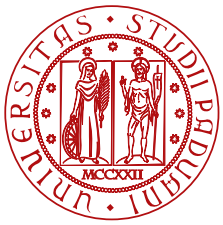

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA MAGISTRALE IN
ICT FOR INTERNET AND MULTIMEDIA - INGEGNERIA PER LE COMUNICAZIONI MULTIMEDIALI E INTERNET

Design, Simulation and Validation of Data Offloading Techniques to Support Realistic Vehicle-to-Satellite Communication

Relatore
Prof. Marco Giordani

Laureando
Alexander Bonora

Correlatore
Dott. Traspadini Alessandro

ANNO ACCADEMICO 2022-2023

Data di laurea 12/12/2023

Abstract

Data offloading in vehicular networks is a key research area in autonomous driving. As Intelligent Transportation Systems (ITS) continue to expand, vehicles face the challenge of handling more and more data. However, vehicles have limited computational capacity, so some data needs to be transferred to other computing units for processing.
In the context of sixth generation (6G) wireless systems, the research community is exploring the concept of non-terrestrial networks (NTN), where satellites can serve as space edge computing nodes to aggregate, store, and process data from ground vehicles, thus to relieve the strain on terrestrial networks.
In particular in this thesis we will analyze video data transmissions from vehicles to Low-Earth-Orbit (LEO) satellites using wireless communication. Specifically, the long distance between LEO satellites and ground vehicles, the resulting severe path loss, and the limited visibility time of satellites introduce complexity for data offloading, and may be source of transmission delays.
The main goal of this thesis is to create a simulator that can realistically simulate communication between a ground vehicle and multiple satellites and evaluate the performance, measured in terms of the end-to-the delay for data offloading and processing. Simulation results are given considering the real Starlink orbital parameters, and as a function of several parameters including the type of application and the density of the constellation. Finally, we provide guidelines on the optimal setup and transmission parameters to reduce data offloading delays.

Sommario

L'offloading dei dati nelle reti veicolari è un'area di ricerca fondamentale per la guida autonoma. Con la continua espansione dei sistemi di trasporto intelligenti (ITS), i veicoli devono affrontare la sfida di gestire un numero sempre maggiore di dati. Tuttavia, i veicoli hanno una capacità di calcolo limitata, quindi alcuni dati devono essere trasferiti ad altre fonti.
Nel contesto dei sistemi wireless di sesta generazione (6G), la comunità di ricerca sta esplorando il concetto di reti non terrestri (NTN), in cui i satelliti possono fungere da nodi di calcolo nello spazio per aggregare, memorizzare ed elaborare i dati provenienti dai veicoli terrestri, alleggerendo così la pressione sulle reti terrestri.
In questa tesi analizzeremo i trasferimenti di dati video da veicoli a satelliti in orbita bassa (LEO) utilizzando la comunicazione wireless. In particolare, la lunga distanza tra i satelliti LEO e i veicoli terrestri, il conseguente aumento della path loss e il limitato tempo di visibilità dei satelliti introducono complessità per l'offloading dei dati e possono essere fonte di ritardi nella trasmissione.
L'obiettivo principale di questa tesi è quello di creare un simulatore in grado di riprodurre realisticamente la comunicazione tra un veicolo di terra (GV) e più satelliti e di valutarne le prestazioni, misurate in termini di ritardo end-to-end per il trasferimento e l'elaborazione dei dati. I risultati delle simulazioni sono presentati considerando i veri parametri orbitali di Starlink e una funzione di diversi parametri, tra cui il tipo di antenna utilizzata dal trasmettitore e la densità della costellazione. Inoltre, la tesi offre indicazioni riguardo alla configurazione ottimale e ai parametri di trasmissione per minimizzare la latenza durante il trasferimento dei dati.

Contents

List of Figures

Chapter 1

Introduction

1.1 Intelligent Transportation Systems (ITS)

In recent years, there has been a rapid increase in the number of vehicles in urban areas, leading to a rise in traffic incidents underlining the need of intelligent transportation systems. To address this issue and enhance driving safety while improving traffic conditions, the concept of the Internet of Vehicles (IoV) has been gaining prominence [1]. Within the IoV framework, the focus of recent research has been on vehicle-to-everything (V2X) communication, to enable vehicles to exchange useful data about road conditions and the surrounding environment, and achieve cooperative perception [2].

Specifically, V2X communication leverages Long-Term Evolution (LTE) and 5G technologies, enabling the efficient transmission of large volumes of data with manageable latency. Technically, V2X communication includes various aspects, such as vehicle-to-vehicle (V2V) communication, vehicle-to-pedestrian (V2P) communication, vehicle-to-infrastructure (V2I) communication, and vehicle-to-network (V2N) communication [3]. These different modes of communication enable vehicles, pedestrians, and infrastructure elements to gather information about their surroundings and exchange this information with nearby data collectors [4].

In this context, the ability of intelligent vehicles to perceive and understand the environment is paramount to guarantee safety. Equipped with an array of sensors, these vehicles gather crucial information on obstacles, street signs, other vehicles, and pedestrians, directly influencing safety and efficiency. Among the common sensors used are Light Detection and Ranging (LIDAR) and cameras. LIDAR utilizes laser pulses to create a detailed 3D map, excelling in providing precise distance and depth information, which is particularly critical for object detection and avoidance. Despite limitations in cost and density, LIDAR's active illumination sensor ensures its effectiveness day and night [5]. Cameras, being cost-effective and adept at capturing high-resolution images, excel in object recognition, lane detection, and identifying colors and

textures. Recent developments in Neural Networks technologies enhance mono-camera setups to generate depth information through inferencing. Stereo vision, utilizing two slightly spaced cameras, produces a 3D perception by analyzing pixel disparity. Combining LIDAR and cameras compensates for their respective strengths and limitations, offering a comprehensive view of the surroundings and creating redundancy for safety[6].

Notably, V2X communication can be leveraged to transmit data to other external infrastructures, e.g., edge servers, with rich computing resources to process data faster [7], [8].

Specifically, data offloading, is the process of delegating the execution of computer applications from a device to remote cloud servers. This offloading can be either full or partial. When an application runs entirely on remote servers in the cloud, it is called full offloading. Conversely, if only a portion of the application runs in the cloud while the rest operates on the mobile device, it is known as partial offloading.

This approach permits faster data processing, since edge servers are generally equipped with more powerful computational units, and can reduce the power consumption onboard battery-powered end devices.

Vehicular offloading emerges as a concept within the Vehicle Edge Computing [9]. At its core, it involves the transfer of resource-intensive computing tasks, e.g., object detection and recognition, from the vehicles to a remote computing center. This approach becomes especially beneficial for ground vehicles with constrained computing and storage capabilities [10]. Leveraging the substantial processing power of cloud computers, VEC aims to minimize data processing latency and, in certain instances, enhance the overall computing capacity of vehicles [11]. However, it is essential to acknowledge the inherent trade-off associated with the VEC paradigm. On one hand, VEC reduces processing delay by offloading resource-intensive tasks to powerful remote computing centers. On the other hand, this introduces a non-negligible delay for both the offloading of data from the vehicle to the remote center and the subsequent return of processed data back to the vehicle. This trade-off becomes a critical aspect to consider when evaluating the feasibility and effectiveness of implementing VEC.

So, the question is whether the reduction in processing time is worth the extra delay caused by sending data back and forth. This trade-off is a crucial factor to consider when deciding if and how to implement VEC [12].

The thesis explores this problem by providing a comprehensive analysis of the trade-off in VEC. By examining the impact of offloading and latency on the overall system performance, the study seeks to offer insights into the conditions under which VEC proves advantageous and the scenarios where its benefits may be compromised.

1.2 Exploring Non-Terrestrial Networks (NTN)

NTNs play an important role in the evolving landscape of 6th generation (6G) wireless networks, , and especially to support IoT and next-generation vehicular networks. [13].

Heading towards 2025, an anticipated 75 billion IoT devices globally are poised to generate a market value of approximately 11.1 trillion USD. This surge in IoT devices, presents a formidable challenge to conventional terrestrial networks. This challenge is particularly pronounced in remote and emergency scenarios where network infrastructure may be absent or unavailable. [14].

To address these challenges, researchers are exploring NTNs as a solution. NTNs employ Unmanned Aerial Vehicles (UAVs), High Altitude Platforms (HAPs), and satellites as aerial/space gateways to aggregate, process, and relay IoT traffic which is highlited in Figure 1.1.

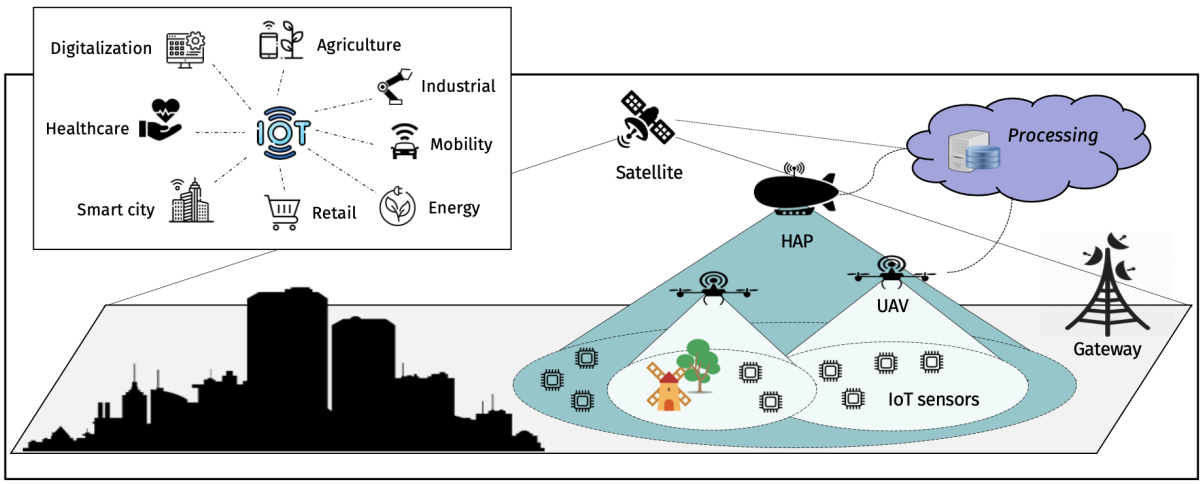

Figure 1.1: The IoT-NTN scenario and relative use cases [14].

Within the context of NTN, and as far as satellite communication is concerned, it is essential to explore the historical roots and possible applications for this rather new technology.

Satellite communication has witnessed remarkable development in recent years, with advancements in microelectronics transforming satellite systems, rendering them more flexible and reprogrammable.

Satellites operate in different orbits, each serving specific purposes. The three main types are Geostationary Orbit (GEO), Low Earth Orbit (LEO), and Medium Earth Orbit (MEO). Understanding their orbits and characteristics is important for comprehending their roles in satellite communication. Figure 1.2 provides a visual representation of the distances of satellites in GEO, MEO, and LEO concerning Earth.

GEO satellites, positioned at 36,000 km above the equator, have an orbital speed that matches the Earth's rotation, so they remain in a fixed position relative to every point on Earth.

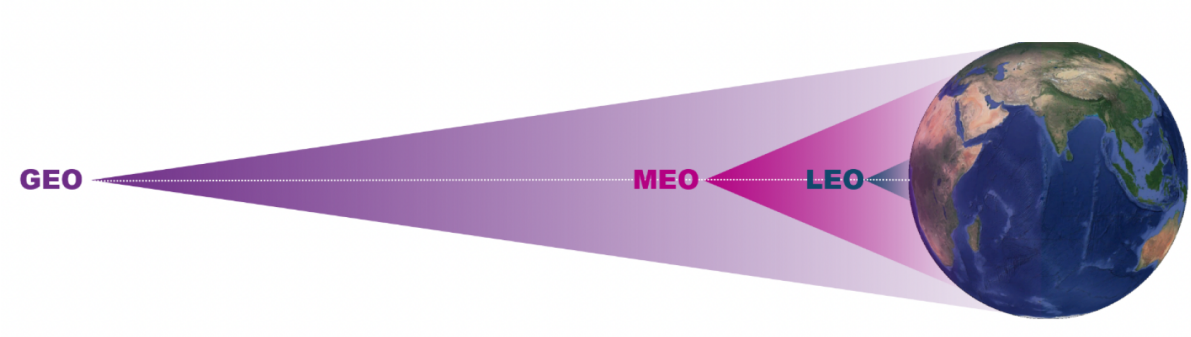

Figure 1.2: Satellite technologies and distances from Earth [15].

These satellites are ideal for telecommunications and weather monitoring, requiring fewer nodes for global coverage. Notably, the European Data Relay System (EDRS) by the European Space Agency (ESA) utilizes GEO satellites to ensure continuous connectivity for transmitting and receiving data [16].

Non-Geostationary Orbit (NGSO) encompasses LEO and MEO. LEO, ranging from 160 km to less than 1000 km in altitude, is closer to Earth's surface. LEO's flexibility allows satellites to follow various paths, commonly used for satellite imaging and hosting structures like the International Space Station (ISS). LEO satellites, moving at a speed of approximately 7.8 km per second, circle Earth in about 90 minutes. While not ideal for standalone telecommunications due to their rapid movement, LEO satellites often form constellations to provide continuous coverage. Similarly, MEO, between LEO and GEO, is commonly employed by navigation satellites like the European Galileo system, supporting navigation across Europe. The rise in NGSO constellations, including both LEO and MEO satellites, is particularly notable for broadband connectivity, offering continuous coverage and supporting various applications, from tracking aircraft to providing smartphone directions[17]. Moreover, the communication delay to/from a LEO satellites is generally small, which can support the requirements of many 5G and 6G applications.

A satellite communication system comprises ground and space terminals, along with essential equipment for communication protocols. The space terminal manages communication routing, access control, and spot-beam systems, offering adaptability and multi-beam capabilities. On the ground, terminals provide gateways with fiber-optic connectivity, serving end-users. Centralized facilities, including monitoring centers (HUBs), configure networks, allocate resources, and ensure user privacy [18].

The modern trend in satellite design favors smaller, lightweight satellites with advanced signal processing capabilities [15] such as LEO satellites, as described in the next section.

These innovations bring about new opportunities, such as seamless integration with 5G communication systems. In the following sections, we will delve into the specifics of Low Earth

Orbit (LEO) satellites.

1.2.1 LEO Satellites in NTN Scenarios

LEO satellites operate at very low altitude, which offers several advantages, including lower signal propagation delays and reduced path losses, making LEO satellites a compelling choice for NTN applications [19].

Especially in the NTN scenario, LEO satellite are able to reduce latency, a critical factor for supporting applications with stringent delay requirements, such as real-time communication, autonomous vehicles, and mission-critical services. In contrast to geostationary satellites, which are positioned at much higher altitudes and introduce higher signal propagation delays, LEO satellites offer a more responsive and efficient communication solution.

However, the support of LEO satellite communication presents several technical challenge, including those associated with the severe path loss, the additional atmospheric channel attenuations, the high Doppler effect, and the large propagation delay experienced in this scenario [20].

In confirmation of this, satellite network operators (SNOs) lead the charge in revolutionizing broadband communications. This rapid growth of SNOs has brought about global connectivity and significantly contributed to bridging the digital divide, especially in remote and underserved areas. SNOs like Starlink, ViaSat, and HughesNet have invested heavily in satellite technologies, moving beyond traditional geosynchronous satellites to innovative LEO.

Notably, in this study we focus on Starlink, a prominent player in the satellite communication arena [21]. With 4,519 Starlink satellites currently in orbit as of July 2023, 4,487 of which are operational, SpaceX plans to launch an additional approximately 12,000 satellites by 2027. Their ambitious vision extends to a megaconstellation comprising as many as 42,000 satellites in the future, a prospect that promises to significantly transform the landscape of satellite communication [22]. Furthermore, the forthcoming V2 satellites within the Starlink constellation are expected to enhance data capacity even further compared to their predecessors. These advanced satellites will offer the capability to deliver services directly to cellular devices, further extending the reach of Starlink's impact [23]. Notably, Starlink satellites are positioned in orbit at an altitude of approximately 550 kilometers above the Earth's surface.

1.3 Synergy of V2X, VEC, and NTN in Shaping ITS

In the rapidly evolving landscape of smart transportation systems, the convergence of V2X communication, VEC, and NTN holds immense potential for shaping the future of intelligent transportation.

Specifically, VEC allows intelligent vehicles to offload some computational tasks (e.g., to perform object detection) to more powerful external computing nodes to achieve faster data processing compared to onboard processing [24]. In this context, VEC servers can be hosted at the NTN nodes, which allows edge processing even in the absence of terrestrial roadside infrastructures, for example in rural/remote areas or in case of emergency. In this context, LEO satellites, with their reduced altitudes and lower signal propagation delays, play a significant role to support NTN-assisted VEC operations [25].

The seamless integration of V2X communication, VEC, and NTN in the context of intelligent transportation sets the stage for a transformative paradigm. However, within this landscape, a specific challenge arises—the efficient offloading of data from vehicles to LEO satellites for real-time object detection [26].

In particular, whether and how LEO-assisted VEC can be realized is not obvious. In particular, challenges such as the non-negligible propagation delay for data offloading, the impact of the dynamic movement of the satellite constellation, the severe degrees of attenuation introduced by the satellite channel may complicate VEC operations, and call for accurate models and architectural designs.

To this aim, in this thesis we formalize comprehensive analysis of the interplay between vehicles and LEO satellites, considering factors such as signal propagation, visibility intervals, and the overall reliability of the communication channel. The goal is to gain insights into the challenges associated with VEC, and to develop strategies that enhance the efficiency of data transmission, ultimately contributing to the broader objective of advancing intelligent transportation systems. Specifically, we evaluate the latency of LEO-assisted VEC offloading, and provide guidelines on the optimal configuration of parameters to support real-time data processing.

1.3.1 Structure of the Thesis

The primary objective of this research is to thoroughly characterize the latency in satellite communication to/from ground vehicles as far as data offloading is concerned. This exploration encompasses various dimensions, including the analysis of satellite constellations, antenna architectures, computational capabilities of satellites, and transmission components.

The methodology adopted involves meticulous simulations to explore the feasibility of real-time communication. To do so, we develop a new simulator that implements VEC on satellites, and featuring the channel model for satellite communication based on 3GPP specifications [27]. The structure of the satellite orbits and the dynamics of satellites are simulated based on real Starlink data [28] [29]. These tools ensure a comprehensive analysis of the impact of antenna gains, packet size, packet inter-arrival time, and server computation capacity on the overall communication scenario.

Specifically, this thesis is structured as follows.

Chapter 2 addresses the problem formulation concerning satellite-based data offloading in VEC scenarios, discussing the existing simulators and possible extentions to accommodate the new system model. The core components of the system model, including TLE data description, 3D structure model, channel model, delay model, and queuing model, are defined, laying the foundation for the development and implementation of the simulator to evaluate the performance of the proposed satellite-based data offloading system.

Chapter 3 provides an overview of the structure of the simulator and its main components, such as Vehicle Object, Satellite Object, and TLE parsing library. It further explains the 3D structure analysis and channel model components, essential for accurate simulations. The client-server structure in VEC scenarios is detailed, emphasizing its importance, and the chapter concludes by guiding the reader on how to build a scenario, outlining the necessary objects, design considerations, and the workflow of scenario building.

Chapter 4 offers a general explanation of the parameters considered in the simulations,with a focus on the vehicle antenna parameters, considering both VSAT and Reflectarray antennas. We also introduce SELMA, a new synthetic dataset for autonomous cars that is introduced to model the VEC application, in particular the packet size and the arrival rate of automotive data during offloading. The chapter explores the impact of elevation angle variation in satellite communication scenarios, setting the stage for the subsequent analysis.

Chapter 5 provides simulation results to evaluate the feasibility and performance of LEO-assisted VEC operations. Simulation results are mainly given in terms of delay, and as a function of several parameters including the antenna configuration, the size and inter-arrival rate of data, the size of the satellite constellation, and the computational capacity of the LEO satellite servers. The chapter concludes with an exploration of the real-time feasibility of the proposed system model, providing valuable findings for further discussion.

Chapter 6 concludes the thesis summarizing the main findings and contributions, and providing suggestions for future work. We demonstrate that the delay improves considering a VSAT antenna rather than a Reflectarray antenna, and applying compression to data before tranmission. For example, we show that the average delay decreases from around 200 ms to only 2.5 ms when the packet size goes from 20 to 5 Mb. Also, the performance improves reducing the rate at which data is transmitted, and considering more powerful computing servers at the satellites. We also perform simulations is dynamic scenarios, therefore considering the impact of the satellite mobility and visibility on the delay.

Chapter 2

Satellite-Based Data Offloading System Model in VEC Scenarios

2.1 Problem Formulation

In addressing the challenge of transferring data from a ground vehicle (GV) to a Low Earth Orbit (LEO) satellite for real-time object detection, it is crucial to understand the reasons for offloading tasks to a server. Vehicles require offloading to enhance the efficiency and safety of 3D flexible coverage in Vehicle-to-Everything (V2X) networks. The success of these networks relies on massive data from on-board sensors, necessitating extensive computational resources. VEC tackles this challenge by offloading resource-intensive tasks to more powerful edge/distributed servers.

When considering cloud and edge computing, the focus is on the unsuitability of cloud computing for this purpose. Cloud computing relies on centralized servers located at a distance from the data source, leading to increased latency and packet round-trip time (RTT) delay. In contrast, edge computing involves processing data closer to the data source, reducing latency. However, ground edge servers still face limitations compared to LEO satellites in terms of computational capabilities.

LEO satellites offer advantages over ground edge servers in terms of global coverage, allowing data offloading from any location on Earth's surface. The system, from frame generation to reception of the processed packet, involves capturing video data on the GV, transmitting it to the LEO satellite, and utilizing the satellite's superior computational capabilities for real-time object detection. The dynamic movements of the satellite constellation, viability of satellite communication, and evaluation of channel conditions in the vehicle-to-satellite link are critical steps in achieving instantaneous feedback on the GV's behavior. This approach highlights the distinct differences compared to a ground edge server, emphasizing the efficiency and feasibility

of leveraging LEO satellites for computational offloading in real-time applications.

In this scenario, we aim to investigate the key parameters affecting the feasibility of computational offloading to the satellite, with the goal of obtaining instantaneous feedback on the GV's behavior. Our primary focus is on reducing packet RTT delay. This involves designing and analyzing a communication system that enables a vehicle, situated anywhere on Earth's surface, to offload data to a satellite. This task includes monitoring the dynamic movements of the satellite constellation, assessing the viability of satellite communication, and evaluating the channel conditions and communication delays in the vehicle-to-satellite link.

2.2 Simulator Representation

In our simulator, the creation of scenarios is pivotal for gaining meaningful insights into VEC systems. Key components, such as Vehicles and Satellites, are fundamental in defining the characteristics and functionalities of the simulation. Additionally, classes like appClient and appServer manage application behaviors on the client and server sides, enriching the simulation experience.

The design philosophy of the simulator revolves around leveraging real-world geospatial data to ensure that scenarios closely mirror actual VEC environments. Latitude and longitude coordinates are employed for precise positioning of vehicles and satellites within the simulation. The simulator places a strong emphasis on comprehensive data collection, recording communication channels, signal quality, elevation angles, line-of-sight probability, distances, and various delays to assess VEC system performance. Scenario results are stored in CSV files, facilitating easy analysis of simulation outcomes.

The simulator was extended to include multiple satellites, mirroring real-world VEC scenarios. The specifications of the Starlink constellation were used as a reference for these satellites, enhancing the realism of the simulation.

Incorporating LEO Satellite Mobility is another crucial aspect, ensuring an accurate representation of the dynamic nature of LEO satellites. Parameters like angular mobility were integrated into the simulator, recognizing their significant impact on signal propagation and coverage.

A fundamental extension to the simulator is the realistic transmission of sensor data, based on the SELMA dataset [30]. This ensures the accurate portrayal of variables influencing the application data rate of the vehicle, such as the number of camera sensors (ranging from 1 to 4), the image size (based on SELMA), and a fixed frame rate (30 fps).

The workflow of scenario building involves several key steps. Initialization includes defining the number of satellites, their parameters, and the range within which vehicles can connect

to satellites. Generating vehicles produces a list of vehicles with specific parameters influencing communication capabilities. Simulation parameters, such as start time, time step, and the total number of simulation iterations, are specified to control the duration and granularity of the simulation. Channel and communication parameters, including packet arrival rate, packet size, computational load, atmospheric conditions, and satellite constellation characteristics, are set. Scenario implementation involves creating an instance of the scenario class corresponding to the specific scenario to be simulated.

During simulation, the simulator iterates over the specified time range, evaluating communication between vehicles and satellites and collecting essential data for analysis. Satellites for analysis are chosen based on a specified range parameter. Simulation results, including time, vehicle and satellite positions, distances, path loss, Signal-to-Noise Ratio (SNR) values, elevation angles, line-of-sight probability, delays, and more, are stored in CSV files for later analysis.

The scenario-building process is crucial for generating realistic VEC simulations and collecting data for subsequent analysis. In the "Results Analysis" section of our thesis, we delve into the outcomes of these simulations, providing valuable insights into VEC technology and strategies. By adhering to these design considerations and workflows, our simulator contributes to the advancement of VEC systems.

2.3 System Model

2.3.1 Orbit Model

The central focus of this thesis revolved around the implementation of a 3D structure for satellite and vehicle communication. The implementation of the model presented multiple challenges. One of the key difficulties arose from the need to calculate real-time geometry for a substantial number of satellites, which posed significant computational demands. To address this issue and ensure the feasibility of the simulator, it became evident that certain simplifications were necessary. Simultaneously, maintaining consistency with real-world results was a priority. The primary simplification employed was the representation of the Earth as a sphere with a fixed radius of $R_{Earth} = 6371$ km. This simplification significantly enhanced the efficiency of the code while retaining a degree of accuracy in line with actual conditions. This 3D structure was in fact designed to realistically compute the transmission process by taking into account the spatial coordinates and positioning of both satellites and vehicles.

The 3D Structure tracks the satellites position and therefore calculates all parameters in a dynamic way. As a premise each vehicle is determined by its $latitude$ and $longitude$ coordinates while the satellite is also defined by its elevation from the Earth's surface ($latitude$, $longitude$, h). For a ground vehicle, the distance d (a.k.a. slant range) can be determined by the satellite

altitude h and elevation angle α as follows [27]:

$$d = \sqrt{(R_{Earth}^2 \sin(\alpha)^2 + h^2 + 2hR_{Earth})} - R_{Earth} \sin(\alpha) \tag{2.1}$$

Using spherical trigonometry [31] the elevation angle α between the GV and the satellite will be calculated in the following way [32]:

$$\cos(\alpha) = \frac{\sin(\theta)}{\sqrt{1 + \left(\frac{R_{Earth}}{R_{Satellite}}\right)^2 - 2\left(\frac{R_{Earth}}{R_{Satellite}}\right)\cos(\theta)}} \tag{2.2}$$

where $R_{Satellite}$ is the distance from the center of the Earth to the satellite.

$$R_{Satellite} = R_{Earth} + h \tag{2.3}$$

While the non-negative angle θ, which represents the angle of the vehicle and satellite as observed from the Earth's center, is determined by the latitude (lat_{GV}) and west longitude (lon_{GV}) of the GV, as well as the north latitude (lat_{SAT}) and west longitude (lon_{SAT}) of the subsatellite point. θ is then computed using the following spherical trigonometry formula:

$$\cos(\theta) = \cos(lat_{\text{SAT}}) \cos(lat_{\text{GV}}) \cos(lon_{\text{GV}} - lon_{\text{SAT}}) + \sin(lat_{\text{GV}}) \sin(lat_{\text{SAT}}) \tag{2.4}$$

The formula is obtained through the law of sines which establishes the following relation:

$$\frac{R_{Satellite}}{\sin(\psi)} = \frac{d}{\sin(\theta)} \tag{2.5}$$

Where ψ is the angle between the line from the GV to the satellite and the line perpendicular to the Earth's surface. While the distance (d) from the GV to the satellite is determined using the law of cosines. The formula is given by:

$$d = R_{Satellite} \sqrt{1 + \left(\frac{R_{Earth}}{R_{Satellite}}\right)^2 - 2\left(\frac{R_{Earth}}{R_{Satellite}}\right)\cos(\theta)} \tag{2.6}$$

Figure 2.1 shows the analyzed scenario, involving a GV and a LEO, highlighting the angles

and distances between the nodes.

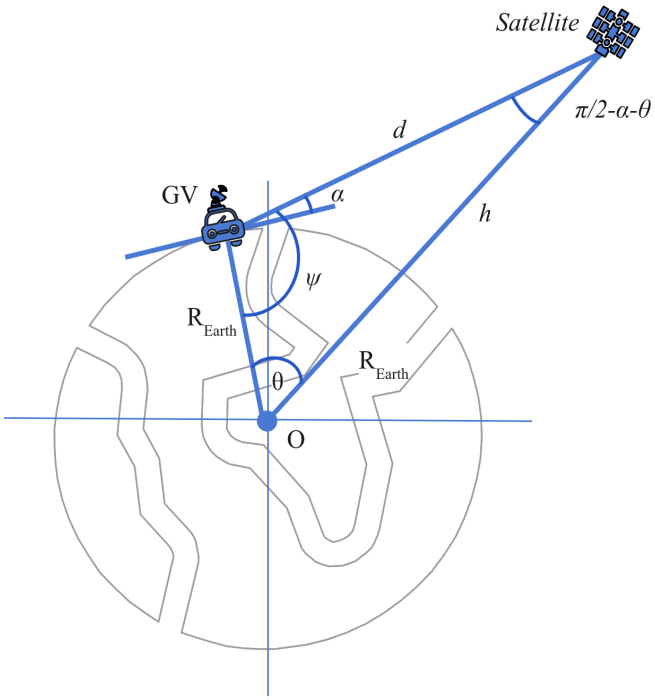

Figure 2.1: Simplified schema of the elevation angle (α) [32]

To address the sensitivity to changes on the horizon surface in satellite availability calculations, a mathematical check has been implemented to reduce mathematical complexity and, consequently, improve the simulator's efficiency. The proposed solution involves limiting satellite availability by defining a surface on Earth and verifying whether the satellite coordinates projected onto the Earth fall within the specified range. This approach significantly improves algorithm performance by allowing the determination of acceptable satellite elevation angles in advance. The sensitivity arises from the need to calculate elevation angles (θ) using a cosine formula, which imposes a condition for θ to be non-negative. Consequently, the calculation must account for various angle scenarios based on the specific condition in play.

Additionally, the $Range$ parameter, representing the distance from the vehicle in its position, plays a crucial role in the satellite availability limitation. This parameter defines a squared surface around the vehicle, taking into consideration the curvature of the Earth. It is noteworthy that users have the flexibility to set the $Range$ parameter in meters, allowing for customization based on specific simulation requirements. This intricate calculation, essential for determining satellite elevation angles, necessitates the efficient derivation of a subset of satellites due to the computational demands, especially when dealing with a potentially large number of satellites (up to 4550) and frequent calculations occurring up to 30 times per second in transmission simulations.

The mathematical expressions for defining the surface on Earth are as follows:

$$\Delta x = \frac{Range}{R_{Earth}} \tag{2.7}$$

$$\Delta y = \Delta x \cdot |\cos(lat_{\text{GV}})| \tag{2.8}$$

The check for satellite coordinates within the specified range is expressed as:

$$(lat_{\text{GV}} - \Delta x < lat_{\text{SAT}} < lat_{\text{GV}} + \Delta x) \tag{2.9}$$

$$(lon_{\text{GV}} - \Delta y < lon_{\text{SAT}} < lon_{\text{GV}} + \Delta y) \tag{2.10}$$

For a visual representation of the problem and the solution, refer to Figure 2.2. This figure provides a helpful schema for understanding the approach taken in the simulator.

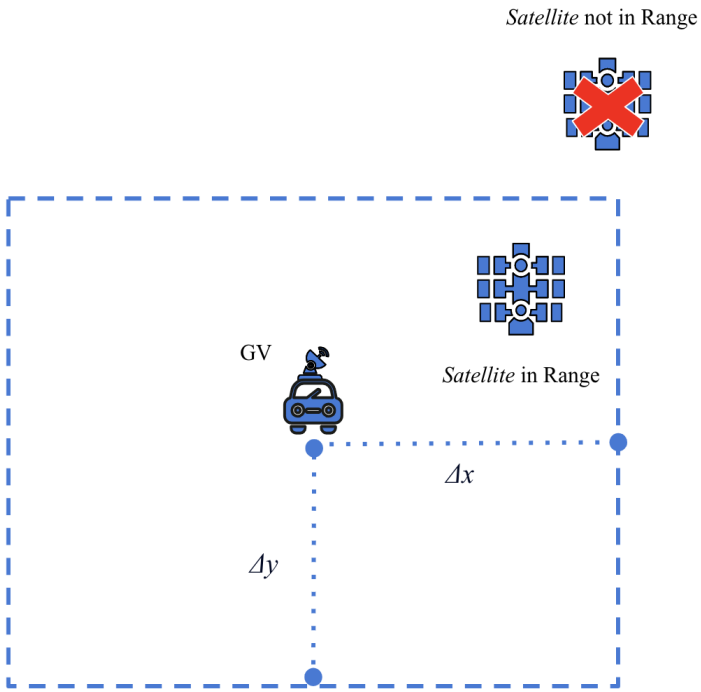

Figure 2.2: Simplified schema from an elevated perspective, illustrating the range computation

2.3.2 TLE Data Description

In our project, Two-Line Element (TLE) data played a crucial role in the real-time computation of satellite positions. This standardized format serves as a comprehensive package of orbital elements, offering essential information for accurate prediction and tracking of satellite locations.

The TLE data, comprising two lines (with an additional line for naming), encapsulates key

details such as satellite identification, orbital parameters, and epoch time. These elements collectively form the foundation for representing the intricate nature of a satellite's orbit.

```
STARLINK-#
1 43065U 17082A 18332.22107755 .00000008 00000-0 21232-4 0 9994
2 43065 98.6528 42.9929 0001247 105.7040 254.4278 14.27265986 48525
```

Analyzing the provided example, the TLE (Two-Line Element) data encompasses:
Line 1:

- **Satellite Identification (STARLINK-#):** Naming convention for the satellite.
- **Catalog Number (43065U):** Unique NORAD catalog number assigned to the satellite.
- **International Designator (17082A):** Represents the launch year and piece of launch.
- **Epoch Time (18332.22107755):** Day of the year and fraction of the day when TLE data was generated.
- **First and Second Time Derivatives:** Parameters related to the satellite's motion.
- **BSTAR Drag Term:** Influences how atmospheric drag affects the orbit.
- **Ephemeris Type (0):** Indicates the type of model used for orbit calculation.
- **Element Number (9994):** Identifier for distinguishing between multiple TLEs for the same satellite.
- **Checksum (4):** Ensures line integrity.

Line 2:

- **Catalog Number (43065U):** Matching the catalog number in Line 1.
- **Orbital Parameters:**
  - **Inclination (98.6528):** Tilt of the satellite's orbit.
  - **Right Ascension of Ascending Node (42.9929):** Angle between the ascending node and vernal equinox.
  - **Eccentricity (0001247):** Measure of orbit ellipticity.
  - **Argument of Perigee (105.7040):** Angle between the ascending node and perigee.
  - **Mean Anomaly (254.4278):** Angle defining the satellite's position in its orbit.

- **Mean Motion (14.27265986):** Average number of orbits completed in a day.
- **Revolution Number at Epoch (48525):** Number of orbits completed since a reference time.
- **Checksum (5):** Ensures line integrity.

This breakdown highlights the individual components of TLE data, showcasing its richness in conveying precise orbital information for accurate satellite positioning in real-time applications [33] [34].

2.3.3 Channel model

Compliant with 3GPP standards, the VEC communication system capitalizes on LE) satellites, employing millimeter waves (mmWaves) to ensure robust connectivity. In satellite missions with complex specifications, the adoption of frequencies spanning from X band to Ka-band has been necessitated by constraints in signal processing and modulation capabilities. Originally designated for radar applications, these frequency bands have found diverse applications in radar, terrestrial, and satellite communications. The classification provided by IEEE designations plays a pivotal role in offering standardized and convenient notations, serving the needs of radar engineers on a global scale.

Among the various radar frequency bands, the Ka-band emerges as particularly noteworthy, covering frequencies from 27 GHz to 40 GHz with a corresponding wavelength ranging between 1.1 to 0.75 centimeters. This specific band facilitates high-speed data communication, ensuring extensive coverage through the deployment of multiple beams, thereby enabling the use of smaller antennas. In the realm of satellite communications, the Ka-band is favored for its capacity to deliver elevated data transmission rates and bandwidth.

The careful selection of carrier frequencies within these bands is pivotal for achieving optimal system performance. High carrier frequencies, characteristic of mmWaves, usher in advantages such as high data rates and low latency, aligning seamlessly with the demands of contemporary communication standards.

Nevertheless, the utilization of high carrier frequencies introduces unique challenges. The shorter wavelengths associated with mmWaves inherently limit transmission range, a consideration that becomes especially pertinent in scenarios where extended reach is imperative. Furthermore, the sensitivity of mmWaves to atmospheric absorption necessitates meticulous system design. Obstacles in the communication path, such as buildings or geographical features, can adversely impact mmWave transmissions, underscoring the need for robust mitigation strategies to ensure reliable and uninterrupted connectivity in the VEC ecosystem [9].

In delving deeper into the specifics of mmWave transmissions, it's essential to explore the frequency bands in detail. The introduction of the Ka-band, encompassing frequencies from 26.5 GHz to 40 GHz, holds particular significance in the context of the VEC system. The Ka-band, falling within the broader mmWave spectrum, presents a compelling option for analysis due to its characteristics.

In the 3GPP context, the SNR between transmitter i and receiver j is computed as:

$$\gamma_{ij} = \text{EIRP}_i + \left(\frac{G_j}{T}\right) - \text{PL}_{ij} - k - B \tag{2.11}$$

where the EIRP is the effective isotropic radiated power, G_j/T is the receive antenna-gain-to-noise-temperature, PL is the path loss, k is the Boltzmann constant, and B is the bandwidth. The EIRP depends on the transmit antenna gain (G_T), power (P_T) and the cable loss (L_C) and it is given by:

$$\text{EIRP}_i = P_{Ti} - L_C + G_{Ti} \tag{2.12}$$

The antenna-gain-to-noise-temperature ($\frac{G_j}{T}$) depends on receiver characteristics: where G_R is the receive antenna gain, N_f is the noise figure, and T_0 and T_a are the ambient and antenna temperatures. Which can be computed as follows:

$$\frac{G_j}{T} = G_{Rj} - N_{fj} - 10\log_{10}[T_{0j} + (T_{aj} - T_{0j}) \cdot 10^{\frac{-N_{fj}}{10}}] \tag{2.13}$$

The free-space path loss (FSPL) model is computed using the carrier frequency f_c and is as follows:

$$\text{FSPL} = 92.45 + 20\log_{10}(f_c) + 20\log_{10}(d) \tag{2.14}$$

For the LEO satellite channel, path loss includes scintillation loss PL_s (due to sudden changes in the refractive index caused by variation of the temperature, water vapor content, and barometric pressure) and the atmospheric absorption loss PL_g (due to dry air and water vapor attenuation). Obtaining:

$$\text{PL} = \text{FSPL} + \text{PL}_g + \text{PL}_s \tag{2.15}$$

In the case of $f_c = 30GHz$ we can consider $PL_s = 0$ (as per 3GPP standard [27]). While for the atmospheric absorption loss it can be calculated using $PL_g(\alpha, f_c) = \frac{A_{\text{zenith}}(f_c)}{\sin(\alpha)}$, α being the elevation angle A_{zenith} corresponding to zenith attenuation for frequencies between 1 and 350 GHz. The calculation of the appropriate zenith attenuation is obtained by specific formulas based on the attenuation considered which can be found on [35].

It is important to notice that this is the PL in case of having LoS between the GV and the satellite, which is considered infinite otherwise.

Finally, the Shannon capacity R_l is given by:

$$R_l = B \log_2 \left(1 + 10^{\left(\frac{\gamma_{ij}}{10}\right)}\right), \quad l \in \{\text{UL, DL}\} \tag{2.16}$$

This channel model is the core of our simulator, crucial for evaluating communication between vehicles and satellites within the VEC framework. Its 3D structure dynamically calculates parameters, enabling a comprehensive analysis.

2.3.4 Delay Model

In this section, we explore the intricacies of processing time within satellite-assisted operations, focusing specifically on the critical aspect of data offloading through wireless information packet transfers.

The RTT is expressed as:

$$RTT = 2\tau_p + t_{\text{UL}} + t_{\text{DL}} + t_{p,SAT} \tag{2.17}$$

Where, τ_p represents the propagation delay, strictly linked to the distance d between the GV and the satellite considered, resulting in $\tau_p = \frac{d}{c_0}$, where c_0 is the speed of light.

The arrival rate of uplink packets is tied to the frame rate of the video. The transmission delays depend on the uplink (UL) (R_{UL}) and downlink (DL) (R_{DL}) channel capacities, along with the respective data size n_{UL} and n_{DL}.

$$t_l = \frac{n_l}{R_l}, \quad l \in \{\text{UL, DL}\} \tag{2.18}$$

In the queuing system, the delay $t_{p,SAT}$ encompasses considerations for both the expected queuing delay E_{QD} and the service time t_{S}. The service time (t_{S}) is defined as the duration for a server to process a packet, determined by the computational load (L) and server capacity (C)

in the form of: $t_{\mathrm{S}} = \frac{L}{C}$. This model captures how the processing requirements depend on both packet size and server capacity.

The expected queuing delay (E_{QD}) is calculated within a D/M/1 queuing model:

$$t_{p,SAT} = E_{\mathrm{QD}} + t_{\mathrm{S}}$$

This equation reflects the total system delay, combining the time spent in the queue (E_{QD}) with the time needed for processing (t_{S}).

Additionally, in the context of satellite-assisted operations, the packet size generated onboard the GV aligns precisely with the bit size of a video frame (n_{UL}), while the processed output, characterized by a size $n_{\mathrm{DL}} \leq n_{\mathrm{UL}}$, is transmitted back to the GV. This comprehensive model provides insights into the capture-to-output delay in satellite-assisted operations.

2.3.5 Queuing Model in Satellite Communication

The queuing model system considered here is specifically tailored for satellite communication and follows the (D/M/1) configuration. In this model, data packets arrive at the satellite station at constant intervals, denoted as β, and are processed with exponentially distributed service times characterized by μ. The system's cost is defined as a combination of expenses incurred due to packet wait times and the idle periods of the satellite service station [36].

Limit Distribution with Increasing Demand

This stability is intricately linked to the arrival rate of data packets, the computational load on the satellite system, and the computational capacity of the satellite itself. The system achieves stability when the product of the service rate (μ) and the inter-arrival time (β) surpasses 1, expressed mathematically as $\mu\beta > 1$.

The scenario envisioned is where data packets regularly arrive at a satellite processing station. The time it takes to process each packet follows a random yet exponential pattern. The critical factor influencing system behavior is the delicate balance between the speed at which packets arrive and how efficiently the satellite system can process them.

When the processing capacity (given by μ) significantly outpaces the arrival rate (given by β), the satellite system stabilizes. This stabilization is described through the concept of a limit distribution, providing a means to understand the long-term, steady-state behavior of the system.

Mathematically, the limit distribution is expressed as:

$$P_i^* = \lim_{n\to\infty} P_i^{(n)} \quad (i = 0, 1, 2, ...)$$

This formula denotes the probability distribution of having a certain number of packets (i) in the satellite system as time progresses. The expression P_i^* is defined by specific rules: the probability is 0 when there are no packets ($i = 0$), and for more than zero packets ($i > 0$), it follows a pattern determined by the stability parameter δ. This mathematical representation offers insights into how the satellite communication system behaves over an extended period of operation.

Where:

$$P_i^* = \begin{cases} 0 & \text{when } i = 0 \\ (1-\delta)\delta^{(i-1)} & \text{when } i > 0 \end{cases} \tag{2.19}$$

Here, δ is the root of the equation:

$$\delta = e^{-\mu\beta(1-\delta)} \tag{2.20}$$

Expected Queuing Delay

The expected queuing delay, denoted as E_{QD}, can be calculated as:

$$E_{\text{QD}} = \frac{1}{\mu} \cdot \frac{\delta}{1-\delta} \tag{2.21}$$

The queuing model plays a vital role in optimizing satellite server operations. It considers factors like arrival rates, computational load, and capacity to strike a balance between minimizing server idle times and customer wait times. This is crucial for efficiently managing satellite servers within the VEC system.

Transitioning to the application model in the next chapter, we will delve into the simulator's main components, the rationale behind their design, and their integration into the VEC landscape. This chapter will further feature a presentation of the simulator code, demonstrating its practical application to offer a lucid understanding of how it contributes to enhancing data processing efficiency in vehicular environments.

Chapter 3

Simulator

The developed simulator provides a versatile platform for exploring VEC systems through satellite edge computing, offering users the ability to simulate diverse scenarios and interactions.

With the inclusion of communication capabilities for vehicles, the simulator enables users to replicate the interactions of a specific vehicle within the VEC system.

Furthermore, users have the flexibility to position vehicles anywhere in the simulated environment, allowing for the exploration of various spatial configurations. This capability is essential for researchers to assess the system's performance under different scenarios, especially in adapting to diverse geographical settings and user-defined parameters.

In the context of satellite communication, our simulator goes beyond static representations. Users can associate satellites with dynamic trajectories, observing how their positions evolve over time and how this impacts the offloading system. The ability to change the timestep enhances this feature, enabling researchers to explore specific time slots with precision.

Moreover, the simulator provides the option to select the number of satellites in the simulation, offering possibilities to study the impact of satellite constellations on communication efficiency and test the scalability of the VEC system with varying satellite densities. This flexibility opens avenues to simulate scenarios with clusters of satellites working in tandem for robust communication coverage or sparse configurations requiring vehicles to adapt to intermittent connectivity.

This simulator is designed to be a robust platform for studying VEC systems through satellite communication, providing a balance between real-world complexity and practical simulation. It allows researchers to explore dynamic interactions, spatial configurations, and the impact of satellite movements on the VEC system's performance, the outline of the UML schema can be seen in 3.1.

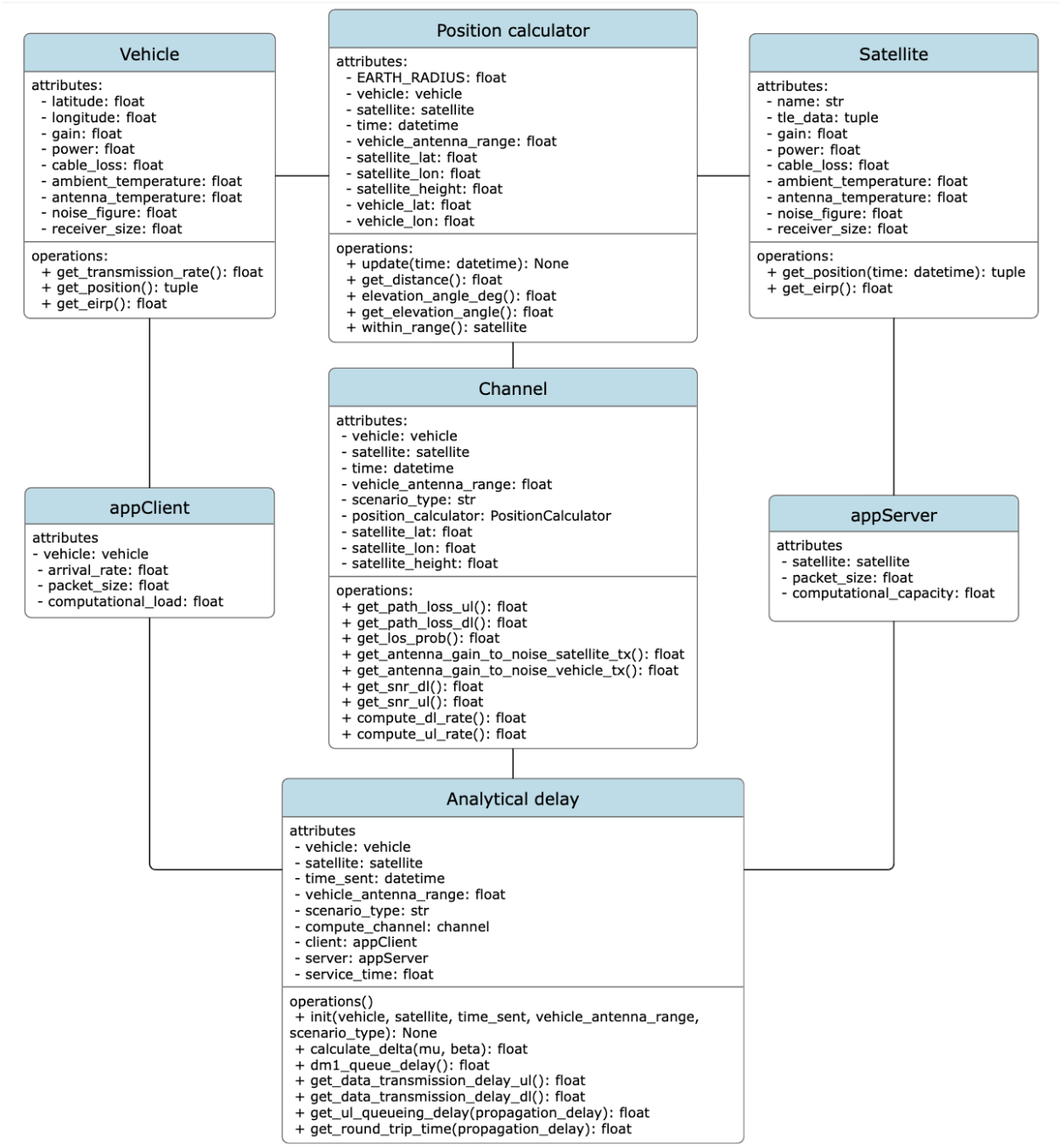

Figure 3.1: UML schema of the simulator

3.1 Main Objects

In this section we will explore the two fundamental components critical to VEC simulations: the Vehicle and the Satellite. These objects have been thoughtfully designed to simplify their representation in the code, capturing the essential real-world characteristics that influence VEC scenarios while ensuring computational efficiency and ease of use.

3.1.1 Vehicle Object

The `Vehicle` class emulates the behavior of vehicles in VEC simulations. It is composed of two key components: its position and antenna parameters. The antenna parameters are specifically tailored to compute the channel model, and they can be fine-tuned based on the desired equipment and communication requirements. In our context, for communication analysis, maintaining a static vehicle position sufficed, as the negligible movement of the vehicle is inconsequential compared to the rapid motion of a satellite. The simplified `Vehicle` object includes the following attributes and methods:

- **Geographical Position (Latitude and Longitude):** In real life, vehicles traverse vast areas, but for our simulations, we focus on their instantaneous positions.
- **Transmission Parameters (Gain, Power, Cable Loss, Antenna Temperature, Noise Figure, Receiver Size):** These parameters encapsulate the core elements of a vehicle's communication system. While real vehicles have diverse and dynamic characteristics, these parameters provide a sufficient approximation for VEC scenarios.

Application and Significance

The Vehicle tool is designed for simulating and analyzing communication scenarios involving vehicles. It allows users to create a virtual representation of a vehicle, specifying its location and communication details. Once set up, users can retrieve information about the vehicle, such as its geographical position and communication setup. The tool also includes built-in methods for calculating communication aspects like data transmission rate and Effective Isotropic Radiated Power (EIRP). Additionally, it can be customized to meet specific simulation needs, providing adaptability for different scenarios without unnecessary complexity. In essence, the Vehicle tool serves as a virtual model to explore and understand how a vehicle communicates in various situations.

3.1.2 Satellite Object

The `Satellite` class represents the satellites in our VEC simulations.

3.1.3 TLE parsing library

Explaining Satellite Position Calculation Method with Skyfield: Skyfield, a powerful Python library, enables users to predict the positions of Earth satellites by utilizing Two-Line Element (TLE) files containing orbital elements published by organizations like CelesTrak. This library

employs the SGP4 satellite propagation routine to generate accurate predictions of satellite positions over time.

To calculate the position of a satellite, the Two-Line Element (TLE) data is crucial. In the satellite class, the Two-Line Element (TLE) data is retrieved by parsing a file that contains information for all satellites. This extracted TLE data is then assigned to the respective attributes of the satellite object.

Once the TLE data is extracted, the code utilizes the Skyfield library to calculate the geocentric coordinates of the satellite at a specified time. The process involves creating a timescale object, converting the input time to Coordinated Universal Time (UTC), and then creating an `EarthSatellite` object using the extracted TLE data. Finally, the latitude, longitude, and height of the subpoint (the point on the Earth's surface directly beneath the satellite) are obtained.

This approach provides a straightforward way to leverage TLE data and the SGP4 satellite propagation routine to estimate the position of Earth satellites. The accuracy of the prediction is subject to limitations as we will explain below.

- Accuracy Limitations: The accuracy of satellite positions is not perfect, and it is limited by the number of decimal places in each field of the TLE.

- Epoch Considerations: Satellite elements quickly go out of date. The "epoch," which is the date on which an element set is most accurate, should be considered. Elements are typically useful for a week or two on either side of the epoch date, and it's advisable to download fresh sets for later dates or pull archived TLEs for earlier dates.

- Dynamic Orbits: A satellite's orbit constantly changes due to factors such as atmospheric drag and the Moon's gravity. The SGP4 propagation routine models these effects, and the true anomaly parameter can vary significantly, especially for satellites with nearly circular orbits.

The fundamental components of satellite placement within the code encompass the Name and Two-Line Element (TLE) Data, which form the cornerstone for achieving optimal accuracy. Each satellite is uniquely identified by its name and associated TLE data, facilitating easy referencing and precise computation of its position. The TLE data serves as critical parameters for the position calculation process. Additionally, the code incorporates crucial transmission parameters, including gain, power, cable loss, and other essential attributes. Although real satellites are characterized by advanced functionalities, these simplified parameters have been tailored to suit the simulation's requirements, ensuring a balance between accuracy and user comprehension without overwhelming complexity.

Application and Significance

The ‘Satellite‘ class is a tool that proves useful in communication analysis simulations. It simplifies the representation of satellites, allowing users to configure virtual satellites with specific names, orbital data, and communication capabilities. In a simulated environment, users can analyze communication paths, assess signal strength, and gain insights into satellite positions. The class enables customization for different scenarios, accommodating changes in the number of satellites, orbital adjustments, or variations in communication parameters. Through the built-in methods, users can evaluate the performance of satellite communication, considering factors like data transmission rates and EIRP. Overall, the ‘Satellite‘ class facilitates a user-friendly approach to understanding and experimenting with satellite communication dynamics in diverse scenarios.

3.2 3D Structure and Channel Model

In this section, ”3D Structure and Channel Model,” we delve into the technical aspects of the 3D spatial structure used for satellite and vehicle communication within the context of VEC. We will also explore the considerations and design choices made in our code to model the communication channels between vehicles and satellites.

3.2.1 Orbit Model

In the realm of Vehicle-to-Satellite (VEC) communication, our `PositionCalculator` class is a vital tool designed to dynamically simulate the intricate 3D spatial dynamics involved in this communication scenario. This class performs key calculations, including distances, elevation angles, and communication ranges between a ground vehicle and a satellite.

Key Features and Functionality

Distance Calculation: The `get_distance` method is at the core of spatial analysis. By considering Earth’s radius and the heights of both the vehicle and the satellite, it computes the spatial separation between them. This distance measurement is fundamental for understanding the physical space between the communicating entities.

Elevation Angle Determination: The `elevation_angle_deg` method calculates the elevation angle in degrees. This angle is pivotal for establishing line-of-sight communication, as it indicates the angle above the horizontal plane from which the satellite is visible to the ground vehicle.

Communication Range Check: The `within_range` method assesses whether the satellite falls within the communication range of the ground vehicle. This check helps determine the feasibility of communication based on predefined antenna ranges, allowing us to understand whether the entities can establish a reliable connection.

Application and Significance

The `PositionCalculator` class serves as a foundational element for spatial analysis in VEC communication scenarios. Its capabilities extend beyond mere distance calculations, playing a crucial role in evaluating parameters that directly impact communication link quality.

Similar to the channel model, our class embraces adaptability. Users can dynamically update simulation parameters such as time and vehicle positions. This flexibility is essential for mirroring real-world scenarios where the positions of vehicles and satellites are dynamic and subject to change.

Despite the inherent complexity of 3D spatial structures, our code deliberately simplifies calculations to focus on key parameters. This strategic simplification ensures that the code remains accessible and efficient while still capturing the essential aspects of VEC communication.

In practical VEC communication scenarios, the `PositionCalculator` class is highly relevant. It addresses the communication challenges between vehicles on Earth's surface and satellites positioned at varying altitudes. By providing accurate spatial insights, including distance and elevation angles, the class becomes instrumental in optimizing communication feasibility and ensuring reliable data transfer between vehicles and satellites.

3.2.2 Channel Model

The channel model encapsulates a comprehensive representation of the communication channel between a ground vehicle and a satellite. This model considers a range of factors influencing communication quality, including atmospheric conditions, antenna characteristics, and path loss. The `channel` class is equipped with methods to calculate crucial parameters for assessing the communication link, making it a valuable tool for analyzing and optimizing satellite communication scenarios.

Key Features and Functionality

A critical aspect of the channel model is its ability to estimate path losses for both uplink and downlink transmissions. Through the `get_path_loss_ul` and `get_path_loss_dl` methods, the model takes into account free space path loss and atmospheric attenuation. This inclusion

of atmospheric conditions, including gas, rain, scintillation, and clouds, enhances the accuracy and realism of the path loss calculations.

The Line-of-Sight (LOS) probability, as determined by the `get_los_prob` method, adds another layer of sophistication to the model. This functionality considers the elevation angle and specific communication scenarios, providing insights into the likelihood of establishing a direct line of sight between the ground vehicle and the satellite. In vehicular edge computing scenarios, where connectivity interruptions can be detrimental, understanding LOS probability is crucial for optimizing communication reliability.

Antenna performance is a key factor in communication quality, and the channel model addresses this through the computation of antenna gain-to-noise ratios for both satellite and vehicle transmissions. The `get_antenna_gain_to_noise_satellite_tx` and `get_antenna_gain_to_noise_vehicle_tx` methods contribute to assessing the signal quality received by the respective antennas.

The model further calculates SNR for downlink and uplink transmissions, considering factors such as EIRP, antenna gain-to-noise ratio, and path loss. These SNR values serve as essential metrics for evaluating the overall performance of the communication link.

Finally, the channel model facilitates the estimation of data rates for both downlink and uplink communications. Leveraging the calculated SNR values, the `compute_dl_rate` and `compute_ul_rate` methods provide insights into the achievable data throughput in the given communication setup. This information is particularly relevant in vehicular edge computing scenarios where efficient data transfer is critical for real-time processing and decision-making. In summary, the presented channel model offers a versatile tool for analyzing and optimizing satellite communication scenarios in the context of vehicular edge computing.

Application and Significance

The `channel` model in our code serves as a central component for evaluating communication quality in V2X systems. By considering factors like SNR, path loss, and transmission delay, this model provides insights into the feasibility and performance of data transfer in dynamic vehicular communication scenarios.

Designed for telecommunications simulations, the `channel` class offers adaptability by receiving parameters from other classes. This flexibility allows users to tailor communication scenarios based on the characteristics and positions of both the transmitter and receiver.

In acknowledging the dynamic nature of real-world communication channels, the `channel` class simplifies complexities for practical simulation purposes. Users can adjust critical parameters, such as signal strength, atmospheric conditions, and antenna characteristics, within the code. This adaptability is crucial for replicating specific V2X scenarios, accommodating a wide

range of dynamic factors influencing communication channels.

Despite its simplified representation, the `channel` class prioritizes essential metrics like SNR and path loss. By focusing on these core factors, the class aligns seamlessly with research objectives, ensuring that simulations accurately reflect the impact of these metrics on data transfer quality.

By receiving and integrating parameters from other classes, the `channel` class becomes a versatile tool that can dynamically adjust its behavior based on the unique characteristics and positions of transmitters and receivers. This structural adaptability enhances the class's utility, making it well-suited for a variety of telecommunications scenarios where the communication environment is ever-changing.

3.3 Client-Server Structure

In this section, we will explore the client-server architecture employed in VEC systems. We'll discuss how this design enhances the offloading process, the main objects involved (Vehicle and Satellite), and the well-thought-out principles underlying the implementation.

3.3.1 Client-Server Architecture in VEC

The client-server architecture acts as the backbone for efficient data offloading and processing. In our specific implementation, the architecture is embodied by two essential objects: the `appClient` and `appServer`, each made to fulfill distinct roles in facilitating the seamless interaction between the vehicle (client) and the satellite (server).

The `appClient` object serves as the embodiment of the vehicle within our VEC system, taking on the responsibility of managing client-side operations. This includes the simulation of parameters such as `arrival_rate`, `packet_size`, and `computational_load`. These parameters are integral in capturing the behavior of the client, considering factors like the rate at which data arrives, the size of data packets, and the computational load of the data sent by the vehicle. The design of the `appClient` object is geared towards providing a comprehensive representation of the vehicle's role in the offloading process, allowing for a nuanced analysis of client-related dynamics.

On the other hand, the `appServer` object is tailored to represent the satellite in our VEC system, taking charge of server-side operations. Crucial parameters such as `computational_capacity` and `packet_size` are incorporated into the design. Here, `computational_capacity` signifies the server's processing power, while `packet_size` accounts for the size of data packets that can be transmitted back to the vehicle. This meticulous

design ensures that the `appServer` object can effectively simulate the server's role in the offloading process, enabling a detailed analysis of how server characteristics impact data transmission and processing.

In essence, the `appClient` and `appServer` objects form the core elements of our VEC system, embodying the client and server entities, respectively. Their thoughtful designs allow for the emulation of real-world scenarios, enabling a simplified exploration of the dynamics involved in data offloading and processing from a vehicle to a satellite. This client-server architecture stands as a foundational framework for understanding and optimizing the communication and computation aspects of VEC systems in the context of satellite connectivity.

Client-server interaction and delay analysis

In the context of VEC systems communicating with satellites, the efficiency of data offloading and processing hinges on the intricate interaction between the client and server entities. This interaction is meticulously captured and analyzed through the `analyticalDelay` class, a core component designed to calculate various delays inherent in the communication process. Tailored to the characteristics and behaviors of the client and server objects, this class serves as a linchpin for understanding the temporal aspects of data transfer in VEC systems.

A pivotal aspect addressed by the `analyticalDelay` class is the queuing delay, which is quantified by the `dm1_queue_delay` method. This calculation takes into account parameters such as arrival rate and service time, providing insights into how the computational load on both the client and server influences queue management. In VEC scenarios, where computational resources are finite, understanding and optimizing queue delays are imperative for enhancing overall system performance.

The queuing delay calculation is particularly relevant in the specific context of vehicular edge computing towards a satellite. This delay represents the time a task or piece of data spends waiting in a queue before being processed, considering both the computational load on the vehicle (client) and the processing capacity of the satellite (server). The arrival rate (β) reflects the rate at which tasks or data arrive at the client for offloading, influenced by factors like the frequency of data generation and demand for computational offloading. The service time (μ), derived from the computational load divided by the computational capacity, represents the time required to process a task or piece of data on the server.

The core of the queuing delay calculation involves solving the root of the equation δ in a mathematical equation, providing a solution used in subsequent queuing delay calculations. This approach is rooted in queuing theory, and the resulting delay is represented in Equation (2.21). If the arrival rate is greater than or equal to the service rate (computed from the reciprocal of service time), the queuing delay is considered infinite, signaling an overloaded system where

tasks accumulate faster than they can be processed.

Uplink and downlink data transmission delays, as computed by the `get_data_transmission_delay_ul` and `get_data_transmission_delay_dl` methods, respectively, further illuminate the complexities of the client-server interaction. These calculations factor in communication channel characteristics influenced by the SNR and packet sizes of both the client and server. The client and server objects, representing the vehicle and satellite, play a crucial role in determining the transmission time, underscoring their significance in the data exchange process.

The `get_total_analytical_delay` method encapsulates the culmination of these delay components, providing a holistic view of the temporal aspects involved in VEC systems. This method combines queue delays, uplink and downlink transmission delays, and propagation delays, offering a comprehensive analytical approach. It takes into consideration the intricacies of the client-server interaction, the dynamics of data transfer, and the physical properties governing satellite-vehicle communication.

In essence, the client-server architecture and the associated `analyticalDelay` class serve as essential tools for modeling and analyzing real-world VEC scenarios. The `appClient` and `appServer` objects encapsulate the main characteristics of vehicles and satellites, allowing for a nuanced analysis of offloading strategies. The division between client and server, coupled with considerations of the 3D structure and channel characteristics, forms the backbone of the VEC system code, providing valuable insights into the temporal intricacies of communication in vehicular edge computing towards a satellite.

3.4 How to Build a Scenario

In this section, we will explore the process of building a scenario within our simulator for VEC systems. Building scenarios is a critical component of our thesis, as it allows us to simulate and analyze a wide range of VEC conditions. The primary goal is to evaluate offloading strategies under various circumstances, making our research more comprehensive and applicable to real-world scenarios.

3.4.1 Workflow of Scenario Building

The workflow of scenario building within our VEC simulator is a meticulous process designed to replicate real-world conditions and dynamics of communication between vehicles and satellites. The initialization phase lays the groundwork by defining the number of satellites, their parameters, and the acceptable range for establishing connections with vehicles. This phase

establishes the foundational framework for the subsequent interactions between mobile entities and communication hubs.

Creating vehicles involves specifying parameters such as latitude, longitude, gain, power, cable loss, ambient temperature, antenna temperature, noise figure, and receiver size. These parameters collectively shape the communication capabilities of vehicles. Gain, representing the amplification factor, influences signal strength, while power impacts energy transmission. Cable loss and ambient temperature affect signal integrity and system performance, emphasizing the importance of striking a balance between these factors. Additionally, considerations like antenna temperature, noise figure, and receiver size directly impact signal quality, noise interference, and sensitivity to weak signals. These nuanced parameters collectively define the vehicles and contribute to the dynamic nature of the simulated VEC scenarios.

Simulation parameters, including start time, time step, and the total number of iterations, are then specified to control the duration and granularity of the simulation. This detailed configuration ensures that the simulator captures the temporal intricacies of the VEC system's behavior over time.

The subsequent step involves setting channel and communication parameters that form the foundation of the simulation. Parameters such as packet arrival rate, packet size, computational load, atmospheric conditions, and the number of satellites in the constellation add depth to the simulation, closely mirroring real-world scenarios. These considerations capture the complexities of communication in VEC systems, providing a rich environment for analysis.

Once the parameters are in place, the scenario is implemented, and the simulator runs the simulation, iterating over the specified time range. During each time step, the simulator evaluates communication between vehicles and satellites, collecting a comprehensive set of data, including positions, distances, path loss, SNR, elevation angles, line-of-sight probability, and various delays.

Crucially, the simulation results are stored in CSV files, facilitating subsequent in-depth analysis. This structured workflow and comprehensive consideration of key design parameters make the VEC simulator a powerful tool for advancing technology and strategies in vehicular edge computing towards a satellite. The simulations provide nuanced insights into the dynamic interplay between vehicles and satellites, offering valuable perspectives for optimizing communication and computational offloading in this complex and evolving domain.

Chapter 4

Scenario and data overview

4.1 General explanation of the parameters considered

In the previous chapter, we discussed the configuration of a foundational scenario for the analysis of vehicle-to-satellite communication. The primary focus of this analysis centers on understanding delays associated with information packets. The core concept of this scenario is to provide flexibility in satellite selection within the vehicle's operational range while concurrently computing communication system parameters.

In the context of the vehicular data offloading we considered using a carrier frequency of 30GHz which aligns with the prevailing trend in satellite communications favoring higher frequencies. Satellite communication commonly employs frequency bands such as UHF, S, X, and Ka, each presenting distinct advantages and trade-offs. The shift towards higher frequencies, exemplified by the 30GHz frequency band, is motivated by the imperative need for elevated data rates. Nevertheless, it is crucial to acknowledge that higher frequencies, such as 30GHz, introduce challenges like heightened atmospheric and rain attenuation, necessitating compensatory measures like augmented power transmission and high-gain antennas. While Ku-, K-, and Ka-band communication systems are more entrenched in larger spacecraft, the decision to operate at 30GHz indicates a preference for the Ka-band or potentially even higher frequencies, foreseeing the advantages of augmented bandwidth despite challenges like rain fade [37]. This aligns with the broader industry trajectory evident in the growing adoption of higher frequencies for data-intensive space missions.

The communication is simulated in a realistic manner, wherein the vehicle transmits a packet whenever one is generated. Moreover, the position of each satellite is dynamically calculated at each step, ensuring that the system's parameters are re-calibrated in real-time, thus providing a comprehensive and accurate depiction of the communication process.

In order to further enhance the realism of the simulation it is possible to select the scenario we

are considering the vehicle to be in. Selecting the scenario influences the possibility of having LoS with the satellite. Dealing with high frequencies and large distances losing the LoS means not being able to send the packet in that instance. The probability of such event in a specific scenario is given by the following table:

Elevation Angle [°]	Dense Urban	Urban	Suburban and Rural
10°	0.282	0.246	0.782
20°	0.331	0.386	0.869
30°	0.398	0.493	0.919
40°	0.468	0.613	0.929
50°	0.537	0.726	0.935
60°	0.612	0.805	0.94
70°	0.738	0.919	0.949
80°	0.82	0.968	0.952
90°	0.981	0.992	0.998

Table 4.1: Elevation angle LoS probability in different environments

Offering options of 20 Mb (SELMA frame size) and a compressed 5 Mb packet, a more compact version of the former. The choice of packet size is of paramount importance as it directly influences computational load. Larger packets necessitate 150 GFLOPs for processing, while smaller packets require 60 GFLOPs, which is computed as the average between the computational performance of two popular object detectors, namely Gaussian YOLO and SqueezeDet+ [38].

Variations in computational capacity are also taken into account, considering a lower capacity of 5000 GFLOPs for one scenario and a higher capacity of 10000 GFLOPs for a more advanced satellite [39]. Additionally, alignment of the communication's time step with the inter-packet arrival time is determined by the SELMA dataset's frame rate, typically set at 30 frames per second (FPS). Nevertheless, simulations at a reduced frame rate of 10 FPS are also explored.

Another pivotal parameter under consideration is the density of the satellite constellation, affording the flexibility to choose between a modest constellation of 1000 satellites or the maximum of 4550 satellites. The flexibility of choice enables us to examine how the arrangement of satellites impacts communication. It is anticipated that having more satellites will lead to improved performance, as the vehicle can select a more strategically positioned satellite for communication purposes.

4.2 Satellite Antenna Parameters

The selection of satellite antenna parameters is a crucial factor influencing the overall performance of the system. In the process of choosing these parameters, default values were employed

for a generic antenna, including Power (3 dBW), Cable Loss (0), Ambient Temperature (290 K), Antenna Temperature (150 K), Noise Figure (1.2 dB), and Receiver Size (1). Notably, the antenna gain was specifically set at 43.2 dB to match the precise specifications of the Starlink satellite as documented in [40].

4.3 Vehicle Antenna Parameters

In the context of vehicle communication systems, various antenna parameters play an important role in determining the performance of the system. When choosing the antenna parameters some were chosen with default values (Power: 3 dBW, Cable Loss: 0, Ambient Temperature: 290 K, Antenna Temperature: 150 K, Noise Figure: 1.2 dB, Receiver Size: 1) meter except for the Gain so that when a new antenna was considered we could examine how that value influenced the results.

4.3.1 VSAT

Description: A Very-Small-Aperture Terminal (VSAT) is a type of satellite ground station with a dish antenna that is smaller than 3.8 meters. VSATs are commonly used for two-way satellite communication, particularly in remote areas or for specialized applications. The Gain value considered was taken from a site selling VSAT antennas to mount on vehicles. After scrolling several antenna the most realistic value was chosen.

- **Gain:** 43.2 dBi

4.3.2 Reflectarray

Description: A Reflectarray antenna consists of an array of unit cells illuminated by a feeding antenna. It is used to reflect electromagnetic waves in a specific direction with a focus on achieving a thinner form factor compared to traditional parabolic reflectors. The value chosen is of a Single-Layer Reflectarray antenna for an internet vehicle.

- **Gain:** 27.12 dBi

Usage: Reflectarray antennas are designed to focus a beam in a manner similar to parabolic reflectors but with a thinner form factor. They are used in applications where space constraints or portability is a concern, making them suitable for various vehicle communication systems.

4.3.3 Gain Differences

One notable difference between the VSAT and the Reflectarray antennas is the gain. The gain of an antenna represents its ability to direct and concentrate radiated power in a specific direction. In this context, it is expressed in decibels relative to an isotropic radiator (dBi).

The Large VSAT antenna has a higher gain of 43.2 dBi, indicating that it can focus and amplify signals more effectively. This higher gain is advantageous for long-range and high-data-rate communication. It is well-suited for scenarios where the satellite signal needs to be received with high efficiency and reliability, such as in satellite Internet access.

On the other hand, the Reflectarray antenna has a gain of 27.12 dBi. While it has a lower gain compared to the VSAT, the Reflectarray's design allows it to achieve a focused beam with a much thinner form factor. This characteristic is advantageous when space constraints or portability is a consideration. The Reflectarray's gain, though lower, is still sufficient for many vehicle communication systems.

The choice between these antennas depends on the specific requirements of the communication system, considering factors like range, data rate, and the available space for the antenna. It's important to select an antenna with the appropriate gain to meet the system's needs while considering practical constraints.

4.4 SELMA Dataset and Its Relevance for Modeling Packet Size and Arrival Rates

Accurate scene understanding from multiple sensors mounted on cars is a key requirement for autonomous driving systems. Nowadays, this task is mainly performed through data-hungry deep learning techniques that need very large amounts of data to be trained. Due to the high cost of performing segmentation labeling, many synthetic datasets have been proposed. We introduce SELMA, a novel synthetic dataset for semantic segmentation that contains more than 30K unique waypoints acquired from 24 different sensors including RGB, depth, semantic cameras and LiDARs, in 27 different weather and daytime conditions, for a total of more than 20M samples. SELMA is based on CARLA, an open-source simulator for generating synthetic data in autonomous driving scenarios, that was modified to increase the variability and the diversity in the scenes and class sets, and to align it with other benchmark datasets. In our scenario, we found a dataset that accurately represents what a vehicle's sensors capture while it's in motion. Using the dataset's frame size is a practical way to organize how information is transmitted. Additionally, taking into account the dataset's frames per second (FPS) helps us estimate the amount of data the vehicle would send over a specific period.

4.4.1 Introduction to ITS and importance of the SELMA dataset

Recent advancements in the automotive industry have paved the way for Connected Intelligent Transportation Systems (C-ITSs) with the goal of enhancing safety and driving efficiency. These systems can significantly reduce traffic accidents, with some estimates suggesting a potential reduction of up to 90%. Additionally, they can improve traffic management through techniques like smart platooning, cruise control, and traffic light coordination. C-ITSs also hold the promise of improving fuel economy and contributing to a substantial 60% reduction in carbon emissions. This progress has spurred significant research efforts in this area.

To achieve these objectives, future connected vehicles will be equipped with a range of sensors, including Light Detection and Ranging (LiDAR) and RGB camera sensors. LiDAR sensors are known for their ability to generate highly accurate 3D omnidirectional environmental maps, making them suitable for geometry acquisition under various weather and lighting conditions. On the other hand, RGB cameras offer advantages such as cost-effectiveness, high resolution, and frame rates, albeit with sensitivity to illumination and visibility conditions.

Sensor fusion is seen as a promising solution to enhance scene understanding, although it entails additional processing overhead for gathering and combining data from multiple sensors.

In this context, the SELMA dataset is presented as a new synthetic dataset for autonomous driving. Created using a modified version of the CARLA simulator, SELMA stands out as one of the most comprehensive and diverse datasets for designing, prototyping, and validating autonomous driving models, particularly for complex tasks like semantic segmentation.

The SELMA dataset includes data collected in 30,909 distinct locations from 7 RGB cameras, 7 depth cameras, 7 semantic cameras, and 3 LiDARs, each paired with semantic information. The multimodal nature of SELMA promotes data diversity and complementarity, enhancing the accuracy and performance of learning tasks. Additionally, acquisitions are generated under various weather, daytime, and viewpoint conditions across 8 maps, resulting in 216 unique settings. The CARLA simulator has been enhanced to increase the realism of weather conditions and visual variability[30].

The dataset also provides semantic labeling for both camera and LiDAR data into 36 distinct classes, ensuring compatibility with common benchmark datasets like Cityscapes. The accuracy and realism of the dataset are validated through baseline experiments, demonstrating the superior performance of deep learning models trained on SELMA when tested in real-world conditions compared to models trained on other synthetic datasets.

For our specific use case, we primarily focused on utilizing the cameras within the SELMA dataset. Our objective was to evaluate the computational requirements for object detection scenarios, particularly when transitioning from vehicle-based to satellite-based processing. By concentrating on camera data, we were able to assess the performance and feasibility of offloading

critical tasks to a satellite environment. The rich and diverse nature of SELMA's camera data allowed us to conduct precise assessments and simulations, confirming its suitability for our research objectives.

4.4.2 Analyzing Frame Sizes in SELMA Dataset

In order to analyze frame sizes and related information in the SELMA dataset, a Python script was developed. This script uses the OpenCV library for video processing and Pandas for data management. The script's main purpose is to extract and record frame sizes for all the videos in the dataset, organized by the source camera.

The script's architecture and functionalities are designed to systematically process and collect frame-related data from video files in the SELMA dataset. It begins by specifying a root directory, serving as the starting point for the search for video files within the dataset. This root directory is where the script initiates the exploration of video folders.

A critical component of the script is the use of a Pandas DataFrame named 'data' for data collection. This DataFrame is initialized as empty and acts as a structured container for organizing frame-related information, specifically frame numbers and their corresponding sizes in bits.

To uniquely identify and categorize the source of each video, the script incorporates a function called 'extract_camera_name.' This function is responsible for parsing the folder structure and extracting the camera name, generating a unique identifier for each camera.

The core functionality of the script lies in its recursive processing of videos. It systematically traverses through all folders and subfolders under the specified root directory. For each encountered video file with the '.mp4' extension, the script opens the file for processing. It then iterates through the video frames, measuring the size of each frame in bits and recording both the frame number and size in the 'data' DataFrame.

The organized data is saved for each camera into CSV files specific to that camera. These CSV files are stored in a designated 'output_directory.' Each file includes information about frame numbers and their corresponding sizes in bits, ensuring a structured and separated storage of frame data by camera source.

To provide insights into the progress of the script, print statements are incorporated to track which video is currently being processed. After saving data for a particular camera, the 'data' DataFrame is reset, ensuring a clean slate for the subsequent camera's data collection. This careful organization and separation of data contribute to the script's efficiency in handling large video datasets while maintaining clarity in data storage and categorization.

This script serves as a tool for analyzing frame sizes within the SELMA dataset, helping understand the data characteristics and use this information to improve realism of the simulator, so

that it is possible to optimize communication systems or evaluate data transmission requirements for autonomous vehicles.

4.5 Analyzing Elevation Angle Variation in Satellite Communication

This section will help us understand how to set the $Range$ parameter, which is the distance in meters from the vehicle position when using the formulas (2.7)-(2.10) in Chapter 2. It is important to know that a certain degree of knowledge is needed to best understand how the range influences the availability of satellites.

The below explained script is designed to analyze how the elevation angle changes of a satellite changes when increasing the $Range$. This is done to understand when the satellite is below horizon and to reduce the relevant satellites for our communication.

The primary objective of this script is to accurately identify satellites that exceed the communication range of a vehicle. It focuses on understanding the relationship between the elevation angle and range for a specific satellite elevation. This information is essential for gaining insights into the dynamics of satellite-vehicle interactions in real-world scenarios.

A critical component of the script is a function that determines if a point is within a specified range from a reference point. The function adjusts the satellite's position until it goes beyond the designated range, at which point the range is updated. This function is pivotal in pinpointing satellites that have surpassed the anticipated communication range of the vehicle.

Upon execution, the script generates a CSV file comprehensive data. These file serves as concrete records, illustrating the correlation between elevation angles and range for the specified satellite elevation. This dynamic data is invaluable when setting the $Range$ parameter in the considered scenario.

In short, the script is tuned to perform a specific task: tracking satellites that move out of range, determining the elevation angle when they exceed the expected range, and adjusting the range to identify the next elevation angle. This focused approach ensures that the script not only yields interesting observations but also offers practical insights for enhancing satellite-vehicle interactions, particularly in situations where maintaining precise communication links is critical and requires continuous refinement.

$Range$ **(m)**	**Elevation angle (α) [°]**
150000	$\approx 78°$
350000	$\approx 56°$
550000	$\approx 45°$
750000	$\approx 34°$
950000	$\approx 26°$
1150000	$\approx 20°$
1340000	$\approx 15°$
1550000	$\approx 13°$
1750000	$\approx 10°$
1950000	$\approx 7°$
2150000	$\approx 4°$

Table 4.2: Range influence on elevation angle in the case of 550 km elevated satellites

This script functions as a tool for analyzing the correlation between the $Range$ parameter and elevation angles. Its purpose is to elucidate the evolution of elevation angles as the variable $Range$ increases. This analysis aids in the precise selection of visible satellites for the vehicle, thereby reducing the computational cost of the code and excluding uninteresting satellites. The provided data pertains to satellites at an elevation of 550 km, a common altitude for Starlink satellites, and is summarized in Table 4.2.

Chapter 5

Performance evaluation

In this chapter, we conduct an in-depth examination of satellite and signal behavior within an urban setting, with a specific focus on the dynamics of elevation angles and their implications for communication reliability. Our simulations meticulously track the movements of designated satellites relative to the Earth's surface. We explore the probability of establishing direct ground-to-satellite communication, considering both elevation angles and the configuration of satellite constellations. The research expands upon delay analysis, delving into the effects of fluctuating packet sizes, frame rates, and server capacity on RTT and across diverse setups as a function of the value of the packet size, the frame rate and the server capacity. The chapter concludes with an evaluation of how changes in elevation angles influence the percentage of data processable by the satellite within a given interval. This thorough analysis yields valuable insights crucial for the optimization of satellite communication systems, particularly in the dynamic context of urban environments.

5.1 Scenario configuration

In our investigation, we explored various setups to enhance our understanding. Specifically, simulations results are given as a function of the type of antenna at the vehicle (VSAT or Reflectarray), the size of the transmitted data packets, the rate at which video frames are sent (FPS), and the satellite computational capacity. In particular, we investigate the following setups.

1. **Default Setup:** 20 Mb packet size sent by the vehicle to the satellite, 30 FPS rate at which the vehicle sends the packets and satellites' computational capacity of 5000 GFLOPs.

2. **Packet Compression Setup:** 5 Mb packet size sent by the vehicle to the satellite, 30 FPS rate at which the vehicle sends the packets and satellites' computational capacity of 5000 GFLOPs.

3. **Reduced Frame Rate Setup:** 20 Mb packet size sent by the vehicle to the satellite, 10 FPS rate at which the vehicle sends the packets and satellites' computational capacity of 5000 GFLOPs.

4. **Increased Computational Capacity Setup:** 20 Mb packet size sent by the vehicle to the satellite, 10 FPS rate at which the vehicle sends the packets and satellites' computational capacity of 10000 GFLOPs.

Our primary focus is on evaluating RTT across different equipment configurations, specifically considering the impact of atmospheric attenuation in the ka-band frequency. The higher frequency range of the ka-band presents both opportunities and challenges, as highlighted in the Channel model of Chapter 2.3, emphasizing the importance of assessing RTT for optimizing communication efficiency.

The simulation involved 100 vehicles engaged in communication cycles over a total duration of 2 seconds. During this period, each vehicle attempted to establish a connection twice, simulating real-world scenarios with intermittent connectivity. To ensure an optimal communication scenario, the vehicle's $Range$ was configured to only consider satellites with an elevation angle above 70 degrees. This careful satellite selection process aimed to enhance the quality of communication links, contributing to a more realistic evaluation of system performance.

5.2 RTT in Optimal Communication Scenario

In our investigation into vehicle-to-satellite communication, we conducted an in-depth analysis focusing on RTT delay graphs within an optimal communication scenario. This scenario involved strategically gathering data for satellites within a $Range$ of 150 km from the vehicles position, ensuring an optimal connection. The chosen $Range$ assures to only get satellites above 70 degrees elevation angle, taking into consideration 4.2 and the satellites having different altitudes with respect to Earths' surface. To simulate real-world conditions, we deployed 100 vehicles across the Earth, taking into account continent sizes and assigning more vehicles to larger continents. Systematically, we collected different metrics for each time slot during data transmission, in a CSV file.

For each vehicle, a dedicated CSV file was generated, containing data for every transmitted packet. The key criterion for determining the optimal communication scenario for each vehicle was based on selecting the satellite with the highest SNR. This process ensured that each vehicle was connected to the most efficient satellite, thus establishing an optimal communication setup.

Upon optimizing communication for each vehicle and selecting the most appropriate satellites based on SNR, we proceeded to aggregate the delay values. This involved calculating the

average delay across all individual transmissions, providing a comprehensive overview of the overall performance of the vehicle-to-satellite communication system in the specified optimal conditions.

Figure 5.1 the graph illustrates the RTT as a function of the type of antenna. Ideally, the RTT should be less than the Inter-Packet Interval (IPI) to facilitate real-time data offloading. In our analysis, the IPI is defined as the frame rate of the video, which is 0.033 seconds for a 30 FPS video and 0.1 seconds for a 10 FPS video. This consideration aligns with the objective of simulating real-time object detection within the context of ITS. These RTT delay graphs provide valuable insights into the performance of different configurations in achieving timely data offloading under optimal communication conditions.

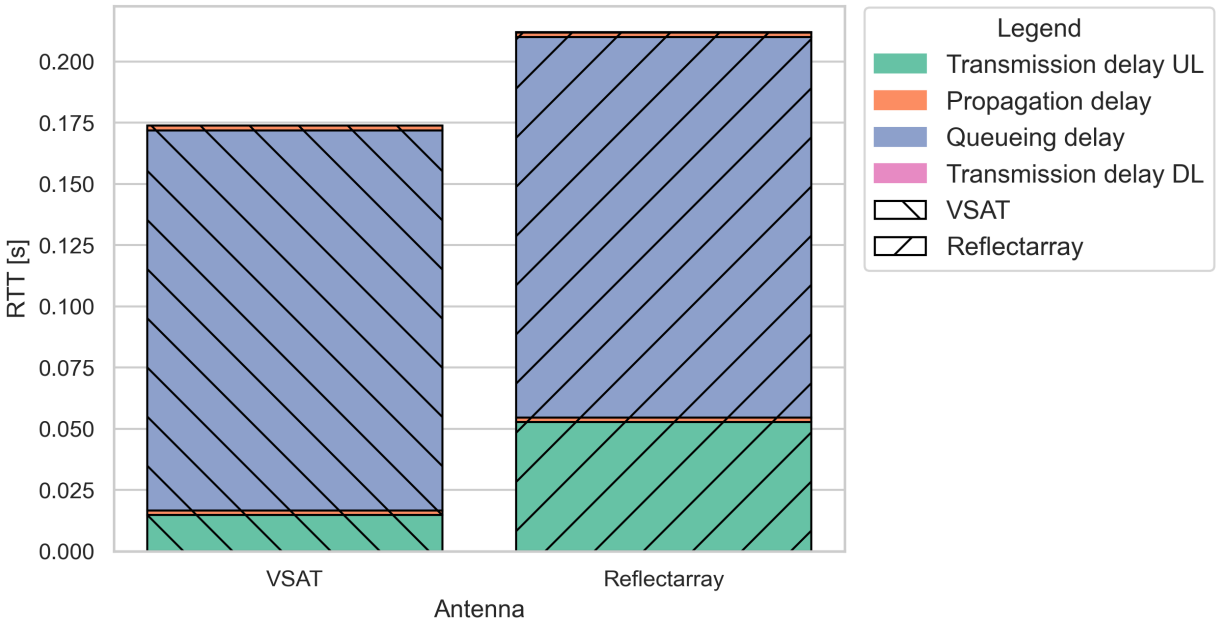

Figure 5.1: RTT delay considering a setup of: 20 Mb packet size, 30 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

What stands out the most at first glance is the high impact of the transmission and queuing delay. The queuing delay is attributed to a combination of factors, with two primary contributors standing out. Firstly, the large packet size of 20 Mb significantly impacts the time required for processing. Secondly, the demand for processing, particularly in the context of image analysis through object detection, exacerbates the delay. Object detection tasks involve intricate algorithms and resource-intensive computations, placing a substantial burden on the processing system. As anticipated, the Reflectarray antenna exhibits significantly higher transmission delays, more than 3 times the transmission delay of the VSAT, influencing the overall RTT. It

is noteworthy that the transmission delay for the downlink was also depicted. Given that the size of the DL packet (i.e., the bounding boxes of the detected objects) sent from the satellite to the ground vehicle is very small, 0.1 Mb, and much smaller than the size of the raw frame sent from the ground vehicle to the satellite, the transmission delay in the downlink path is nearly negligible. As a result, the only relevant delay for the downlink path is the return propagation delay. Looking at the numerical results we can already notice how the RTT delay of packet would greatly exceed the threshold of 0.033s for optimal real-time communication. The results don't exclude the possibility of offloading part of the data to the satellite, in fact by reducing the IPI we also reduce the strain on the satellite which would reduce the overall delay.

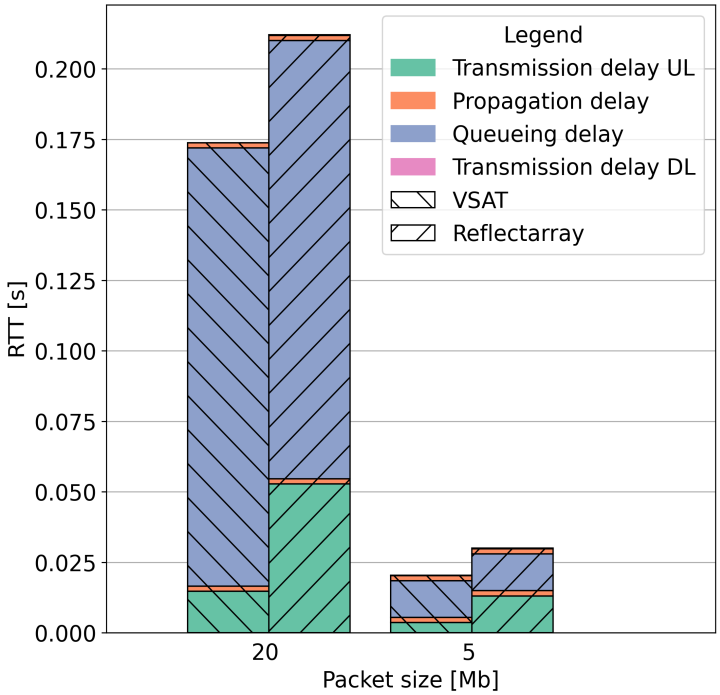

Figure 5.2: RTT delay with compressed transmitted packets. Considering a setup of: 5 Mb packet size, 30 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

In Figure 5.2 we examine a situation in which the frame of the video of 20 Mb (1280 x 640) is compressed to a 5 Mb frame (640 x 320) before transmitting it to the satellite, analyzing how the delay is impacted by the operation of compression. While packet compression undeniably contributes to a notable reduction in delay, it may introduce some challenges, especially if the quality of the resulting data after compression is too low to jeoparize accurate object detection [41]. Looking at the raw number of the tables we notice that the RTTs of both the VSAT and Reflectarray are below 0.033 s (VSAT=0.018, Reflectarray=0.028). Several conclusion can be made based on how the data is interpreted. Imagining an ideal scenario in which the data would be recorded and sent in the size of 5 Mb, real-time communication would be feasible for both

antennas. While considering the problem realistically the need of compression would inherently enhance the delay, meaning that we would have 0.033-RTT seconds to compress the image at the vehicle and the decompress it at the satellite, making the feasibility of complete offloading an optimization task between several intermediate processing steps. Much more reasonable would be to consider partial offloading in order to avoid the intricacies of having to deal with such challenging task.

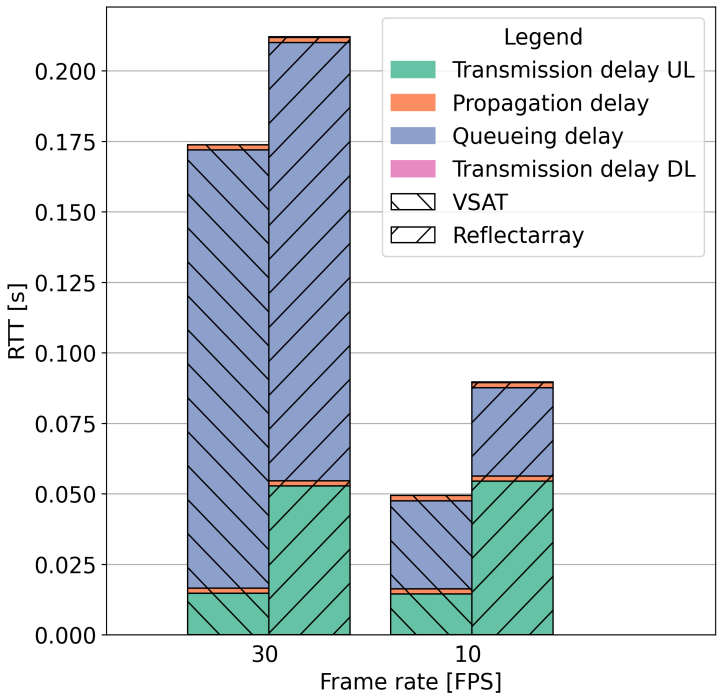

Figure 5.3: RTT delay with reduced packet inter-arrival time. Considering a setup of: 20 Mb packet size, 10 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

While reducing the frame rate from 30 FPS to 10 FPS as seen in Figure 5.3 can indeed mitigate delays in the queuing system. However, this approach reduces the reactiveness of the system to possible changes in the environment, which may be critical in ITS scenarios. Optimal performance requires considering not only frame rate but also the nature of the information being conveyed. Ensuring that the trade-offs between improved delays and potential drawbacks are carefully evaluated to achieve a well-balanced and effective system. Notice that reducing the FPS is equivalent to reducing the amount of data that is offloaded to the satellite. Specifically, reducing the frame rate from 30 to 10 fps is equivalent to consider data generated at 30 fps, with only 1/3 of this data to be offloaded to the satellite.

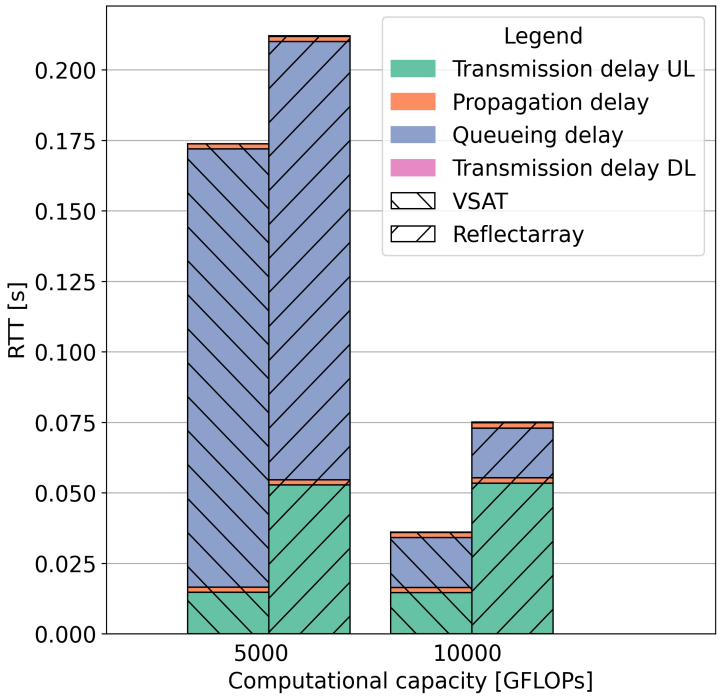

Figure 5.4: RTT delay with increased server computation capacity. Considering a setup of: 20 Mb packet size, 30 FPS and a computational capacity of the satellite of 10000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

In the context of optimizing system efficiency, increasing the computational capacity of satellites emerges as an important factor aimed at mitigating queuing delays. The ongoing evolution of Starlink satellites, transitioning to the enhanced Version 2 (V2) with augmented computational capabilities, underscores the significance of this approach. While the perceptible improvements in the queuing system resulting from an increased computational capability are noteworthy, an examination of the associated dynamics highlights the indispensable role of antenna selection. Despite the improvements made in computational efficiency, it is important to acknowledge that temporal constraints linked to data transmission endure. Figure 5.4 provides a visual representation, illustrating the impact of upgraded computational infrastructure—from 5000 GFLOPs to 10000 GFLOPs. This depiction vividly showcases a significant reduction in delays attributed to enhanced computational capacity. While estimating the real values of the computational capacity of a single satellite remains a challenge in itself some considerations can be made. Looking at the vehicle with a VSAT mounted antenna on it we reach an overall RTT of 0,034 s which is extremely close to the threshold we set for real-time communication. With the little difference between IPI and RTT we can make some realistic consideration about complete offloading of data in real-time. While RTT is still slightly higher than the IPI the difference is almost negligible, considering the context of an ITS system where real-time responsiveness is essential in order to enhance driving safety, returning a packet with a 1 ms delay would be

still considered tolerable. While for the Reflectarray it would be interesting to evaluate an improvement of its performance by reducing the offloading on the satellite having now a higher computational capacity. It is easy to notice that such consideration can't be made. Looking at the transmission delay in UL of the Reflectarray, we notice that it is 0.053 s, already higher than the threshold value. This underlines the statements made at the beginning of the graphs analysis which involves balancing the queuing and transmission delay evaluating beforehand which one is the bottleneck.

5.2.1 Observations

In the realm of ITS, the efficient offloading of data from a vehicle to a satellite is critical for real-time object detection. Examining the graphs related to this process reveals identifiable delay elements influenced by various configurations. Let's delve into the specific aspects that impact the performance of this data offloading process.

Antenna Efficiency: Shifting to a less efficient antenna can significantly increase the overall delays in data offloading. Consider a scenario where a vehicle, equipped with a less efficient antenna, transmits data to a satellite. The smaller antenna gain may result in a weaker signal, leading to increased transmission time and potentially compromising the real-time nature of object detection. Therefore, maintaining high-quality, efficient antennas is crucial for minimizing delays in data transmission.

Packet Compression: The impact of packet compression is substantial in the context of offloading data for real-time object detection. Compression, while reducing the amount of data to be transmitted, introduces a computational workload and processing time. This trade-off becomes crucial as the computational workload can potentially offset the benefits of reduced transmission time. Compression can potentially lead to extended IPIs in vehicles due to the additional time spent compressing the packet, thereby impacting the overall delay in transmitting data to the satellite for object detection.

Queuing System Bottleneck: In situations where the queuing system is the bottleneck, modifying packet size becomes a key factor. If the queuing system is struggling to handle the incoming data, either reducing the frame rate or enhancing the computational capacity can significantly improve the overall delay. For example, optimizing the inter packet interval based on the capabilities of the queuing system can prevent congestion and ensure timely data transmission to the satellite for object detection.

Balancing Transmission and Queuing System Delay: Effective minimization of both queuing system delay and transmission delay simultaneously requires a careful balance. For instance, if the queuing system delay is dominant, adjustments in frame rate or computational capacity are essential. On the other hand, if transmission delay is the primary concern, optimiz-

ing packet size becomes paramount. Striking the right balance ensures that data offloading is not only timely but also efficient in supporting real-time object detection in ITS.

Offloading optimization: In Figure 5.3 we noticed the duality of the analysis, which would be interesting to further develop in combination to other improvements. We noticed that even by offloading 33.3% of the packets towards the satellite we still wouldn't be able to receive them in reasonable amount of time for real-time data offloading. Therefore considering the improvements done by only partially offloading data towards a satellite in the default setup, lets evaluate how the changes would affect the offloading in combination with an improved satellite computational capacity. As we have seen for the VSAT it is already able to process almost all the data of the vehicle on its own. While considering the more interesting case of the Reflectarray we try to evaluate how a 33.3% offloading would affect it's RTT. Knowing that we improved the signal from 0.21 s of the default setup to 0.088 s by offloading only part of the data we apply the same improvement by making a proportion on the queuing delay knowing that reducing the FPS only affects that specific delay. Now considering the default setup as: Reflectarray antenna, sending 20 Mb packets with a satellite computational capacity of 10000 GFLOPs and knowing that the queuing delay of such setup is 0.018 s we apply a proportion: $\frac{0.031}{0.15} = \frac{x}{0.018}$. With the result of x being 0.004 s, the operation applied would improve a lot the queuing delay, but in accordance with the analysis done for Figure 5.4 this improvements mean nothing knowing that the transmission delay in uplink is already higher than the 0.033 s threshold. Therefore for the Reflectarray considering a packet compression in order to to reduce the transmission delay would lead to better results. Applying the same concept as previously applying a proportion between transmission and queuing delay.

Transmission delay UL (t_{UL}):

$$\frac{0.013}{0.031} = \frac{t_{\text{UL}}}{0.018}$$

Queuing delay ($t_{p,\text{SAT}}$):

$$\frac{0.013}{0.053} = \frac{t_{p,\text{SAT}}}{0.053}$$

Adding the new calculated delays $t_{\text{UL}} = 0.0075$ and $t_{p,\text{SAT}} = 0.013$ to the old we obtain the new RTT=0.022 s, demonstrating how the packet reduction could be applied to a new setup in order to highly improve its RTT.

5.3 Satellite Dynamics and Signal Behaviour

In our study, we examined how the elevation angles of three specific satellites, each associated with a unique vehicle, change dynamically relative to the Earth's surface—a crucial aspect of satellite communication systems.

The simulation commences by selecting a satellite for analysis, identifying it by its name, and positioning a vehicle beneath it. The script then selects the appropriate Starlink satellite from the TLE data based on the provided name as seen in Chapter 3. This combination of satellite and vehicle initializes a specific scenarios, which is simulated over defined time period, considering factors such as start time, end time, and time step.

In our initial investigation, we focused on understanding the dynamic changes in elevation angles. We selected a known satellite's position at a specific moment and placed a vehicle beneath it, capturing the resulting movement by choosing a start time and end time aligned with the satellite's trajectory.

The data for this specific scenario was captured using a time step of 1 second, tracking the satellite from the moment it entered a range of 2000 km on Earth from the vehicles' position, observing its elevation angle from 6 degrees to almost 90 degrees and back to 6 degrees.

The simulation output is stored in CSV files, providing insights into satellite behavior, particularly the dynamics of elevation angles and their implications for satellite communication systems.

Figure 5.5 represents the satellite movements from the Earth's surface perspective. Notably, as a satellite approached an elevation angle of 90 degrees, the rate of change in its angle decreased. Conversely, as it moved towards the horizon, the elevation angle showed reduced variability. This is because of the relative motion of the satellite with respect to and observer on the Earth's surface. Conversely the changes in elevation angle over time would appear smoother to an observer placed at the centre of the satellites' orbit. As expected, our analysis indicated that a satellite typically remains visible for around 10 minutes.

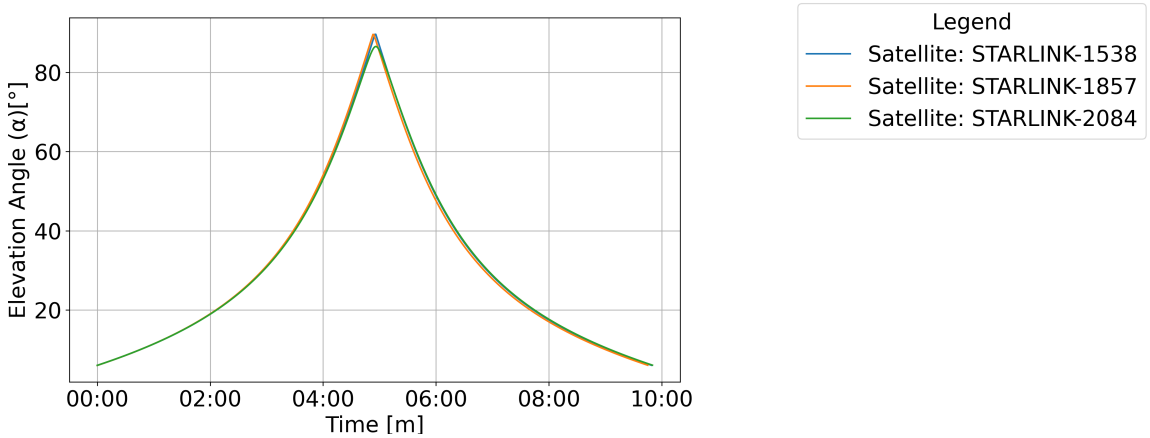

Figure 5.5: Elevation angle over time for 3 different satellites

By exploring these elevation angle dynamics, we gain insights into satellite behavior and its interaction with the Earth's surface. This knowledge is crucial for optimizing satellite commu-

nication systems, ensuring efficient and reliable connectivity.

In the next phase of our study, we studied how the SNR changes over time, specifically correlating these changes with shifts in elevation angles. We visually represented these dynamics through two distinct graphs, Figure 5.6 and 5.7. Our investigation focused on the Ka-band, using the channel model outlined in Chapter 2.

To add diversity to our exploration, we assessed signal performance using two different antennas. Initially, we used the default equipment, a VSAT antenna, for the analysis. Subsequently, we switched to a specialized Reflectarray antenna designed for the internet of vehicles.

This exploration helps us understand how the quality of the signal varies over time, considering different elevation angles and antenna types. It contributes valuable insights into optimizing satellite communication systems for reliable and efficient connectivity.

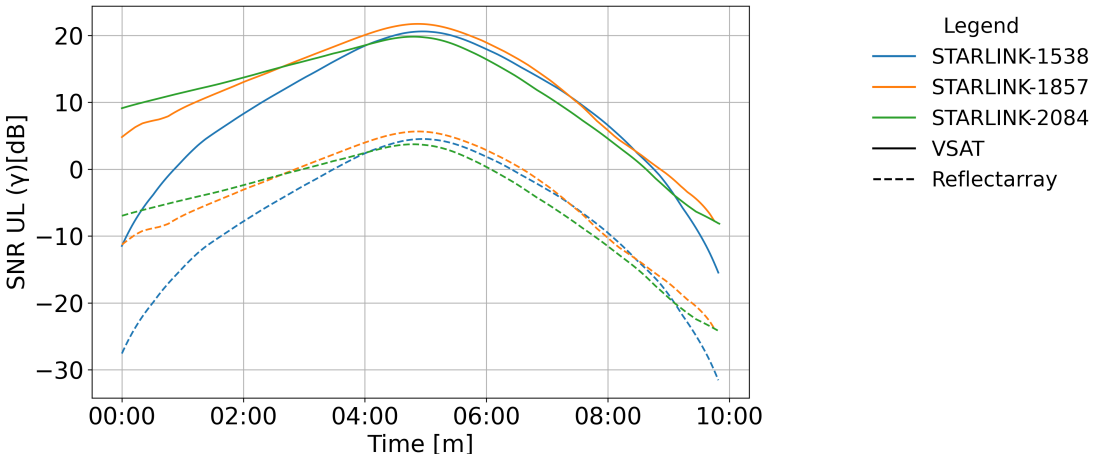

Figure 5.6: Comparison of SNR between a VSAT and a Reflectarray.

Figure 5.6 displays how the SNR changes over time. the SNR decreases as the elevation angle decreases because of the resulting larger distance with respect to the ground vehicles, which makes the path loss increase accordingly.

Comparing VSAT and Reflectarray antennas, the VSAT's higher antenna gain is evident, leading to better SNR values. Both antennas show similar patterns, with the VSAT consistently maintaining strong SNR. It hits about 15 dB at a 30-degree elevation, peaking at approximately 25 dB at 90 degrees. These signal fluctuations align with the satellite's position and elevation angle, as seen in Figure 5.7.

Analyzing Figures 5.6 and 5.7 reveals slight signal variations among different satellites, mainly due to atmospheric losses considered in the simulator. The impact is noticeable at the graph's edges in Figure 5.6 and around 30-degree angles in Figure 5.6. Atmospheric losses affect path loss more at lower angles, where signals travel through a greater atmospheric distance.

Moreover it is important to notice how the atmospheric loss impacts differently based on the satellite chosen, the reason being that the simulator considers the atmospheric loss in accordance to the receiver position, in this case the satellite.

Examining the link between GV and satellites requires careful consideration of real-world scenarios where physical structures, like buildings, may obstruct the communication path. The presence of high-rise buildings in densely populated areas can disrupt the direct line of sight between ground vehicles and satellites, impacting data transmission. In the context of mmWave and the great distances of satellite communication, this challenge becomes even more pronounced, being mmWaves particularly sensitive to obstacles. Therefore the loss of LoS would attenuate the signal to the point that the link would be considered lost.

For instance referencing 4.1:

- At 30 degrees elevation:
  - Dense Urban: 46.8% line of sight probability
  - Urban: 61.3% line of sight probability
  - Suburban and Rural: 92.9% line of sight probability

Figure 5.7 highlights how the signal evolves with respect the problems' geometry. Conversely, to the VSAT the Reflectarray, with its lower gain, exhibited less reliability. The maximum dB for this antenna can be seen reaching 6 dB, which could be considered already a scarce link quality to begin with, it is therefore important to draw a line where the geometry of the communication makes the communication unreasonable. In fact looking at the graph it is possible to examine how the SNR varies with respect to the elevation angle showcasing when the Reflectarrays' communication becomes critical. The graphs of the SNR changes with respect to time and elevation angle helps us to effectively understand how to implement the correct technology in order to take maximal advantage when establishing a communication between a vehicle and a satellite. This becomes particularly important in the context of offloading information from a vehicle to a satellite to enhance processing power. In fact understanding the graphs help us determine the intervals in which it is most efficient to establish a communication also taking into consideration the available devices. This information is valuable for maximizing the effectiveness of data transfer between vehicles and satellites.

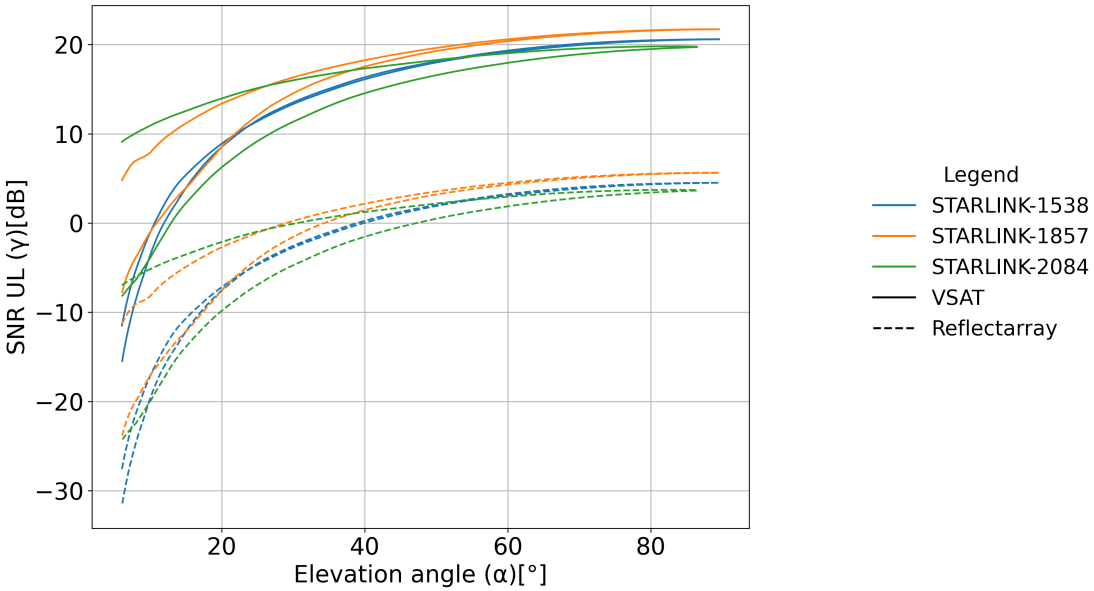

Figure 5.7: Comparison of SNR between a VSAT and a Reflectarray.

In Figure 5.8 we plot the delay with respect to the elevation angle. We can see that the delay is very large for small values of the elevation angle due to the resulting small SNR, as illustrated in Figure 5.7.

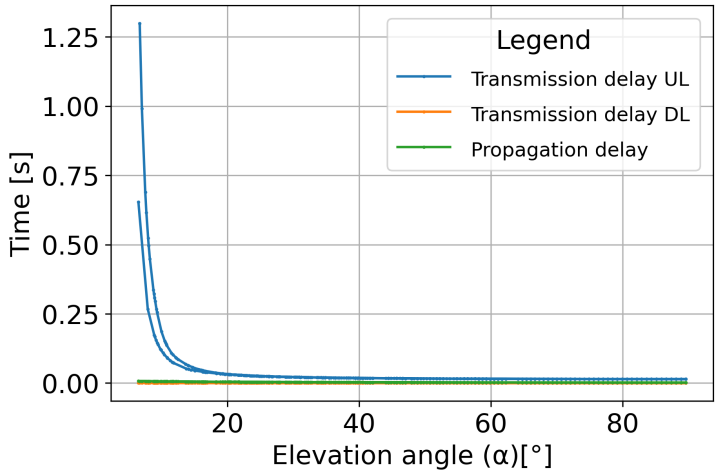

Figure 5.8: Delay comparison: propagation delay, transmission delay in uplink and downlink

Figure 5.9 represents a zoom of Figure 5.8 to better see the difference in terms of delay. The results obtained from the graphs confirm the choices made during the optimization process in Chapter 5.2.

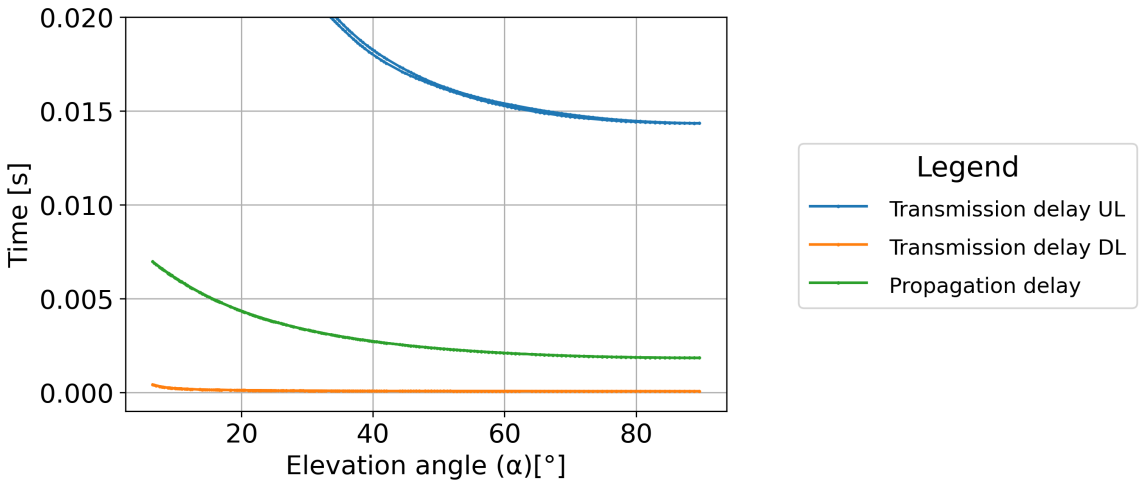

Figure 5.9: Normalized propagation delay, transmission delay in uplink and downlink

In Figure 5.10 we explore the transmission delays in vehicle-to-satellite communication considering satellite-specific dynamics. To facilitate comparison across varying delays and to discern any notable differences, the delay values have been normalized with respect to their maximum value measured, approximately around 6 degrees. The uplink transmission delay exhibits a remarkably steep slope from 15 degrees to 0 degrees elevation angle, indicating a rapid increase in delay as the satellite approaches the horizon. This heightened sensitivity is attributed to the presence of larger packet sizes in the uplink dew to the equations used in Chapter 2 (2.18), contributing to an exponential rise in transmission time. Similarly, the downlink transmission delay also demonstrates a steep slope from 15 degrees to 0 degrees elevation angle, although not as pronounced as in the uplink. The consistent steepness in both cases aligns with the observation that downlink packets are considerably smaller. Importantly, the propagation delay, determined by the distance between the satellite and the vehicle, shows the least steep slope, suggesting a more consistent delay across different elevation angles. The overall inverse exponential nature of the plots underscores the expected decrease in delays as the elevation angle increases, indicating a more direct line of sight between the vehicle and the satellite.

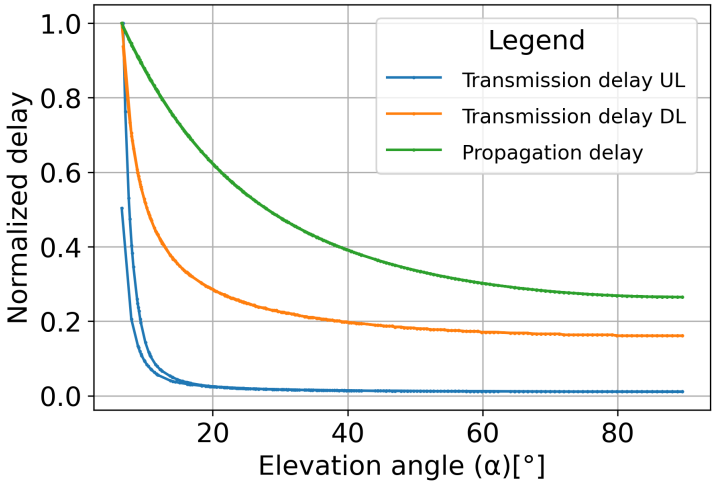

Figure 5.10: Normalized propagation delay, transmission delay in uplink and downlink

Exploring the influence of the number of satellites on elevation angles is crucial for understanding satellite communication reliability. Intuitively, it seems more satellites would enhance the communication experience.

To investigate this, we considered a scenario deploying 1000 vehicles globally, aiming to establish communication twice within a 2-second interval. We assessed elevation angles during communication, filtering data to identify the best angles at each interval. Subsequently, we plotted a graph depicting the cumulative probability of having a certain elevation angle based on the considered satellite constellation.

Analyzing the results of Figure 5.11, the steeper curve of the 4550-satellite constellation indicates a quicker approach to high elevation angles. Conversely, a more linear curve suggests a more random likelihood of achieving a specific elevation angle. The data supports the idea that a superior satellite constellation increases the likelihood of obtaining favorable elevation angles.

In simpler terms, a well-structured satellite network enhances the probability of achieving good elevation angles during communication, ultimately improving the reliability of satellite-based connectivity.

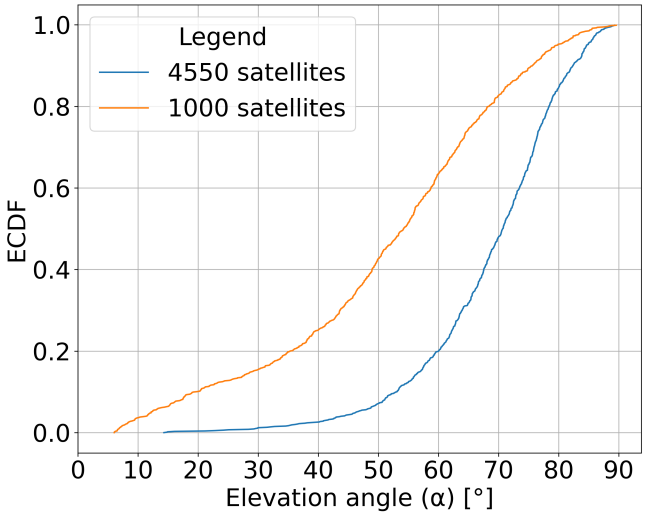

Figure 5.11: Cumulative probability of having a certain elevation angle between 4550 and 1000 satellite constellations

Our in-depth exploration of satellite and signal behavior in vehicle-to-satellite communication systems provides valuable insights into optimizing communication strategies. The analysis of elevation angle dynamics showcases the intricate interplay between transmission conditions and satellite performance. The observed patterns in satellite movements and signal-to-noise ratio fluctuations highlight the importance of considering factors such as elevation angles and antenna types for reliable and efficient connectivity. Notably, the examination of different satellite constellations reinforces the idea that a well-structured and populated satellite network, exemplified by the 4550-satellite constellation, significantly enhances the probability of achieving favorable elevation angles during communication.

5.4 Real-time Feasibility

The objective of the analysis is to establish how many packets the satellite is able to process within 1 second, taking into consideration the impact of the LoS in the communication. Relative to the actual number of packets sent from the vehicle.

In the default setup, offloading data to the satellite appears challenging as it is possible to see in Figure 5.12. At a frame rate of 30 frames per second, with a packet size of 20 Mb, and when equipped with a VSAT antenna, the system successfully returns 16.67% of the total number of packets within an interval of 1 second. On the other hand, if a Reflectarray antenna is employed, the percentage of offloadable packets slightly decreases, with only 13.55% of the packets being successfully transmitted within the same time frame. This results are in line with those obtained in Section 5.2.

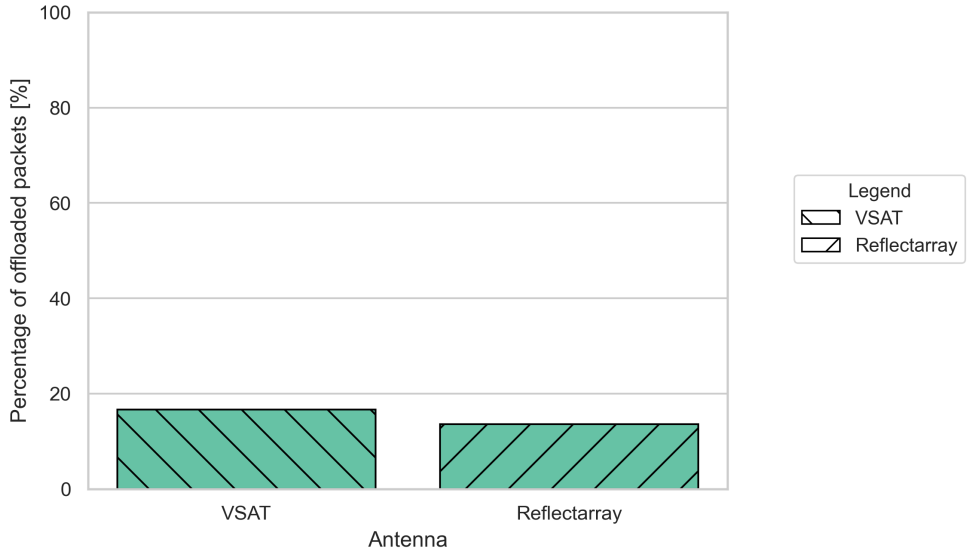

Figure 5.12: Satellite processing percentage of incoming data within a 1 second interval. Considering a setup of: 20 Mb packet size, 30 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

Consistently with the findings in Chapter 5.2, compressing data before transmission significantly improves communication as we can see from Figure 5.13, making complete offloading to a satellite more feasible. The histogram shows us that it is possible to offload 100% of the packets to the satellite but is important to notice that some might not be sent due to the line of sight blocking the signal, therefore resulting in 94.79% for VSAT and 95.37% for the Reflectarray instead of the 100% offloaded data.

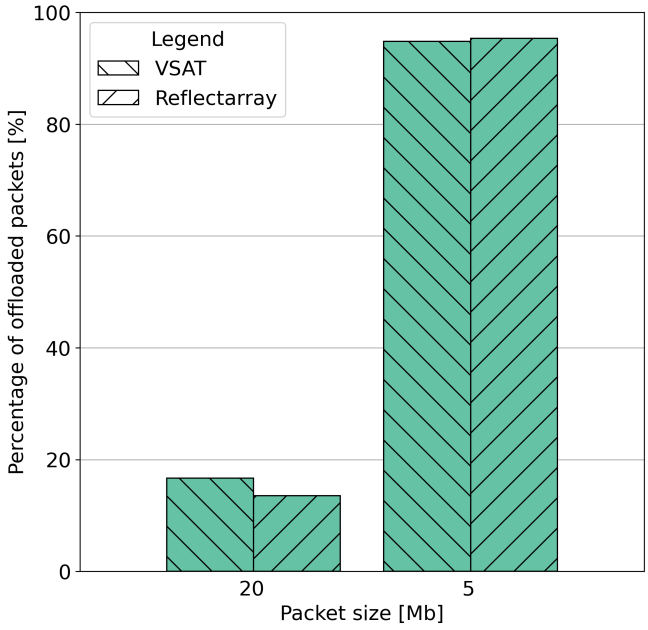

Figure 5.13: Satellite processing percentage of incoming data within a 1 second interval compressing transmitted packets. Considering a setup of: 5 Mb packet size, 30 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

With reduced frame rate from 30 to 10 FPS, both VSAT and the Reflectarray can process and return 95.90% and 96.46% of packets in a second, respectively.

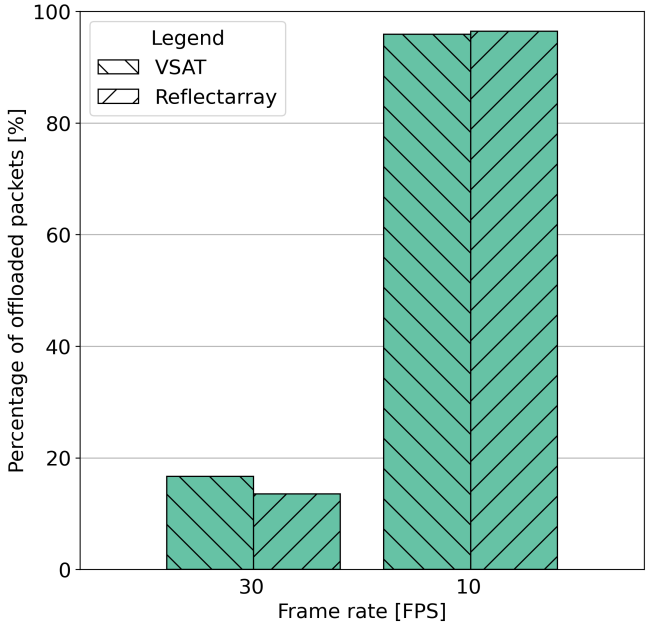

Figure 5.14: Satellite processing percentage of incoming data within a 1 second interval reducing FPS. Considering a setup of: 20 Mb packet size, 10 FPS and a computational capacity of the satellite of 5000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

Increasing the computational capacity of the server enables the processing of 93.12% of packets. However, for the Reflectarray, only 43.59% of the packets can be processed by satellite in 1 second, less than half of the other antenna.

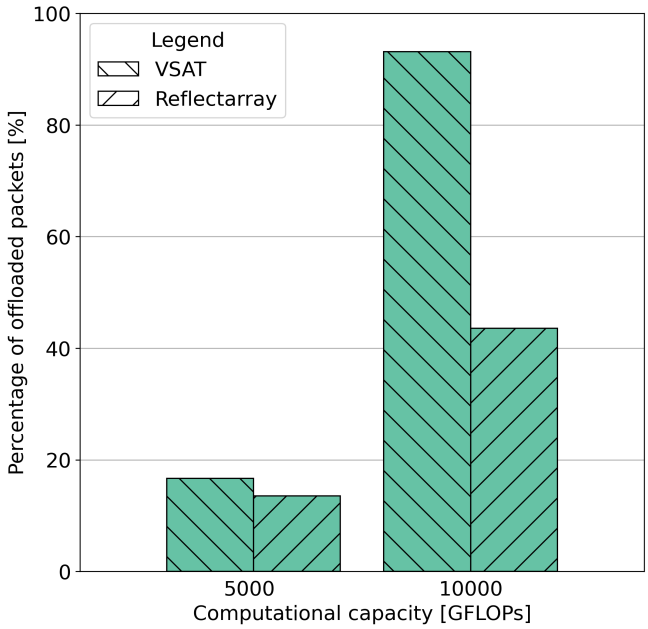

Figure 5.15: Satellite processing percentage of incoming data within a 1 second interval increasing server computational capacity. Considering a setup of: 20 Mb packet size, 30 FPS and a computational capacity of the satellite of 10000 GFLOPs. The setup is tried on two different antennas: VSAT and Reflectarray.

The results obtained from the graphs are consistent with the results obtained in Section 5.2. These findings underscore the importance of considering multiple parameters when optimizing satellite offloading.

Chapter 6

Conclusion

In this thesis, we have explored the paradigm of satellite offloading within the context of VEC, as a solution for the satellite to process automotive data generated from ground vehicles. The focus on satellite communication technologies aimed at understanding their influence on the RTT and how it affects real-time communication in the context of having to process partial or total data transmitted by the vehicle.

The simulations have revealed the impact of the antenna design on the feasibility of real-time communication. We showed that with a Reflectarray antenna the RTT was as high as 200 ms, while with a VSAT antenna it could be reduced to less than 175 ms. Then, we recognized the importance of data compression, and observed that reducing the data size from 20 to 5 Mb could reduce the RTT to only 2.5 ms. Another way to improve the RTT was to reduce the inter-packet interval, that is the frame rate for data offloading. Specifically, the RTT goes from 200 ms with a frame rate of 30 fps to only 75 ms with a frame rate of 10 fps. Finally, we explored the impact of the computational capacity, and showed that the RTT can be as small as 75 ms with a capacity of 10000 GFLOPs.

Then, we considered real Starlink traces to monitor the time-varying evolution of the satellites, and evaluate the impact of the satellite dynamics on the delay. We showed via simulations that the visibility period of a single satellite station is less than 10 minutes, and the SNR is larger than 0 dB, which is a lower-bound threshold to enable data offloading, only when the elevation angle between the satellite and the ground vehicle is between 70 and 90 degrees. In this range, the RTT is around 10 ms, or lower when increasing the number of satellites in the constellation.

Finally, we run additional simulations to evaluate the probability of the satellite to receive and process data frames in real time. It turned out that this is feasible only if data compression is applied before transmission, and under certain configurations in terms of frame rate, antenna design, and computational capacity.

As part of our future work, the aim is to develop an algorithm that dynamically determines

the optimal offloading factor based on the circumstances, effectively distributing computational tasks between vehicles and LEO satellites. The offloading factor will be tuned to minimize overall delays while balancing energy consumption.

Bibliography

[1] X. Xu, Y. Xue, X. Li, L. Qi, and S. Wan, "A computation offloading method for edge computing with vehicle-to-everything," *IEEE Access*, vol. 7, pp. 131 068–131 077, 2019. doi: 10.1109/ACCESS.2019.2940295.

[2] T. Higuchi, M. Giordani, A. Zanella, M. Zorzi, and O. Altintas, "Value-anticipating v2v communications for cooperative perception," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1947–1952. doi: 10.1109/IVS.2019.8814110.

[3] X. Zhang, H. Gao, M. Guo, G. Li, Y. Liu, and D. Li, "A study on key technologies of unmanned driving," *CAAI Transactions on Intelligence Technology*, vol. 1, no. 1, pp. 4–13, 2016, issn: 2468-2322. doi: https://doi.org/10.1016/j.trit.2016.03.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2468232216000044.

[4] M. Giordani, A. Zanella, T. Higuchi, O. Altintas, and M. Zorzi, "Performance study of lte and mmwave in vehicle-to-network communications," in *2018 17th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2018, pp. 1–7. doi: 10.23919/MedHocNet.2018.8407093.

[5] S. Galea, D. Seychell, and M. Bugeja, "A survey of intelligent transportation systems based modern object detectors under night-time conditions," in *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, 2020, pp. 265–270. doi: 10.1109/ICISS49785.2020.9316076.

[6] G. Melotti, C. Premebida, and N. Gonçalves, "Multimodal deep-learning for object recognition combining camera and lidar data," in *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2020, pp. 177–182. doi: 10.1109/ICARSC49921.2020.9096138.

[7] Y.-J. Choi, J. Hur, H.-Y. Jeong, and C. Joo, "Special issue on v2x communications and networks," *Journal of Communications and Networks*, vol. 19, no. 3, pp. 205–208, 2017. doi: 10.1109/JCN.2017.000037.

[8] N. Kumar, J. J. P. C. Rodrigues, and N. Chilamkurti, "Bayesian coalition game as-a-service for content distribution in internet of vehicles," *IEEE Internet of Things Journal*, vol. 1, no. 6, pp. 544–555, 2014. doi: 10.1109/JIOT.2014.2374606.

[9] A. Traspadini, M. Giordani, and M. Zorzi, "Uav/hap-assisted vehicular edge computing in 6g: Where and what to offload?" *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pp. 178–183, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247025914.

[10] A. Traspadini, M. Giordani, G. Giambene, and M. Zorzi, "Real-time hap-assisted vehicular edge computing for rural areas," *IEEE Wireless Communications Letters*, vol. 12, no. 4, pp. 674–678, 2023. doi: 10.1109/LWC.2023.3238851.

[11] J. Feng, Z. Liu, C. Wu, and Y. Ji, "Ave: Autonomous vehicular edge computing framework with aco-based scheduling," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 12, pp. 10 660–10 675, 2017. doi: 10.1109/TVT.2017.2714704.

[12] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[13] M. Giordani and M. Zorzi, "Non-terrestrial networks in the 6g era: Challenges and opportunities," *IEEE Network*, vol. 35, no. 2, pp. 244–251, 2021. doi: 10.1109/MNET.011.2000493.

[14] D. Wang, A. Traspadini, M. Giordani, M.-S. Alouini, and M. Zorzi, "On the performance of non-terrestrial networks to support the internet of things," *2022 56th Asilomar Conference on Signals, Systems, and Computers*, pp. 881–887, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:254246282.

[15] N. Muhammad, M. Danish, A. Syed, and Pasha, "Satellite communication: State-of-the-art and future challenges," Jun. 2023.

[16] M. Tropea, "State-of-the-art in satellite communication networks," *Electronics*, vol. 11, no. 9, 2022.

[17] European Space Agency. "ESA - European Space Agency." Accessed on Date of Access. (Year of Access), [Online]. Available: https://www.esa.int/Enabling_Support/Space_Transportation/Types_of_orbits.

[18] M. Furqan and B. Goswami, "Satellite communication networks," in Jan. 2022, pp. 1–22, isbn: 978-981-4585-87-3. doi: 10.1007/978-981-4585-87-3_70-1.

[19] A. Chaoub, M. Giordani, B. Lall, *et al.*, "6g for bridging the digital divide: Wireless connectivity to remote areas," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 160–168, 2022. doi: 10.1109/MWC.001.2100137.

[20] A. Guidotti, S. Cioni, G. Colavolpe, *et al.*, "Architectures, standardisation, and procedures for 5g satellite communications: A survey," *Computer Networks*, vol. 183, p. 107 588, 2020, issn: 1389-1286. doi: https://doi.org/10.1016/j.comnet.2020.107588. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912862031224X.

[21] A. Raman, M. Varvello, H. Chang, N. Sastry, and Y. Zaki, *Dissecting the performance of satellite network operators*, 2023. arXiv: 2310.15808 [cs.NI].

[22] S. Liu, Z. Gao, Y. Wu, *et al.*, "Leo satellite constellations for 5g and beyond: How will they reshape vertical domains?" *IEEE Communications Magazine*, vol. 59, no. 7, pp. 30–36, 2021. doi: 10.1109/MCOM.001.2001081.

[23] T. Pultarova, E. Howell, D. Dobrijevic, and A. Mann, "Starlink satellites: Everything you need to know about the controversial internet megaconstellation," 2023, Last updated August 02, 2023.

[24] F. Hawlader, F. Robinet, and R. Frank, "Vehicle-to-infrastructure communication for real-time object detection in autonomous driving," in *2023 18th Wireless On-Demand Network Systems and Services Conference (WONS)*, 2023, pp. 40–46. doi: 10.23919/WONS57325.2023.10061953.

[25] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2092–2104, 2020. doi: 10.1109/TVT.2019.2959410.

[26] S. S. Shinde and D. Tarchi, "Network selection and computation offloading in non-terrestrial network edge computing environments for vehicular applications," in *2022 11th Advanced Satellite Multimedia Systems Conference and the 17th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, 2022, pp. 1–8. doi: 10.1109/ASMS/SPSC55670.2022.9914757.

[27] 3rd Generation Partnership Project, "3GPP TR 38.811 V15.4.0 (2020-09)," Technical Specification Group Radio Access Network, Technical Report 3GPP TR 38.811, 2020, Study on New Radio (NR) to Support Non-Terrestrial Networks (Release 15).

[28] *Celestrak - supplemental two-line elements*, https://celestrak.org/NORAD/elements/supplemental/.

[29] *United states space force*, https://www.spaceforce.mil/.

[30] P. Testolina, F. Barbato, U. Michieli, M. Giordani, P. Zanuttigh, and M. Zorzi, "Selma: Semantic large-scale multimodal acquisitions in variable weather, daytime and viewpoints," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, pp. 7012–7024, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248299911.

[31] S. S. Lee and C. D. Hall, "An analytic solution of full-sky spherical geometry for satellite relative motions," *Scientific Reports*, vol. 11, no. 1, p. 9075, 2021.

[32] M. Geyer, "Geometric analysis of an observer on a spherical earth and an aircraft or satellite," John A. Volpe National Transportation Systems Center (U.S.), Tech. Rep., Sep. 2013. [Online]. Available: https://rosap.ntl.bts.gov/view/dot/10098.

[33] J. Liu, W. Long, Y. Wu, J. Xu, J. Sang, and X. Lei, "Tle orbit determination using simplex method," *Geodesy and Geodynamics*, vol. 14, no. 5, pp. 438–455, 2023.

[34] J.-L. Lefebvre, *Space Strategy, First Edition*. ISTE Ltd and John Wiley & Sons, Inc., 2017.

[35] International Telecommunication Union (ITU), "Attenuation by atmospheric gases and related effects," ITU-R, Tech. Rep. P.676-12, 2019, Radiowave Propagation.

[36] B. Jansson, "Choosing a good appointment system-a study of queues of the type (d, m, 1)," *Operations Research*, vol. 14, no. 2, pp. 292–312, 1966. [Online]. Available: https://www.jstor.org/stable/168256.

[37] N. S. S. T. Program. "State of the art report on communications." (), [Online]. Available: https://www.nasa.gov/smallsat-institute/sst-soa/soa-communications/.

[38] W. Li and K. Liu, "Confidence-aware object detection based on mobilenetv2 for autonomous driving," *Sensors*, vol. 21, no. 7, 2021.

[39] P. Cassará, A. Gotta, M. Marchese, and F. Patrone, "Orbital edge offloading on mega-leo satellite constellations for equal access to computing," *IEEE Communications Magazine*, vol. 60, no. 4, pp. 32–36, 2022. doi: 10.1109/MCOM.001.2100818.

[40] I. Merino-Fernandez, S. L. Khemchandani, J. del Pino, and J. Saiz-Perez, "Phased array antenna analysis workflow applied to gateways for leo satellite communications," *Sensors*, vol. 22, no. 23, 2022.

[41] A. Varischio, F. Mandruzzato, M. Bullo, M. Giordani, P. Testolina, and M. Zorzi, "Hybrid point cloud semantic compression for automotive sensors: A performance evaluation," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6. doi: 10.1109/ICC42927.2021.9500523.