function
	AUSF	Authentication Server Function
	NRF	Network function Repository Function
	NSSF	Network Slice Selection Function
	PCF	Policy Control Function
	SMF	Session Management Function
	UDM	Unified Data Management
User Plane	UPF	User Plane Function

Figure 2-3: Example of slice-based architecture for vehicular communications

For the automotive vertical, it is important that network functions can be deployed both in the edge and central cloud, according to the requirements they are designed to serve. The edge cloud hosts NFs which need to be allocated in proximity of the UEs, potentially including additional features such as Multi-access Edge Computing (MEC) and storage facilities. The central cloud, on the other hand, contains the slice-specific network functions for use cases requiring connectivity with a remote public network.

The concept of multi-tenancy is leveraged in vehicular applications, wherein the tenant is the company, vertical, or service provider offering the services supported by one slice, or one set of slices. Examples of tenants for automotive applications are mobile network operators, road operators, and automakers.

Road authorities may provide Cooperative Intelligent Transportation System (C-ITS) services like hazard warning, in-vehicle signage, and in general cooperative perception and cooperative manoeuvre services. These services involve information that is both strictly time-sensitive and location-sensitive: messages are in fact transmitted and received by vehicles to spread and acquire safety-critical information about the instantaneous traffic conditions in their surroundings. These services hence require a low latency slice with high reliability, providing timely reception of these messages. For this reason, network resources are foreseen to be mostly allocated in the edge cloud, as close as possible to the road users. Alternatively, sufficient transport network resources towards the central cloud must be allocated for the slice.

Automakers may offer different classes of services to their clients, such as remote maintenance and tele-operated driving. Both require connectivity between the vehicle and the automaker's cloud, although each with completely different service level requirements. In the former case, the machine-type communications, which could be delivered via an eMBB slice, is used to retrieve data from the on-board sensor to plan ahead the maintenance of large vehicular fleets. In contrast, remote driving requires low latency, high data rate, and high reliability in the uplink to provide a real-time video flow and instantaneous sensor data to the remote driver. Similarly, the downlink needs to deliver the driving commands to the vehicle. While both services mostly rely on network functions running in the operator's central cloud, they have completely different degrees of redundancy. Furthermore, the automaker can implement further authentication functions beyond those offered by the network, as well as hosting Vehicle-to-Everything (V2X) application servers in their premises. In any case, network slices can leverage dedicated bearers to provide tailored QoS to specific applications/flows also within the same slice. Distinct applications are assigned to specific bearers (in the RAN), and flows (in the core network), which are treated with a different level of priority.

2.3 Edge Computing

2.3.1 Concept

Multi-access Edge Computing (MEC) is an emerging technology in 5G era which enables the provision of the cloud and IT services within the close proximity of mobile subscribers. It allows the availability of the cloud servers inside or adjacent to the base station. The end-to-end latency perceived by the mobile user is therefore reduced with the MEC platform. The context-aware services are able to be served by the application developers by leveraging the real time radio access network information from MEC. The MEC additionally enables the compute intensive applications execution in the resource constraint devices with the collaborative computing involving the cloud servers.

To address the problem of a long latency, the cloud services should be moved to a proximity of the UEs, i.e., to the edge of mobile network as considered in newly emerged edge computing paradigm. The edge computing can be understood as a specific case of the Mobile Cloud Computing (MCC). Nevertheless, in the conventional MCC, the cloud services are accessed via the Internet connection while in the case of the edge computing, the computing/storage resources are supposed to be in proximity of the UEs (in sense of network topology). Hence, the MEC can offer significantly lower latencies and jitter when compared to the traditional MCC. Moreover, while the MCC is fully centralized approach with farms of computers usually placed at one or few locations, the edge computing is supposed to be deployed in fully distributed manner. On the other hand, the edge computing provides only limited computational and storage resources with respect to the MCC. A high-level comparison of key technical aspects of the MCC and the edge computing is outlined in Table 2-1.

Technical aspect	MCC	Edge computing
Deployment	Centralized	Distributed
Distance to the UE	High	Low
Latency	High	Low
Jitter	High	Low
Computational power	Ample	Limited
Storage capacity	Ample	Limited

Table 2-1: High level comparison of MCC and edge computing concepts.

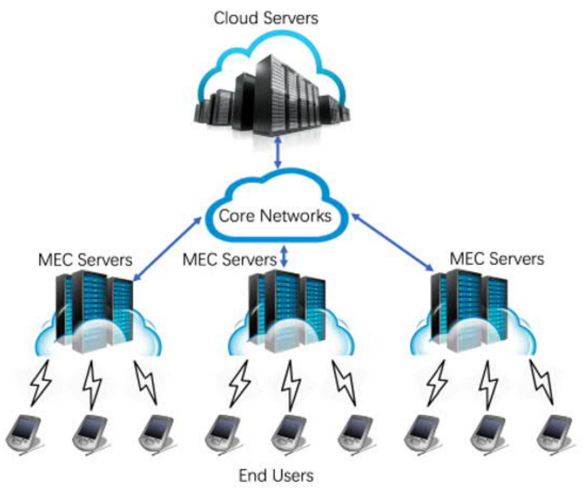


Figure 2-4: The basic edge computing architecture.

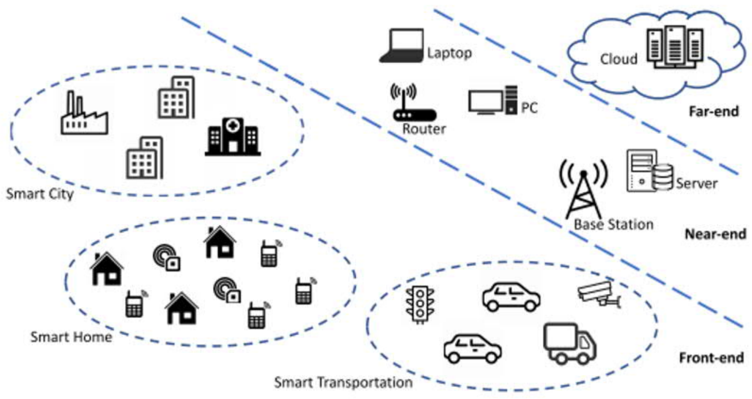


Figure 2-5: A typical architecture of edge computing networks.

Figure 2-4 illustrates the basic architecture of edge computing. Notice that the edge computing servers are closer to the end user than cloud servers. Thus, even though the edge computing servers have less computation power than the cloud servers, they still provide better QoS (Quality of Service) and lower latency to the end users. To study the advantages and disadvantages of edge computing, we will focus on the architectures of both, and compare the two. Obviously, unlike cloud computing, edge computing incorporates edge computation nodes into the network. In this paper, the edge computation nodes are called edge/cloudlet servers. Generally speaking, the structure of edge computing can be divided into three aspects, the front-end, near-end, and far-end, as shown in Figure 2-5. The differences among these areas are described below in detail.

The end devices (e.g., sensors, actuators) are deployed at the front-end of the edge computing structure. The frontend environment can provide more interaction and better responsiveness for the end users. With the computing capacity provided by the plethora of nearby end devices, edge computing can provide real-time services for some applications. Nonetheless, due to the limited capacity of the end devices, most requirements cannot be satisfied at the frontend environment. Thus, in these cases, the end devices must forward the resource requirements to the servers.

The gateways deployed in the near-end environment will support most of the traffic flows in the networks. The edge/cloudlet servers can have also numerous resource requirements, such as real-time data processing, data caching, and computation offloading. In edge computing, most of the data computation and storage will be migrated to this near end environment. In doing so, the end users can achieve a much better performance on data computing and storage, with a small increase in the latency.

As the cloud servers are deployed farther away from the end devices, the transmission latency is significant in the networks. Nonetheless, the cloud servers in the far-end environment can provide more computing power and more data storage. For example, the cloud servers can provide massive parallel data processing, big data mining, big data management, machine learning, etc.

MEC, shown in Figure 2-6, refers to the high performance and telco-grade cloud platform adjacent to the RAN allowing the computation to happen at the network edge. It works on both the downstream data from the cloud service host to the mobile terminal and the upstream data from the mobile terminal to the cloud host. The MEC platform is composed of the standard IT servers and the network devices inside or outside of the base station. The 3rd party applications are deployed and executed within the virtual machines (VMs)[90] interconnected by the network devices. It is also possible to build the MEC platform simply with the standard IT servers where the network device is implemented as the software entity such as Open vSwitch (OVS)[91]. The fundamental functionalities in the MEC platform includes routing modular, network capability exposure modular and management modular. The routing modular is responsible for the packet forwarding among the MEC platform, the RAN and the mobile core network as well as within the MEC platform. The network capability exposure modular enables the authorized exposure of the radio network information service (RNIS) and the radio resource management (RRM). The management modular supports the authentication, authorization. and accounting (AAA) and management of the 3rd applications in the MEC platform. In particular, it involves the orchestration for the application deployment and the authorization on the network capability exposure.

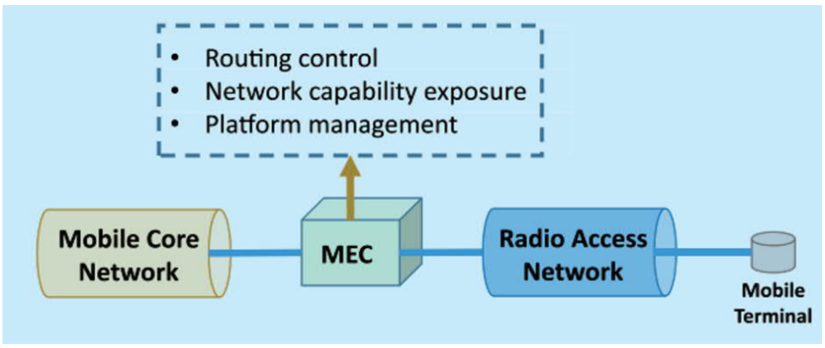


Figure 2-6: MEC architecture.

The routing modular conducts the traffic offloading to the applications hosted in local cloud infrastructure in the following two ways: Pass-through mode where (uplink and/ or downlink) user plane traffic is passed to an application which can monitor, modify or shape it and then send it back to the original connection. End-point mode where the traffic is terminated by the application running in a server. The routing modular should enable the service continuity given mobile terminal mobility. As the mobile terminal switches to the access point connecting the different MEC platform, the routing modular should eliminate the session interruption. The coordination between the routing modular in the various MEC platforms is needed. The routing modular should additionally conduct the traffic forwarding among the internal VMs in MEC. Especially, the network virtualization is supported to facilitate the flexible packet-forwarding backplane where network and security services are allocated to the VM following its needs with the managed programmability.

The network capability exposure modular is to provide a means to securely expose the services and capabilities abstracted from the underlying mobile network to the 3rd party application. The access to network capabilities is through the homogenous Application Programming Interfaces (APIs). Variety of network services such as SMS, MMS, and Location etc. are able to be exposed by invoking the relevant API. With the advent of 5G, the suitable APIs should be introduced for the dynamic customization of the dedicated network slice cater supporting different diverse use cases by the 3rd party application, within the limits set by the operator. The APIs for network capability exposure are called with a set of middleware executed as the Platform as a Service (PaaS) facilities. They are called following the open standards in terms of the framework and reference architecture and the platform services specifications (i.e. RESTful API). There are two approaches for the API enabling in the network capability exposure.

One is to have the 3rd party application establish the standalone session with the specific PaaS facility.

The other is to enable the API calling with the HTTP Header Enrichment (HHE) wherein the user traffic is inspected by the network capability exposure modular which then calls the API on behalf of the 3rd party application.

The management modular conducts the management on other modular and the local IT infrastructure. It enables the AAA operation on the network capability exposure and the local IT infrastructure resource assigned to the 3rd party application. The management on the local IT infrastructure is mostly deployed as an infrastructure as-a-service (IaaS) such as OpenStack. It consists of the interrelated components controlling the hardware pools of computation, storage, and networking resources to enable the planning and orchestration of IT resource as required by the 3rd party application. The management on the network capability exposure and routing modular is built as the PaaS entity. It includes the creation, deletion, authentication. and registration of the middleware. The standardized environment should be provisioned so that the MEC platform may accommodate the components from the diverse vendors.

The foremost challenge associated with mobile data offloading is user experience in the presence of user mobility. The consistent user experience and service continuity must be ensured, independent of the user location and the edge service delivery location. In other words, the initial offloading should be maintained when the user with the offloaded session changes the point of attachment. It is possible to introduce the data forwarding between the distinct MEC platforms to enable the service continuity. The current 3GPP solutions enable the local breakout with PDN connection granularity requested as the UE. It implies that the UE must be with the knowledge that which application session could be served at the edge of the specific base station. It potentially leads to the complicated update on the UE given the change of the service deployment. Obviously, the 3rd party application provider should be allowed to tune the offloading policy due to its global view on the service deployment. An additional issue with the data offloading is the flexibility in the sense that the versatile forwarding function should be enabled. It potentially requires the routing modular to be implemented with the software defined networking (SDN) switch which is excellent in supporting the diverse forwarding policy but should be extended to support the GTP-U-based user traffic processing and the high-performance traffic forwarding which may exceed 10Gbps in 5G era.

Although the problem of offloading decision is well addressed in the current efforts, the application partitioning is still challenging due to the spatiotemporal resource requirements of heterogeneous mobile devices. It should consider which granularity to dynamically choose for offloading at different hierarchy of mobile terminal and edge cloud to adapt the changes in network. Firstly, the changes in the network bandwidth or latency require the variable application partitioning between the mobile terminal and the MEC server. Specifically, the remote execution of the applications in the MEC server may induce the additional latency caused by the data delivery. The suddenly decreasing bandwidth thereby might result in the poor user experiences. Secondly, the wireless link for the data delivery in the application partitioning requires the lightweight mechanism for the establishment and management of distributed computation entities residing in the mobile client and the MEC server respectively so that the minimal communication overhead is involved. Thirdly, the software engineering practices for application development in the computation offloading need the minimum efforts from application developers to enable the rapid development with the reduced developmental time and cost. Especially, the significant diversity in today’s mobile clients ranging from the high-end smartphone with the powerful CPU and GPU to the wearable devices running with the SoC integrating the constraint CPU and peripheral imposes the necessity in the efficient application developments for the computation offloading.

The cloud infrastructure in MEC is to unite the telco and IT-cloud computing capabilities within the RAN. It should enable the applications to be hosted in a multi-vendor data center environment by leveraging the IT virtualization technology to consolidate multiple capabilities into the Commercial-Off-The-Shelf (COTS) servers, switches and storage. Used as a node in the edge cloud infrastructure, the MEC devices should serve as a reliable computing unit and run the virtualized applications from the developer partners. It should be the computation and networking platform delivering data center performance at the edge of the network specifically built in the extreme environments dedicated for the telecom cellular system where the deployment space is very constraint and generally outdoor. The edge cloud infrastructure therefore should be carefully designed to adapt to the highly difficult environment. For example, the standard data center rack is with the size more than 40U while the edge cloud infrastructure in MEC is generally allowed to be deployed in the rack space no larger than 10U.

3 Potential AI Methods for 5G/MEC Networks

3.1 Introduction

In this section, the major AI methods for 5G and MEC networks are summarized. First of all, the artificial neural network algorithm that is the foundation of deep learning computation is explained. Next, reinforcement learning algorithms are introduced that is useful for discrete-time stochastic sequential decision making. After that, one of the major computer vision algorithms is introduced that is essential for mobile platforms that is called super-resolution. Lastly, Lyapunov optimization framework that is needed for stable execution of deep learning algorithms in mobile and network platforms is introduced.

3.1.1 Artificial Neural Network

Origin and Motivation: Many perspectives can be assumed for analysing Neural Networks (NNs), ranging from a biological perspective to a purely mathematical point of view. I will start to describe NNs from biological foundations and move on to their mathematical properties and applications for machine learning.

An Artificial Neural Network (ANN) is a mathematical model that tries to simulate the structure and functionalities of biological neural networks. Basic building block of every artificial neural network is artificial neuron, that is, a simple mathematical model (function). Such a model has three simple sets of rules: multiplication, summation. and activation. At the entrance of artificial neuron, the inputs are weighted what means that every input value is multiplied with individual weight. The middle section of artificial neuron is sum function that sums all weighted inputs and bias. At the exit of artificial neuron, the sum of previously weighted inputs and bias is passing through activation function that is also called transfer function (Figure 3-1).

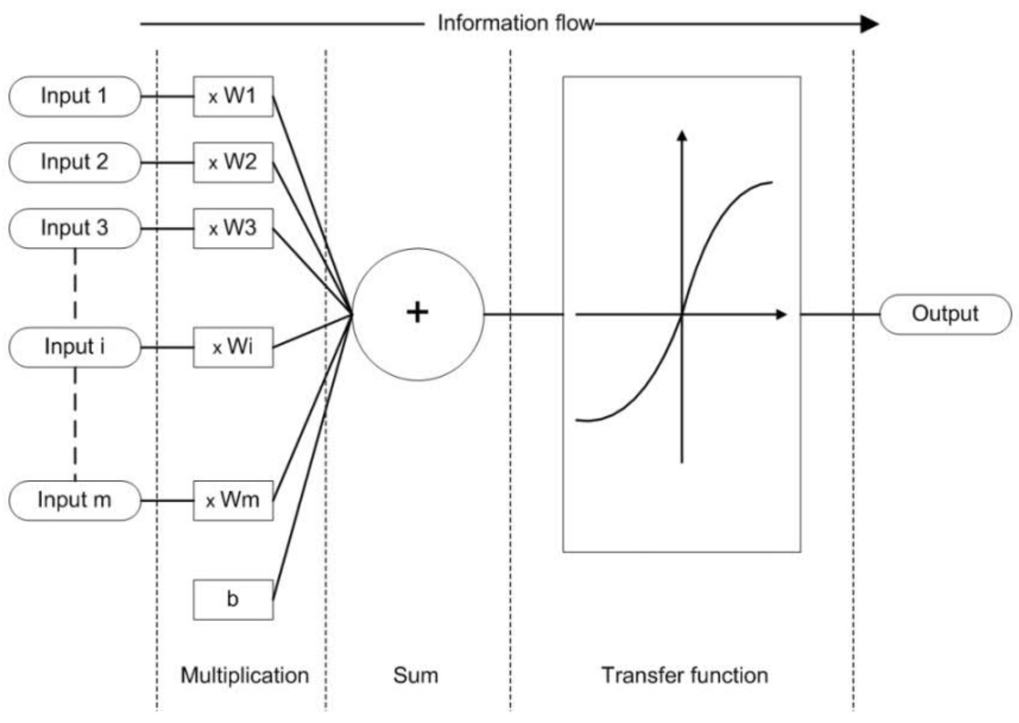


Figure 3-1: Working principle of an artificial neuron. [59]

Although the working principles and simple set of rules of artificial neuron looks like nothing special the full potential and calculation power of these models come to life when we start to interconnect them into artificial neural networks (Figure 3-2). These artificial neural networks use simple fact that complexity can grow out of merely few basic and simple rules.

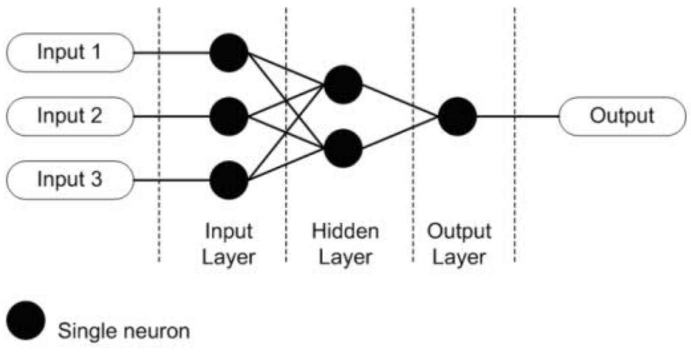


Figure 3-2: Example of simple artificial neural network. [59]

In order to fully harvest the benefits of mathematical complexity that can be achieved through interconnection of individual artificial neurons and not just making system complex and unmanageable we usually do not interconnect these artificial neurons randomly. In the past, researchers have come up with several “standardised” topographies of artificial neural networks. These predefined topographies can help us with easier, faster, and more efficient problem solving. Different types of artificial neural network topographies are suited for solving different types of problems. After determining the type of given problem, we need to decide for topology of artificial neural network we are going to use and then fine-tune it. We need to fine-tune the topology itself and its parameters.

Fine-tuned topology of artificial neural network does not mean that we can start using our artificial neural network, it is only a precondition. Before we can use our artificial neural network, we need to teach it solving the type of given problem. Just as biological neural networks can learn their behaviour/responses on the basis of inputs that they get from their environment the artificial neural networks can do the same. There are three major learning paradigms: supervised learning, unsupervised learning and reinforcement learning. We choose learning paradigm similar as we chose artificial neuron network topography - based on the problem we are trying to solve. Although learning paradigms are different in their principles, they all have one thing in common; on the basis of “learning data” and “learning rules” (chosen cost function) artificial neural network is trying to achieve proper output response in accordance to input signals.

After choosing topology of an artificial neural network, fine-tuning of the topology and when artificial neural network has learnt a proper behaviour, we can start using it for solving given problem. Artificial neural networks have been in use for some time now and we can find them working in areas such as process control, chemistry, gaming, radar systems, automotive industry, space industry, astronomy, genetics, banking, fraud detection, etc. and solving of problems like function approximation, regression analysis, time series prediction, classification, pattern recognition, decision making, data processing, filtering, clustering, etc., naming a few. [60], [61], [62], [63]

3.1.1.1 Artificial neuron

Artificial neuron is a basic building block of every artificial neural network. Its design and functionalities are derived from observation of a biological neuron that is basic building block of biological neural networks (systems) which includes the brain, spinal cord, and peripheral ganglia. Similarities in design and functionalities can be seen in Figure 3-3 where the left side of a figure represents a biological neuron with its soma, dendrites, and axon and where the right side of a figure represents an artificial neuron with its inputs, weights, transfer function, bias and outputs.

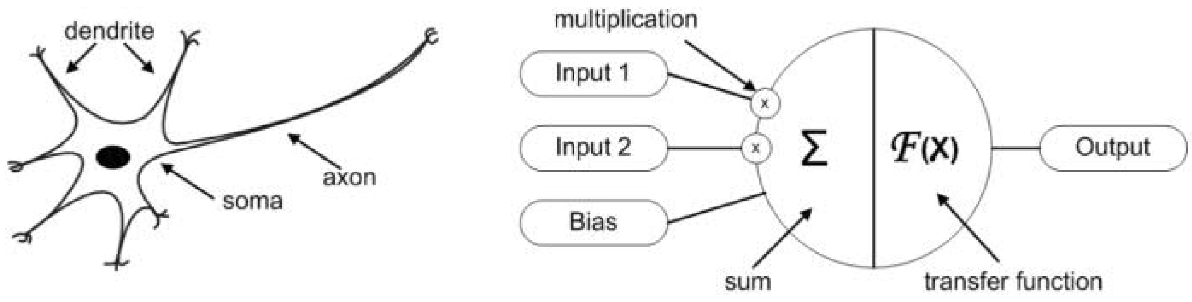


Figure 3-3: Biological and artificial neuron design. [59]

In case of biological neuron information comes into the neuron via dendrite, soma processes the information and passes it on via axon. In case of artificial neuron, the information comes into the body of an artificial neuron via inputs that are weighted (each input can be individually multiplied with a weight). The body of an artificial neuron then sums the weighted inputs, bias and “processes” the sum with a transfer function. At the end an artificial neuron passes the processed information via output(s). Benefit of artificial neuron model simplicity can be seen in its mathematical description below:

$$y(k0 = F\left(\sum_{i=0}^{m} w_i(k)x_i(k) + b\right)$$

Where:

- $x_i(k)$ is input value in discrete time k where i goes from 0 to m,
- $w_i(k)$ is weight value in discrete time k where i goes from 0 to m,
- b is bias
- F is a transfer function
- $y_i(k)$ is output value in discrete time k.

As seen from a model of an artificial neuron and its equation the major unknown variable of our model is its transfer function. Transfer function defines the properties of artificial neuron and can be any mathematical function. We choose it on the basis of problem that artificial neuron (artificial neural network) needs to solve and in most cases we choose it from the following set of functions: $Step\ function, Linear\ function\ and\ Non-linear(Sigmoid)\ function.$

Step function is binary function that has only two possible output values (e.g. zero and one). That means if input value meets specific threshold the output value results in one value and if specific threshold is not meet that results in different output value. Situation can be described with next equation.

$$y = \begin{cases} 1\ if\ w_i x_i \geq threshold \\ 0\ if\ w_i x_i < threshold \end{cases}$$

When this type of transfer function is used in artificial neuron, we call this artificial neuron perceptron. Perceptron is used for solving classification problems and as such it can be most commonly found in the last layer of artificial neural networks. In case of linear transfer function artificial neuron is doing simple linear transformation over the sum of weighted inputs and bias. Such an artificial neuron is in contrast to perceptron most commonly used in the input layer of artificial neural networks. When we use non-linear function the sigmoid function is the most commonly used. Sigmoid function has easily calculated derivate, which can be important when calculating weight updates in the artificial neural network.

3.1.1.2 Artificial Neural Networks

When combining two or more artificial neurons we are getting an artificial neural network. Even if single artificial neuron has almost no usefulness in solving real-life problems the artificial neural networks have it. In fact artificial neural networks are capable of solving complex real-life problems by processing information in their basic building blocks (artificial neurons) in a non-linear, distributed, parallel and local way.

The way that individual artificial neurons are interconnected is called topology, architecture, or graph of an artificial neural network. The fact that interconnection can be done in numerous ways results in numerous possible topologies that are divided into two basic classes. Figure 3-4 shows these two topologies; the left side of the figure represent simple feedforward topology (acyclic graph) where information flows from inputs to outputs in only one direction and the right side of the figure represent simple recurrent topology (semicyclic graph) where some of the information flows not only in one direction from input to output but also in opposite direction. While observing Figure 3-4 we need to mention that for easier handling and mathematical describing of an artificial neural network we group individual neurons in layers. On Figure 3-4 we can see input, hidden and output layer.

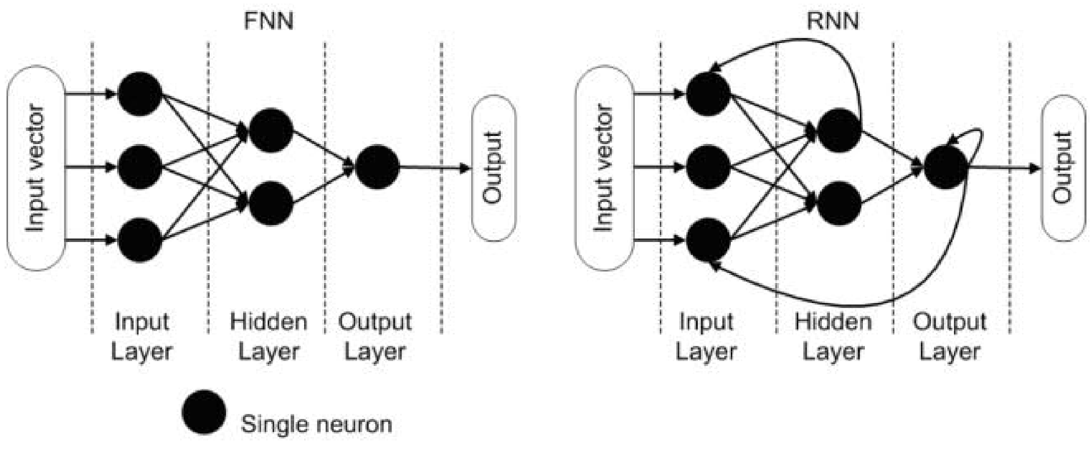


Figure 3-4: Feed-forward (FNN) and recurrent (RNN) topology of an artificial neural network. [59]

When we choose and build topology of our artificial neural network, we only finished half of the task before we can use this artificial neural network for solving given problem. Just as biological neural networks need to learn their proper responses to the given inputs from the environment the artificial neural networks need to do the same. So the next step is to learn proper response of an artificial neural network and this can be achieved through learning (supervised, un-supervised or reinforcement learning). No matter which method we use, the task of learning is to set the values of weight and biases on basis of learning data to minimize the chosen cost function.

3.1.2 Reinforcement Learning

Reinforcement Learning (RL) is the area of machine learning that deals with sequential decision-making. In this chapter, we describe how the RL problem can be formalized as an agent that has to make decisions in an environment to optimize a given notion of cumulative rewards. It will become clear that this formalization applies to a wide variety of tasks and captures many essential features of artificial intelligence such as a sense of cause and effect as well as a sense of uncertainty and nondeterminism. This chapter also introduces the different approaches to learning sequential decision-making tasks and how deep RL can be useful.

A key aspect of RL is that an agent learns a good behaviour. This means that it modifies or acquires new behaviours and skills incrementally. Another important aspect of RL is that it uses trial-and-error experience (as opposed to e.g., dynamic programming that assumes full knowledge of the environment a priori). Thus, the RL agent does not require complete knowledge or control of the environment; it only needs to be able to interact with the environment and collect information. In an offline setting, the experience is acquired a priori, then it is used as a batch for learning (hence the offline setting is also called batch RL).

This is in contrast to the online setting where data becomes available in a sequential order and is used to progressively update the behaviour of the agent. In both cases, the core learning algorithms are essentially the same but the main difference is that in an online setting, the agent can influence how it gathers experience so that it is the most useful for learning. This is an additional challenge mainly because the agent has to deal with the exploration/exploitation dilemma while learning. But learning in the online setting can also be an advantage since the agent is able to gather information specifically on the most interesting part of the environment. For that reason, even when the environment is fully known, RL approaches may provide the most computationally efficient approach in practice as compared to some dynamic programming methods that would be inefficient due to this lack of specificity.

3.1.2.1 Formal framework

The reinforcement learning setting

The general RL problem is formalized as a discrete time stochastic control process where an agent interacts with its environment in the following way: the agent starts, in a given state within its environment $s_0 \in$S, by gathering an initial observation $\omega_0 \in \Omega$. At each time step t, the agent has to take an action $a_t \in$A. As illustrated in Figure 3-5, it follows three consequences: (i) the agent obtains a reward $r_t \in$R, (ii) the state transitions to $s_{t+1} \in S$, and (iii) the agent obtains an observation $\omega_{t+1} \in \Omega$. This control setting was first proposed by [64] and later extended to learning by Barto *et al*., 1983 [65]. Comprehensive treatment of RL fundamentals are provided by Sutton and Barto, 2017 [66]. Here, we review the main elements of RL before delving into deep RL in the following chapters.

The Markov property

For the sake of simplicity, let us consider first the case of Markovian stochastic control processes (Norris, 1998) [67].

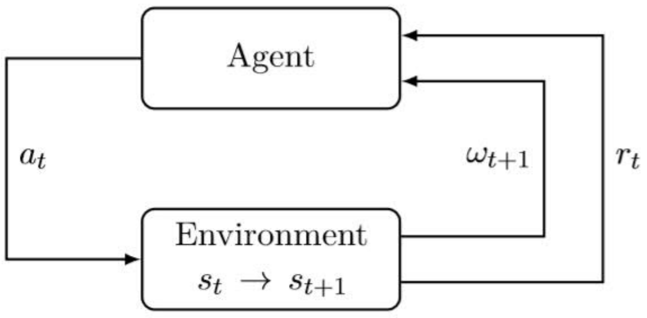


Figure 3-5: Agent-environment interaction in RL. [69]

Definition 3.1. A discrete time stochastic control process is Markovian (i.e., it has the Markov property) if

- $\mathbb{p}(\omega_{t+1}|\omega_t, a_t) = \mathbb{p}(\omega_{t+1}|\omega_t, a_t, \ldots, , \omega_0, a_0)$, and
- $\mathbb{p}(r\_t \ / \omega\_t, a\_t\ ) = \mathbb{p}(r\_t\ |\omega_t, a_t, \ldots, , \omega_0, a_0.$

The Markov property means that the future of the process only depends on the current observation, and the agent has no interest in looking at the full history. A Markov Decision Process (MDP) [68] is a discrete time stochastic control process defined as follows:

Definition 3.2. An MDP is a 5-tuple *($\mathcal{S}$, $\mathcal{A}$,T,R,γ)* where:

- $\mathcal{S}$ is the state space,
- $\mathcal{A}$ is the action space,
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to$ [0,1] is the transition function (set of conditional transition probabilities between states),
- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}$ is the reward function, where $\mathcal{R}$ is a continuous set of possible rewards in a range $R_{max} \in \mathbb{R}^+$.g., [0,R_{max}
- $\gamma \in$ [0,1) is the discount factor.

The system is fully observable in an MDP, which means that the observation is the same as the state of the environment: $\omega_t = s_t$. At each time step *t*, the probability of moving to s_{t+1} is given by the state transition function $T(s_t, a_t, s_{t+1})$ and the reward is given by a bounded reward function $R(s_t, a_t, s_{t+1}) \in \mathcal{R}$. This is illustrated in Figure 3-6.

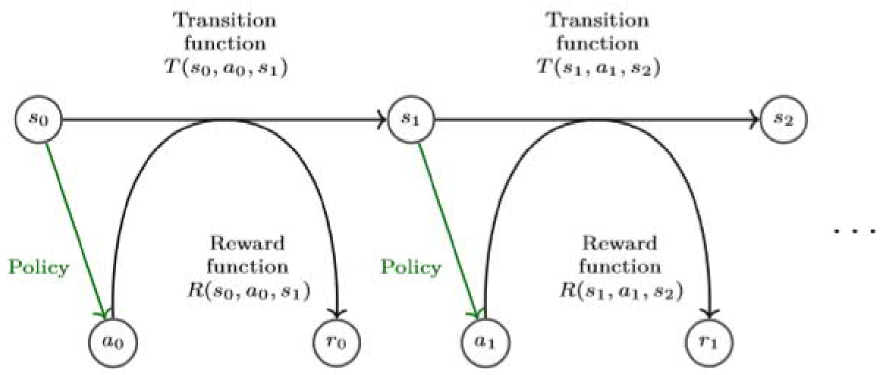

Transition function $T(s_0, a_0, s_1)$
Transition function $T(s_1, a_1, s_2)$
s_0
s_1
s_2
Policy
Reward function $R(s_0, a_0, s_1)$
Policy
Reward function $R(s_1, a_1, s_2)$
a_0
r_0
a_1
r_1
. . .

Figure 3-6: Illustration of a MDP. At each step, the agent takes an action that changes its state in the environment and provides a reward. [69]

Different categories of policies

A policy defines how an agent selects actions. Policies can be categorized under the criterion of being either stationary or non-stationary. A nonstationary policy depends on the time-step and is useful for the finite horizon context where the cumulative rewards that the agent seeks to optimize are limited to a finite number of future time steps [70]. In this introduction to deep RL, infinite horizons are considered and the policies are stationary.

Policies can also be categorized under a second criterion of being either deterministic or stochastic:

- In the deterministic case, the policy is described by $\pi(s) : \mathcal{S} \to \mathcal{A}$.
- In the stochastic case, the policy is described by $\pi(s, a) : \mathcal{S} \times \mathcal{A} \to [0,1]$ where $\pi(s, a)$ denotes the probability that action *a* may be chosen in state *s*.

The expected return

Throughout this survey, we consider the case of an RL agent whose goal is to find a policy $\pi(s, a) = \Pi$, so as to optimize an expected return $V^{\Pi}(s): \mathcal{S} \to \mathbb{R}$ (also called V-value function) such that

$$V^{\Pi}(s, a): \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, \pi\right],$$

where:

- $r_t = \mathbb{E}_{a \sim \pi(s_t, \cdot)} R(s_t, a_t, s_{t+1}),$
- $\mathbb{p}(s_{t+1} | s_t, a_t) = T(s_t, a_t, s_{t+1})$ with $a \sim \pi(s_t, \cdot),$

From the definition of the expected return, the optimal expected return can be defined as:

$$V^*(s) = max_{\pi \in \Pi} V^{\Pi}(s).$$

In addition to the V-value function, a few other functions of interest can be introduced. The Q-value function $Q^{\pi}(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as follows:

$$Q^{\Pi}(s, a): \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a, \pi\right].$$

This equation can be rewritten recursively in the case of an MDP using Bellman's equation:

$$Q^{\Pi}(s,a): \sum_{s^{'} \in S} T\left(s,a,s^{'}\right)\left(R\left(s,a,s^{'}\right) + \gamma Q^{\pi}\left(s^{'},a = \pi\left(s^{'}\right)\right)\right).$$

Similarly to the V-value function, the optimal Q-value function Q∗(s,a) can also be defined as

$$Q^{*}(s,a) = max_{\pi \in \Pi} Q^{\Pi}(s,a).$$

The optimal V-value function $V^{*}(s)$ is the expected discounted reward when in a given state s while following the policy π^{*} thereafter. The optimal Q-value $Q^{*}(s,a)$ is the expected discounted return when in a given state s and for a given action a while following the policy π^{*} thereafter.

It is also possible to define the advantage function

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s).$$

This quantity describes how good the action a is, as compared to the expected return when following directly policy π.

Note that one straightforward way to obtain estimates of either $V^{\pi}(s)$, $Q^{\pi}(s,a)$ or $A^{\pi}(s,a)$ is to use Monte Carlo methods, i.e. defining an estimate by performing several simulations from s while following policy π.In practice, we will see that this may not be possible in the case of limited data. In addition, even when it is possible, we will see that other methods should usually be preferred for computational efficiency.

3.1.2.2 Different components to learn a policy

An RL agent includes one or more of the following components:

- a representation of a $value\ function$ that provides a prediction of how good each state or each state/action pair is,
- a direct representation of the $policy$ $\pi(s)$ or $\pi(s,a)$, or
- a $model$ of the environment (the estimated transition function and the estimated reward function) in conjunction with a planning algorithm.

The first two components are related to what is called model-free RL. When the latter component is used, the algorithm is referred to as model based RL. A combination of both and why using the combination can be useful. A schema with all possible approaches is provided in Figure 3-7.

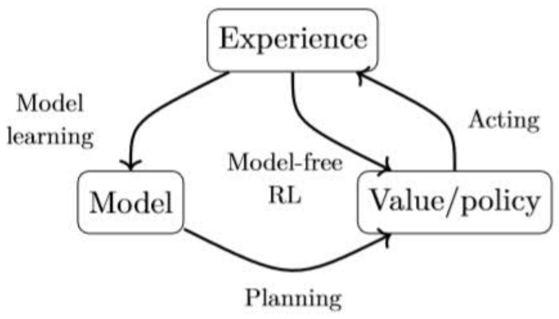


Figure 3-7: General schema of the different methods for RL. The direct approach uses a representation of either a value function or a policy to act in the environment. The indirect approach makes use of a model of the environment. [69]

For most problems approaching real-world complexity, the state space is high-dimensional (and possibly continuous). In order to learn an estimate of the model, the value function or the policy, there are two main advantages for RL algorithms to rely on deep learning:

- Neural networks are well suited for dealing with high-dimensional sensory inputs (such as times series, frames, etc.) and, in practice, they do not require an exponential increase of data when adding extra dimensions to the state or action space.
- In addition, they can be trained incrementally and make use of additional samples obtained as learning happens.

3.1.2.3 Different settings to learn a policy from data

We now describe key settings that can be tackled with RL.

Offline and online learning

Learning a sequential decision-making task appears in two cases: (i) in the offline learning case where only limited data on a given environment is available and (ii) in an online learning case where, in parallel to learning, the agent gradually gathers experience in the environment. In both cases, the core learning algorithms are essentially the same. The specificity of the batch setting is that the agent has to learn from limited data without the possibility of interacting further with the environment. In the online setting, the learning problem is more intricate and learning without requiring a large amount of data (sample efficiency) is not only influenced by the capability of the learning algorithm to generalize well from the limited experience. Indeed, the agent has the possibility to gather experience via an exploration/exploitation strategy. In addition, it can use a replay memory to store its experience so that it can be reprocessed at a later time. In both the batch and the online settings, a supplementary consideration is also the computational efficiency, which, among other things, depends on the efficiency of a given gradient descent step. All these elements will be introduced with more details in the following chapters. A general schema of the different elements that can be found in most deep RL algorithms is provided in Figure 3-8.

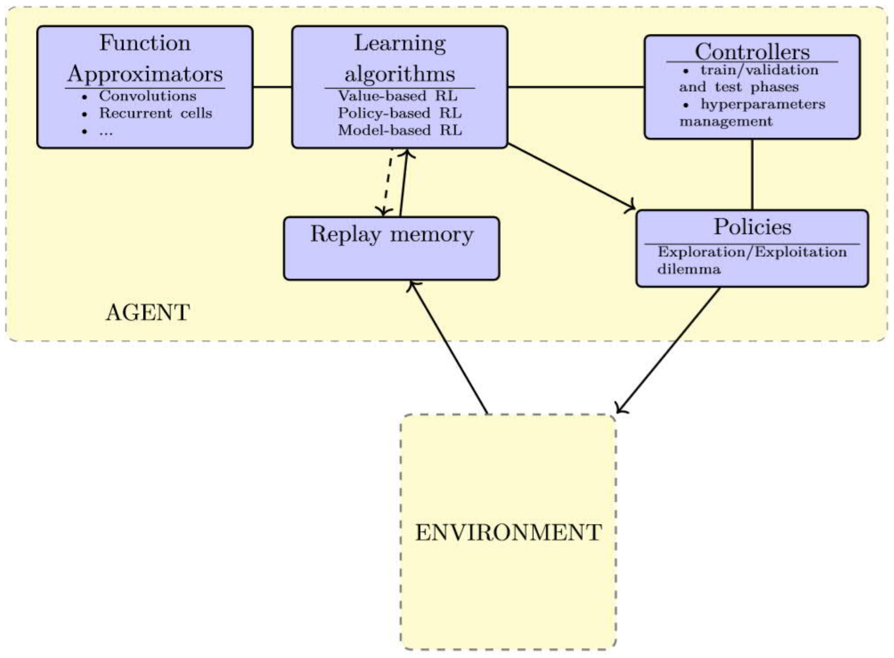


Figure 3-8: General schema of deep RL methods [69]

Off-policy and on-policy learning

According to Sutton and Barto, 2017 [66], 《on-policy methods attempt to evaluate or improve the policy that is used to make decisions, whereas off-policy methods evaluate or improve a policy different from that used to generate the data》. In off-policy based methods, learning is straightforward when using trajectories that are not necessarily obtained under the current policy, but from a different behavior policy $\beta(s,a)$. In those cases, experience replay allows reusing samples from a different behavior policy. On the contrary, on-policy based methods usually introduce a bias when used with a replay buffer as the trajectories are usually not obtained solely under the current policy π. As will be discussed in the following chapters, this makes off-policy methods sample efficient as they are able to make use of any experience; in contrast, on-policy methods would, if specific care is not taken, introduce a bias when using off-policy trajectories.

3.1.3 *Super-Resolution*

According to the image priors, single-image super resolution algorithms can be categorized into four types – prediction models, edge-based methods, image statistical methods and patch based (or example-based) methods. These methods have been thoroughly investigated and evaluated in Yang *et al.*'s work [71]. Among them, the example-based methods [72], [73], [74], [75] achieve the state-of-the-art performance.

The internal example-based methods exploit the self-similarity property and generate exemplar patches from the input image. It is first proposed in Glasner's work [75], and several improved variants [76], [77] are proposed to accelerate the implementation. The external example-based methods [74], [78], [79], [80], [81], [82], [83], [84], [85], [86] learn a mapping between low/high resolution patches from external datasets. These studies vary on how to learn a compact dictionary or manifold space to relate low/high-resolution patches, and on how representation schemes can be conducted in such spaces. In the pioneer work of Freeman *et al*. [87], the dictionaries are directly presented as low/high-resolution patch pairs, and the nearest neighbour (NN) of the input patch is found in the low-resolution space, with its corresponding high-resolution patch used for reconstruction. Chang *et al.* [79] introduce a manifold embedding technique as an alternative to the NN strategy. In Yang *et al.*'s work [84], [85], the above NN correspondence advances to a more sophisticated sparse coding formulation. Other mapping functions such as kernel regression [73], simple function [75], random forest [82] and anchored neighbourhood regression [74], [88] are proposed to further improve the mapping accuracy and speed. The sparse coding-based method and its several improvements [74], [83], [88] are among the state-of-the-art SR methods nowadays. In these methods, the patches are the focus of the optimization; the patch extraction and aggregation steps are considered as pre/post-processing and handled separately.

The majority of SR algorithms [74], [78], [79], [81], [83], [85], [86] focus on gray-scale or single-channel image super-resolution. For colour images, the afore mentioned methods first transform the problem to a different colour space (YCbCr or YUV), and SR is applied only on the luminance channel. There are also works attempting to super-resolve all channels simultaneously. For example, Kim and Kwon [73] and Dai *et al.* [89] apply their model to each RGB channel and combined them to produce the final results. However, none of them has analyzed the SR performance of different channels, and the necessity of recovering all three channels.

3.1.4 Lyapunov Optimization Framework

In this section, the Lyapunov optimization theory which aims at time-average penalty function minimization subject to queue stability. Notice that the time-average penalty function minimization can be equivalently converted to time-average utility function maximization [10].

The Lyapunov optimization theory can be used when the trade-off exists between utility and stability. For example, it can be obvious seen that the trade-off exists when current decision-making is optimal in terms of the minimization of penalty function whereas the operation of the decision takes a lot of time, i.e., thus it introduces delays (i.e., queue-backlog increases in the system). Then, the optimal decision can be dynamically time-varying because focusing on utility maximization (i.e., penalty function minimization) is better when the delay in the current system is not serious (i.e., queueing delay is small or marginal). On the other hand, the optimal decision will be for the delay reduction when the delay in the current system is large. In this case, the decision should be for delay reduction while sacrificing certain amounts of utility maximization (or penalty function minimization) [10].

Suppose that our time-average penalty function is denoted by $P(\alpha[t])$ and it should be minimized, and our control action decision-making is denoted by $\alpha[t]$. Then, the queue dynamics in the system, i.e., $Q[t]$, can be formulated as follows:

$Q[t+1] = \max\{Q[t] + a(\alpha[t]) - b(\alpha[t]), 0\}$ and $Q[0] = 0$

where $a(\alpha[t])$ is an arrival process at $Q[t]$ at t when our control action decision-making is $\alpha[t]$. In this equation, $b(\alpha[t])$ is a departure/service process at $Q[t]$ when our control action decision-making is $\alpha[t]$ at t.

In this section, control action decision-making should be made in each unit time for time-average penalty function minimization subject to queue stability. Then, the mathematical program for minimizing time-average penalty function, $P(\alpha[t])$ where the control action decision-making at t is $\alpha[t]$ can be presented as follows:

$$min\colon \lim_{\{t\to\infty\}} \sum_{\{\tau=0\}}^{\{t-1\}} P(\alpha[\tau])$$

subject to queue stability:

$$\lim_{\{t\to\infty\}} \sum_{\{\tau=0\}}^{\{t-1\}} Q[\tau] < \infty$$

In this equation, $P(\alpha[t])$ stands for the penalty function when a control action decision-making is $(\alpha[t])$ at t. As mentioned, the Lyapunov optimization theory can be used when trade-off between utility maximization (or penalty function minimization) and delays. Based on this nature, drift-plus-penalty (DPP) algorithm is designed for maximizing the time-average utility subject to queue stability. Here, the Lyapunov function is defined as $L(Q[t]) = \frac{1}{2}(Q[t])^2$, and let $\Delta(.)$be a conditional quadratic Lyapunov function which is formulated as $E[\mathrm{L}(\mathrm{Q}[\mathrm{t}+1]) - \mathrm{L}(\mathrm{Q}[\mathrm{t}])|\, \mathrm{Q}[\mathrm{t}]]$, which is called as the drift on t. This dynamic policy is designed to achieve queue stability by minimizing an upper bound of our considering penalty function on DPP which is given by

$$\Delta(Q[t] + VE[P(\alpha[t])]$$

where V is a trade-off coefficient. The upper bound on the drift of the Lyapunov function at t is derived as follows:

$$L(Q[t+1]) - L(Q[t]) = \frac{1}{2}(\,Q[t+1]^2 - Q[t]^2)$$

$$\leq \frac{1}{2}(a(\alpha[t])^2 + b(\alpha[t])^2) + Q[t]\big(a(\alpha[t]) - b(\alpha[t])\big).$$

Therefore, the upper bound of the conditional Lyapunov drift can be derived as follows:

$$\Delta\big(Q(t)\big) = E[L(Q[t+1]) - L(Q[t])\,|\,Q[t]]$$

$$\leq C + E\left[\,Q[t](a(\alpha[t]) - b(\alpha[t])|Q[t]\right],$$

where C is a constant given by

$$\frac{1}{2}E\,[a(\alpha[t])^2 + b(\alpha[t])^2|\mathrm{Q}[\mathrm{t}]] \leq C$$

which supposes that the arrival and departure process rates are upper bounded. Due to the fact that C is a constant, minimizing the upper bound on DPP is as follows:

$$VE[P(\alpha[t])] + E\big[Q[t]\cdot\big(a(\alpha[t]) - b(\alpha[t])\big)\big].$$

Finally, the dynamic control action decision-making $\alpha[t]$ in each unit time t for time-average penalty function $P(\alpha[t])$ minimization subject to queue stability can be formulated as follows based on the Lyapunov optimization theory.

$$\alpha^*[t+1] \leftarrow \arg\min_{\{\alpha[t]\in A\}}\big[V\cdot P(\alpha[t]) + \;Q[t]\cdot\big(a(\alpha[t]) - b(\alpha[t])\big)\big]$$

where A is the set of all possible control actions and $\alpha^*[t+1]$ is the optimal control action decision-making for next time slot.

Algorithm 1 Stabilized Time-Average Penalty Function Minimization

```
Initialize:
 1: $t \leftarrow 0$;
 2: $Q[t] \leftarrow 0$;
 3: Decision Action: $\forall \alpha[t] \in \mathcal{A}$
Time-Average Penalty Function Minimization subject to Stability
 4: while $t \leq T$ do // $T$: operation time
 5:     Observe $Q[t]$;
 6:     $\mathcal{T}^* \leftarrow \infty$;
 7:     for $\alpha[t] \in \mathcal{A}$ do
 8:         $\mathcal{T} \leftarrow V \cdot P(\alpha[t]) + Q[t] \cdot (a(\alpha[t]) - b(\alpha[t]))$;
 9:         if $\mathcal{T} \leq \mathcal{T}^*$ then
10:             $\mathcal{T}^* \leftarrow \mathcal{T}$;
11:             $\alpha^*[t+1] \leftarrow \alpha[t]$;
12:         end if
13:     end for
14: end while
```

Figure 3-9: Pseudo-code for stabilized time-average penalty function minimization.

In order to verify whether the given framework works correctly or not, following two example cases can be considerable.

Case 1: Suppose $Q[t] \approx \infty$. Then,

$$\alpha^*[t+1] \leftarrow \arg \min_{\{\alpha[t] \in A\}} \left[V \cdot P(\alpha[t]) + Q[t] \cdot \big(a(\alpha[t]) - b(\alpha[t])\big)\right]$$

$$\approx \arg \min_{\{\alpha[t] \in A\}} [a(\alpha[t]) - b(\alpha[t])].$$

Then, this shows that control action decision-making should works as follows, i.e., (i) the arrival process should be minimized and (ii) the departure process should be maximized. The both cases are for stabilizing the queue, i.e., it should be beneficial when $Q[t] \approx \infty$.

Case 2: Suppose $Q[t] = 0$ Then,

$$\alpha^*[t+1] \leftarrow \arg \min_{\{\alpha[t] \in A\}} \left[V \cdot P(\alpha[t]) + Q[t] \cdot \big(a(\alpha[t]) - b(\alpha[t])\big)\right]$$

$$= \arg \min_{\{\alpha[t] \in A\}} V \cdot P(\alpha[t]).$$

Then, this shows that control action decision-making should works for minimizing the given penalty function. This is semantically reasonable because focusing on our main objective is possible because stability does not need to be considered because $Q[t] = 0$.

The pseudo-code of the proposed time-average penalty function minimization algorithm is presented in Figure 3-9. From (line 1) to (line 3), all variables and parameters are initialized. The algorithm works in each unit time as shown in (line 4). In (line 5), current queue-backlog Q[t] is observed. From (line 7) to (line 13), the main computation procedure for solving given framework is described [10].

Up to now, the time-average penalty function minimization is considered. Based on the theory, the dynamic control action decision-making $\alpha[t]$ in each unit time t for time-average utility function $U(\alpha[t])$ maximization subject to queue stability can be formulated as follows.

$$\alpha^*[t+1] \leftarrow \arg \max_{\{\alpha[t]\in A\}} \left[V \cdot U(\alpha[t]) - Q[t] \cdot \left(a(\alpha[t]) - b(\alpha[t])\right)\right]$$

where A is the set of all possible control actions and $\alpha^*[t+1]$ is the optimal control action decision-making for next time slot.

4 Use Cases of AI methods for 5G/Edge Computing Networks

4.1 Super-Resolution Control for MEC Networks

As explained, the Lyapunov optimization theory is a scalable, self-configurable, low-complexity algorithm which can be used in many applications. In this section, the use of Lyapunov optimization for deep learning and computer platforms is discussed. Finally, its related performance evaluation results are presented.

Lyapunov Control over Departure Processes

As illustrated in Figure 4-1, stabilized real-time computer vision platforms should be equipped with queues in order to handle bursty traffics. If the queue is busy or near-overflow, the departure process should be accelerated. Thus, the simplest model should be used for reducing corresponding computation. On the other hand, if the queue is empty, deep learning computation accuracy can be improved with more sophisticate models because we have enough time to conduct the computation. Thus, multiple models are desired in order to select one depending on queue backlog.

In Figure 4-1, multiple models exist and it can be seen that the simplest model (i.e., low-resolution model) is able to conduct fast computation whereas it presents low learning accuracy. On the other hand, the most sophisticate model (i.e., high-resolution model) is good for accurate learning performance whereas it introduces computation delays. Thus, the trade-off exists between performance and delays, i.e., Lyapunov optimization theory based dynamic model selection decision-making algorithm can be designed as follows [10].

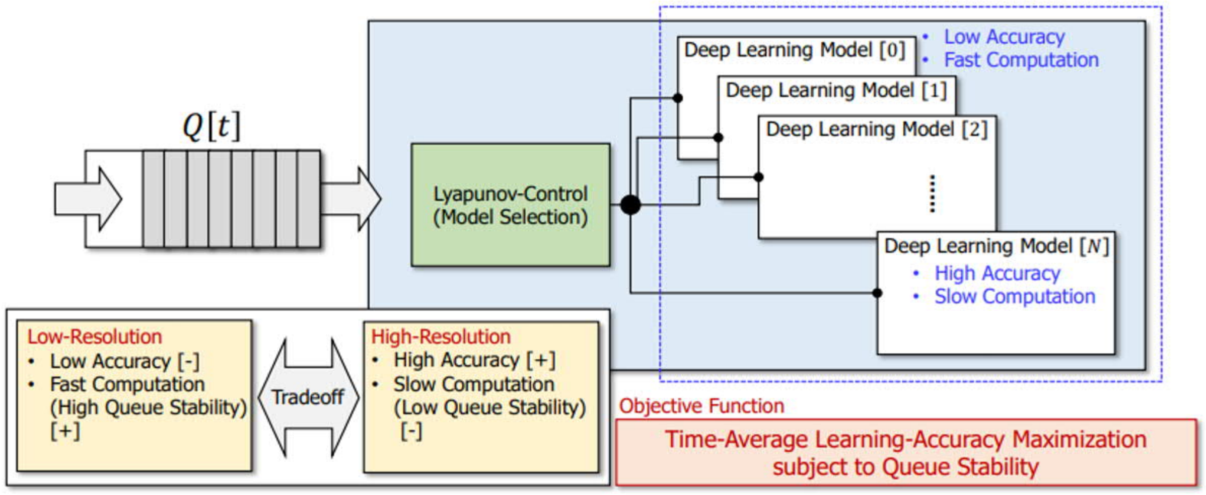


Figure 4-1: Lyapunov control over departure processes in real-time computer vision platforms for time-average learning accuracy maximization subject to queue stability.

$$\alpha^*[t+1] \leftarrow \arg\max_{\{\alpha[t]\in A\}} \left[V \cdot A(\alpha[t]) - Q[t] \cdot \left(a(\alpha[t]) - b(\alpha[t])\right)\right]$$

and this can be re-formulated as follows due to the fact that the arrival process is out of control:

$$\alpha^*[t+1] \leftarrow \arg\max_{\{\alpha[t]\in A\}} \left[V \cdot A(\alpha[t]) + Q[t] \cdot b(\alpha[t])\right]$$

where $A(\alpha[t])$ stands for the learning-accuracy when the model selection decision is $\alpha[t]$ at t. Here, A is the set of all possible deep learning models and $\alpha^*[t+1]$ is the optimal control action decision-making for next time slot.

Lyapunov Control over Arrival Processes

The stabilized real-time computer vision platform in previous section is novel and scalable, however it has burden because multiple deep learning models should be implemented in a single platform.

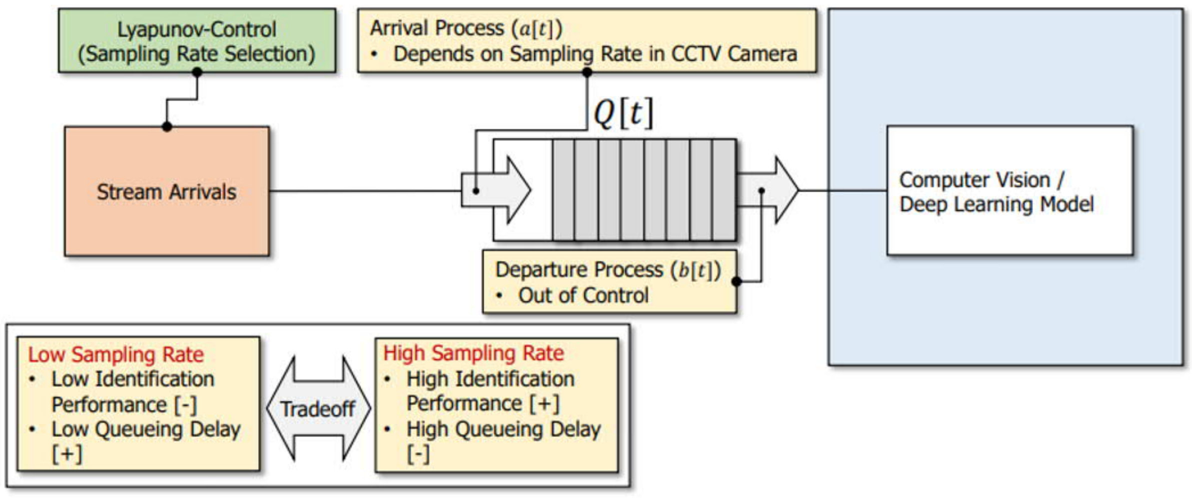

Lyapunov-Control (Sampling Rate Selection)
Arrival Process ($a[t]$)
• Depends on Sampling Rate in CCTV Camera
$Q[t]$
Stream Arrivals
Computer Vision / Deep Learning Model
Departure Process ($b[t]$)
• Out of Control
Low Sampling Rate
• Low Identification Performance [-]
• Low Queueing Delay [+]
Tradeoff
High Sampling Rate
• High Identification Performance [+]
• High Queueing Delay [-]

Figure 4-2: Lyapunov control over arrival processes in real-time computer vision platforms for time-average learning accuracy maximization subject to queue stability.

Depth (# Hidden Layers)	0	4	6	8	11	14	17	20
PSNR (dB)	30.400	32.560	33.010	33.229	33.379	33.435	33.495	33.523
SSIM	0.8682	0.9100	0.9160	0.9180	0.9200	0.9200	0.9210	0.9220
Processing Time (CPU-only)	0.0020	0.3210	0.5468	0.7725	0.9940	1.3170	1.6220	1.9600
Processing Time (CPU+GPU)	0.0010	0.0100	0.0120	0.0152	0.0189	0.0224	0.0262	0.0305

Table 4-1: Trade-off between utility and delay obtained from super-resolution performance measurement results (processing time have measured on 512 x 768 images)

Thus, a new dynamic control algorithm with a single deep learning model is also needed for resource-limited systems. As illustrated in Figure 4-2, our considering system has a single computer vision and deep learning model in computing platforms. In addition, the queue is in front of the system. Thus, the departure process is not controllable any more. In this case, the arrival process should be controllable in order to control the queue dynamics for stability. Therefore, the arrival image/video streams should be controlled by handling sample rates. If high-frequency sampling is available, more signals will be generated and then the results will be enqueued. Thus, the arrival process increases. This is beneficial because it increases computer vision performance due to the fact that more images/videos can be obtained especially in surveillance applications. On the other hand, i.e., if low-frequency sampling is conducted, the computer vision performance can be degraded whereas the number of arrival process data decreases which is beneficial in terms of stability. Eventually, the trade-off between computer vision performance and delays can be observed. Finally, Lyapunov optimization theory-based sampling rate selection decision-making algorithm can be designed as follows.

$$\alpha^*[t+1] \leftarrow \arg\max_{\{\alpha[t]\in A\}} \left[V \cdot A(\alpha[t]) - Q[t] \cdot \left(a(\alpha[t]) - b(\alpha[t])\right)\right]$$

and this can be re-formulated as follows due to the fact that the departure process is out of control:

$$\alpha^*[t+1] \leftarrow \arg\max_{\{\alpha[t]\in A\}} [V \cdot A(\alpha[t]) - Q[t] \cdot a(\alpha[t])]$$

where $A(\alpha[t])$ stands for the learning-accuracy when the sample rate selection decision is $\alpha[t]$ at t. Here, A is the set of all possible sample rates and $\alpha^*[t+1]$ is the optimal control action decision-making for next time slot.

In this section, the performance evaluation results of the proposed algorithm are presented. The data-intensive simulation-based evaluation is performed and then the results are presented in Figure 4-3. In addition, Table 4-1 shows the performance of super-resolution depending on the number of hidden layers. If the number of hidden layers is maximum (i.e., 20 in this research), the PSNR and structural similarity (SSIM), one of the widely used performance metrics in super-resolution, values are maximum [10].

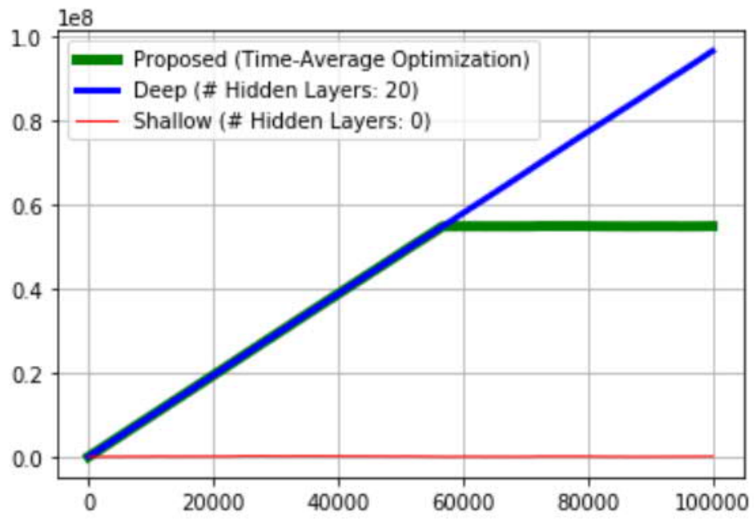


Figure 4-3: Performance Evaluation: Queue-Backlog (x-axis: unit time, y-axis: queue-occupancy (unit: bits)

However, the computation times (for CPU-only and CPU-GPU) become slow. As illustrated in Figure 4-3, if the models are static (i.e., Deep or Shallow), the curves show that the two models are not efficient. The deep model cannot handle the overflow situations, thus, the queue diverges. On the other hand, the shallow model is too fast, thus, the queue is always empty. This is obviously positive for stability where the performance in terms of super-resolution performance is the lowest. Thus, it might be better if the algorithm allows certain amounts of delays in order to enhance the quality of super-resolution. The proposed algorithm is initially following Deep model because the queue is idle during the initial phases. If the queue becomes filled with certain amounts of images (i.e., near threshold), it starts the control, i.e., self-adaptive, near the unit time of 5, 800. Thus, the proposed algorithm starts to select super resolution models which can handle delays. Thus, it is true that the proposed algorithm is better than the other two static algorithms. For the proposed self-adaptive stabilized algorithm, the evaluation with two processing capabilities (CPU-only platform vs. CPU-GPU platform), it can be observed that the CPU-GPU platform selects the maximum-performance super-resolution model (i.e., 20 hidden layers in Table 1) 4.36 times more than the CPU-only platform. It means that the proposed algorithm is self-adaptive depending on the hardware/platform requirements. This is obviously beneficial in terms of system engineers because they do not need to conduct trial-and error-based system parameter tuning any more.

4.2 Deep Reinforcement Learning based Video Caching for Vehicular Networks

Within few years, it has been expected that tens of exabytes of global data traffic be handled on daily basis, and on-demand video streaming will account for about 70% of them. In on-demand video streaming services, a relatively small number of popular contents is requested at ultra-high rates and playback delay is one of the most important measurement criteria of goodness. To deal with the characteristics, wireless caching technologies have been studied for video streaming services by storing popular videos in caching helpers located nearby users during off-peak time. Therefore, it is obvious that storing and streaming of video files are of major research interests in wireless caching networks.

There have been major research results for caching popular files in stochastic wireless caching networks. The major goal of those research results was to design the optimal caching policies according to the popularity distribution of contents and wireless network topology. The probabilistic caching policy was proposed in to adapt characteristics of the stochastic network. Many probabilistic caching methods have been proposed depending on various optimization goals, e.g., maximization of cache hit probability, average success probability of content delivery, and delivery rate of multiple content requests. However, these works on the caching policy do not consider the identical content with different qualities.

Since video files can be encoded to multiple versions which differ in the quality levels, the video caching policies having different quality levels have been widely studied. Many researchers have proposed the static content placement policies under the consideration of differentiated quality requests for the same content, given probabilistic quality requests or minimum quality requirements. The above works are focused only on the content placement problem with different qualities, however, the delivery policy of contents with different qualities has not yet been studied much.

For video delivery/streaming, there are some necessary decisions to be made: 1) which caching node will deliver the video, 2) which quality of video will be provided, and 3) how many video chunks will be transmitted. The first one is called node association problem, and in most research contributions that do not consider different quality levels for the same file, the file-requesting user is allowed to receive the desired video from the caching node under the strongest channel condition. The node associations for video delivery in heterogeneous caching networks have been studied. On the other hand, when videos with different qualities are independently cached, more elaborate node association algorithm is necessary, because decision on the content quality relies on the node association. In this case, the video delivery policy was proposed in to pursue time-average video quality maximization while avoiding playback delays.

Since dynamic video streaming allows each chunk to have a different quality depending on time-varying network conditions, some researchers addressed transmission schemes which serve the video by dynamically selecting the quality level. The scheduling policies which maximize the network utility function of time-averaged video quality in small-cell networks and device-to-device networks were proposed. Some authors considered scalable video coding (SVC) and proposed dynamic resource allocation and quality selection under the pricing strategy for interference. While the video delivery policies are operated at the base station (BS) side, the decision policy of video quality level at user sides was not considered. This scenario is consistent with the practical real-world software implementation of dynamic adaptive streaming over HTTP (DASH), in which users dynamically choose the most appropriate video quality. Even though it can choose the video quality at the user side, however it cannot dynamically change the video quality without updates of node association.

Further, control of the amount of receiving chunks depending on stochastic network states has been largely neglected in above existing research about video delivery. Even though the authors of other papers maximize the long-term time average video quality under the various constraints, their metrics representing video quality is obtained by averaging the number of quality selections at each time slot. This method would be not enough to evaluate the user's quality of service, especially when the transmission rate varies over the video streaming service time. In practice, when channel experiences deep fading and only the low-quality video is available, it would not be the best choice to receive as many chunks as the channel condition can provide. Rather than receiving many low-quality chunks, the user could prefer to wait channel conditions to be better and then to receive high quality chunks, if it guarantees no playback delay. Therefore, by considering decision process of combinations of video quality and chunk amounts, we can formulate the optimization problem which maximizes the average video quality per each received chunk. There has been not many research yet considering the average video quality per chunk as a performance metric as well as aiming at adjustments of receiving chunk amounts depending on caching node and user states.

This report addresses dynamic video delivery in two different timescales for wireless caching networks with consideration of user mobility. There are some existing works of mobility-aware caching in which user mobility is known at the centralized server or randomness of user mobility is probabilistically modelled. Meanwhile, content delivery in wireless caching networks with moving users has not been explored yet. When the delivery decisions are made at the user side, it is reasonable to utilize the knowledge of user mobility for dynamic content delivery. In addition, joint optimization of long-term content caching and short-term delivery was studied; however, content delivery decisions in different timescales has not been investigated.

This report proposes a video delivery policy in the wireless caching network for dynamic streaming services. The main contributions are as follows [11]:

- This report proposes a dynamic video delivery policy depending on stochastic network states. The proposed policy makes three different but necessary decisions for the streaming user: 1) caching node for video delivery, 2) video quality and 3) the quantity of video chunks to receive. To the best of the authors' knowledge, no research has yet considered all of those video delivery decisions.

- Caching node association and decisions of video quality and amounts of receiving chunks are conducted in different timescales. Since wireless link activation for video delivery is time-consuming, it is reasonable that caching node association is performed slower than decisions of video quality and amounts of receiving chunks. The optimization framework of video delivery policy in two different timescales is constructed based on frame based Lyapunov optimization theory and Markov decision process (MDP).

- The proposed content delivery method reflects user mobility. Therefore, the user enables to associate with the caching node based on estimation of the short-term decisions on quality and receiving chunk amounts in future. Note that the knowledge of user mobility can be obtained because delivery decisions are made at the user side.

- The proposed technique maximizes the average streaming quality while averting playback latency, and can control the trade-off between video quality and playback delay. Different from last studies, we adopt the long-term average video quality per each received chunk as a performance metric.

- We perform simulations to verify the proposed video delivery policy and to show the advantages of using Lyapunov optimization theory and MDP.

4.2.1 Network model

The wireless caching network model is described in this section. In addition, the user queue model and channel model are introduced and the dependency of the necessary video delivery decisions on the user queue and channel models is explained. The inter-user interference is roughly included by using distance-based interference management [11].

Wireless Caching Network Model

This report considers wireless caching network model where a user requests certain video file for one of caching nodes around the user, as shown in Figure 4-4. The BS has already pushed popular video files during off-peak hours to caching nodes which have the finite storage size. Since we focus on video delivery, the caching policy is out of scope for this paper and only the desired video is considered. Suppose that the desired video has L quality levels. Therefore, there are L types of caching nodes, and the type-l caching nodes can deliver the video of any quality $q \in \mathcal{L}_l$, where $\mathcal{L}_l = \{1,\cdots, l\}$ is the set of qualities which the type-l caching node can provide. Thus, the type-L caching nodes can provide all quality levels from the quality set $\mathcal{L}_L$. Note that simple definition of $\mathcal{L}_l = \{1,\cdots, l\}$ is assumed, but the proposed technique can be coordinated with any arbitrary quality set as long as multiple versions of the same video having different qualities are stored in caching nodes.

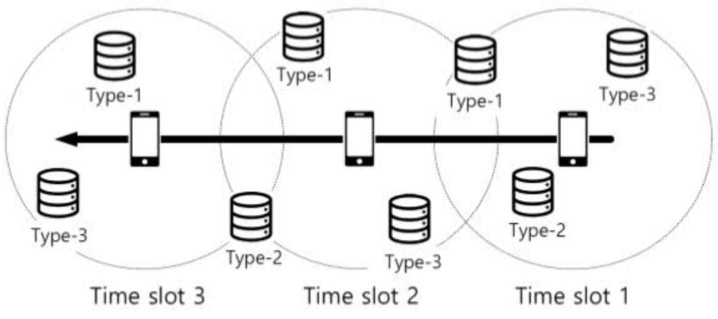


Figure 4-4: Network model

The identical files of different qualities are stored in multiple caching nodes, and the type-l caching nodes are distributed by the independent Poisson Point Processes (PPPs) with intensity λ_{p_q}, where pq indicates the caching probability of the requested video encoded to provide any quality $q \in \{1,\cdots, l\}$. Suppose that the caching policy is already determined, i.e., all pq for all q are given. In addition, videos of different qualities have different file sizes and N_q denotes the file size of quality q in bits, satisfying $N_m < N_q$ for all $m, q \in t_k$ and $m < q$.

User mobility is also captured in network model. The user is moving towards certain direction and periodically searches for a caching node to receive the desired video file. As shown in Figure 4-4, geological distribution of caching nodes around the user varies at each time slot, so the caching node decision should be appropriately updated. Further, this paper also considers how many chunks of which quality level to be requested from the user depending on the stochastic network environment. When there are other users who exploit the wireless caching network with the same resource, the target streaming user is interfering with them. We adopt the distance-based interference management to limit the interference power lower than certain threshold [11].

4.2.2 Dynamic Video Delivery Policies

The dynamic video delivery policy in two different timescales is described in this section. When the data rate of the delivery link not in outage is varying, i.e., the number of receiving video chunks at each discrete slot is not fixed, video quality per received chunk is the reasonable performance metric. Therefore, the problem that maximizes quality per received chunk with the constraints for playback delay and channel capacity is formulated.

Video Delivery Decisions

The goal of this paper is to find the appropriate three decisions at each slot t in the network model: 1) caching node for video delivery $\alpha(t)$, 2) video quality level $q(t)$, and 3) the quantity of receiving chunks $M(t)$. However, to update the caching node association, the time-consuming process is required in which the user sends the request signal for video delivery and the caching node approves it. Therefore, new caching node association is hardly performed as frequent as receiving chunks, and we suppose that the decision on $\alpha(t)$ is made at larger timescale than decisions on $q(t)$ and $M(t)$.

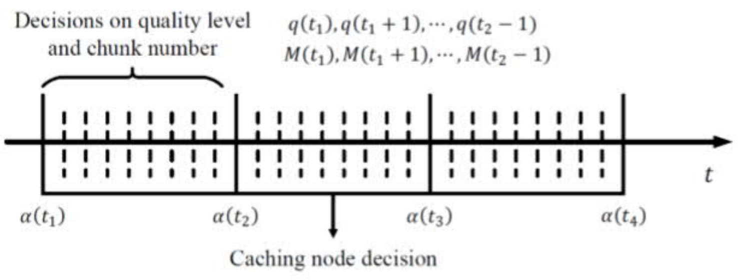

Decisions on quality level and chunk number
$q(t_1), q(t_1+1), \cdots, q(t_2-1)$
$M(t_1), M(t_1+1), \cdots, M(t_2-1)$
t
$\alpha(t_1)$
$\alpha(t_2)$
$\alpha(t_3)$
$\alpha(t_4)$
Caching node decision

Figure 4-5: Different timescales for decisions on $\alpha(t), q(t)$ *and* $M(t)$.

In this sense, the user decides $q(t)$ and $M(t)$ at time slots $t \in \{0,1,2 \cdots\}$, but caching node decisions are performed attime slots $t = 0, T, 2T, \cdots$, where$T$ is the time interval for caching node association. The time slot for the k-th caching node decision is denoted by $t_k = (k-1)T$ for $k \in \{1,2,\cdots\}$. Different timescales of decisions on $\alpha(t), q(t)$ *and* $M(t)$ are described in Figure 4-5. Let the k-th frame for caching node decision be $T_k = \{t_k, t_k + 1, \cdots, t_k + T - 1\}$. As shown in Figure 4-5, after associating with the caching node $\alpha(t_k)$ at time t_k, decisions on quality level $q(t)$ and chunk amounts $M(t)$ are performed over $t \in T_k$ to receive the desired video from $\alpha(t_k)$. Therefore, $q(t) \in \mathcal{L}_l(\alpha(t_k))$ and $M(t)\, N_{q(t)} \leq B(\alpha(t_k), t)$should be satisfied for $t \in T_k$, wherel($\alpha(t_k)$) is the type of the caching node $\alpha(t_k)$.

The user can make the candidate set of caching nodes denoted by $\mathcal{A}(t_k)$, and $\alpha(t_k) \in \mathcal{A}(t_k)$. All caching nodes in $\mathcal{A}(t_k)$. should be outside the radius R_U of all existing users to limit the interference power lower than ρ. To avoid the situation in which no caching node can deliver the desired video, i.e., $\mathcal{A}(t_k)$. > 0, the cachingnodeswhich provideSINRs larger than γmin are assumed to be outside the radius R_U of all existing users. $\mathcal{A}(t_k)$. consists of up to L caching nodes, i.e., $|\mathcal{A}(t_k)| \leq L$, in which each caching node in $\mathcal{A}(t_k)$ belongs to different types. If there are several nodes of type-l, the user takes one of them whose channel condition is the strongest. There is no reason to choose another type-l caching node while leaving the node with the strongest channel if another streaming user does not request the video from that strongest node. In addition, $|\mathcal{A}(t_k)| < L$ means that $L - |\mathcal{A}(t_k)|$caching node types do not exist around the user. Suppose that the new streaming user is already generated outside the radius R_N of all existing caching nodes and the INR Υ is observed at the new user. Also, another user's link activation is banned around the target user due to the interference issue. Then, we just consider the node association problem of the new streaming user with respect to the candidate set $\mathcal{A}(t_k)$ while the INR Υ is observed.

```
Algorithm 1 Dynamic Video Delivery Decisions on $\alpha$, $q$, and $M$ in Different Timescales
Precondition:
 1:  • $V$: parameter for streaming quality-delay trade-offs
     • $\tilde{Q}$: threshold for queue backlog size
     • $K$: the number of caching node decisions
     • $T$: the time interval of updating caching node decision
 2: $t = 0$ // $KT - 1$: number of discrete-time operations
 3: while $k \leq K$ do
 4:     $t_k = (k-1)T$: time for the $k$-th caching node decision
 5:     Observe $S(t_k)$ and find $\mathcal{A}(t_k)$
 6:     Compute $\mathcal{D}_k(\alpha(t_k), Q(t_k))$ by using dynamic programming equation (35) and store $\Theta_k^*(t, s, b)$ for every $\alpha(t_k) \in \mathcal{A}(t_k)$, $s \in \mathcal{S}$ and $b \in \mathcal{B}$.
 7:     Make a decision of $\alpha^*(t_k)$ by using (22)
 8:     for $t = t_k : t_k + T - 1$ do
 9:         Observe $S(t)$ and $B_k(t)$
10:         Make a decision of $\Theta_k^*(t, S(t), B_k(t))$
11:     end for
12: end while
```

4.2.3 Simulation Results

No. of quality levels (L)	3
Default PPP intensity (λ)	0.4
Time interval of caching node decisions (T)	5
Caching probabilities ($\mathbf{p} = [p_1, \cdots, p_L]$)	$[\frac{4}{7}, \frac{2}{7}, \frac{1}{7}]$
Transmit SNR (Ψ)	25 dB
INR (Υ)	5 dB
Path loss exponent (β)	2
Shadowing variance (σ_w^2)	4 dB
Minimum probability of finding caching node ($\eta_{\min}$)	0.99
Queue departure (c)	1
User radius (R)	50 m
Bandwidth ($\mathcal{W}$)	1 MHz
Coherence time (t_c)	5 ms
End cost coefficient (A)	10^4
End cost coefficient (μ)	1
$\tilde{Q}$	100
$B_{\max}$	52 kbits
V	0.015

Table 4-2: System parameters

In this section, we show that the proposed algorithm for dynamic video delivery works well with video files of different quality levels in wireless caching network with user mobility. Simulation parameters are listed in Table 4-2, and these are used unless otherwise noted. The proposed technique can be applied to any distribution model for caching nodes, but we suppose that caching nodes which store the desired content are modelled as an independent PPP with the intensity of λ, which is generally assumed for researches of wireless caching networks. Then, the PPP intensity of type-l caching nodes becomes λ_{p_l}, where p_l denote the caching probability of the video which can be encoded into any quality in $\mathcal{L}_l$. Therefore, larger p_l, more caching nodes of type-l around the streaming user. Based on the network model described in Figure 4-5, the user is slowly moving towards certain direction. In practice, the channel condition between the user and the caching node delivering the desired video could be varying due to Doppler shift as the user is moving, but this effect is not captured in this report Peak-signal-to-noise ratio (PSNR) is considered as a video quality measure, and quality measures and file sizes depending on quality levels are supposed as $\mathcal{P}(q)$ = [34, 36.64, 39.11] dB and N(q)=[2621, 5073, 10658] Kbits which are obtained from real-world video traces. Since we assume that η_{min} = 0.99 and $t_0 \mathcal{W} \log_2(1 + \gamma_{min}) = N(1)$, the minimum intensity of PPP distributions to satisfy the performance criterion of $\{\gamma_{min}, \eta_{min}\}$ becomes λ_{min} = 0.0352. Therefore, even for the least distributed chunks of the highest quality, i.e., q =3, the intensity of caching node distribution is sufficiently large, i.e., λ_{p_3} = 0.573 > λ_{min}.

performance comparisons with ‘Strongest’ and ‘Highest-Quality’ can show the effects of caching node decision based on Lyapunov optimization. Also, comparison with [11] (M. Choi et al., “Wireless video caching and dynamic streaming under differentiated quality requirements,” IEEE J. Sel. Areas Commun., vol. 36, no. 6, pp. 1245–1257, Jun. 2018.) can show that the proposed scheme is appropriate for the practical performance metric (i.e., average video quality per played chunk) and specify the advantage of using MDP and dynamic programming for decisions on video quality and the amounts of receiving chunks.

4.2.3.1 Caching Node Distribution

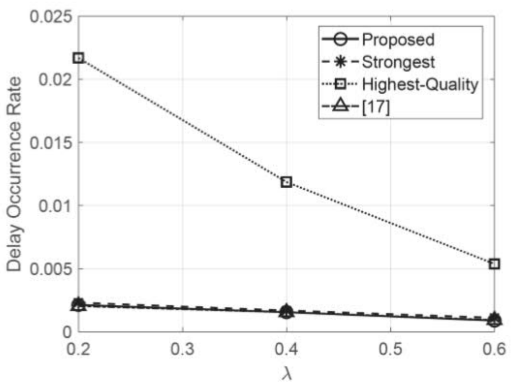


Figure 4-6: Delay occurrence rates over λ.

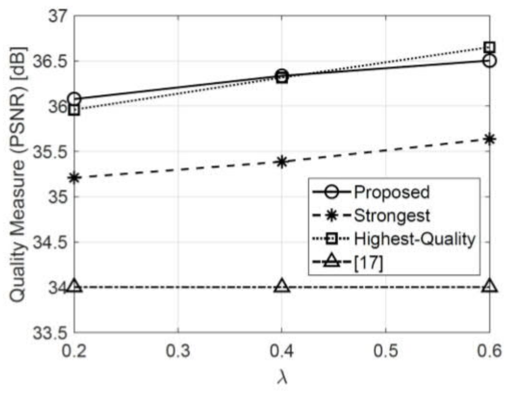


Figure 4-7: Video quality measures over λ.

At first, impacts of the PPP intensity, i.e. how many caching nodes are distributed around the streaming user, are shown in Figure 4-6 and Figure 4-7, which give the plots of playback delay occurrence rates and average video quality measures per received chunk versus λ, respectively. As λ grows, there becomes an increase chance that the user can find the caching node around itself that can deliver many high-quality chunks; therefore, both delay and quality performances of all schemes become improved.

Since the caching node whose channel condition is the strongest can deliver the largest number of chunks to the user, ‘Strongest’ provides much lower delay occurrence rates than ‘Highest-Quality’ in Figure 4-6. However, ‘Strongest’ does not consider the video quality when associating with the caching node; therefore, its quality measure is worse than ‘Highest Quality’. On the other hand, ‘Highest-Quality’ can provide the higher quality measure to the user compared to ‘Strongest’, however, the user can suffer from frequent playback delays especially when λ is small.

Meanwhile, the proposed technique determines to associate with the caching node by balancing the video quality and channel condition; therefore, the proposed one can provide quite high quality as ‘Highest-Quality’ while limiting playback delays as ‘Strongest’ and the scheme in [11], as shown in Figure 4-6 and Figure 4-7, respectively. Thus, the proposed scheme can be said to smooth out the tradeoff between quality and playback delay and to fairly achieve both goals. Also, note that ‘Strongest’ and ‘Highest-Quality’ determines the caching node depending on their purposes, i.e., the channel condition and quality of cached contents, without consideration of user mobility. Even though the channel condition of the caching node chosen by ‘Strongest’ is the strongest at tk, after that it could not be the strongest due to time-varying channels and user mobility. Therefore, ‘Strongest’ is not the best choice for reducing playback delays, and the proposed one and the work of [11] can show almost the same delay performance as ‘Strongest’. Similarly, with the knowledge of user mobility and estimated the future decisions on quality and receiving chunk amounts, the proposed scheme provides as high quality measure as ‘Highest-Quality’.

The scheme proposed in [11] aims at limiting the playback delay using the same constraint of queue stability based on Lyapunov optimization; therefore, its delay performance is as good as ‘Strongest’ and the proposed one. However, the average quality measure per received chunk maintains the lowest value. Since maximization of the average of quality levels chosen at every slot in [11] treats receiving one chunk and multiple chunks of the same quality as the identical case, the actual playback quality could be significantly degraded when the user receives very many low-quality chunks to avoid the playback delay if the channel capacity is sufficiently large. In practice, the average video quality per playing chunk is more appropriate for a performance metric representing the users’ satisfactions; therefore, we can see from the results in Figure 4-7 that adjustments of receiving chunk amounts are very important.

4.2.3.2 Uniform and Nonuniform Caching Probabilities

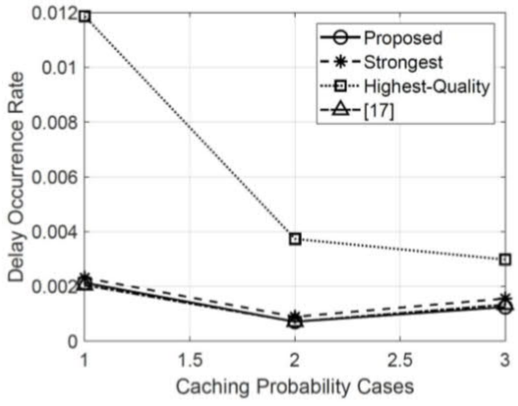


Figure 4-8: Delay occurrence rates over caching policies

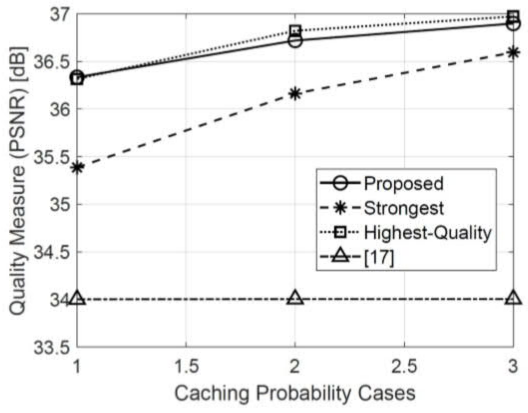


Figure 4-9: Video quality measures over caching policies.

We set three cases of caching probabilities for the video file with different quality levels, as follows:

- $Case\ 1$: $p_1 = 4/7$, $p_2 = 2/7$, $p_3 = 1/7$
- $Case\ 2$: $p_1 = 1/3$, $p_2 = 1/3$, $p_3 = 1/3$
- $Case\ 3$: $p_1 = 1/7$, $p_2 = 2/7$, $p_3 = 4/7$

Note that Case 2 corresponds to the uniform caching probability case and Case 1 and Case 3 are nonuniform. In Case 3, the streaming user is more likely to receive high-quality video than other cases, on the other hand, Case 1 represents an environment where there are not many caching nodes which can provide the high-quality video around the user. The performances of playback delay and quality measure depending on those cases of caching probabilities are shown in Figure 4-5 and Figure 4-6, respectively.

In Figure 4-8, delay incidence of ‘Highest-Quality’ definitely decreases as p1 decreases and p3 grows, because the caching nodes storing the high-quality video are likely to be near to the streaming user. However, since the distribution density of all caching nodes does not change, i.e. $\lambda \sum_{l=1}^{L} p_l = \lambda$, the delay performances of all the other schemes are not influenced much by different caching policies. For the ‘Strongest’ scheme, any caching probability case can deliver as many low-quality chunks of small size as possible when there are too few chunks in queue, so the playback delay is about to occur. In this sense, the proposed technique and [11] show almost the same delay performances as ‘Strongest’, because both schemes strongly limit the playback delay compared to quality improvement.

The average quality measures of all schemes increase as p1 decreases and p3 grows as shown in Figure 4-9. Even though ‘Highest-Quality’ pursues the video quality, its average quality measure per received chunk does not differ much from that of the proposed one for any caching probability case similar to Figure 4-7. Especially in Case 3, caching nodes storing the highest-quality video are distributed more than nodes of other types, therefore the caching node whose channel condition is the strongest would be highly probable to be type 3. Thus, the difference between quality performances of ‘Strongest’ and ‘Highest-Quality’ or the proposed scheme is not large.

The performance rankings in Figure 4-8 and Figure 4-9 among comparison techniques are consistent with the results of Figure 4-4 and Figure 4-5. Compared to those comparison schemes, the proposed technique provides quite high average video quality, while limiting delay occurrence rate as low as ‘Strongest’ and [11]. In addition, [11] still provides as low delay occurrence rates as the proposed one but its average quality per chunk is too poor to achieve user satisfaction.

4.2.3.3 System Parameter V

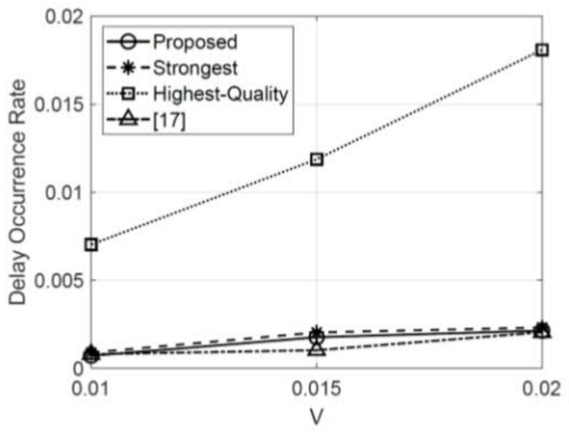


Figure 4-10: Delay occurrence rates over V.

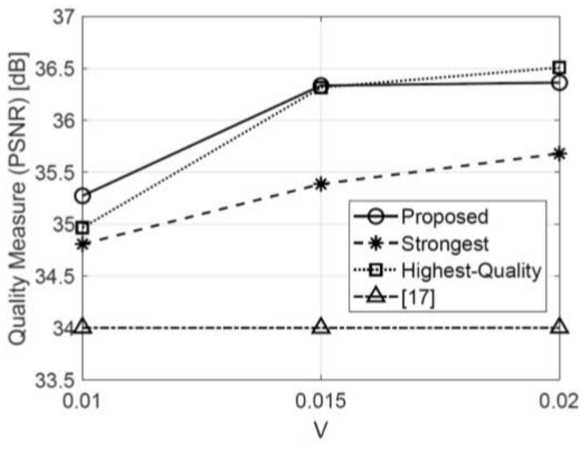


Figure 4-11: Video quality measures over V.

Since V has a role to weigh quality maximization compared to averting playback delay in Lyapunov optimization problem, delay occurrence rates increase and the expected quality measures of all techniques become improved, as V grows, as shown in Figure 4-10 and Figure 4-11, respectively. Therefore, we can control the tradeoff between video quality and playback latency by adjusting the system parameter V. Among comparison techniques, the proposed scheme improves the quality performance sufficiently while minimizing the increase in delay incidence by taking large V. As we've seen in Sections VI-A and VI-B, the proposed technique provides higher average video quality than 'Strongest' and [11], and much better delay performance than 'Highest-Quality'

4.2.3.4 SINR Level

The delay and quality performances over INR levels are shown in Figure 4-12 and Figure 4-13, respectively. It is easily expected that quality performances decrease, and delay occurrence rates increase as INR grows for all comparison techniques. Almost all of the performance rankings among comparison techniques remain as seen former subsections, but the performance of 'Highest-Quality' is influenced by INR levels more than the proposed one and 'Strongest'. We can expect that 'Highest Quality' becomes more difficult to accumulate video chunks in the queue as the INR grows, therefore the quality level chosen by the user becomes increasingly degraded. Rather, 'Strongest' is not significantly affected by INR changes compared to 'Highest-Quality', because the channel condition of its caching node is much stronger than that of the node chosen by 'Highest-Quality'. The proposed scheme still achieves the improved video quality while guaranteeing very low delay occurrence rate.

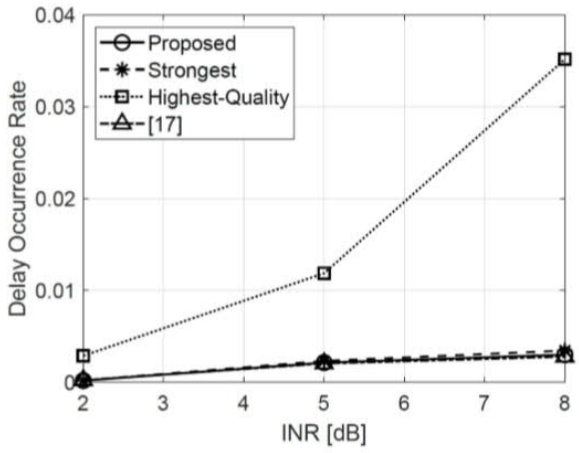


Figure 4-12: Delay occurrence rates over ϒ.

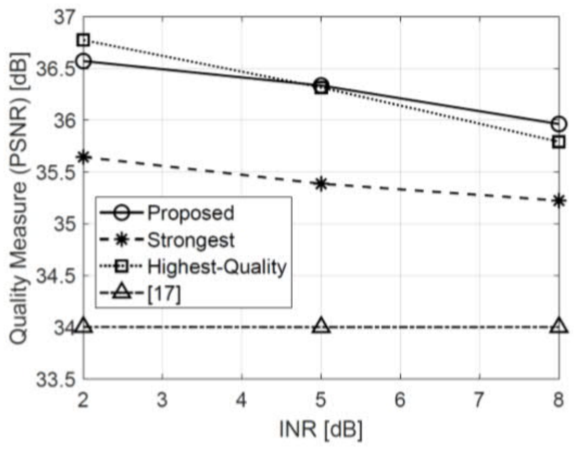


Figure 4-13: Video quality measures over ϒ.

4.2.4 *Conclusion*

This report studies the dynamic delivery policy of video files of various quality levels in the wireless caching network. When the caching node distribution around the streaming user is varying, e.g. the user is moving, the streaming user makes decisions on caching node to receive the desired file, vide quality, and the number of receiving chunks. The different timescales are considered for the caching node association and decisions on quality and the number of receiving chunks. The optimization framework of those video delivery decisions conducted on different timescales is constructed based on Lyapunov optimization theory and Markov decision process. By using dynamic programming and the frame-based drift plus-penalty algorithm, the dynamic video delivery policy is proposed to maximize average streaming quality while limiting playback delay quite low. Further, the proposed technique can adjust the trade-off between performances of video quality and playback delay by controlling the system parameter of V.

4.3 Application Mobility with Optimal Routing

Optimal Routing is part of the PriMO-5G architecture [7] and its main goal is to ensure low latency paths between servers and UEs (server-to-UE), and between UEs (UE-to-UE communication). In the same time, it ensures IP address preservation: the IP address of the UE does not change. After certain UE mobility events, when the base station is changing, the Session Management Function (SMF) may decide to relocate the User Plane Function (UPF). At this point, if the UE communicates with local edge server, it will still communicate with one in the old location. To be able to fully utilize low latency paths, the application server (instance) should potentially also be relocated from the old location to a new location closer to the UE. In this chapter, we describe how Optimal Routing can be complemented with application server relocation to fully utilize the benefit of the low latency paths.

4.3.1 *Optimal Routing Overview*

For a full overview on Optimal Routing, see D2.1 [3].

Optimal Routing introduces two new elements into the 5GC. In the control plane, the Location Register (LR) keeps the UE IP address to UPF mappings. In the user plane, an IAP (IP Announcement Point) advertises IP address ranges and takes downlink packets, queries the LR if needed, stores the reply in its local cache, and encapsulates the packets and sends them to the UPF where the session is processed. Whenever the mapping changes, e.g. after handovers with UPF change, the SMF notifies the LR, which in turns notifies all the IAPs that still have the previous reply in their local caches.

The Optimal Routing architecture is shown in a schematic illustration in Figure 4-14 In the figure, two remote servers communicate with the UE, and the flows go through different IAPs. Again, for more information on how the LR and IAP works, please see D2.1 [3].

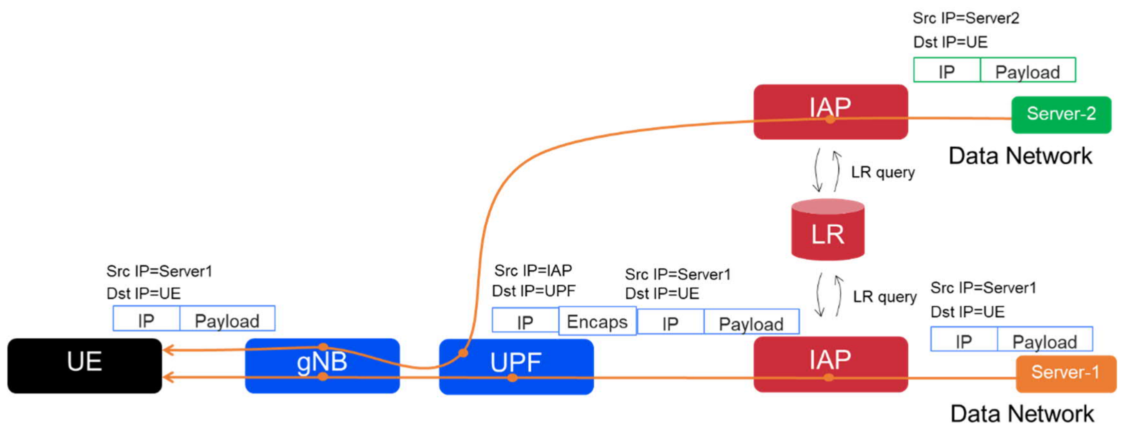


Figure 4-14: Optimal Routing

4.3.2 *The need for application relocation in edge with Optimal Routing*

Consider the scenario of two local (edge) sites, and a UE is communicating with an edge server in Local Site 1. Apart from this communication, the UE may communicate with other servers (e.g. in central locations or in the Internet), and also with other UEs. The communication in downlink direction is following the path, from an application in Local Site 1 (Server) via IAP-L1, UPF-L1, gNB to the UE. This is depicted in the left side of Figure 4-15. After a UE handover, base station change and UPF relocation, the UE still communicates with the server at Local site 1, and therefore the latency is potentially increasing.

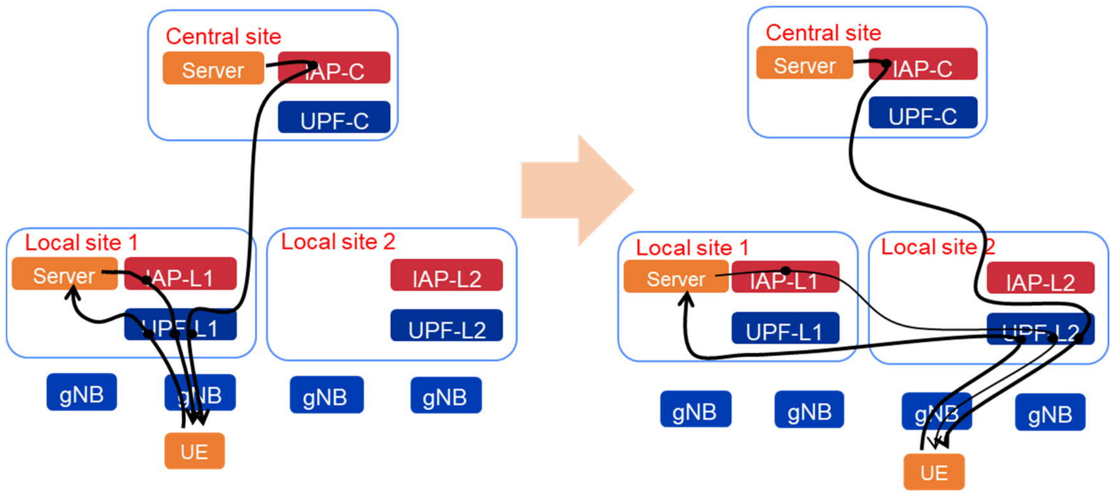


Figure 4-15: After UE mobility, the UE still talks at the edge server at Local site 1

To address the latency problem, the application state needs to be transferred (migrated) from Local site 1 to Local site 2. This means that the UE will be now served from the Local site 2. This is depicted in Figure 4-16.

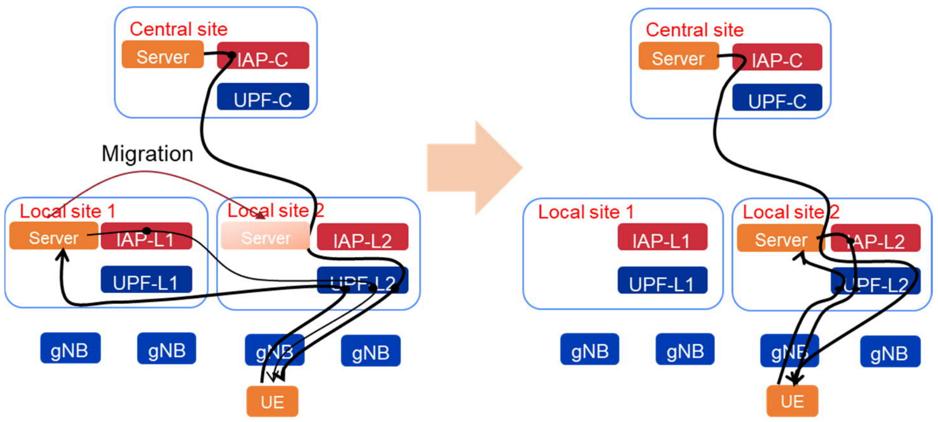


Figure 4-16: The application migration enables the UE to communicate with an edge server in Local site 2

4.3.3 Application Mobility Design

In the edge cloud, a Service Mobility Framework called Service Mobility Orchestrator is proposed. It is shown on Figure 4-17.

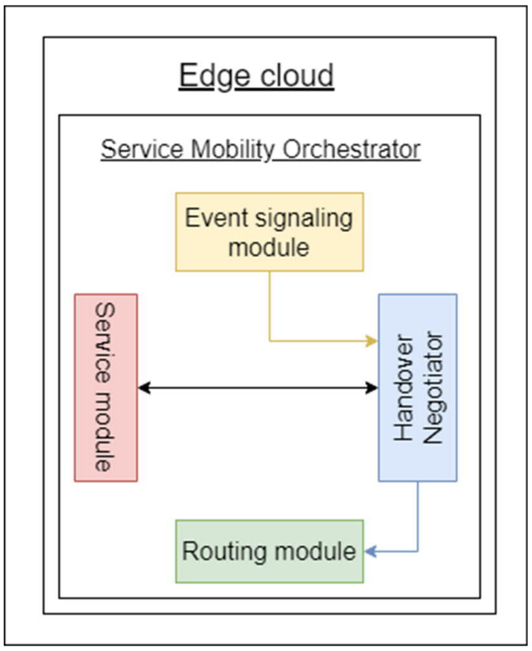


Figure 4-17: The overview of the Service Mobility Orchestrator framework [4]

The Service Mobility Orchestrator (SMO) has the following functional entities:

- Event Signalling Module: Acts as entry point, and receives handover trigger from the 5GC and forwards it to the Handover Negotiator. The handover trigger is coming from the SMF, via SMF event exposure.

- Handover Negotiator: Acts as controller, and communicates with all the other components in the SMO, as well as with peer SMO handover negotiator.
- Service Module: It keeps track of its own services, keeps track which UE is served by which service, and provides live migration feature. In our implementation, LXD [5] is used as service module. It offers an API to create, delete LXC containers. It utilizes CRIU (Checkpoint/Restore in Userspace) [6] to handle the live migration process.
- Routing module: It provides accessibility to the different services during the relocation procedure. For example, in a transient phase, where uplink packets are already arriving to the new edge cloud, but the relocation is not complete, it routes the packets back to the old edge cloud, to keep the communication on-going.

The relocation itself is divided into a number of phases in the Handover Negotiator module:

- Initiate: this phase handles the handover trigger and performs a preliminary decision for service migration.
- Validate: this phase checks and ensures that there are enough resources to accept an incoming service
- Prepare: this phase synchronizes the source and destination involved in service migration
- Execute: this phase finds the service that is serving the UE and performs live migration from the source to the target edge cloud
- Complete: this phase frees up all old and temporary resources

4.3.4 Experimentations

In our experimentation, we have used an NGINX server, which is hosted in LXC container. A UE with a video client can access the service by establishing a TCP connection towards it. On the top of TCP, RTMP (Real-Time Messaging Protocol) is used for streaming the video from the service to the UE.

We have tested our implementation with two different setups:

- OpenStack testbed with 5GC/Optimal Routing prototype
- Physical machine with 2 VMs as edge clouds but without 5GC components

The OpenStack testbed is shown in Figure 4-18. It consists several Virtual Machines. Two edge sites are modelled, Site A and Site B. The edge sites have 3 VMs: one hosting the IAP, one the UPF, and a third is hosting the edge cloud: the SMO and the services (containers) serving the UEs.

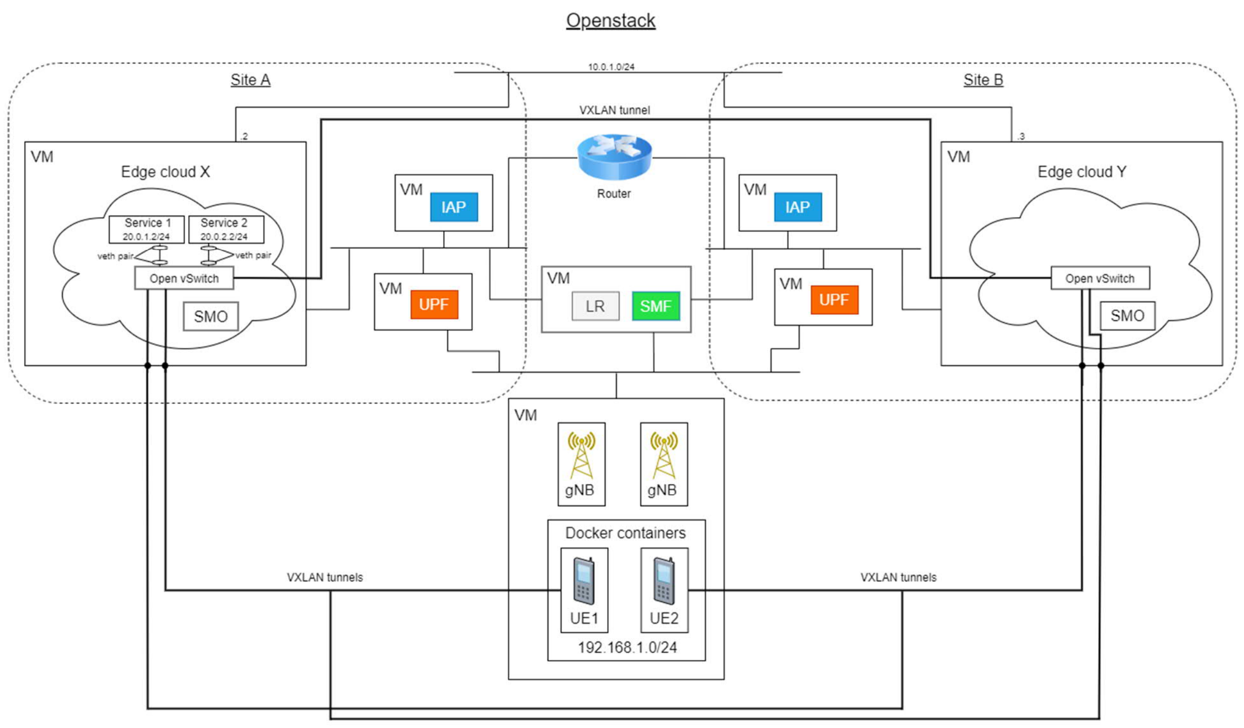


Figure 4-18: OpenStack testbed with 5GC/Optimal Routing prototype [79]

The second setup is shown on Figure 4-19. Here, only the edge clouds are modelled, again, as different VMs (Site A and Site B). They host the services, and the SMO. Also, this setup is using a simplified emulated SMF component. Finally, there is a direct link for live migrating the containers.

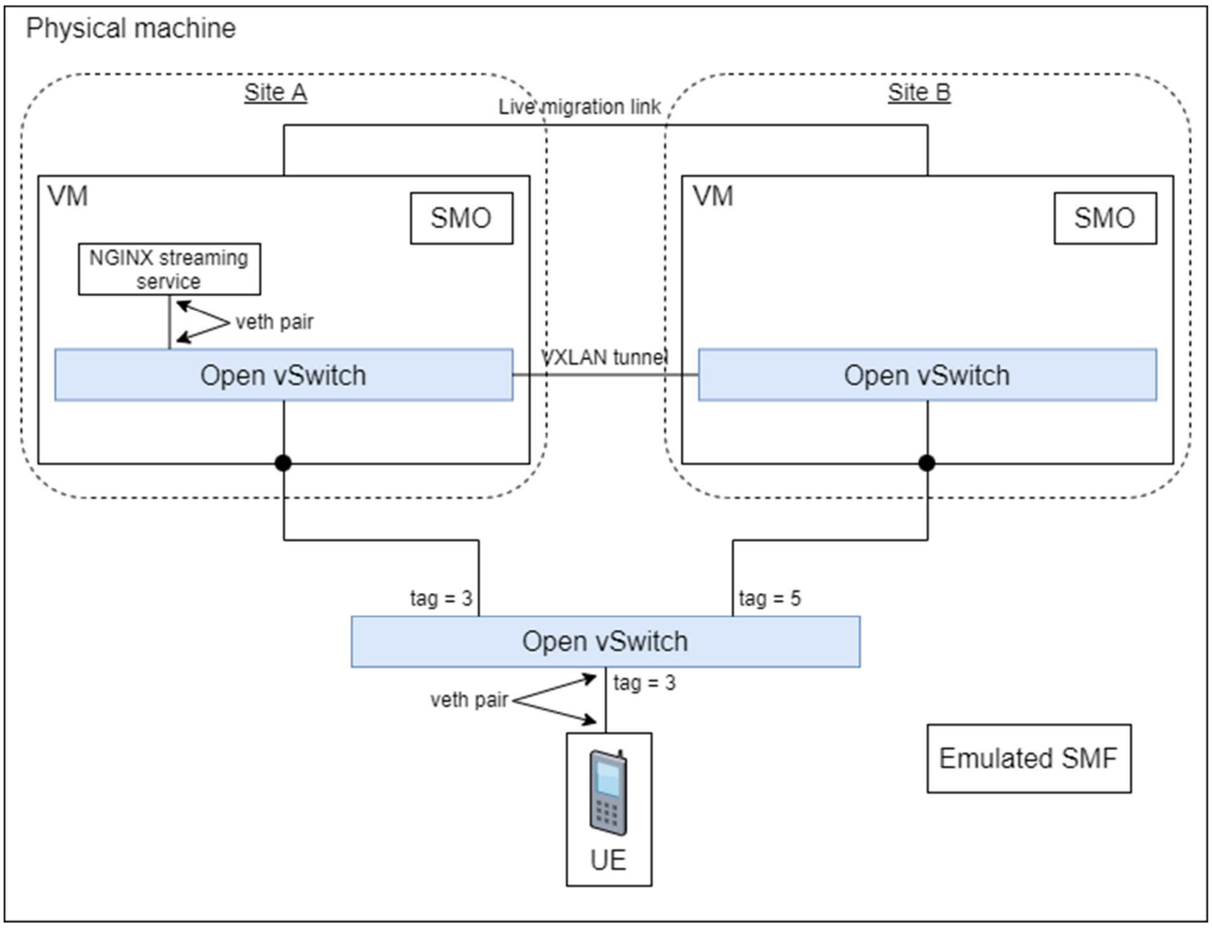


Figure 4-19: Physical machine with 2 VMs as edge clouds but without 5GC components [4]

In both setups, we have performed 50 relocations to test the characteristics of our solution. The following KPIs are of interest:

- Time spent in each of the 5 relocation phases (minimum, average, 90th percentile, maximum, standard deviation)
- Service downtime
- Total migration time

4.3.4.1 Results – OpenStack testbed

Relocation phase	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Initiate	38	169	92,74	30,07	129,30
Validate	2	21	6,92	3,99	12,1

Prepare	18	89	40,54	17,40	62,4
Execute	8674	15360	11377,24	1519,23	13325,5
Complete	41	188	96,08	42,36	141,3

Table 4-3: Time spent in each of the relocation phases in the source [4]

Relocation phase	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Initiate	1	28	6,7	5,4	13,4
Validate	1	13	2,78	2,16	4,1
Prepare	17	313	43,64	43,32	62,3
Execute	16	57	29,36	11,38	46,3
Complete	168	969	348,16	185,05	612,9

Table 4-4: Time spent in each of the relocation phases in the target [4]

KPI	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Total migration time	8966	15816	11767,66	1523,17	13627
Downtime	5619,96	9143,86	7199,23	992,75	8603,99

Table 4-5: Total migration time and downtime [4]

The results obtained in the OpenStack testbed can be seen in Table 4-3, Table 4-4 and Table 4-5, respectively. We observe that the most time is spent in the Execute phase in the source edge cloud. This is related to how CRIU works: it performs iterative dumping of pages and transfers them to the target. We have found that this takes a long time in this setup. The reasons may be: the size of the container, the number of TCP connections, internals of the dumping algorithm. As the container has a low size (<30 MB), and only one TCP connection is established, we believe that the reason is in CRIU internals. We have also tested without a running video session, where similar results were obtained. In another test, we have used a simple container only pinging the UE, where the results again were high.

Ultimately, a partial test revealed that creating a stateful snapshot with CRIU from the nginx container takes ca. 3 seconds on a VM with 2 vCPU, while it takes 0.7 seconds on a VM with 1 vCPU.

These learnings made us to design the second testbed, where the SMO performance can be better obtained.

4.3.4.2 Results – Physical machine testbed

Relocation phase	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Initiate	19	85	39,66	14,6	57,3
Validate	1	8	1,48	1,45	2
Prepare	32	83	44,18	12,61	59,60
Execute	2438	3001	2607,9	111,09	2707,7
Complete	36	86	49,1	10,1	67

Table 4-6: Time spent in each of the relocation phases in the source [4]

Relocation phase	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Initiate	<1	22	1,92	3,19	3
Validate	<1	3	<1	<1	1
Prepare	18	69	31,3	8,82	42,4
Execute	18	44	21,74	5,34	25,5
Complete	85	236	99,32	21,85	111,0

Table 4-7: Time spent in each of the relocation phases in the target [4]

KPI	Min (ms)	Max (ms)	Mean (ms)	Standard dev. (ms)	90th percentile (ms)
Total migration time	2649	3217	2835,26	122,31	2966,5
Downtime	1255,79	2909,97	1679,18	493,16	2622,81

Table 4-8: Total migration time and downtime [4]

The results obtained in our physical machine testbed can be seen in Table 4-6, Table 4-7 and Table 4-8, respectively. The most important KPI, the Downtime reduced in average to 1.68 seconds, compared to the 7.2 seconds obtained in the OpenStack setup. The total migration time is also reduced to 2.83 seconds from 11.77 seconds. This can be explained by the availability of direct inter-connections between the VMs and running the containers that are live migrated in a VM with 1 vCPU.

4.3.5 Discussion and Conclusions

There are several opportunities to introduce and exploit AI techniques.

By utilizing NWDAF [2], the SMF can subscribe to mobility predictions. This way, the SMF may prepare for the UPF relocation, and give a notification to the Application Function earlier, which can initiate the application service relocation before the user changed its location, and therefore ensuring low latency between the edge server and the UE.

Another opportunity is utilizing AI techniques in the Handover Initiator module of the SMO to be able to make better decision whether to move the application service or not to the new location.

In this work, we have explored application service mobility and built a Service Mobility Orchestrator that acts as an Application Function and receives notifications from the SMF if the user plane path changes. After receiving the trigger, it may relocate the application service, by live migrating the container serving the UE. The solution complements Optimal Routing, and helps to maintain low latency between edge applications and the UEs, even when mobility is considered.

5 AI-MEC System Design Specification and Integration

5.1 Introduction

UAV are able to be connected to cellular networks as a new user equipment. Furthermore, the UAV devices can take a role of mobile base station, thus it can be used for coverage extension, spectral efficiency improvements, and user quality of experience improvements. Therefore, there are many research results for the use of UAV device in wireless and cellular networks. In this case, AI-based schemes are widely and actively used for the dynamic optimization under the consideration of energy-constraints and dynamic-unexpected high mobility.

5.2 UAVs types and characteristics

A. **Payload:** The payload is defined as the maximum weight that a drone can carry. When this payload becomes larger, the UAV sizes are getting bigger and the battery capacity of UAVs becomes bigger. However, the flight time becomes lower.[13]

B. **Flying Mechanism:** The flying mechanism can be classified into multi-rotor drones (refer to Figure 5-1), fixed-wing drones (refer to Figure 5-2), and hybrid fixed/rotary wing drones (refer to Figure 5-3). The multi-rotor drones are capable for vertical take-off and landing; and it can hover over a fixed location to provide continuous cellular coverage for certain areas [14]. However, the mobility is limited and the power consumption during flight is relatively high. The fixed-wing drones are able to do glide over the air. Therefore, it is more energy-efficient and is able to carry heavy items. However, it is not able to conduct hovering. The hybrid fixed/rotary wing drones is the combination of multi-rotor drones and fixed-wing drones.

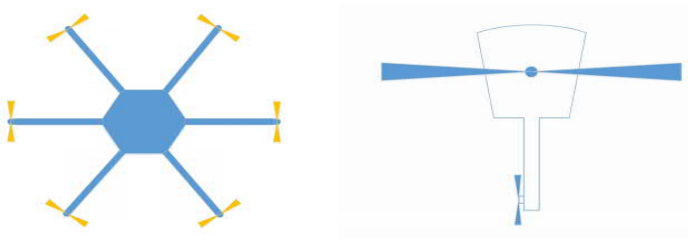


Figure 5-1: Conceptual illustration for multi-rotor drones.

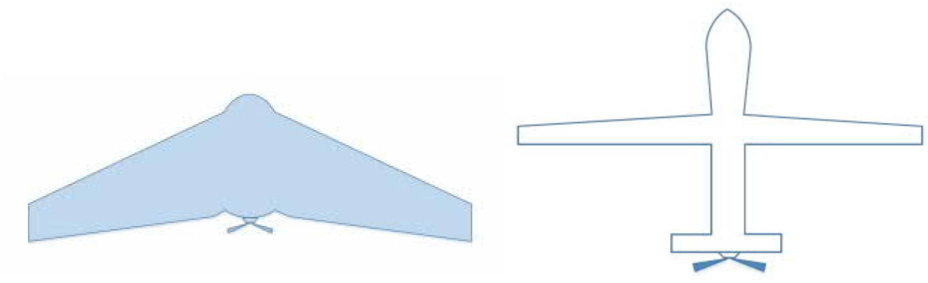


Figure 5-2: Conceptual illustration for fixed-wing drones.

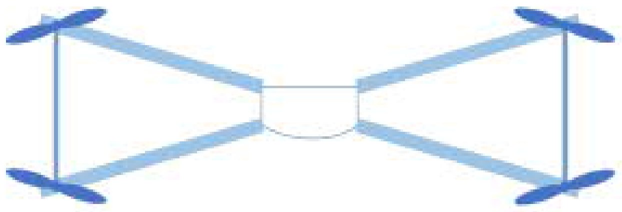
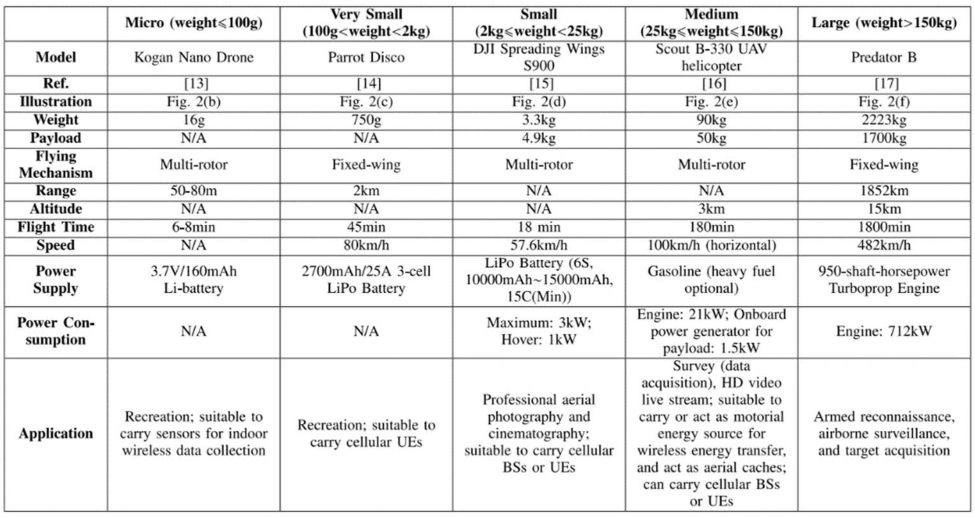


Figure 5-3: Conceptual illustration for hybrid fixed/rotary wing drones.

C. **Range and Altitude:** a UAV BS needs to vary its altitude to maximize the ground coverage and satisfy different quality of service (QoS) requirements [15]. Low-altitude platforms (LAPs) are usually employed to assist cellular communications since they are more cost-effective and allow fast deployment and provide short-range line-of-sight (LOS) links that can significantly enhance the communication performance. High-altitude platforms (HAPs) such as balloons can also provide cellular connectivity. HAPs have a wider coverage and can stay much longer in the air, using HAPs in cellular communications may cause total network outage due to extremely large inter-cell interference [16][17][18].

D. **Speed and Flight time:** When a UAV BS/relay flies in a designated trajectory to maximize its energy and spectral efficiency, its speed needs to be carefully considered if the trajectory requires frequent turns (trade-off between drone's speed and turning agility) [19]. The maximum time a drone can spend in the air without recharging or refuelling is referred to as its flight time or endurance, the limited endurance of the existing off-the-shelf UAVs is currently one of the major practical factors restricting their full-scale deployment in cellular networks.

E. **Power supply:** A drone's power supply significantly determines its endurance.



	Micro (weight⩽100g)	**Very Small (100g<weight<2kg)**	**Small (2kg⩽weight<25kg)**	**Medium (25kg⩽weight⩽150kg)**	**Large (weight>150kg)**
Model	Kogan Nano Drone	Parrot Disco	DJI Spreading Wings S900	Scout B-330 UAV helicopter	Predator B
Ref.	[13]	[14]	[15]	[16]	[17]
Illustration	Fig. 2(b)	Fig. 2(c)	Fig. 2(d)	Fig. 2(e)	Fig. 2(f)
Weight	16g	750g	3.3kg	90kg	2223kg
Payload	N/A	N/A	4.9kg	50kg	1700kg
Flying Mechanism	Multi-rotor	Fixed-wing	Multi-rotor	Multi-rotor	Fixed-wing
Range	50-80m	2km	N/A	N/A	1852km
Altitude	N/A	N/A	N/A	3km	15km
Flight Time	6-8min	45min	18 min	180min	1800min
Speed	N/A	80km/h	57.6km/h	100km/h (horizontal)	482km/h
Power Supply	3.7V/160mAh Li-battery	2700mAh/25A 3-cell LiPo Battery	LiPo Battery (6S, 10000mAh~15000mAh, 15C(Min))	Gasoline (heavy fuel optional)	950-shaft-horsepower Turboprop Engine
Power Consumption	N/A	N/A	Maximum: 3kW; Hover: 1kW	Engine: 21kW; Onboard power generator for payload: 1.5kW	Engine: 712kW
Application	Recreation; suitable to carry sensors for indoor wireless data collection	Recreation; suitable to carry cellular UEs	Professional aerial photography and cinematography; suitable to carry cellular BSs or UEs	Survey (data acquisition), HD video live stream; suitable to carry or act as motorial energy source for wireless energy transfer, and act as aerial caches; can carry cellular BSs or UEs	Armed reconnaissance, airborne surveillance, and target acquisition

Table 5-1: Characteristics of different drone types.

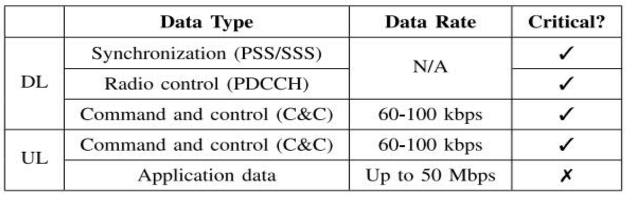
5.3 Standardization: Enabling UAV cellular communication

The cellular industry has recognized the importance of providing support to low-altitude UAVs for enabling beyond LOS control and establishing a reliable communication.

5.3.1 3GPP Study Item

i. UAV Traffic Requirements:

The 3GPP identified the traffic types that cellular networks should cater for UAVs flying between ground level and 300 meters.



	Data Type	Data Rate	Critical?
DL	Synchronization (PSS/SSS)	N/A	✓
	Radio control (PDCCH)		✓
	Command and control (C&C)	60-100 kbps	✓
UL	Command and control (C&C)	60-100 kbps	✓
	Application data	Up to 50 Mbps	✗

Table 5-2: UAV communication requirements.

The traffic types can be classified into three categories: 1) synchronization and radio control, 2) command & control for reliably controlling, and 3) application data

ii. Channel modelling:

In order to characterize the performance of existing cellular networks when serving both ground and aerial devices, the 3GPP developed a UAV-specific statistical channel model building. This channel model complements those developed by the academic community. These academic models are classified into three different categories (Air to Ground (A2G), Air to Air (A2A), and Ground to Air (G2A)) [20].

The 3GPP propose models for rural-macro (RMa), urban-macro (UMa), and urban-micro (UMi) BS deployments. The main UAV-related features are UAV spatial placement which is defined as five different cases depending on the density of UAVs in the network by the 3GPP, LOS probability (between ground BSs and UAVs), Path loss which decreases as UAVs increase their height, Shadowing, Fast-fading model [20].

iii. UAV performance analysis:

UAVs are more likely to undergo downlink and uplink interference problems than GUEs. This is mainly due to two factors, namely, that flying UAVs are likely to be in line-of-sight with a large number of base stations (BSs), and that the majority of these BSs are down tilted, since their deployment has been optimized for providing coverage to GUEs. This impacts all phases of the cellular communication, i.e., 1) association and handover, 2) downlink transmissions, and 3) uplink transmissions.

1) Association and handover: In contrast to GUEs, flying UAVs do not generally associate to their physically closest BS. This is because cellular BSs generally focus their main antenna beam towards the center of their ground coverage area [21]. Instead, the association of airborne devices is dominated by the sidelobes of their directive BS antennas [22], [23]

2) Downlink transmissions: a UAV receives line-of-sight transmissions from a large number of BSs when increasing its altitude. UAVs flying above the building clutter generate/perceive interference towards/from a multiplicity of line-of-sight BSs. Instead, UAVs flying at low altitudes only generate/perceive interference towards/from nearby BSs [24].

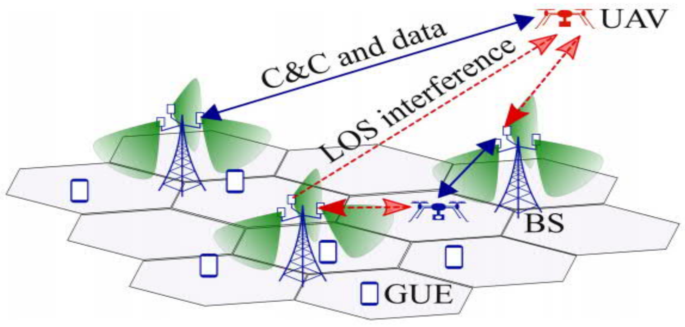


Figure 5-4: Illustration of the UAV interference challenge in cellular networks

3) Uplink transmissions: UAVs' good propagation conditions with a multiplicity of BSs also impact the network's uplink performance. This is because airborne devices transmitting data towards their serving BS can generate strong interference to a variety of ground BSs, an issue that becomes critical in networks with a large number of UAVs [25].

iv. Enhancing UAV Communications:

To address the interference challenges, the 3GPP examined a number of complementary interference mitigation techniques.

1) Association and handover

UAV location and flight plan knowledge can be leveraged to facilitate the handover procedure, e.g., by anticipating BS candidates for a potential handover. Enhancement of existing report mechanisms through the definition of UAV-specific handover triggering conditions and an optimized control of the reporting load [26].

2) Downlink transmissions:

The full dimension MIMO (FD-MIMO) multiantenna BSs defined in LTE Release 13 enhance the performance of UAV communications thanks to a) their beamformed transmissions, which allow reducing the amount of interference generated towards the constrained spatial regions where UAVs lie, and b) their spatial multiplexing capabilities, which in turn enable a better utilization of the precious time/frequency resources.

UAVs with directional antennas and beamforming capabilities contribute to reduce the number of downlink interferers perceived by aerial devices. These interference mitigation gains can be further complemented with a boost of the useful signal power in UAVs beam steering towards their serving BS [12].

Cooperative multipoint (CoMP) can convert the harmful line-of-sight BS interferers into useful signal contributors.

3) Uplink transmissions:

Uplink power control is essential to harmonize the coexistence among GUEs and aerial devices.

Full dimension MIMO (FD-MIMO) can also benefit uplink transmissions by enabling spatial separation of ground and aerial users, which have clearly distinguishable propagation characteristics.

UAVs with directional antennas and beamforming capabilities generate a diminished amount of interference to the GUE-generated uplink transmissions owing to a potential decrease in both the number of interfered BSs and the UAV transmission power [27].

updated information about the flying status of a mobile device is required to effectively implement some of the above-mentioned solutions. The 3GPP considered a variety of device- and network-based solutions to acquire this information. Among others, these include the use of explicit UAV identification signalling, the exploitation of mobility history information, or the employment of measurement reports from mobile devices [12].

5.3.2 3GPP Work Item: Main Outcomes and Way Forward

- Uplink power control, by augmenting the existing fractional power control mechanisms through a) the assignment of a UAV-specific path loss compensation factor, and b) the range extension of the P0 parameter [28]
- Signalling, to identify the status of airborne devices. For instance, the technical specification (TS) 36.331 now defines new reporting events that are triggered when the UAV height is above or below a BS-configurable threshold [28]
- Interference detection, by allowing UEs to trigger a measurement report when a configurable number of reference signals received from neighbouring cells satisfy specific power-related conditions [28]
- Subscription-based access, to prevent the non-authorized connection of cellular-connected UAVs. This is enabled through new network signalling from the core network to cellular BSs, allowing the latter to determine whether the UE subscription includes aerial functionalities [12]
- Mobility, where new radio resource control (RRC) signalling was included to facilitate the flight plan communication from UAVs to their serving BS. This information can be subsequently exploited to facilitate handovers [12]

5.3.3 UAV Standardization Outside the 3GPP

Other standardization bodies have also considered the particular characteristics of UAVs in the definition of new specifications:

The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) defined the work item Y.UAV.arch for providing a functional architecture for UAVs and UAV controllers using IMT-2020 networks. [29]

The European Telecommunications Standards Institute (ETSI) technical report (TR) 103 373 aims at identifying UAV-specific use cases and understanding whether new spectrum rules are required for enabling them [30], [31].

The Institute of Electrical and Electronics Engineers (IEEE) defined the Drones Working Group in 2015. This group aims to develop a standard for consumer drones, primarily with the intention of addressing privacy and security concerns. With this purpose, the standard is currently focused on specifying 1) the taxonomy and definitions related to UAVs, and 2) the requirements, systems, methods, testing and verification required to preserve the privacy and security of people and properties within range of the UAVs. [33], [34].

5.4 Aerial base stations: challenges and opportunities

Because UAV BSs can be quickly deployed at optimum locations in 3D space, they can potentially provide much better performance in terms of coverage, load balancing, spectral efficiency, and user experience compared to existing ground-based solutions.

The deployment of aerial BSs, however, faces several practical issues.

5.4.1 Placement Optimization for Aerial BSs:

The problem of optimum placement is more challenging for aerial BSs compared to the conventional terrestrial BSs because an aerial BS can be placed at many different heights in the sky [34], [35]. However, the coverage as well as the UL and DL channels change with the altitude of the BS. Some researchers considered the height of the aerial BS as a variable in their optimization formulation thus treating it as a 3D placement problem, while others essentially solved 2D placement problems for constant heights. Optimizations also differed in whether the backhaul, interference from other BSs, and existence of terrestrial BSs in the same coverage area were considered in problem formulation. Finally, the objectives in of these optimizations varied from maximizing the system capacity to minimizing the required number of aerial BSs to minimizing the total transmit power of the entire aerial system [12].

5.4.2 Mobility Optimization for Aerial BSs:

Mobility is an intrinsic feature and capability of aerial BSs, which provide additional opportunities to dynamically improve their placements in response to user movements on the ground. To exploit the mobility features of aerial BSs, the practical hardware limitations of the UAVs must be considered. Based on the transportation methods of aerial BSs, there are two types of mobilities

1) UAVs are used only to transport a BS to a particular ground location where the BS auto starts to serve the users. If the BS needs to be relocated, it must shut down first before being transported to the new location

2) UAVs continue to carry the BSs and the BSs can continuously serve the ground users while they are flying [36]

In both types, although a single UAV can perform plenty of tasks, multiple UAVs can form a cooperative group to achieve an objective more efficiently, and to increase the chance of successful task operation. Additionally, the robustness of the communications will increase by employing cooperative UAVs. Designing cruising aerial BSs requires autonomous mobility control algorithms that can continuously adjust the movement direction or heading of the BS in a way that maximizes system performance. These algorithms must also insure that multiple aerial BSs cruising in an area can maintain a safe distance from each other to avoid collisions [12].

5.4.3 Power-Efficient Aerial BSs:

A critical problem of Aerial BSs is their short lifetime due to the battery depletion problem. Power is consumed by both communications (electronics) and mobility (mechanical).

1) Reducing Communication Energy

   minimize the transmission power: The deployment of UAVs was optimized to minimize the total transmit power for UAVs while satisfying the users' data-rate requirements [37]. And reducing the number of transmissions to decrease the energy consumption of a UAV is addressed

   improve communication energy efficiency: The solution is to develop optimal transmission schedule of UAVs, especially when UAVs are flying in a predetermined trajectory. The frequent need of UAVs to recharge interrupts the data collection, as a result, prolonging the UAVs lifetime is critical [38].

2) Reducing Mechanical Energy

   To reduce mechanical energy of UAVs, first, an energy consumption model is needed [39], [40]. (ex. $E = (\beta + \alpha . h)t + P_{max}(h/v)$

   According to these models, one solution to control the energy consumption of UAVs is to regulate their height. However, changing the height might reduce the performance of UAVs. There is a trade-off between energy consumption and coverage radius. One of the major advantages of these methods is that they have targeted the mechanical energy consumption of UAVs, which is considered as the main source of energy consumption for UAVs.

5.4.4 Recharging of Aerial BSs:

Separate from reducing the energy consumption of UAVs, another attractive solution to combat the short lifetime of UAVs is to consider charging locations for them and replace exhausted UAVs with the fully charged ones. Comparing to the methods that focus on reducing the energy consumption, this solution is more costly and complex, as replacing/charging points must be designed in urban areas. Moreover, the battery consumption of UAVs needs to be monitored regularly. [39], [41]

5.4.5 Fronthauling and Access Communication Links:

Assuming that it is feasible to efficiently fly a BS attached to a UAV in terms of load, dynamic positioning and power consumption, a crucial remaining aspect is the establishment of a communication link between the UAV and the terrestrial network. This link, conventionally referred to as fronthaul, is crucial to guarantee enough bandwidth and reliability in the access link between the UAV and the user terminals. Due to the need of keeping the computational complexity low, thus the associated energy consumption, it is foreseen aerial BS will generally work as relay nodes that requires the implementation of a reduced number of protocol stack layers, with the simplest configuration that involves only layer1 and substantially works as an amplify and forward node. Moreover, another crucial aspect to minimize the use of the spectrum will be the possibility for aerial BS to allow fronthaul and access communications on the same frequencies [42].

5.4.6 Paradigm-Shifting Cost Model of Aerial BSs:

With flying drones carrying cellular BSs, site-related costs could be significantly reduced or even completely removed. While the costs of BSs and network testing for drone-cells should increase compared to employment of terrestrial BSs due to the purchase and operation of drones. Overall, drone-mounted BSs have the potential to slash down the CAPEX (Costs of conventional cellular operators usually include capital expenditure) of cellular operators by significantly reducing the site-related costs, which currently account for the lion’s share of the CAPEX.

5.5 Prototyping and field tests

5.5.1 Facebook Aquila

One interesting example of hight altitude drone BS is the Facebook Aquila project, which aims at providing Internet coverage in remote areas directly from the sky [43]. The main component of the system architecture is an unmanned autonomous aircraft, named Aquila, which is capable of flying at an altitude of 18-20km over a defined trajectory to create a communication coverage region of about 100 km. Aquila is self-powered through solar panels integrated on wings wider than a Boeing 737 and has light weight to increase the flying time. Moreover, it counts with a control system to adjust the GPS-based route and monitoring the most important flying parameters (like heading, altitude, airspeed, etc.)

5.5.2 Google Loon

The Google Loon project aims at bringing Internet connectivity in remote areas [44]. This is achieved by adopting stratospheric balloons to relay radio communication links from ground stations to users’ LTE phones out of the coverage of traditional ground cellular communication infrastructures. Unlike the Facebook Aquila, the communication may be directly relayed to the end user and not to ground access points and the positioning of the balloons in the sky is controlled to generate the required coverage area at the ground.

5.5.3 Nokia F-Cell

It is an interesting example of drone base stations prototyping for low altitude applications [45]. The fundamental problem that F-Cell tries to solve is the high cost associated to the deployment and installation of a large number of small cells. F-Cell is an innovative solar-powered, self-configured and auto-connected drone deployed small cell served by a massive MIMO wireless backhaul. The key innovations proposed by F-Cell can be summarized in the following three aspects, removing the need for a wired power supply, the need for a wired backhaul, and the constraint of a fixed deployment.

5.5.4 Eurecom Perfume

Perfume project has developed the concept of "autonomous aerial cellular relay robots", where UAVs act as a relay base stations capable to enhance connectivity and throughput performance for off-the-shelf commercial terminals. The key target of the Perfume project is to design machine learning algorithms able to find and constantly update the optimal 3D position of flying wireless relays using fine-grained information of their LOS conditions together with other radio measurements. [46]

5.5.5 Huawei Digital Sky

cellular networks are involved to ensure C&C between drones and ground control stations. In particular, one interesting use case is a remotely operated passenger carrying drone (a taxi drone), which is controlled through live high quality video streaming transmitted over a 4.5G cellular network directly to the operation room [47].

5.6 Regulation

The evolution of UAV regulations significantly contributes to the integration of UAV into national and international aviation systems. In this section, socio-technical concerns of drone operations are outlined, and we explain the main criteria that constitute the current UAV regulation frameworks.

5.6.1 Socio-Technical Concerns of Drones

Emerging technologies have enabled the widespread use of drones and their strong operational capabilities. As a result, there are increasing concerns regarding privacy, data protection and public safety from national and international aviation authorities. To understand the motivation of the development of UAV regulations, socio-technical concerns of drones need to be analysed.

- Privacy: The operation of drones can be a serious threat to the privacy of both individuals and busi nesses. Moreover, high manoeuvrability and sensitive on-board instruments have made drones ev en more capable of privacy breaches [48]. Although every country has legislation to protect the pri vacy of the public's and citizens', rules might be out-of-date due to the rapid development of emerg ing technologies. Therefore, the operation of drones needs to be regulated to further protect the pri vacy.

- Data Protection: During their operation, drones are usually equipped with sensors that collect pers onal data such as images, videos and location data. According to data protection laws, citizens' p ersonal information should be protected from abuses. Moreover, as a result of high mobility, drone s indiscriminately collect and store a mass of data, which is against data protection principles. Con sequently, operation of drones should also be governed to protect personal information [49].

- Public Safety: Compared to traditional manned aircraft, drones are usually insufficiently maintained and more likely to encounter pilot errors.

– UAV collisions with manned aircraft might lead to engine shutdowns or damaged surfaces, risking the loss of control. Therefore, in many countries there are constraints such as maximum allowed flight heights and minimum distances to airports for drones' operation [50].

– UAV collisions with terrain usually cause loss of control, which might hurt civilians on the ground. Hence some countries forbid drones to fly over certain areas such as specific urban areas with high population density [50].

5.6.2 *Criteria based on Socio-Technical Concerns of Drones*

Based on the socio-technical concerns specified above, UAV regulations are framed and developed. Current UAV regulations are mainly based on six criteria, as explained below [51].

• Applicability: Applicability describes the scope that UAV regulations apply to. For example, drones are usually classified into groups based on weight or purpose, which might be treated differently by UAV regulations.

• Technical requirements: Technical requirements specify the mandatory instruments or techniques for drones. For example, collision avoidance mechanism could be a typical one.

• Operational limitations: UAV's operation is usually restricted by many factors. Typical operational limitations include maximum flight height, minimum distance to airport and individuals, prohibited areas, etc.

• Administrative procedures: Certain procedures and documents might be required before a UAV is allowed to operate, which include registration, operational certificate and insurance.

• Human resource requirements: For certain categories of UAV and purposes of operation, the pilot needs to be qualified.

• Implementation of ethical constraints: This criterion appoints the demands for data and privacy protection when operating drones.

5.6.3 *UAV Communications Regulation*

• The Electronic Communications Committee (ECC) within the European Conference of Postal and Telecommunications Administrations (CEPT) produced the ECC 268 report discussing the technical, regulatory aspects and the needs for spectrum regulation for UAVs. This report paves the way to the harmonisation across Europe of the frequencies dedicated to UAV communications. Additionally, ECC 268 discusses the communications requirements of both professional UAV use cases, which could benefit from using individual licensed spectrum, and non-professional UAV applications, where unlicensed bands may suffice for short range communications [12].

• The Federal Communications Commission (FCC) in the U.S. received in Feb. 2018 a petition for rule-making from the Aerospace Industries Association (AIA) [167]. The petition is seeking to allow the secure communication of C&C and non-payload data between UAVs and licensed pilots in the 5030-5091 MHz band.

5.7 Security

Security is a very important issue for any digital system. For a UAV-aided wireless communication system, due to its unmanned nature and required remote wireless communication, security is an even more serious problem. Moreover, cellular UEs served by terrestrial BSs might suffer from strong interference due to LOS links, if a UAV is manipulated by attackers. Therefore, it is significant to ensure the security of UAV systems when drones are used for cellular communications.

5.7.1 Cyber Security

5.7.1.1 Use cases:

When drones are employed for cellular communications, they can serve as cellular UEs or flying cellular BSs/relays, as shown in Figure 5-5. In case of UAV UEs, drones can be directly controlled either by terrestrial BSs or by ground control stations (GCSs) through non-cellular connections, which are mainly Wi-Fi connections [52]. There are three categories of radio links, which are satellite connection, cellular connection and Wi-Fi connection respectively. It is known that compared to cellular networks and GPS networks, Wi-Fi networks are more insecure due to the unreliable and vulnerable security techniques [12]. Since GPS signals are broadcast and the signal format is specified to the public, it is easier to attack satellite connections than cellular connections where encryption keys and scrambling code are exchanged end-to-end. Therefore, risk levels of cellular connections, satellite connections and Wi-Fi connections are evaluated as low, medium and high, respectively.

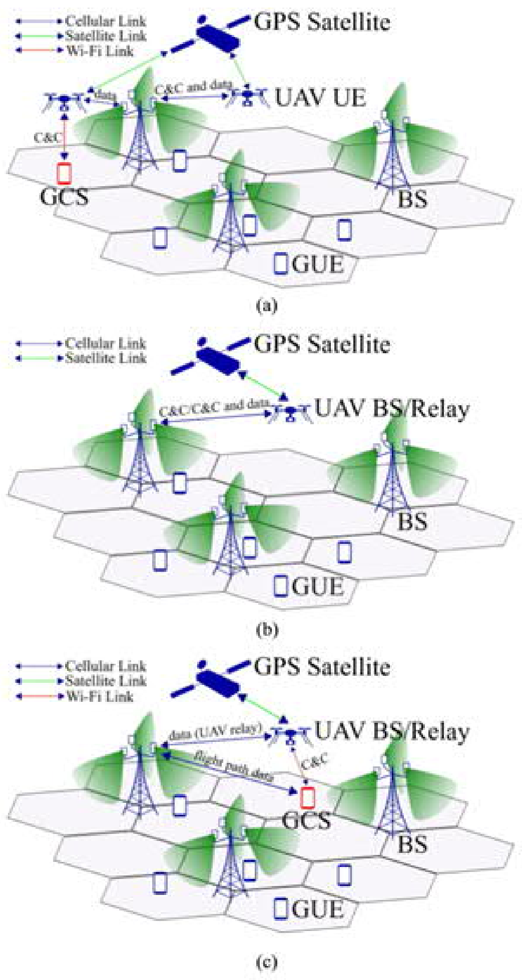


Figure 5-5: Use cases for cellular UAVs. (a) UAV UEs. (b) UAV BSs/relays controlled by terrestrial BSs. (c) UAV BSs/relays controlled by third party GCSs.

A. Threat identification and countermeasure

After assessing the overall risk levels of the radio links in the use cases above, threats for these radio links are identified and listed. Corresponding countermeasures are also presented.

• Jamming: Adversaries generate interference signals in the same frequency band to disrupt the reception process, which is a common way of integrity attacks. For jamming attacks, increasing the signal to noise ratio (SNR) could be a typical defence solution. However, this is always limited by how much power the transmitter can provide and how to lower the noise at the receiver by effective receiver algorithms

• Eavesdropping: Since both cellular and Wi-Fi connections employ wireless channels, adversaries might be able to obtain the transmitted information directly from the open environment. Eavesdropping breaches the confidentiality aspect of security. Encryption and physical layer security techniques could be used as a protective mechanism [53].

• Hijacking: Hijacking here refers to attacks that adversaries take over a radio link. For example, radio links between drones and GCSs in our scenarios are all Wi-Fi connections. To launch hijacking attacks, adversaries could first use reauthentication management frames to disconnect the association between a drone and the corresponding GCS. Then the drone can be remotely controlled by adversaries via 802.11 protocols. There are several security techniques against reauthentication attacks. Effective detection algorithms could be applied and transmitted frames could be encrypted. Alternatively, a Wi-Fi access point (AP) could be hidden by disabling service set identifier (SSID) broadcasting and Wi-Fi UEs could be restricted to those with certain media access control (MAC) addresses.

• Spoofing: Adversaries can pretend to be some entity using false information. A typical spoofing attack to drones is GPS spoofing. By transmitting false GPS signals with higher power than the authentic ones, drones could be taken over by adversaries. To prevent GPS spoofing attacks, defence solutions such as jamming to-noise sense and multi-antenna defence could be employed [54], [55].

• Denial of Service (DoS): For a DoS attack, adversaries will send excessive requests to the server, which causes network congestion. As a result, the legitimate users will lose their service.

B. Physical Security

Besides cyber-attacks, adversaries could also launch physical attacks to drones. To laun ch physical attacks, adversaries first need to obtain access to drones. First, adversaries can access a drone on the ground or capture a flying drone. Second, adversaries can co ntrol drones by successfully launching cyber-attacks as introduced before [56].

• Low: Attackers aim to disassemble the captured drone to access its internal data. To defend such attacks, self-destruction mechanisms could be applied on drones, which will be enabled under pre-defined circumstances. However, self-destruction mechanism should only be triggered when necessary due to its strong side effects, loss of both data and drones.

• Medium: Attackers could access data through higher standard interfaces and also access the embedded system. In this case, information stored on the drone needs to be encrypted. However, encryption may only delay the time taken by adversaries to obtain their desired data.

• High: Adversaries are capable of launching advanced attacks such as side-channel attacks, fault injection attacks and software attacks to retrieve desired information from a drone. To deal with such attacks, superior cryptographic mechanisms and secure key management should be equipped by drones.

5.8 Lessons learned

1) UAV types and characteristics

one of the main drawbacks of the current UAV technology is still the presence of a significant trade-off between flying time, carried payload and associated costs. In the future, a joint optimization of these three important metrics is desirable. Although the research community is already working in this direction, it seems that several years are still required to close this gap.

2) Standardization

With the introduction of aerial users, future cellular communication systems must embrace also the third dimension, with BSs able to point towards the sky and maybe dedicate specific resources to this type of new users. For this reason, massive MIMO 5G technology is foreseen as a strong candidate for introducing the required 3D spatial communication flexibility, thus pushing for an increasing focus in the 3GPP standardization activity to enhance and complement massive MIMO with UAV-dedicated solutions.

3) Aerial base stations

We foresee benefits such as the avoidance of overprovisioned fixed network infrastructures to cope with hardly predictable data traffic peaks in time and the reduction of CAPEX and OPEX associated to site acquisition and maintenance or wiring. Current major concerns and research activities are focused on optimal placement and mobility of aerial BSs, power efficiency, recharging, and security.

4) Prototyping and field tests

Innovative approaches include those using futuristic autonomous solar-powered aircrafts, autonomous balloons, and self-powered aerial small cells. Although we find all these solutions extremely interesting and inventive, little can be said on their effectiveness in a wider cellular architecture with a mixture of aerial and ground BSs. Moreover, none of them have proven to be a cost-effective solution ready for wide adoption into an imminent product.

5) Regulation

Not only technical challenges are associated with UAV communications, but also the one related to privacy, public safety, administrative procedures, and licenses. Rules have been put in place in the majority of the countries in the world with the aim to control and limit the use of drones. However, we are still far from an unified view on a basic common set of rules to be adopted world-wide.

6) Security

With increasing number of drones flying in the sky, security becomes an extremely important requirement for drones not only to prevent actual falling and injuring people, but also to protect the data they are capturing and transferring to the ground network against possible hijacking.

5.9 Future research directions

A. UAV simulator:

Real experiments with UAVs are inherently difficult due to tough regulations and need for large open space. It would be useful to develop publicly available simulators for UAVs, which would allow researchers to accurately simulate different types of drones according to their hardware specifications and subject to location-specific obstacles. There exist sophisticated open license simulators for ground vehicles, such as SUMO. The extensions would allow UAV communications researchers to enjoy similar level of simulation support currently availed by researchers working on vehicular communications.

B. Advanced UAV Mobility Control Based on Image Processing and Deep Learning

With image processing and deep learning, UAVs can be programmed to identify the optimal hovering location or the optimal flying direction which would provide the best signal propagation between the UAV and the target ground BS (when UAV is acting as an aerial UE), or between the UAV and a ground UE (when the UAV is acting as an aerial BS), while considering real-life obstacles, such as buildings, rooftop cranes etc.

C. UAVs Antennas

Because of the drones' ability to move in any direction with different speed, a new antenna design for airborne communication is required to achieve high data rate. One alternative to have high data rate transmission between UAVs and ground base stations is to have a tracking antenna installed on UAVs

D. Aerial UE Identification

One of the main challenges in introducing aerial UEs in LTE and 5G is identifying that an aerial UE has the proper certification to connect to the cellular networks. To this end, 3GPP proposed both UE-based and network-based solutions to indicate that a UE is airborne. In the UE-based solutions, the UE can report information such as altitude, flight mode and etc. With network-based techniques, different characteristics of the UE such as the mobility history, handover, etc. can help in UE detection. As stated before, since having a LOS is more probable for aerial UEs, they experience different radio conditions and interference than ground users. Developing more advance and intelligent algorithms and utilizing various characteristics in future will lead in more precise aerial UE detection.

E. Overcoming the Issue of Physical Reliability of UAVs

By nature, UAV-BSs are expected to suffer from a physical reliability issue that does not concern existing terrestrial BSs. The physical reliability issues will eventually affect the quality of wireless service and the overall quality of experience for the users. The physical reliability issue therefore must be addressed adequately before UAVs are integrated into cellular systems.

F. Mobile Edge Computing With UAV-BSs

Cellular networks are moving towards a new paradigm called mobile edge computing where the BSs provide not only communications, but also computing services to mobile users. This paradigm essentially brings the so-called cloud services closer to the mobile users thus reducing the latency for many real-time compute-intensive applications, such as augmented reality, speech recognition, and so on [57]. Several issues arise when edge computing is supported through UAV-BSs. To support computing services, UAV-BSs will have to be fitted with significant computing platforms, such graphical processing units (GPUs), which will increase both payload and energy consumption of UAV. Energy optimizations for UAVs therefore will have to factor in the computing tasks as well to ensure that UAV batteries can last longer. Computing session continuity is another issue that will have to be considered by the UAV path planning optimizations as the mobility of UAV-BSs can cause serious disruptions to ongoing computing tasks for mobile users. Indeed, researchers have realized such challenges and started exploring edge computing challenges and opportunities for UAVs [58].

G. UAV-Supported Caching

Owing to UAV's inherent high mobility, aerial caching provides a superior solution to deliver the contents more efficient than traditional static caching by addressing the user mobility problem. The proactive deployment of UAV-supported caching improves users' QoS in a cloud radio access network (CRAN). The UAV-supported caching could achieve higher multimedia data throughput in IoT system. Major challenges for aerial caching include location optimization of UAVs, user-UAV association, the contents to cache at UAVs and efficient power control strategy.

6 Conclusions

The main objective of this deliverable is to provide AI-MEC system design specification and integration report. In section 2, 5G and mobile edge computing (MEC) network architectures are discussed. In section 3, potential AI and deep learning algorithms those are widely used in 5G and MEC networks are discussed such as artificial neural networks, reinforcement learning, super-resolution deep learning, deep reinforcement learning, and system design with Lyapunov optimization framework. In section 4, the use cases of AI methods for 5G and MEC networks are discussed. First of all, stabilized super-resolution control is discussed. After that, deep reinforcement learning based video caching for vehicular networks is presented. Third, multi-agent deep reinforcement learning based surveillance UAV coordination is discussed. Finally, application server relocation with Optimal Routing is described. In section 5, detailed AI-MEC system design specification and integration are presented. The discussions are all related to standards, characteristics, prototyping, and regularization.

Reference

[1] J. Choi, “Secure transmissions via compressive sensing in multicarrier systems,” IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1315–1319, 2016.

[2] 3GPP-23228 3GPP Technical Specification 23.228, Architecture enhancements for 5G System (5GS) to support network data analytics services, Release 16, 2019.

[3] PRI19-D21 PriMO-5G Deliverable D2.1, Initial design of MEC and Network Slice Manager, April 2019, https://primo-5g.eu/download/389/

[4] Ali Symeri, Application Server Mobility and 5G Core Network, MSc Thesis, KTH, 2019, Industrial supervisor: Ericsson AB, http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A1334568&dswid=6091

[5] https://linuxcontainers.org/lxd/introduction/

[6] https://www.criu.org/Main_Page

[7] PRI20-D12 PriMO-5G Deliverable D1.2, End-to-end PriMO-5G Network Architecture, June 2020, https://primo-5g.eu/download/609/

[8] https://5g-ppp.eu/wp-content/uploads/2019/07/5G-PPP-5G-Architecture-White-Paper_v3.0_PublicConsultation.pdf

[9] V. François-Lavet, "An introduction to deep reinforcement learning," arXiv preprint arXiv:1811.12560, 2018. (3.2)

[10] S. Park, D. Kim, and J. Kim, "Dynamic Decision-Making for Stabilized Deep Learning Software Platforms," Advances in Deep Learning. IntechOpen, 2020. (3.4)(4.1)

[11] M. Choi, et al, "Markov decision policies for dynamic video delivery in wireless caching networks," IEEE Transactions on Wireless Communications, vol. 18, no. 2, pp. 5705-5718, 2019. (4.2)

[12] A. Fotouhi, et al, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," IEEE Communications Surveys & Tutorials, vol. 21, no.4, 3417-3442, 2019. (5)

[13] L. Tang and G. Shao, “Drone remote sensing for forestry research and practices,” J. Forestry Res., vol. 26, no. 4, pp. 791–797, 2015.

[14] K. Nonami, F. Kendoul, S. Suzuki, W. Wang, and D. Nakazawa, Autonomous Flying Robots. Tokyo, Japan: Springer-Verlag, 2010.

[15] A. Al-Hourani, S. Kandeepan, and S. Lardner, “Optimal LAP altitude for maximum coverage,” IEEE Wireless Commun. Lett., vol. 3, no. 6, pp. 569–572, Dec. 2014.

[16] M. Ding, P. Wang, D. López-Pérez, G. Mao, and Z. Lin, “Performance impact of LoS and NLoS transmissions in dense cellular networks,” IEEE Trans. Wireless Commun., vol. 15, no. 3, pp. 2365–2380, Mar. 2016.

[17] T. Ding et al., “Uplink performance analysis of dense cellular networks with LoS and NLoS transmissions,” IEEE Trans. Wireless Commun., vol. 16, no. 4, pp. 2601–2613, Apr. 2017.

[18] M. Ding and D. López-Pérez, “Please lower small cell antenna heights in 5G,” in Proc. IEEE Glob. Commun. Conf. (GLOBECOM), Dec. 2016, pp. 1–6.

[19] A. Fotouhi, M. Ding, and M. Hassan, “Understanding autonomous drone maneuverability for Internet of Things applications,” in Proc. WoWMoM Workshop Internet Things, Jun. 2017, pp. 1–6.

[20] “Study on channel model for frequencies from 0.5 to 100 GHz (release 14),” 3GPP, Sophia Antipolis, France, Rep. 38.901, May 2017.

[21] J. Yang et al., “Optimal base station antenna downtilt in downlink cellular networks,” IEEE Trans. Wireless Commun., vol. 18, no. 3, pp. 1779–1791, Mar. 2019.

[22] “Technical specification group radio access network; Study on enhanced LTE support for aerial vehicles (release 15),” 3GPP, Sophia Antipolis, France, Rep. 36.777, Dec. 2017.

[23] G. Geraci, A. Garcia-Rodriguez, L. Galati Giordano, D. López-Pérez, and E. Bjoernson, “Supporting UAV cellular communications through massive MIMO,” in Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops), May 2018, pp. 1–6.

[24] R. Amorim et al., “Radio channel modeling for UAV communication over cellular networks,” IEEE Wireless Commun. Lett., vol. 6, no. 4, pp. 514–517, Aug. 2017.

[25] H. Li et al., “Performance analysis of the access link of drone base station networks with LoS/NLoS transmissions,” in Proc. INISCOM, Aug. 2018, pp. 1–7.

[26] Ericsson, New WID on Enhanced Support for Aerial Vehicles, document RP-172826 RAN#78, 3GPP, Sophia Antipolis, France, Dec. 2017.

[27] T. Ding et al., “Uplink performance analysis of dense cellular networks with LoS and NLoS transmissions,” IEEE Trans. Wireless Commun., vol. 16, no. 4, pp. 2601–2613, Apr. 2017.

[28] Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol Specification, 3GPP Standard 36.331, Sep. 2018.

[29] Functional Architecture for Unmanned Aerial Vehicles and Unmanned Aerial Vehicle Controllers Using IMT-2020 Networks, ITU-T, Geneva, Switzerland, 2017.

[30] “Use cases and spectrum considerations for UAS (unmanned aircraft systems),” ETSI, Sophia Antipolis, France, Rep. 103 373, Feb. 2018.

[31] Next Generation Protocols (NGP); Scenarios Definitions (v1.1.1), ETSI Standard GS NGP 001, Nov. 2016.

[32] Standard for Consumer Drones: Taxonomy and Definitions, IEEE Standard P2025.1, Oct. 2015.

[33] Standard for Consumer Drones: Privacy and Security, IEEE Standard P2025.1, Oct. 2015.

[34] D. López-Pérez et al., “On the downlink performance of UAV communications in dense cellular networks,” in Proc. IEEE Glob. Commun. Conf. (GLOBECOM), Dec. 2018, pp. 1–7.

[35] C. Liu et al., “Performance analysis for practical unmanned aerial vehicle networks with LoS/NLoS transmissions,” in Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops), May 2018, pp. 1–6.

[36] Eurecom. ERC Perfume Project. Accessed: Apr. 2, 2019. [Online]. Available: http://www.ercperfume.org/about/

[37] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Drone small cells in the clouds: Design, deployment and performance analysis,” in Proc. IEEE GLOBECOM, Dec. 2015, pp. 1–6.

[38] K. Li et al., “Energy-efficient cooperative relaying for unmanned aerial vehicles,” IEEE Trans. Mobile Comput., vol. 15, no. 6, pp. 1377–1386, Jun. 2016.

[39] L. D. P. Pugliese, F. Guerriero, D. Zorbas, and T. Razafindralambo, “Modelling the mobile target covering problem using flying drones,” Optim. Lett., vol. 10, no. 5, pp. 1021–1052, 2016.

[40] D. Zorbas, L. D. P. Pugliese, T. Razafindralambo, and F. Guerriero, “Optimal drone placement and cost-efficient target coverage,” J. Netw. Comput. Appl., vol. 75, pp. 16–31, Nov. 2016.

[41] V. Sharma, K. Srinivasan, H.-C. Chao, K.-L. Hua, and W.-H. Cheng, “Intelligent deployment of UAVs in 5G heterogeneous communication environment for improved coverage,” J. Netw. Comput. Appl., vol. 85, pp. 94–105, May 2016.

[42] “Study on integrated access and backhaul,” 3GPP, Sophia Antipolis, France, Rep. TR 38.874, May 2018.

[43] Facebook. Building Communications Networks in the Stratosphere. Accessed: Apr. 2, 2019. [Online]. Available: https:// code.facebook.com/posts/993520160679028/building-communications -networks-in-the-stratosphere/

[44] Google X. Balloon-Powered Internet for Everyone. Accessed: Apr. 2, 2019. [Online]. Available: https://www.google.com/intl/enUS/loon/

[45] Nokia. (Oct. 2016). F-Cell Technology From Nokia Bell Labs Revolutionizes Small Cell Deployment by Cutting Wires, Costs and Time. [Online]. Available: https://www.nokia.com/news/releases/ 2016/10/03/f-cell-technology-from-nokia-bell-labs-revolutionizes-smal l-cell-deployment-by-cutting-wires-costs-and-time/

[46] J. Chen and D. Gesbert, “Optimal positioning of flying relays for wireless networks: A LOS map approach,” in Proc. IEEE Int. Conf. Commun. (ICC), Paris, France, May 2017, pp. 1–6.

[47] Huawei. Connected Aerial Vehicle Live. Accessed: May 20, 2018. [Online]. Available: http://www.huawei.com/en/industry-insig hts/innovation/xlabs/use-cases/mbbf2017-connected-aerial-vehicle-live

[48] B. Liu, M. Ding, T. Zhu, Y. Xiang, and W. Zhou, “Using adversarial noises to protect privacy in deep learning era,” in Proc. IEEE Glob. Commun. Conf. (GLOBECOM), Abu Dhabi, UAE, Dec. 2018, pp. 1–6

[49] R. L. Finn, D. Wright, L. Jacques, and P. De Hert, “Study on privacy, data protection and ethical risks in civil remotely piloted aircraft systems operations: Final report,” Eur. Comission, Brussels, Belgium, Rep. 39, Feb. 2015.

[50] B. Canis, Unmanned Aircraft Systems (UAS): Commercial Outlook for a New Industry, Congressional Res. Service, Washington, DC, USA, 2015

[51] C. Stöcker, R. Bennett, F. Nex, M. Gerke, and J. Zevenbergen, “Review of the current state of UAV regulations,” Remote Sens., vol. 9, no. 5, p. 459, 2017

[52] I. Jawhar, N. Mohamed, J. Al-Jaroodi, D. P. Agrawal, and S. Zhang, “Communication and networking of UAV-based systems: Classification and associated architectures,” J. Netw. Comput. Appl., vol. 84, pp. 93–108, Apr. 2017

[53] Y. Zeng, R. Zhang, and T. J. Lim, “Wireless communications with unmanned aerial vehicles: Opportunities and challenges,” IEEE Commun. Mag., vol. 54, no. 5, pp. 36–42, May 2016.

[54] D. Borio and C. Gioia, “Real-time jamming detection using the sumof-squares paradigm,” in Proc. IEEE Int. Conf. Localization GNSS (ICL-GNSS), Gothenburg, Sweden, 2015, pp. 1–6.

[55] J. Magiera and R. Katulski, “Detection and mitigation of GPS spoofing based on antenna array processing,” J. Appl. Res. Technol., vol. 13, no. 1, pp. 45–57, 2015.

[56] K. Namuduri, S. Chaumette, J. H. Kim, and J. P. G. Sterbenz, UAV Networks and Communications. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[57] E. Ahmed et al., “Bringing computation closer toward the user network: Is edge computing the solution?” IEEE Commun. Mag., vol. 55, no. 11, pp. 138–144, Nov. 2017.

[58] S. Jeong, O. Simeone, and J. Kang, “Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning,” IEEE Trans. Veh. Technol., vol. 67, no. 3, pp. 2049–2063, Mar. 2018.

[59] Krenker A, Bešter J, Kos A (2011). Introduction to the artificial neural networks. In: Suzuki K (ed), Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech, pp. 1–18.

[60] Gurney, K. (1997). An Introduction to Neural Networks, Routledge, ISBN 1-85728-673-1 London

[61] Kröse B.; Smagt P. (1996). An Introduction to Neural Networks, The University of Amsterdam, Amsterdam.

[62] Pavešić N. (2000). Razpoznavanje vzorcev: uvod v analizo in razumevanje vidnih in slušnih signalov, Fakulteta za elektrotehniko, ISBN 961-6210-81-5, Ljubljana

[63] Rojas R. (1996). Neural Networks: A Systematic Introduction, Springer, ISBN 3-540-60505-3, Germany.

[64] Bellman, R. 1957. “Dynamic Programming”.

[65] Barto, A. G., R. S. Sutton, and C. W. Anderson. 1983. “Neuronlike adaptive elements that can solve difficult learning control problems”. IEEE transactions on systems, man, and cybernetics. (5): 834–846

[66] Sutton, R. S. and A. G. Barto. 2017. Reinforcement Learning: An Introduction (2nd Edition, in progress). MIT Press.

[67] Norris, J. R. 1998. Markov chains. No. 2. Cambridge university press. O’Donoghue, B., R. Munos, K. Kavukcuoglu, and V. Mnih. 2016. “PGQ: Combining policy gradient and Q-learning”. arXiv preprint arXiv:1611.01626.

[68] Bellman, R. 1957. “A Markovian decision process”. Journal of Mathematics and Mechanics: 679–684.

[69] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare and Joelle Pineau (2018), “An Introduction to Deep Reinforcement Learning”, Foundations and Trends in Machine Learning: Vol. 11, No. 3-4. DOI: 10.1561/2200000071.

[70] Bertsekas, D. P., D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas. 1995. Dynamic programming and optimal control. Vol. 1. No. 2. Athena scientific Belmont, MA.

[71] Yang, C.Y., Ma, C., Yang, M.H.: Single-image super-resolution: A benchmark. In: European Conference on Computer Vision, pp.372–386 (2014)

[72] Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: IEEE International Conference on Computer Vision. pp.349–356 (2009)

[73] Kim, K.I., Kwon, Y.: Single-image super-resolution using sparse regression and natural image prior. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(6), 1127–1133 (2010)

[74] Timofte, R., De Smet, V., Van Gool, L.: Anchored neighborhood regression for fast example-based super-resolution. In: IEEE International Conference on Computer Vision. pp. 1920–1927 (2013)

[75] Yang, J., Lin, Z., Cohen, S.: Fast image super-resolution based on in-place example regression. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1059–1066 (2013)

[76] Freedman, G., Fattal, R.: Image and video upscaling from local self-examples. ACM Transactions on Graphics 30(2), 12 (2011)

[77] Yang, C.Y., Huang, J.B., Yang, M.H.: Exploiting self-similarities for single frame super-resolution. In: IEEE Asian Conference on Computer Vision, pp. 497–510 (2010)

[78] Bevilacqua, M., Roumy, A., Guillemot, C., Morel, M.L.A.: Lowcomplexity single-image super-resolution based on nonnegative neighbor embedding. In: British Machine Vision Conference (2012)

[79] Chang, H., Yeung, D.Y., Xiong, Y.: Super-resolution through neighbor embedding. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)

[80] Dai, D., Timofte, R., Van Gool, L.: Jointly optimized regressors for image super-resolution. In: Eurographics. vol. 7, p. 8 (2015)

[81] Freeman, W.T., Pasztor, E.C., Carmichael, O.T.: Learning lowlevel vision. International Journal of Computer Vision 40(1), 25–47 (2000)

[82] Schulter, S., Leistner, C., Bischof, H.: Fast and accurate image upscaling with super-resolution forests. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 3791–3799 (2015)

[83] Yang, J., Wang, Z., Lin, Z., Cohen, S., Huang, T.: Coupled dictionary training for image super-resolution. IEEE Transactions on Image Processing 21(8), 3467–3478 (2012)

[84] Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution as sparse representation of raw image patches. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8 (2008)

[85] Yang, J., Wright, J., Huang, T.S., Ma, Y.: Image super-resolution via sparse representation. IEEE Transactions on Image Processing 19(11), 2861–2873 (2010)

[86] Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: Curves and Surfaces, pp. 711–730 (2012)

[87] Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-based superresolution. Computer Graphics and Applications 22(2), 56–65 (2002)

[88] Timofte, R., De Smet, V., Van Gool, L.: A+: Adjusted anchored neighborhood regression for fast super-resolution. In: IEEE Asian Conference on Computer Vision (2014)

[89] Dai, S., Han, M., Xu, W., Wu, Y., Gong, Y., Katsaggelos, A.K.: Softcuts: a soft edge smoothness prior for color image superresolution. IEEE Transactions on Image Processing 18(5), 969–981 (2009)

[90] Buyya R, Broberg J, Goscinski A, et al. "Cloud Computing Principles and Paradigms," Cloud Computing, 2011.

[91] Pfaff B, Pettit J, Koponen T, et al. "The design and implementation of open vSwitch," Networked Systems Design and Implementation, 2015.