University of Milan

Faculty of Science and Technology

Computer Science Master's Degree

Performance evaluation of Tele-Operated Driving service in a heterogeneous urban environment

Advisor: Christian Quadri

Thesis by:

Davide Busolin

Student ID: **988414**

Academic Year 2022/2023

People think computers will keep them from making mistakes.
They're wrong. With computers you make mistakes faster.
— Adam Osborne

Contents

List of Figures

Introduction

Self-driving vehicles have been the topic of extensive research and experimentation for a number of years, with auto makers and tech companies announcing major breakthroughs and the viability of the technology for public use getting closer and closer. There have been some public tests and recent service deployments, like the San Francisco robo-taxi service.

Nevertheless, even as constant training for AI models makes them better by the day and advancements in sensor technology make them more reliable, there will be situations where human intervention is required, be it a software crash, a sensor malfunction, or simply an edge case that is not contemplated by the training the model has received. Until recently, this has been accomplished by simply having a human in the vehicle at all times ready to intervene, but a possible novel solution is called *Tele-Operated Driving*. Tele-Operated Driving (ToD) is a service that allows an operator, human or machine, to remotely take control of a Host Vehicle (HV) and complete the required driving tasks. This type of service is enabled by advancements in cellular network technology such as 5G. In the case of a human operator, a virtual cockpit is set up to reproduce the vehicle's environment as it is actively captured by on-board sensors and cameras, which are also used for the autonomous driving tasks. The operator then sends back control inputs that are actuated by the vehicle.

Studies were performed using previous technological iterations, i.e. 4G LTE. In [7], *Liu et al.* conducted experiments on remote driving over an LTE network, collecting data from both a scaled-size prototype and real-world measurements of publicly accessible mobile networks. Using a low-definition video feed, they registered RTT values between 56ms and 358ms, averaging around 100ms but with a lot of variance. In their experiments with human candidates, they concluded that with current technology at the time, remote driving was still a questionable endeavor, mainly due to the variability of network conditions.

Previous work [4] has laid the groundwork for realistic simulation of both vehicle physics and a 5G communication network by creating a middleware that interfaces OMNeT++, INET, and the Simu5G frameworks with the CARLA driving simulator. The goal of this thesis is to carry out a performance evaluation of the ToD infrastructure against a set of scenarios, that take the form of a route that the HV must navigate which includes both urban and highway sections and some experiments with obstacles that force the vehicle to perform an emergency stop. Each of those scenarios is tested under a set of parameter configurations that influence the need for resources of the Host Vehicle and the amount of background other users generate, putting an additional load on the network.

The ToD simulator framework needed expanding in order to allow for the appropriate scenarios to be simulated as part of this thesis work. To accomplish this, features were added that allow an obstacle to be placed on the map either since the start of the simulation or only when the HV is within a certain distance from its spawn point. A novel background generation model was included as well, in the form of *background vehicles*, one or more, that drive around the world in the driving simulator much like the HV, but are linked to background generators in the network simulator that then effectively move around, differently from the default method of users in fixed positions around each base station.

The simulation results show that Tele-Operated driving over 5G is feasible but heavily influenced by the radio conditions, most importantly the network delay which is affected by factors including the required bandwidth, the channel quality, the volume of other traffic on the shared medium, and the handovers between different base stations.

This thesis is organized as follows: Chapter 1 explains the concept of Tele-Operated Driving in detail as defined by sector associations, with its characteristics, variations, and proposed requirements. Chapter 2 describes the software foundation used to perform the simulations required for the performance evaluation, starting from an overview of the two simulators used for the network and driving side and continuing with the middleware that ties them together. Chapter 3 details the simulated scenarios including the chosen map from the driving simulator and parameter configurations. Finally, Chapter 4 exposes the results of the analysis performed on the simulation data, together with the description of the evaluated metrics.

Chapter 1

Tele-Operated Driving (ToD)

1.1 Introduction

In order to talk about Tele-Operated driving, an introduction must be made about Autonomous Driving. SAE International and ISO have a Work In Progress standard, of which the latest version was published in 2021 [8], defining six levels for (motor) vehicle automation, from no automation (level 0) to full automation (level 5). They are laid out as follows:

- *Level 0*: No Driving Automation
- *Level 1*: Driver Assistance, e.g., Adaptive Cruise Control (ACC) for longitudinal speed control in order to maintain a set distance from a preceding vehicle.
- *Level 2*: Partial Driving Automation, such as a Parking Assistance Feature which automatically performs the required controls to parallel park a vehicle under the supervision of the driver.
- *Level 3*: Conditional Driving Automation, for example an autonomous feature that is activated by the diver and executes an overtake in order to pass a slower-moving vehicle.

- *Level 4*: High Driving Automation, which still necessitates some input from a human driver, be it onboard or remote.
- *Level 5*: Full Driving Automation, capable of operating on all mapped roads in the US that are navigable by a human driver.

Tele-Operated Driving comes into play in level 4, where some unforeseen circumstance makes the autonomous system unable to react and thus human intervention, remote in this case, is required.

1.2 Definitions

Tele-Operated Driving is a service that involves a remote operator (ToD operator), be it human or a machine, to remotely execute driving activities on a vehicle (Host Vehicle), over a communications network. The 5GAA divides driving tasks into three types of activities [1]:

- *Strategic level operation*: travel planning (e.g. definition of goals, route), considering available options, costs, and risks involved.
- *Tactical (or maneuvering) level operation*: e.g. speed selection, lane selection, object and event response selection, and maneuver planning.
- *Operational (or control) level operation*: e.g. longitudinal and lateral control, object and event detection and classification.

This classification is hierarchical: each level relies on more and more input from the environment and allows for a tighter time interval for an output. Figure 1 provides a diagram that shows its structure.

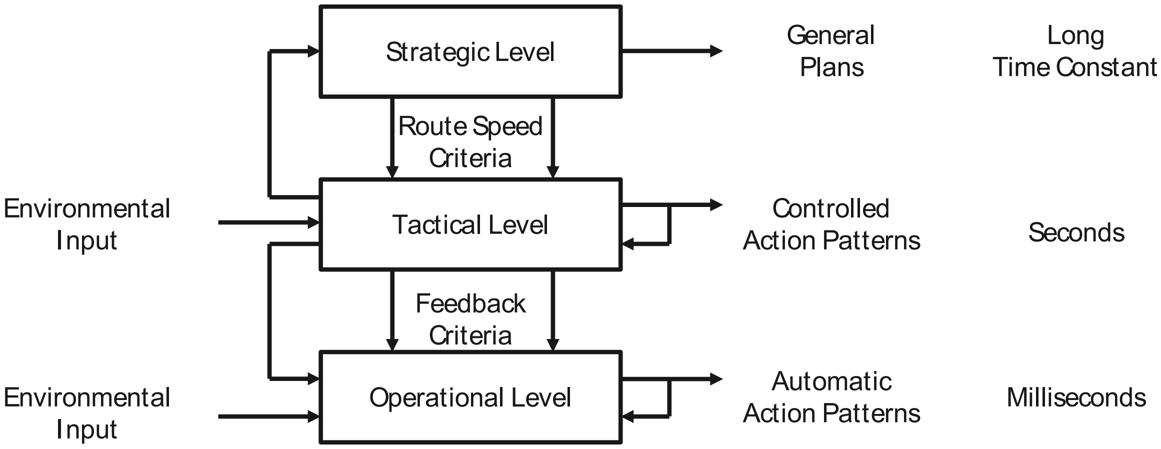


Figure 1: *The hierarchical structure of driving tasks, from [1]*

The 5GAA also defines four levels for the involvement of the Tele-Operated Driving operator with the act of driving, ranging from none to total:

- *Non-ToD*: the ToD operator takes no role in the act of driving and all levels mentioned above are performed by the in-vehicle driver, be it human or not.
- *Dispatch ToD*: the ToD operator acts as the Dispatcher, which performs the Strategic Level operations of driving, while the Tactical and Operational level operations are performed by the in-vehicle user.
- *Indirect Control ToD*: the ToD operator acts as the Indirect Controller, or Remote Assistant, which performs the Tactical Level operations and, if needed, the Strategic Level operations.
- *Direct Control ToD*: the ToD operator acts as the Direct Controller, or Remote Driver, which performs all or part of the real-time Operational Level functions. If needed, the operator may also perform Tactical and Strategic level operations.

Table 1 summarizes the distinction between ToD types with respect to the driving activity levels.

ToD Type	Driving Activity		
	Strategic Operation	Tactical Operation	Operational level Operation
Non-ToD	In-Vehicle user or system	In-Vehicle user or system	In-Vehicle user or system
Dispatch ToD	ToD operator	In-Vehicle user or system	In-Vehicle user or system
Indirect Control ToD	ToD operator (if needed)	ToD operator	In-Vehicle user or system
Direct Control ToD	ToD operator (if needed)	ToD operator (if needed)	ToD operator

Table 1: *Tod Operator engagement in the act of driving in different types of ToD, adapted from [1]*

The focus of this thesis' performance evaluation will be the simulation of a Direct-Control ToD service, with operational level operation.

1.3 Use cases

The variety of applicable use cases for Tele-Operated Driving is endless. This section will introduce some of them.

The main use case, as defined by the 5GCroCo project [2] and depicted in Figure 2, involves the event of a ToD operator taking direct control of an otherwise-autonomous vehicle in the case of an unexpected road blockage or difficult section that it cannot handle by itself. The vehicle comes to a safe stop and control is handed over to the remote operator, which from their virtual cockpit connects to the video and sensor feed and provides the required controls to overcome the difficult situation. After the obstacle is avoided, the operator brings the vehicle to a stop once again and the Autonomous Driving function takes over and continues on its task.

The 5GAA [9] lists several more use cases, including:

- Remote driving in the context of a car-sharing service for a trip from airport to city center.
- Tele-operated Driving for automated parking.
- Remote driving for vehicles in situations where human access is not possible or too dangerous.

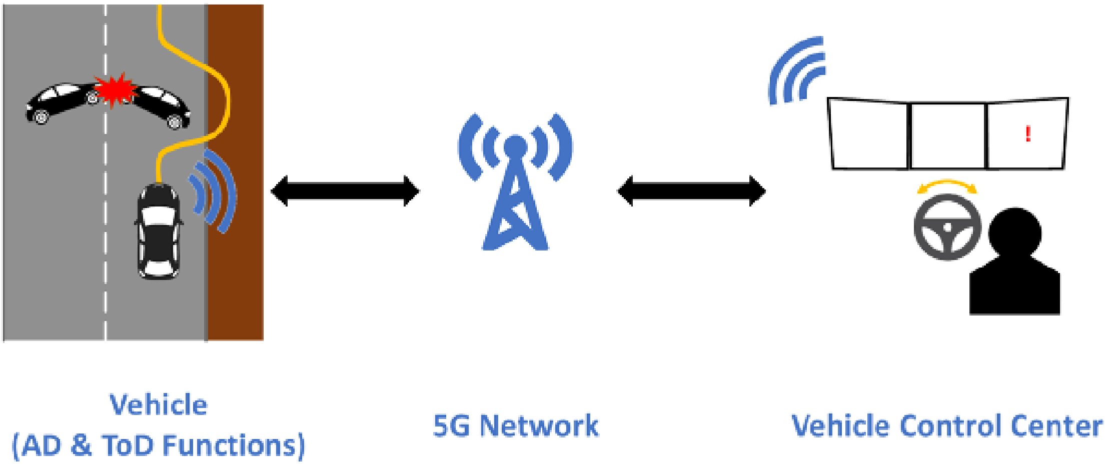


Figure 2: *Remotely controlled maneuvering around a road blockage, from [2]*

1.4 Functional requirements

In [1] and [9] a series of functional requirements are described for the host vehicle, the ToD operator, the communication network and the infrastructure. This thesis will focus on the first three categories and requirements relevant to Direct-Control ToD.

1.4.1 Host Vehicle requirements

The Host Vehicle must meet the following functional requirements in order to engage in Tele-Operated Driving:

- The vehicle should keep track of its geographical position and report it to the ToD Service Provider when required.

- The vehicle should be able to verify and calibrate its perception sensors and ensure they are in a proper state for engaging in the ToD service.

- The vehicle should be able to detect any malfunctioning of the ToD system and override the controls in order to bring itself to a minimal risk condition, e.g. pull over safely to the side of the road. For example, in the case of a loss of connectivity, a physical intrusion in the vehicle, or any other fault leading to incorrect actuation data.

- The vehicle should share sensor information with the ToD operator, which includes video streams, LiDAR, and RADAR data, when required.

- If regulations require it, the vehicle should be able to store communication relevant to the ToD service for later evaluation, inspection and analysis.

- The vehicle should be able to report the operations and status of the ToD service to the in-vehicle system or user, e.g. whether the ToD operator is currently engaged in the service.

- The vehicle sub-system should take steps to protect the ToD service from security threats and malicious actors trying to gain control of the vehicle's movement.
- The vehicle should be able to process the remote actuator commands.
- The vehicle should receive maneuver instructions from the ToD operator and execute them, according to its internal security checks.

1.4.2 ToD operator requirements

The ToD operator subsystem must be able to fulfill the following requirements in order to allow for safe operation:

- The ToD operator should be able to acquire all ToD-related parameters of the vehicle in a real-time manner, e.g. ToD type, vehicle capability, and sensor data.
- The ToD operator should be able to detect the malfunctioning of a ToD system and bring the vehicle to a minimal risk condition.
- The ToD operator should be able to disengage the ToD function in a safe manner at the end of its tasks.
- The ToD operator subsystem should be able to obtain sensor data and information from the vehicle, weather conditions, and infrastructure data, and display it to the ToD operator to enhance the remote driving experience.
- The ToD operator subsystem should be able to authenticate and authorize the human ToD operator and machine ToD apps/entities.

- The ToD operator should be able to send commands to the vehicle for the purpose of remote control.
- The ToD operator should be able to change the ToD type when required.
- The ToD operator should be able to monitor the network parameters and be swiftly informed about changes in its performance.
- The ToD operator should be able to monitor the latency and quality of the video signal at all times.
- The ToD operator sub-system should be able to display real-time video around the vehicle to the operator with acceptable quality in order to provide a good user experience.

1.4.3 Communication network requirements

The communication network is essential for allowing data exchange between HV and ToD operator. In particular, the following criteria must be met:

- The communication network should provide the ability for the ToD operator to establish a mutually authenticated and secure communication session with the vehicle.
- The communication network should guarantee service continuity during the operation of the ToD service.
- The communication link should be reliable and encrypted.
- The communication session should be kept active even when roaming different networks and across national borders. The ToD operator and vehicle should be informed in advance in case this is not possible.

- The communication network should be able to notify the vehicle and the ToD operator about sudden changes in network conditions.
- The communication network should inform the vehicle and the ToD operator about its measured latency information.

1.5 Technical requirements

Other than functional requirements, it is important to establish some technical requirements in order to obtain a reliable and safe operation.

1.5.1 Latency

The 5GAA defines service level latency as: "*measurements of time from the occurrence of the event in scenario application zone to the beginning of the resulting actuation*" [9]. Network latency, while being the most crucial aspect and the one subject to the most variability, as it will be shown later, only comprises a part of the chain of delay-inducing processes, which include on the uplink (Vehicle to Operator):

- The elapsed time between an event happening in the world and the sensors acquiring the raw data.
- The elapsed time between the sensors capturing the event and them transmitting the data to the vehicle's communication unit, which takes some time to process it.
- The network latency in relaying the transmitted information to the ToD operator's location, including the mobile network and backend systems.
- The latency induced by the ToD Operator App to decode, process and present the received information

- The time the ToD operator needs to perceive and react to the presented information.

Correspondingly, on the downlink (Operator to Vehicle):

- The time the ToD operator (if human) takes to move the controls.
- The elapsed time between the controls being moved and the movement getting registered by the sensors.
- The elapsed time between the movement being captured and it being processed and stored into a packet to be transmitted back to the vehicle.
- The communication latency induced by the network from the ToD operator's location to the vehicle, through backend systems and the mobile network.
- The time the vehicle's systems take to decode the data and send the appropriate instructions to the onboard actuators.
- The elapsed time between the actuators executing the required action and the vehicle's motion occurring as a result.

In the case of video transmission, the delay that occurs between photons of visible light hitting the lens of a camera and the event being portrayed on a display after encoding, transmission, processing and decoding is called *Glass-to-Glass (GTG) delay.* Bachhuber and Steinbach [3] have proposed a system that accurately measures GTG delay and provides an image representation of the steps that influence the time an event takes from when it happens to when it's reproduced on the user's display, Figure 3.

A consideration can be made about the variability of each delay: All but the network and the ToD operator's (in the case of a human) contributions can basically be regarded as fixed values, and once properly tuned and in the absence of faults are

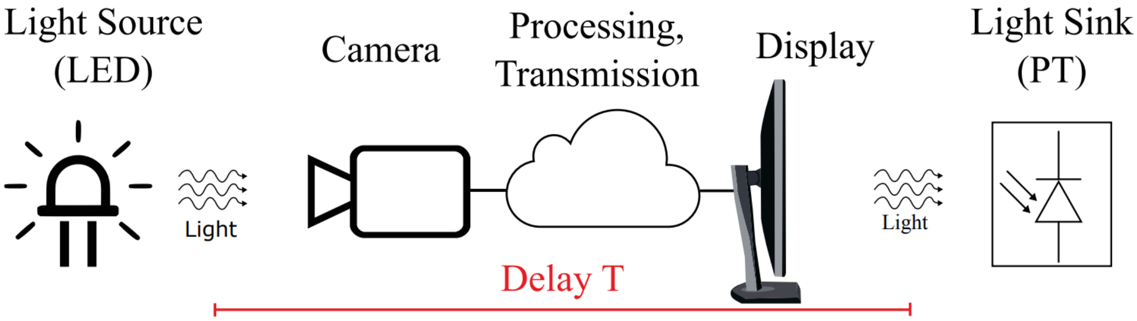


Figure 3: *GTG delay measurement principle, from [3]*

not to be concerned about. It is very important then to keep network delay in check as it can make or break the service.

The 5GAA [9] defines a requirement for the maximum total latency from HV to Operator of 100ms and from Operator to HV of 20ms for a ToD use case up to 50km/h (where the HV will move 0.27m within 20ms).

In a study carried out by Neumeier et al. [10], tests indicated that a latency of around 300ms led to a deteriorated driving experience for the candidates, and found a link between added latency and the detriment of the driver's confidence. They also showed that the tolerable latency is severely affected by the environment.

The sensitivity to latency of the service is mostly determined by the difficulty level of the environment around the vehicle and the speed it is traveling at: a low-speed scenario on a straight section of road will be less impacted by a latency spike than a high-speed highway section in the presence of other vehicles and on a curve.

1.5.2 Sensors

In order to allow for the operation of ToD services, the host vehicle must equip a vast array of sensors and fulfill the technical requirements for automated driving according to SAE levels 3, 4, and 5. The 5GAA provides a list of the most relevant ones [9].

HD Digital maps

High-Definition digital maps act as a sensor by themselves by providing details about the environment, possibly also about real-time conditions. They allow to ‘see’ around curves or in fog. HD maps support sensor data and show much more than just roads and routes. They capture the three-dimensional environment at a centimeter-accurate resolution and can be compared to the data provided by the onboard sensors from the host vehicle. Information captured by HD maps may include lane models, traffic signs, road infrastructure, and lane geometry. As the automation level increases, so does the requirement for resolution of these maps.

Precise positioning

Precise positioning refers to a series of techniques for correcting GNSS errors in order to provide even higher level of position accuracy in the vehicles, ideally with as small an error as a few centimeters. Automated driving and ToD operations require at least lane-level precision with very high integrity. This is especially crucial when bad weather conditions, such as heavy snow, impair the other sensors’ performance.

LiDAR capabilities

LiDAR - Light Detection and Ranging, is a sensor technology used to actively monitor the environment and perceive short-term changes, creating a 3D map of the host vehicle’s surroundings. It can operate even under difficult conditions, e.g. in darkness, between 50 and 500m, and scan angles up to 180 degrees.

RADAR capabilities

RADAR technology is very well known and like LiDAR, it is also used to actively monitor the environment. It’s however limited to around 250m and has a narrower scanning angle. On the other hand, RADAR systems are very effective over short distances, starting at 0.2m, and have low processing demands for detecting a wide variety of environmental values, e.g. angle, distance, speed, and material parameters.

Ultrasonic capabilities

Ultrasonic technology is used to actively monitor the near-field environment of the vehicle, under 1m. It's useful for avoiding collisions while parking or navigating tight spaces.

Video camera capabilities

In addition to the aforementioned sensors, it is important that the Host Vehicle also provide a video stream of its surroundings. This is crucial in the case of a human ToD operator but it is useful in the case of a machine operator that might be able to acquire some new information using computer vision that the sensors might not have yet captured. Cameras are most effective in clear and well-lit conditions, as they are rendered almost useless in heavy fog or dark environments. Video cameras can additionally be installed on the infrastructure, such as RSUs, lamp posts and traffic masts, in order to provide additional data from a different point of view.

Microphone and loudspeakers

Microphones and loudspeakers are useful to monitor and interact with the environment through audio. They can be used to communicate with a passenger, detect relevant outside communication and pick up instructions e.g. from police commands or emergency vehicle sirens.

Chapter 2

ToD Simulator

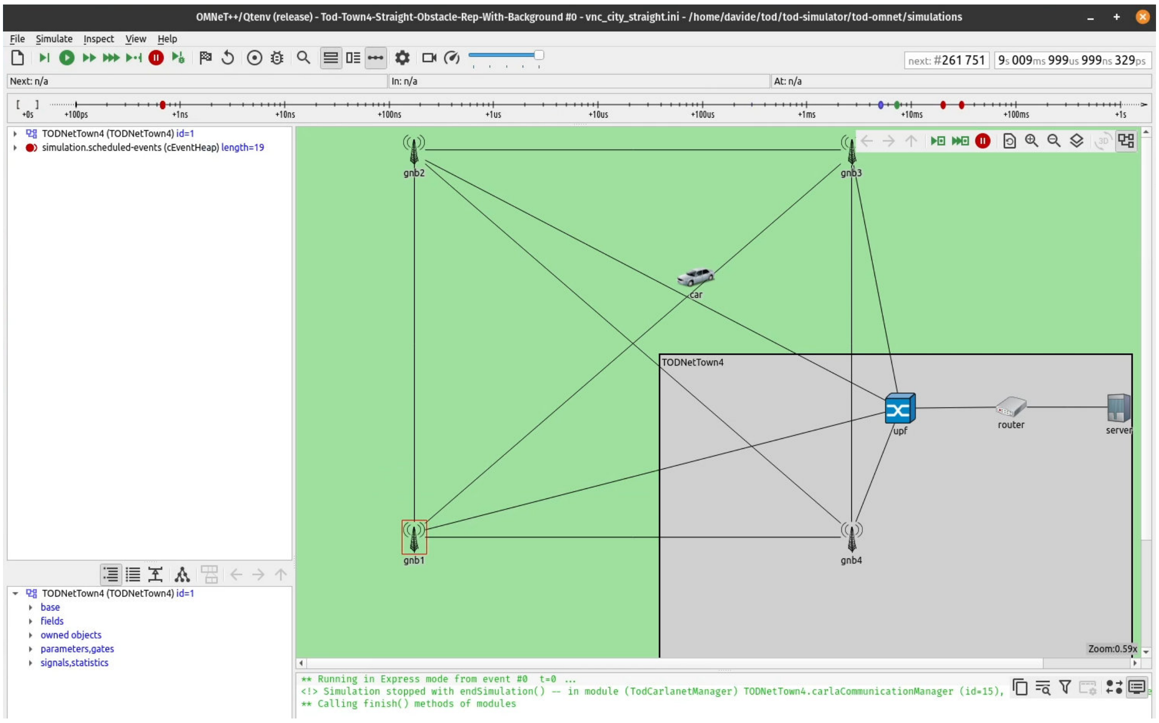
This chapter will elaborate on the ToD simulation framework described in [4], which composes the basis for the work carried out over the course of this thesis.

2.1 Network simulator: OMNeT++

OMNeT++ is a versatile discrete event simulation framework that has been widely used in the research community, particularly in the field of network simulations. It is based on C++ and provides a modular and extensible architecture that allows users to create complex simulations of communication networks, such as the Internet or wireless networks.

The modular architecture of OMNeT++ enables users to build simulations by using pre-existing components and creating custom ones. This makes it possible to create simulations that are tailored to specific research needs and to easily modify and extend existing simulations. Its support for discrete event simulation, which models the behavior of a system as a series of discrete events, allows for the accurate modeling of network behaviors and for the simulation of large-scale networks.

OMNeT++ provides a feature-rich IDE, shown in Figure 4, and includes built-in support for visualizing simulation results, which are useful for analyzing the behavior



OMNeT++/Qtenv (release) - Tod-Town4-Straight-Obstacle-Rep-With-Background #0 - vnc_city_straight.ini - /home/davide/tod/tod-simulator/tod-omnet/simulations

next: #261 751 | 9s 009ms 999us 999ns 329ps

Next: n/a | In: n/a | At: n/a

TODNetTown4 (TODNetTown4) id=1
simulation.scheduled-events (cEventHeap) length=19

TODNetTown4 (TODNetTown4) id=1
- base
- fields
- owned objects
- parameters,gates
- signals,statistics

```
** Running in Express mode from event #0  t=0 ...
<!> Simulation stopped with endSimulation() -- in module (TodCarlanetManager) TODNetTown4.carlaCommunicationManager (id=15),
** Calling finish() methods of modules
```

Figure 4: *OMNeT++ GUI (Qtenv) showing the basic network used for simulations*

of a network and identifying performance bottlenecks. This makes it easier to understand the behavior of complex communication systems and to identify potential areas for improvement [11].

The simulator itself lacks the appropriate modules for TCP/IP and other Internet-related protocols, as well as the ones needed to simulate the 5G radio communications. They are provided by INET and Simu5G.

INET [12] is an open-source model library that provides protocols, agents, and other models for simulating communication networks. It contains models for the Internet stack (TCP, UDP, IPv4, IPv6, etc.) and some wireless and wired link layer protocols, such as Ethernet and PPP, among others. It is used as a base for expansion by several other simulation frameworks for the purpose of simulating vehicular networks, overlay and peer-to-peer networks, LTE or 5G.

Simu5G [13] [14] is a 5G New Radio (NR) simulator for the OMNeT++ simulation framework based on the now-deprecated SimuLTE library by the same developers. It incorporates all models from INET and allows researchers to simulate TCP/IP networks including 5G NR interfaces. Simu5G provides models for the data plane of 5G RAN and core networks including the radio links, in both Frequency Division Duplexing and Time Division Duplexing modes, with various types of base stations. It also includes all models from SimuLTE for 4G/LTE networks.

2.2 Driving simulator: CARLA

CARLA (CAR Learning to Act) is an open-source simulator developed from the ground up to support autonomous driving research. It is designed to create a high-fidelity 3D environment that mirrors real-world urban environments, allowing users to test and develop autonomous driving algorithms in a safe, controlled, and cost-effective manner. It leverages Unreal Engine for rendering, which provides high-quality graphics and realistic lighting effects.

The environment provided by CARLA includes city streets, highways, intersections, and various types of roads. It can be customized with different road layouts, buildings, lighting conditions, and weather settings.

In addition, it simulates the behavior of various types of vehicles, including cars, trucks, and buses, with realistic physics and dynamics. This includes accurate modeling of vehicle acceleration, braking, steering, and collision detection. It supports a variety of sensors commonly used in autonomous driving, including cameras, lidars, radars, and GPS.

CARLA provides a Python client API that allows developers to interact with the simulation programmatically. This API enables tasks such as spawning and controlling vehicles, retrieving sensor data, and creating custom scenarios [15].

By default, the CARLA server runs the simulation as quickly as it can, in what is called `asynchronous mode`. The feature that makes it possible for it to work together with OMNeT++ is its `synchronous mode`. As the name implies, it allows the server to be kept in sync with the client, and to use all the time it needs to compute each simulation step. This is ideal for slower client applications as well. In the case of the ToD simulation framework, as many components are being simulated in both programs and thus simulated time is much slower than real time on common PC hardware, asynchronous mode would be unusable.

2.3 ToD Simulation Framework

The ToD simulation framework described in [4] is used as the basis for this thesis' experiments and expansion. This section provides a description of its architecture and inner workings.

2.3.1 Co-simulation architecture

In order to make the co-simulation model work, it is firstly essential to clarify the separation of concerns for the two simulators involved - The OMNeT++ network simulator and the CARLA driving simulator - of which the names provide an intuitive description.

OMNeT++ is responsible for simulating the network connection between the HV and the ToD operator, that is, the protocol stack for the UE on the vehicle, the gNodeBs, UPF, and the ToD remote host. It is also in charge of generating the timed events of the application and interacting with the CARLA application components by message passing.

CARLA is responsible for implementing the physical world, including all vehicle dynamics, simulation of sensors and actuators, traffic lights, roads, and pedestrians. It runs the application layer of the ToD driving simulator, including the operator, which is derived from the BehaviorAgent provided by the CARLA library [5]. It is able to calculate a path from the starting point on the map to one or more destinations, following roads and speed limits, and drive the vehicle using the data from its camera and sensor array.

A message-oriented middleware is distributed into two separate components, each integrated into one simulator, which communicate over a network connection. This also allows the two sides to run on separate systems if necessary. They operate in a request-reply model, with requests performed on the OMNeT++ side and replies sent by the CARLA side. Figure 5 shows a schematic representation of the framework's architecture.

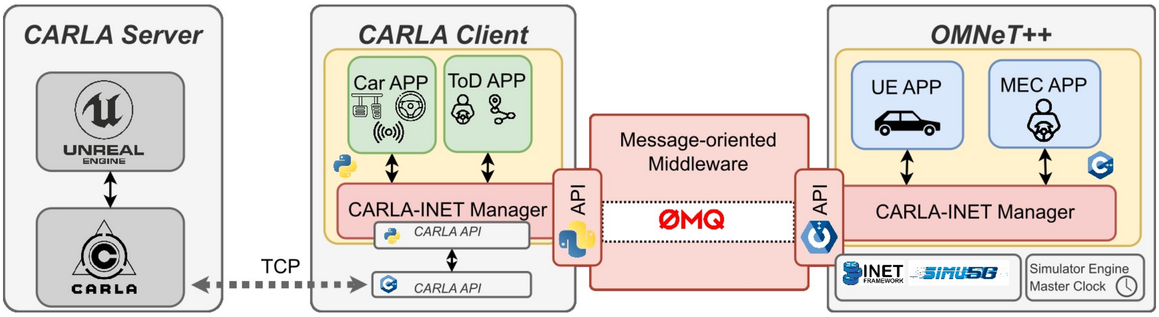


Figure 5: *Architecture of the ToD simulation framework, from [4]*

Each of the two simulators comes with an internal clock, which keeps the simulation time. They also have different needs for its resolution, as the events taking place in the network simulation are much more frequent than those in the driving simulation (picoseconds as opposed to milliseconds). Because of this, the main clock is kept by the component on the OMNeT++ side and it is in charge of sending messages that trigger a world tick to its CARLA correspondent at the required interval.

2.3.2 The simulation process

At the start of each simulation, the OMNeT++ component sends the CARLA component an initialization message containing the required configuration parameters. These include the run ID, random seed, list of nodes, actor IDs and types, and name of the route configuration file. The CARLA component replies with its initial timestamp and the position of each actor that is configured to be synchronized on both sides.

Over the duration of the simulation, the OMNeT++ component schedules a repeating event which sends a message to the CARLA component with the purpose of triggering a simulation step. This message is received by the CARLA side and a simulation step is triggered. A reply is sent back containing each node's position to be updated on the network simulator side. Different types of messages exist to make the ToD service work, which include the request to generate and send the vehicle sensor data to the operator, the computation of the instructions to send back to the vehicle, and the reply with the controls to be applied. They are simulated as if they were going through the network and are subject to the delay caused by varying network conditions over the vehicle's movement with respect to the base stations.

Within each response sent from CARLA to the OMNeT++ side, the status of the simulation is included, and its purpose is to report whether or not the simulation is due to finish and for which reason, such as a time limit having been reached or a collision having occurred. At that point, the OMNeT++ component sends a finish simulation message, the CARLA simulator terminates its side of the simulation and replies with an acknowledgment, and the network simulator finally terminates its side, marking the end of the whole simulation process.

2.3.3 Additions

In order to carry out some of the scenarios discussed in this thesis, modifications to the aforementioned framework were introduced, the most relevant ones are:

- A feature was added so that the route configuration file can contain instructions for where to place an obstacle, which blueprint it uses, which way it faces, and whether it is present in the simulated world from the start of each simulation run or it is spawned when the HV's center point is within a set distance from its configured spawn point (Euclidean distance). A configuration option is also available to specify whether the obstacle vehicle will have CARLA's basic autopilot enabled. Scenarios with this kind of configuration were considered but, due to the agent's limitations, they were not explored further.

- A new kind of actor was introduced: the background vehicle, which is used in some of the experiments. The background moving vehicles follow the same implementation as the operator for the HV, but instead of being run through the network they are contained within the CARLA application layer, and ticks at regular intervals are sent to them by the ToD simulator middleware's environment handler. Their operation is thus not affected by any delay and the camera and sensor data are instantly available to the operator at all times. They are linked to background-inducing network nodes in the OMNeT++ simulation so that their location is updated at all times in the simulated network as well.

Chapter 3

Simulated Scenarios

This chapter is dedicated to the description of the various simulated scenarios that were tested using the ToD simulation framework, after a brief introduction on the chosen map from CARLA. They are divided into two sections, as their nature is twofold: one that involves the Host Vehicle completing a trip over a predetermined route with varying levels and types of network background, and one that tests the feasibility of the reaction to obstacles by the ToD operator and how it is influenced by growing levels of background traffic.

3.1 Simulation map

The chosen map from CARLA used in all simulations is `Town04`, shown in Figure 6, which consists of a small town rounded by a multi-lane highway laid in a "figure of 8" shape, which intersects with itself in the center of the map in a cloverleaf interchange [5].

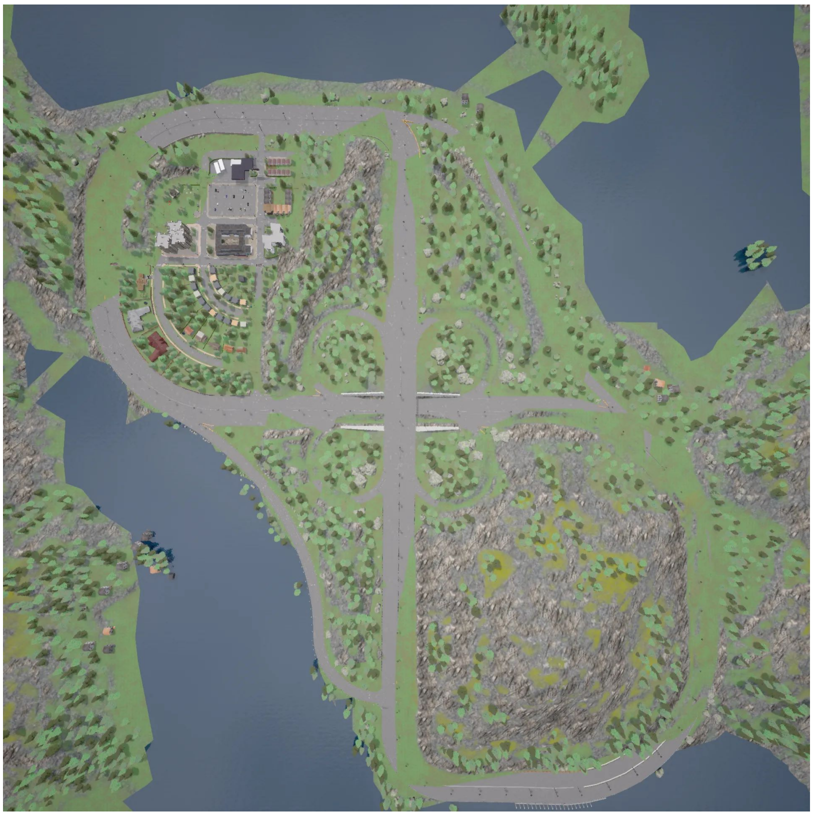


Figure 6: *Aerial view of CARLA map Town04, from [5]*

3.2 City trip

This scenario takes the route specified in [4] and tests it against varying network configurations and loads.

The selected route goes as follows: It starts on the north section of the "ring road" going south, takes a slip road heading south-west with a speed limit of 60km/h for the first smooth curve, crosses a junction with a 30 and then 40km/h limit into the highway again heading east at 90km/h towards the center of the map. Once there, the speed limit decreases to 60km/h and then 30km/h into the slip road going north. Then, after a short 90km/h stretch of highway it turns right into the city where the final 30km/h section begins. Once there it makes a left and leads into the center of the city where the end waypoint is. A drawing of the route on the map color-coded with the different speed limits is visible in Figure 7.

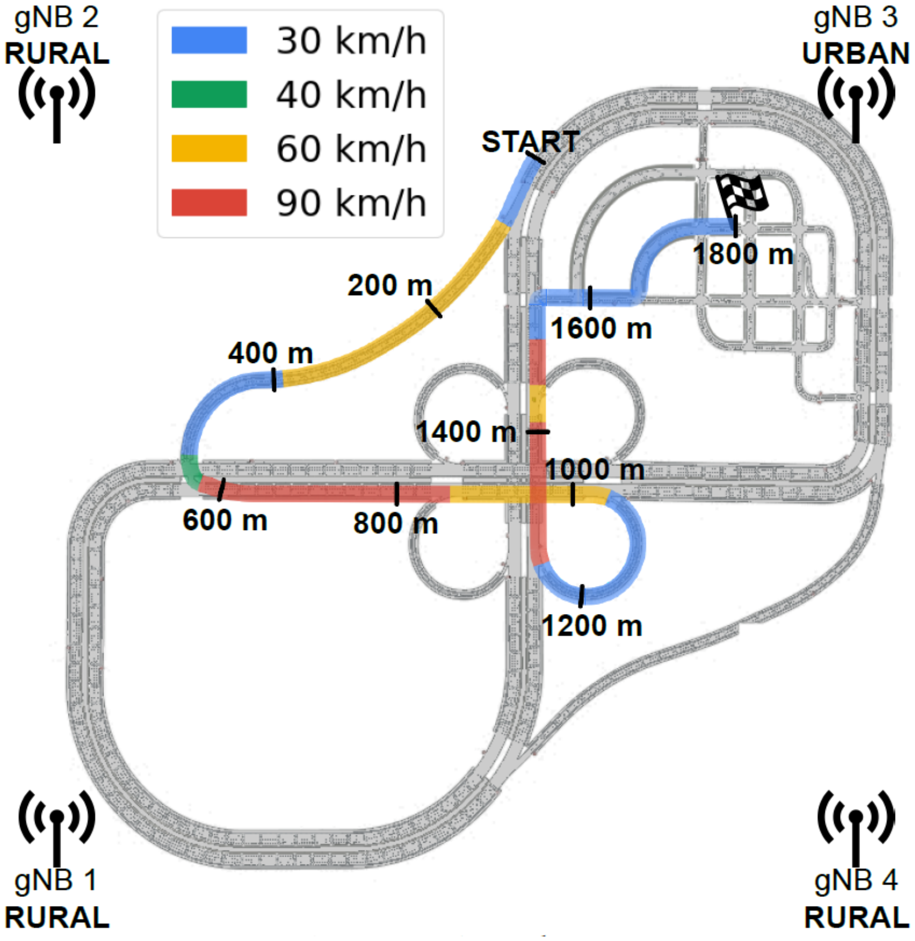


Figure 7: *City trip route to follow overlaid on the road map of Town04 showing speed limits, from [4]*

3.2.1 Background

The base scenario takes into consideration the presence of a single user on the network, which is far from a realistic simulation, so a way to generate some background traffic was needed. The background was simulated in two different ways:

Static background

This takes the same form as the simulations described in [4], where up to three nodes are spawned in the OMNeT++ simulation around each base station and they produce a comparable amount of traffic as the same number of other HVs running a ToD service.

Moving background

The static background simulation works and it will be shown that it does have an influence over the feasibility of certain parameter configurations, but it does so in a pretty linear and predictable fashion. The following introduces a different perspective on background generation: instead of keeping the same number of background users in fixed positions around each base station, up to three CARLA actors are roaming around the same area as the Host Vehicle, to better simulate scenarios where other users are on the road and putting varying load on the base stations, influenced by their distance from the base station and channel quality, making their impact on the overall service less predictable.

In order to keep consistency on the route across simulations, the background vehicle routes are set up so that they will not interfere with the HV by making it slow down or stop.

3.3 Reaction to obstacles

This section describes the different simulated scenarios that involve the presence of an obstacle on the HV's path prompting a reaction by the ToD operator, both present from the start and suddenly appearing in an urban environment, and suddenly appearing in a highway or extra-urban environment.

3.3.1 Static obstacle

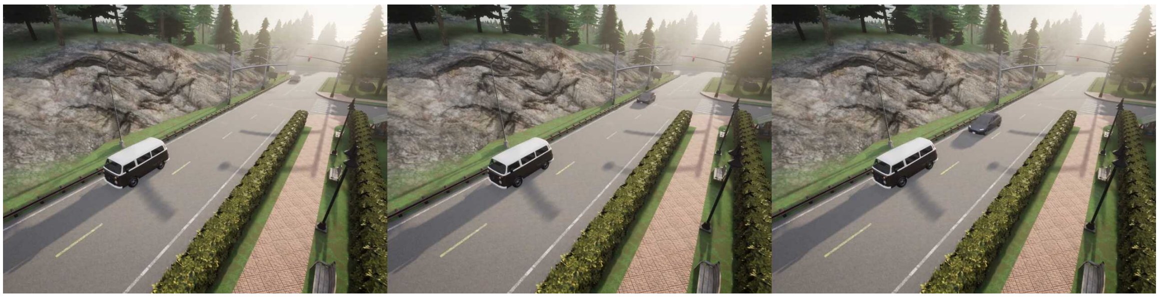


Figure 8: *City static obstacle scenario rendered by CARLA*

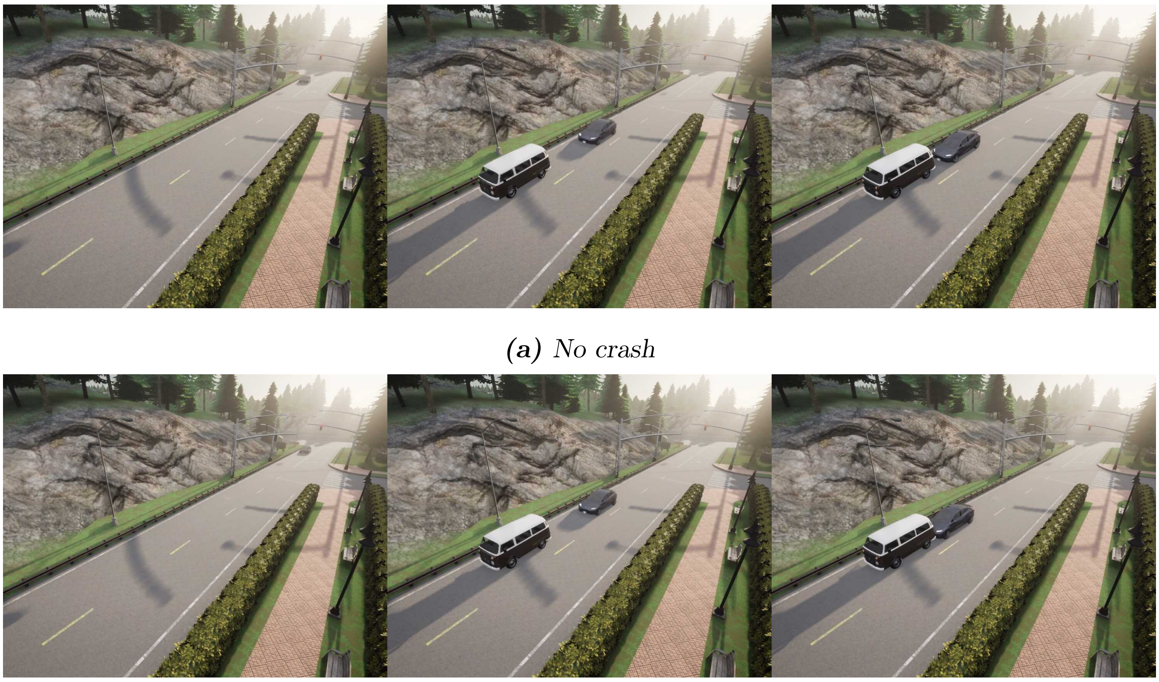
In this scenario, an obstacle is placed on the vehicle's path from the start of the simulation (Figure 8). The route is very short as is the overall duration of simulations. The vehicle reaches a speed of 30km/h and then has to brake in order to avoid a collision.

3.3.2 City sudden obstacle

The nature of this scenario is the same as Section 3.3.1 except the obstacle only appears in the world when the Host Vehicle is 10m away from the obstacle's spawn point (Figure 9). This is a scenario similar to a distracted pedestrian or a pet suddenly crossing the street without looking.

3.3.3 Highway sudden obstacle

This scenario follows the purpose of evaluating the feasibility of the reaction by the ToD operator to an obstacle suddenly appearing on the Host Vehicle's path. The event takes place on a highway section, where the obstacle appears in the world when the HV is 44m away from its spawn point and traveling at a speed of 90km/h.



(a) *No crash*

(b) *Crash*

Figure 9: *City sudden obstacle scenario rendered by CARLA*

3.4 Simulation Parameters

The complex nature of the simulators, particularly OMNeT++, offers immense parameter customization down to the most specific details. Table 2 contains a relevant subset of parameters that make up the overall configuration and this section will explain the most relevant ones considered when defining the parameter study.

3.4.1 Number of background users

This parameter regulates how many background users are simulated around each base station as the static background and how many users are roaming the streets as the moving background. The latter is only used in some of the tested city trip scenarios.

All other cases use the static background. For the purpose of this thesis between 0 and 3 of them will be considered.

3.4.2 Numerology

Numerology in the context of 5G NR (New Radio) defines the chosen subcarrier spacing (Δf) as per the following formula [16]:

$$\Delta f = 2^{\mu} \times 15\text{kHz}$$

Subcarrier Spacing (Δf) determines the frequency granularity of the system. Smaller subcarrier spacings provide higher frequency resolution but may result in larger symbol durations. Conversely, larger subcarrier spacings offer better spectral efficiency but with coarser frequency resolution. For our data transmissions, we will only consider μ values of 0 and 1, i.e. 15 and 30 kHz.

The symbol duration is inversely proportional to the subcarrier spacing. Transmissions are organized in time slots, each of those containing 14 symbols. For numerology 0, a time slot lasts 1ms, while for numerology 1 it's half that: 0.5ms, because of the wider subcarrier spacing. More specifically, the symbol duration is 66.7µs for a numerology index of 0 and 33.3µs for a numerology index of 1. It will be shown that this helps greatly with overall latency.

3.4.3 Packet size & Frame rate

These two parameters are tightly related as together they make up the effective bitrate of the data transmissions. The packet size refers to the size in kilobytes of each status packet sent by the Host Vehicle to the ToD operator through the network. The frame rate, expressed in frames per second or fps, defines how many times per second this transmission is made.

A bigger packet size represents a higher quality video stream and a higher frame rate allows the operator to respond to information that is more up-to-date thanks to the higher sampling frequency. The overall bitrate can be calculated using the formula:

$$\text{bitrate (Mbps)} = \frac{\text{status packet size (kB)}}{1024} * 8 * \text{frame rate}$$

The least demanding and lowest quality configuration is the one with 33kB packet size (HD video) and a frame rate of 25fps, which amounts to around 6.5Mbps, while the most demanding and higher quality configuration is the one with 66kB packet size (Full HD video) and a frame rate of 60fps, which amounts to around 31Mbps.

3.4.4 Number of resource blocks

A resource block consists of 12 consecutive subcarriers, and while the work presented in [4] uses a fixed number of resource blocks for both numerology values, this thesis will employ 100 of them for $\mu = 0$ and 50 for $\mu = 1$, so that the available bandwidth stays constant across all simulations (20MHz). It will be shown in Chapter 4 that this brings a change with respect to the original scenarios in more demanding configurations.

Parameter	**Value**
Numerology index (μ)	*0, 1*
Frame rate	*25 fps* *60 fps*
Status packet size	*33 kB* *66 kB*
Instruction packet size	*1000 B*
Encoding image time	*normal(10 ms, 0.5 ms)*
Collection data time	*normal(15ms, 0.1ms)*
Processing status time	*normal(15ms, 0.75ms)*
gNodeB mac scheduler	*Round-robin*
Handover latency	*50 ms*
Target BLER	*0.01*
Resource blocks	*100* if $\mu = 0$, *50* if $\mu = 1$
ToD background cars count	*0, 1, 2, 3*
Background status packet size	*33 kB*
ToD driving simulation time-step	*10 ms*

Table 2: *Simulation parameters used*

Chapter 4

Results

This chapter is dedicated to the discussion of the results obtained after analyzing the recorded data from the simulated scenarios described in Chapter 3.

4.1 Evaluated metrics

A number of metrics were considered in order to evaluate the ToD system's performance over the different parameter configurations and considered scenarios. This section explains them in detail.

4.1.1 Completion percentage

Each simulated scenario is tested over 10 runs, each with a different random seed. The completion percentage refers to how many out of those 10 runs ended successfully and how many did not. This value is a high-level indication of the feasibility of each scenario and completion percentage comparisons can be made over simulations with different parameters to show how big of an impact they have on the overall feasibility of each scenario.

4.1.2 Trajectory error

The trajectory error is defined as the deviation from the optimal trajectory, which is the path followed by the Host Vehicle when the network simulation is disabled, effectively making it as if the operator was sitting in the vehicle.

In the simulated world, as is the case in the real world, even though the same route is followed over different runs, there will be a variation in the time domain that has to be eliminated and the routes aligned before a distance can be meaningfully computed. One of the possible means to accomplish this and the one used in this work is Dynamic Time Warping (DTW) [17]. It is frequently used in speech recognition and other fields where temporal sequences with varying speeds are involved, such as data mining and information retrieval. Figure 10 shows a visualization of the difference between using an Euclidean distance metric versus DTW for two time series. Since the changes in altitude are marginal in the driving simulator scenario, only a 2D set of coordinates is evaluated in computing the distances between routes at each given sample.

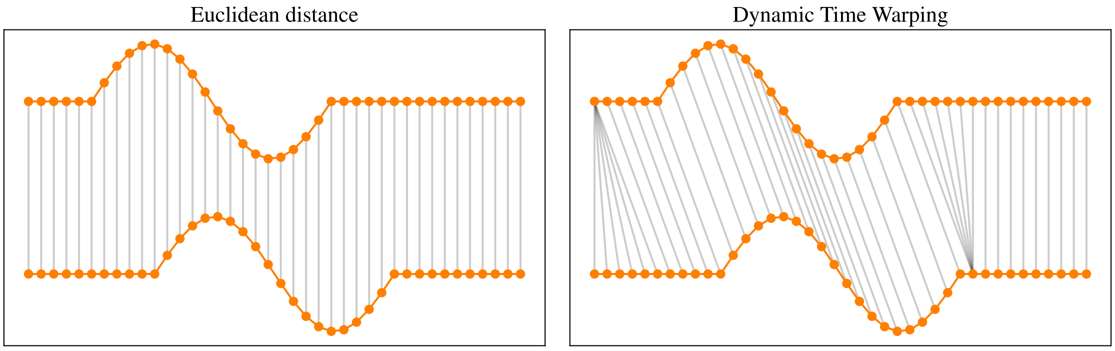


Figure 10: *Comparison between DTW and Euclidean distance, adapted from [6]*

4.1.3 Round-Trip Time

The Round-Trip Time is the time that elapses between the moment a status is sent from the vehicle to the operator and the instruction calculated on that status is received.

4.1.4 Channel Quality Indicator

The Channel Quality Indicator (CQI) is a numerical value between 0 and 15 that describes how good the communication channel is, with 0 being the worst and 15 being the best value. A bad quality channel forces the use of modulation coding schemes that are more resistant to data corruption but less efficient in terms of data rates. It also increases the chance of needing to make retransmissions because of corrupt frames, which in turn increases the network delay and bandwidth usage.

4.1.5 Resource Block Utilization percentage

This is the percentage of available physical resource blocks that the vehicle uses in transmitting its status and sensor data.

4.2 City trip

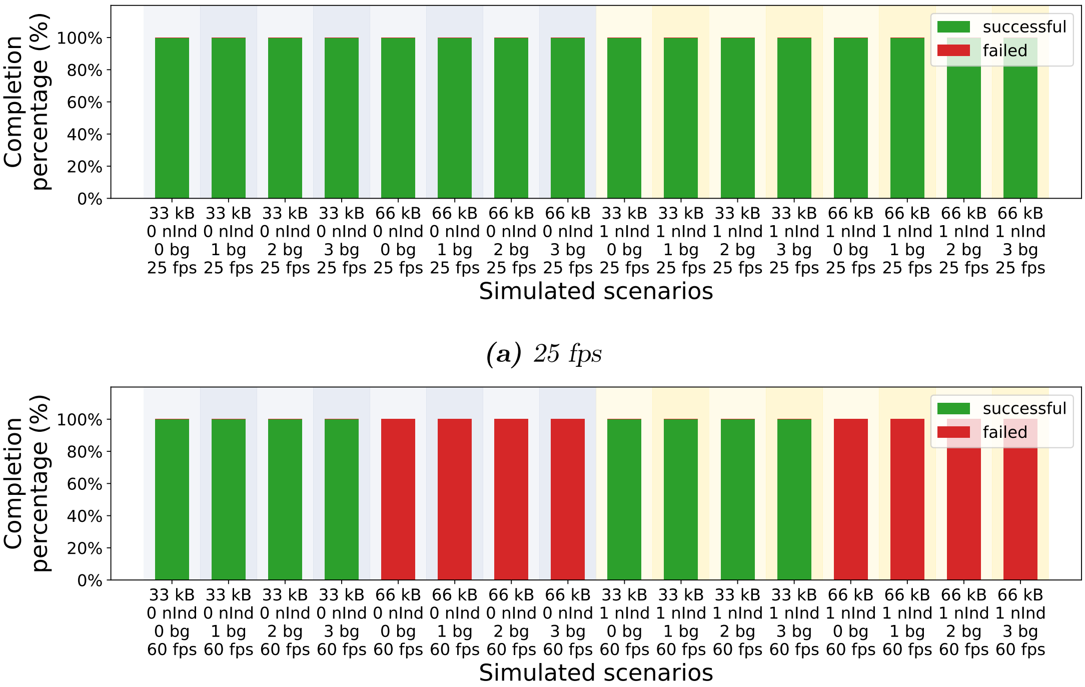
As introduced in Section 3.2.1, this route was considered with both static background and moving background simulation models.

In this context, the completion percentage across the 10 runs for each scenario will be plotted as bars colored in green for the successful share of runs and red for the share of failed runs.

Trajectory error, RTT, CQI, and RB utilization percentage will be traced as subplots sharing the same x-axis corresponding to the distance from the origin point.

Vertical dotted lines are placed in correspondence with the handover from one base station to another.

4.2.1 Static background



(a) *25 fps*

(b) *60 fps*

Figure 11: *City trip with static background - Simulations completion percentage*

Figure 11 shows an overall look at the completion percentage across the 10 runs executed over each of the 32 considered parameter configurations.

This route and parameter configuration is very similar to what is described in [4], with the difference being that the change in the resource block allocation to keep constant bandwidth across all runs made it impossible for the scenarios with 66kB status packet size and 60fps refresh rate to be successfully completed, regardless of numerology and number of background sources, as in the configurations with a numerology parameter μ of 1 the Host Vehicle no longer had double the amount of bandwidth available to it.

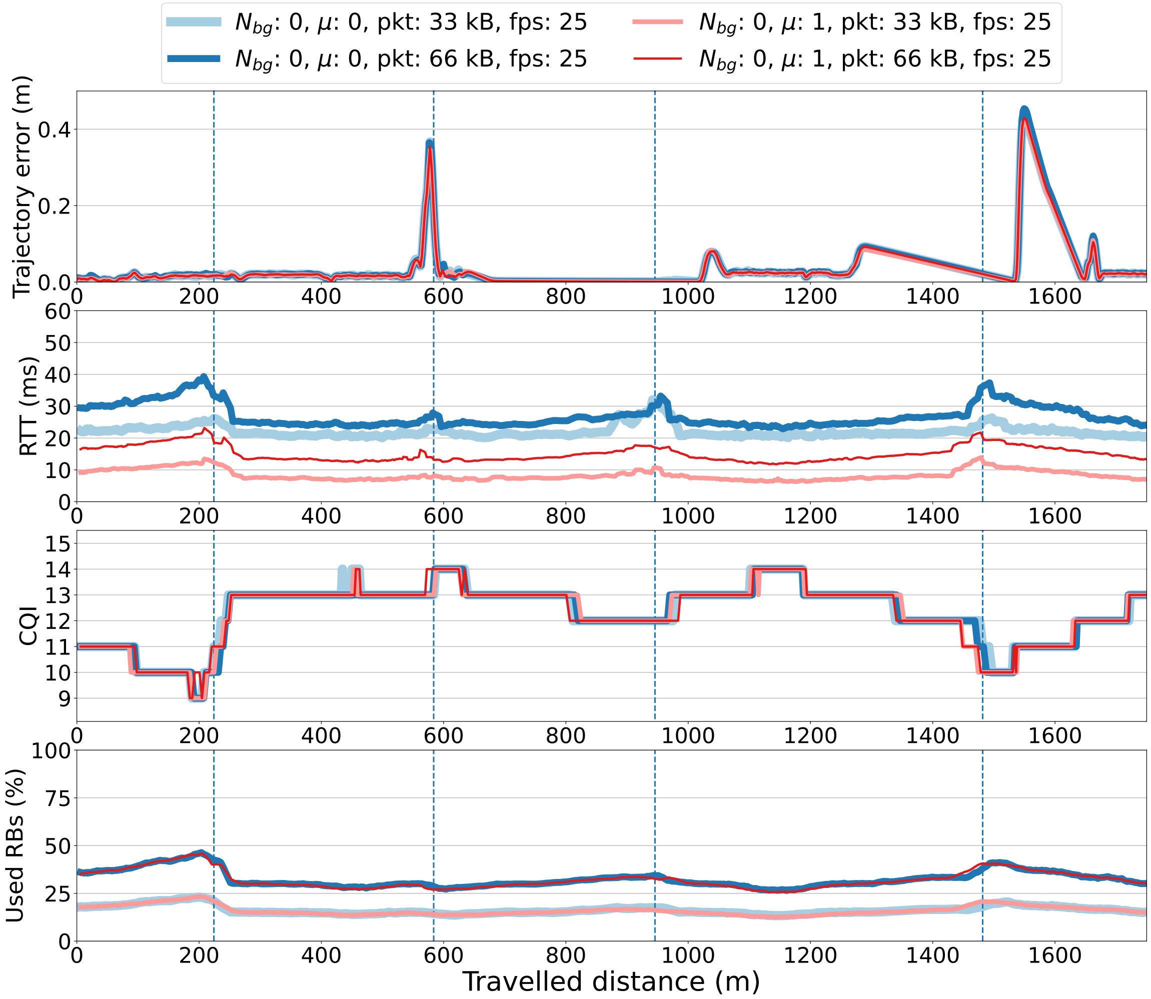


__Figure 12:__ City trip with no background - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps

Figure 12 traces the trajectory error, RTT, CQI and used RBs in the absence of background for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps, each averaged over the respective 10 runs. The error on the trajectory spikes in correspondence with the more difficult parts of the route, i.e. the sections with curves. It can be observed that the dips in channel quality make both the RTT and the percentage of used RBs increase. This is because as channel quality lowers the frequency of corrupted or dropped packets increases and thus more bandwidth is required and a delay occurs.

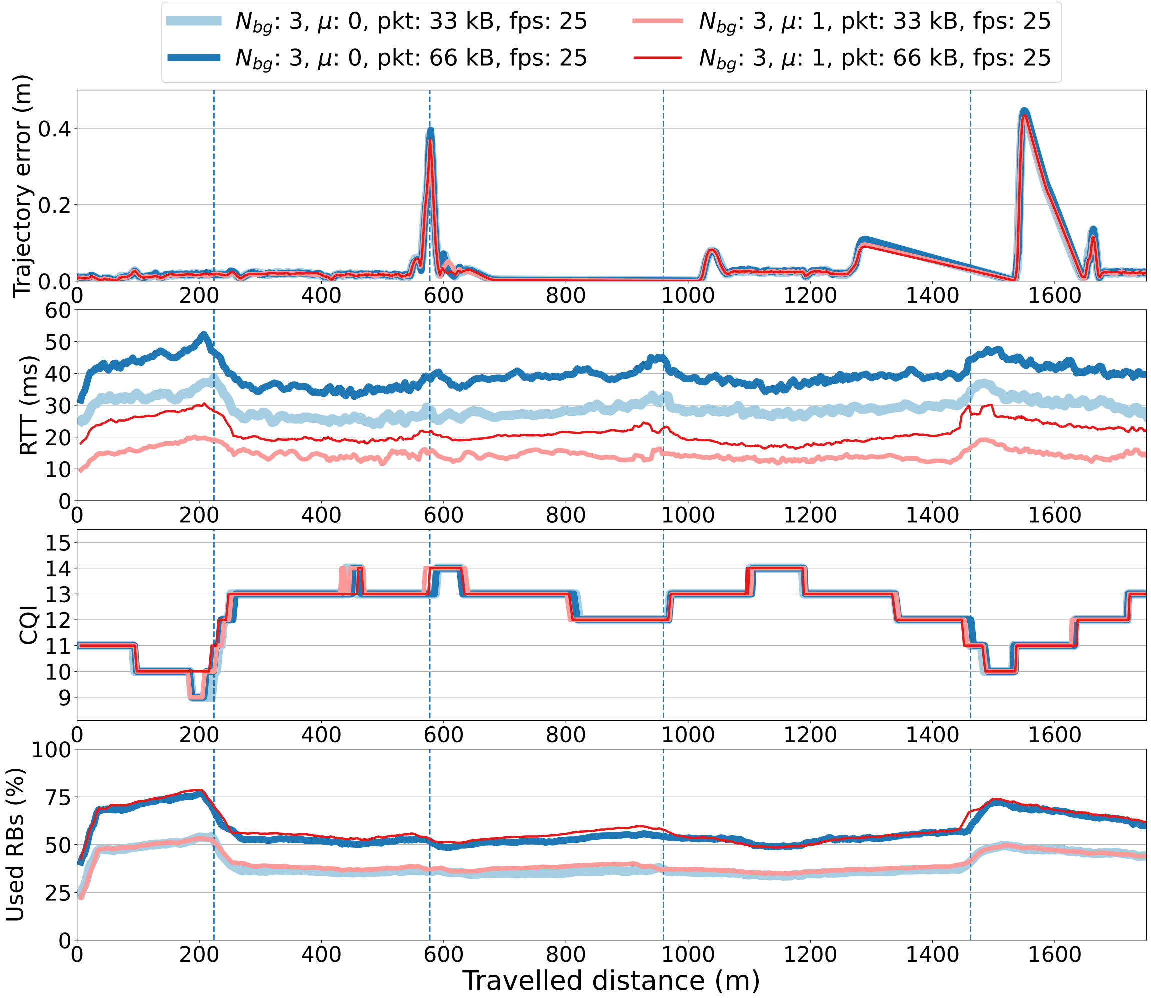


Figure 13: *City trip with 3 static background users - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 13 traces the trajectory error, RTT, CQI and used RBs in the presence of 3 background users near each base station for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps. Here the trajectory error is basically the same as the previous set of plots but the RTT and used RBs are showing an increase. This is a consequence of the background users' activity on the network, but still within the margins for safe operation.

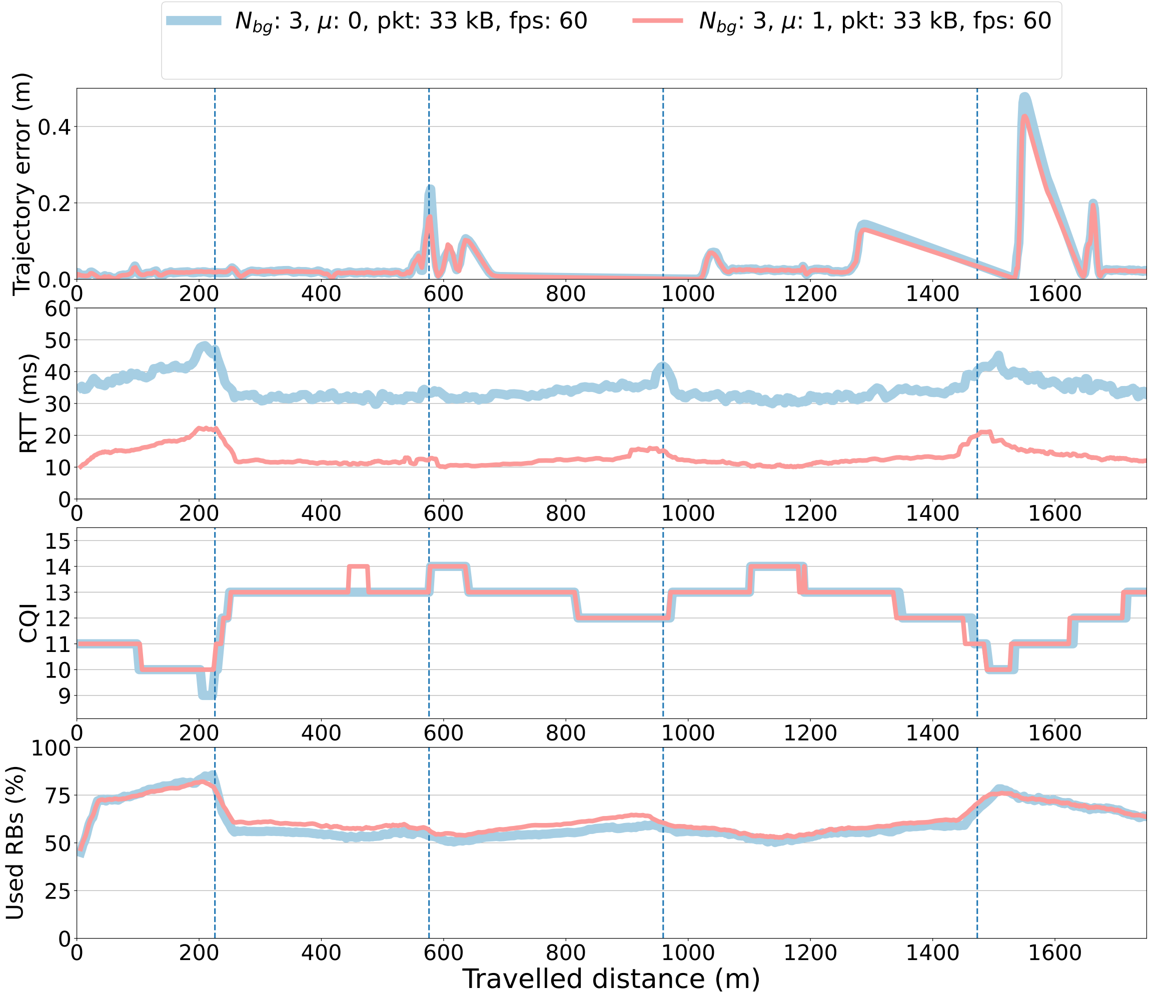


Figure 14: *City trip with 3 static background users - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin, 60 fps*

Figure 14 traces the trajectory error, RTT, CQI and used RBs in the presence of 3 background users near each base station for scenarios with packet size of 33kB, numerology of 0 and 1 and 60fps. The scenarios with a 66kB packet size are absent as each of their repetitions failed. The RTT is within safe values and in fact the trajectory error is also on the safe side. The percentage of used RBs is nearing full utilization in the sections of the route with lower channel quality. This gives a hint towards the reason why the scenarios with a bigger packet size failed.

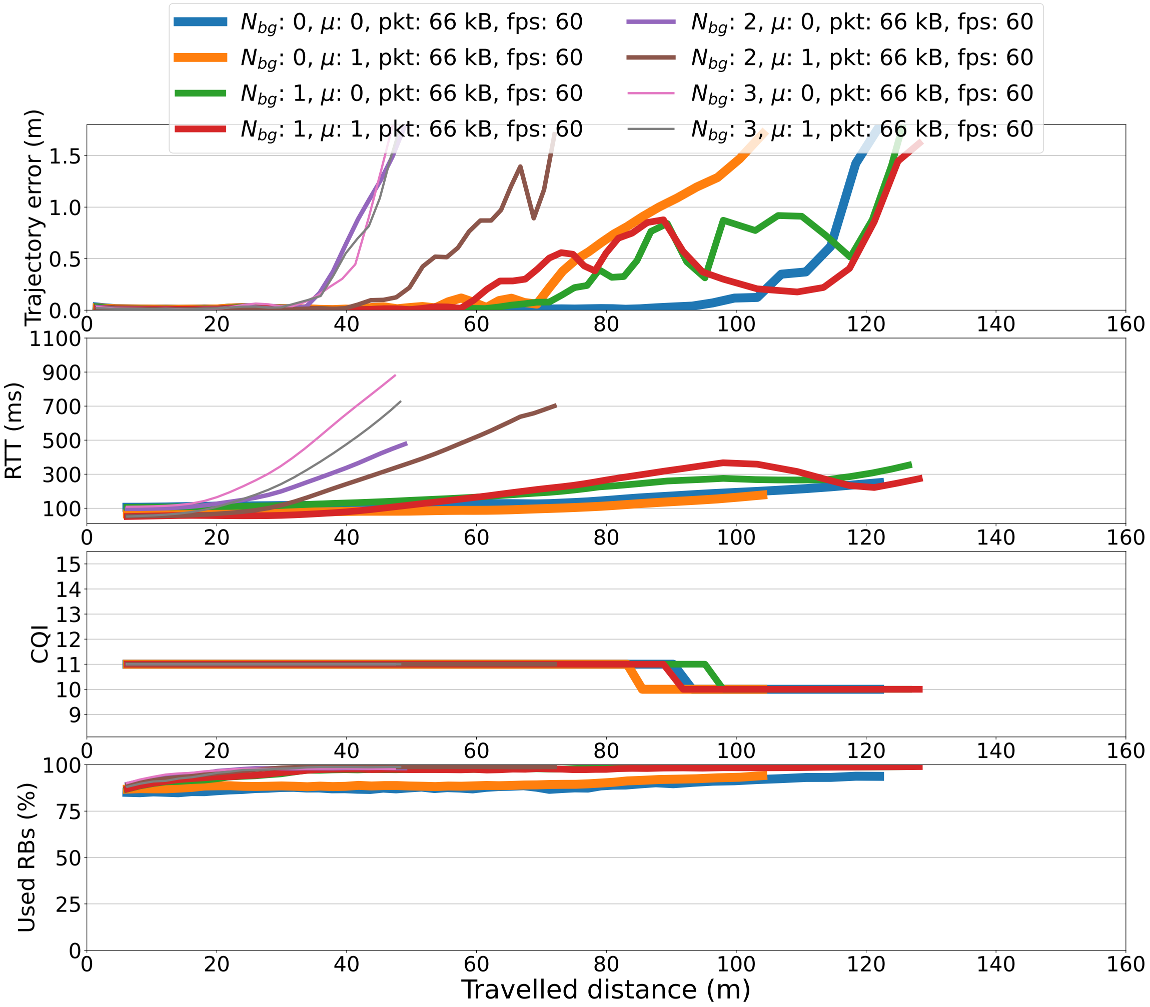
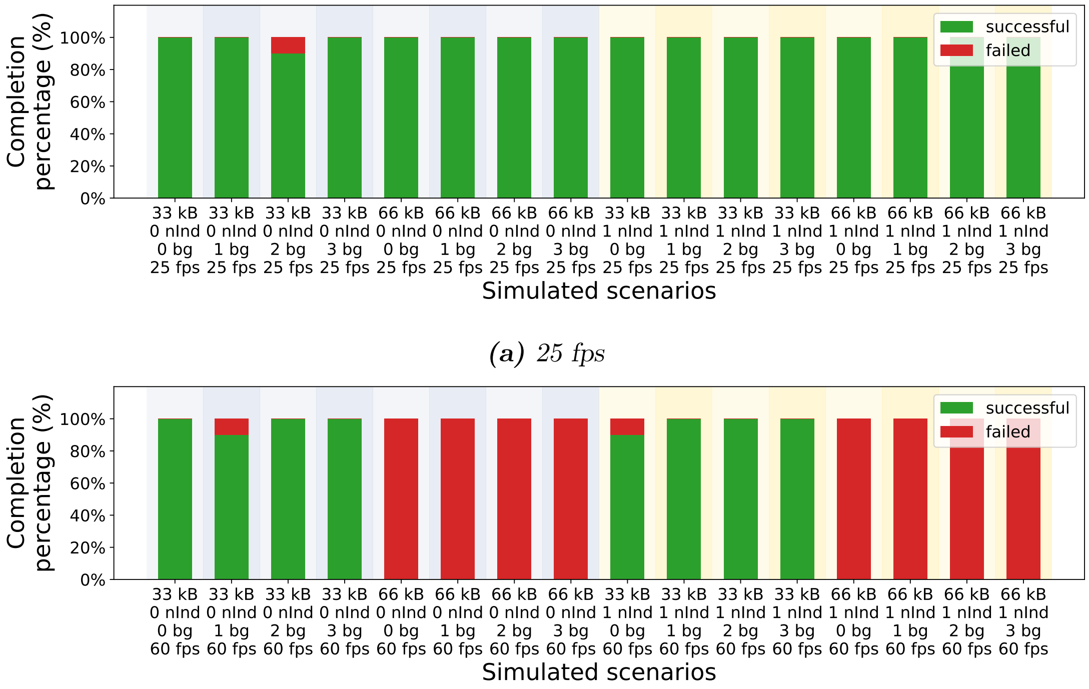


Figure 15: *City trip with static background failed runs - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin*

Figure 15 traces the trajectory error, RTT, CQI and used RBs for the scenarios where the simulations failed, i.e. a crash occurred. This happened at the start of the simulation and the main culprit is the complete exhaustion of the available amount of Resource Blocks in conjunction with the dwindling channel quality. The RTT skyrockets, the ToD operator is unable to keep control of the vehicle and a crash is inevitable.

4.2.2 Moving background



(a) *25 fps*

(b) *60 fps*

Figure 16: *City trip with moving background - Simulations completion percentage*

Figure 16 shows an overall look at the completion percentage across the 10 runs executed over each of the 32 considered parameter configurations.

The difference from the previous section is the nature of the background users, so the scenario with no background corresponds to the one in Figure 12.

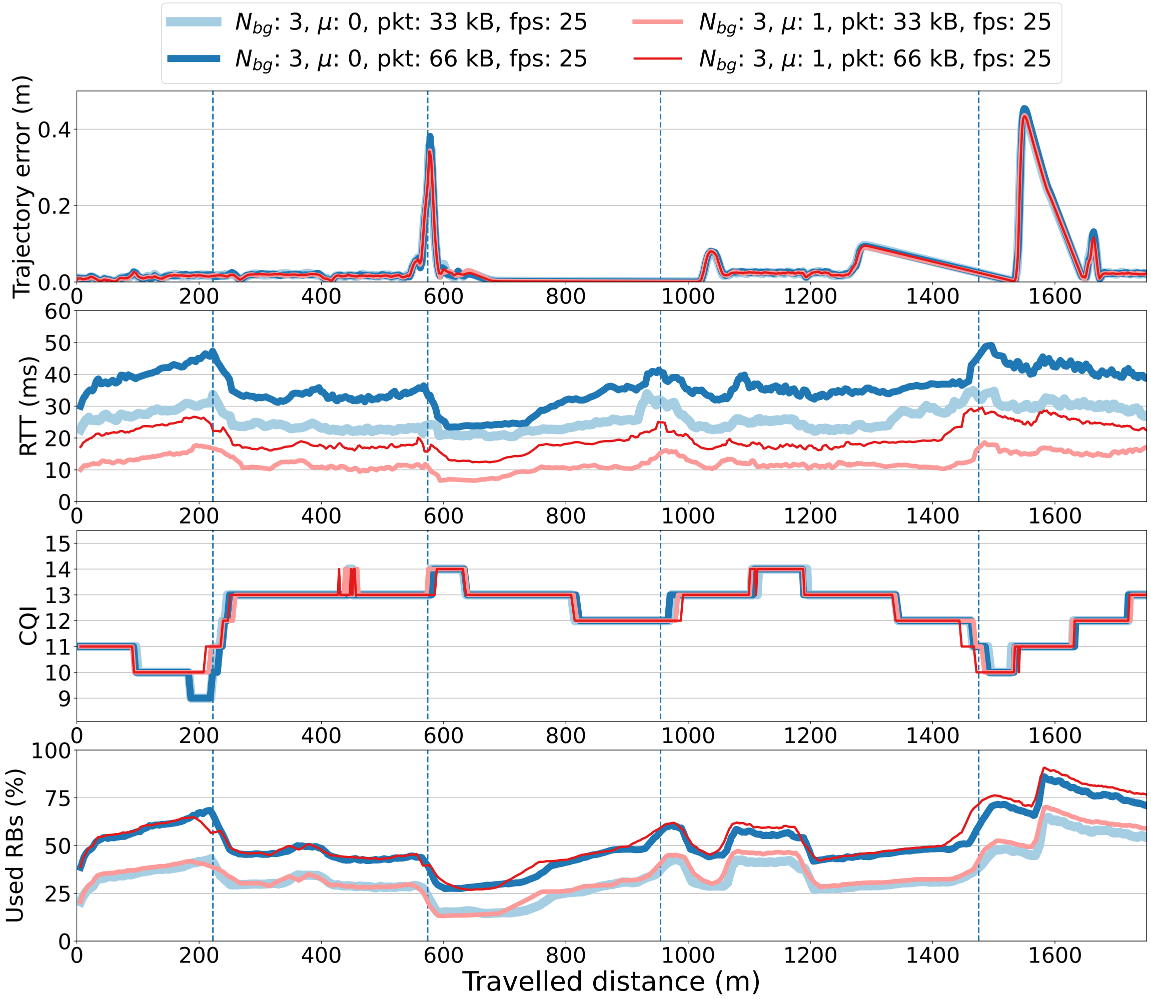


Figure 17: *City trip with 3 moving background users - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 17 traces the trajectory error, RTT, CQI and used RBs in the presence of 3 moving background users for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps.

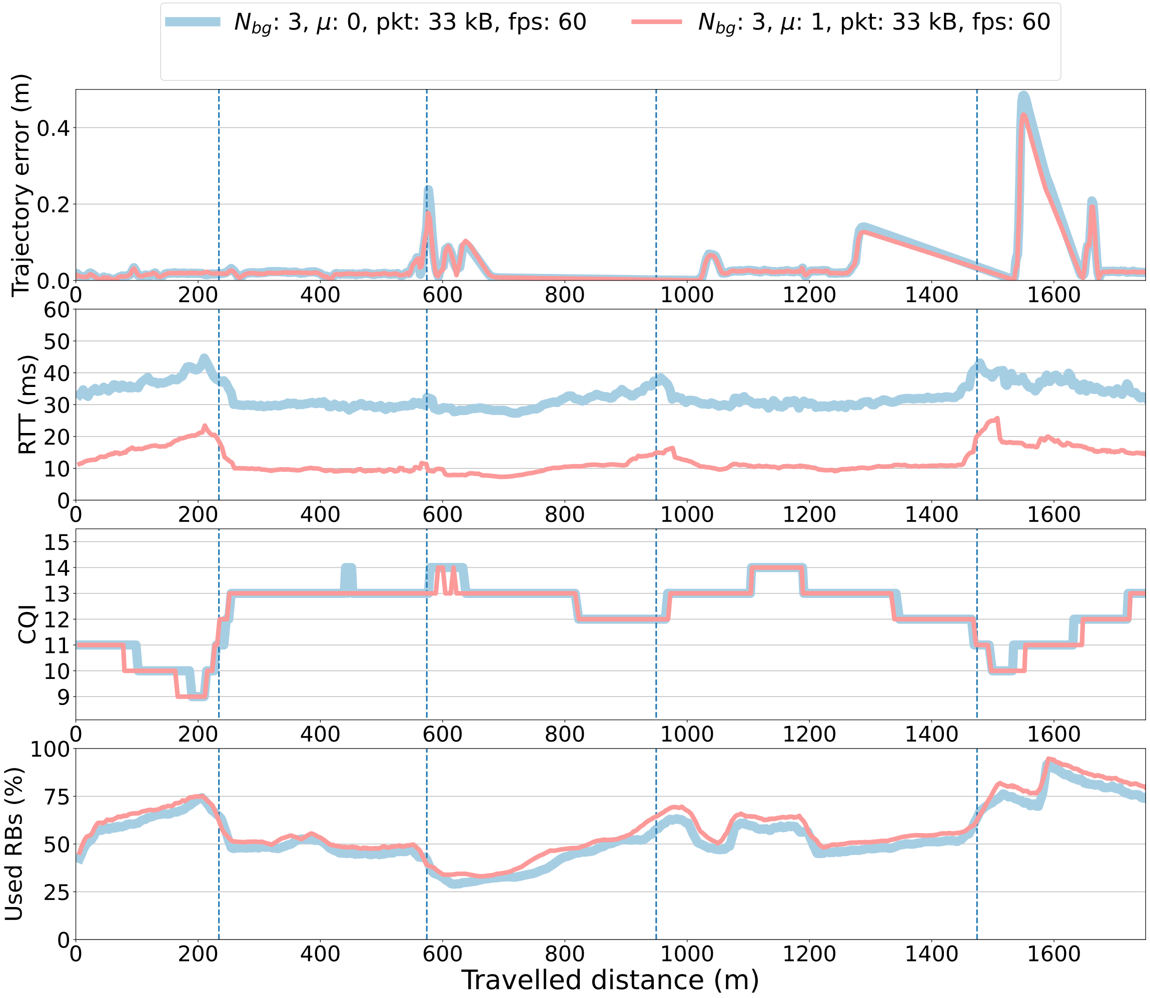


Figure 18: *City trip with 3 moving background users - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin, 60 fps*

Figure 18 traces the trajectory error, RTT, CQI and used RBs in the presence of 3 moving background users for scenarios with packet size of 33kB, numerology of 0 and 1 and 60fps. The scenarios with a 66kB packet size are absent as each of their repetitions failed.

It is noticeable that the moving background users make the network conditions much more challenging and unpredictable. This is because as they are traveling, though around the same area as the Host Vehicle, they will need to switch base stations and will each experience varying channel quality with its consequences. In both figures towards the end of the route the amount of used RBs almost tops out, leading to the danger of a crash.

The effect of randomness is observable in the failures experienced in some scenarios where in theory the available resources are enough, as 90% of the runs succeeded, but still there was a crash due to a sudden spike in latency, likely influenced by unforeseen conditions that the network couldn't efficiently handle.

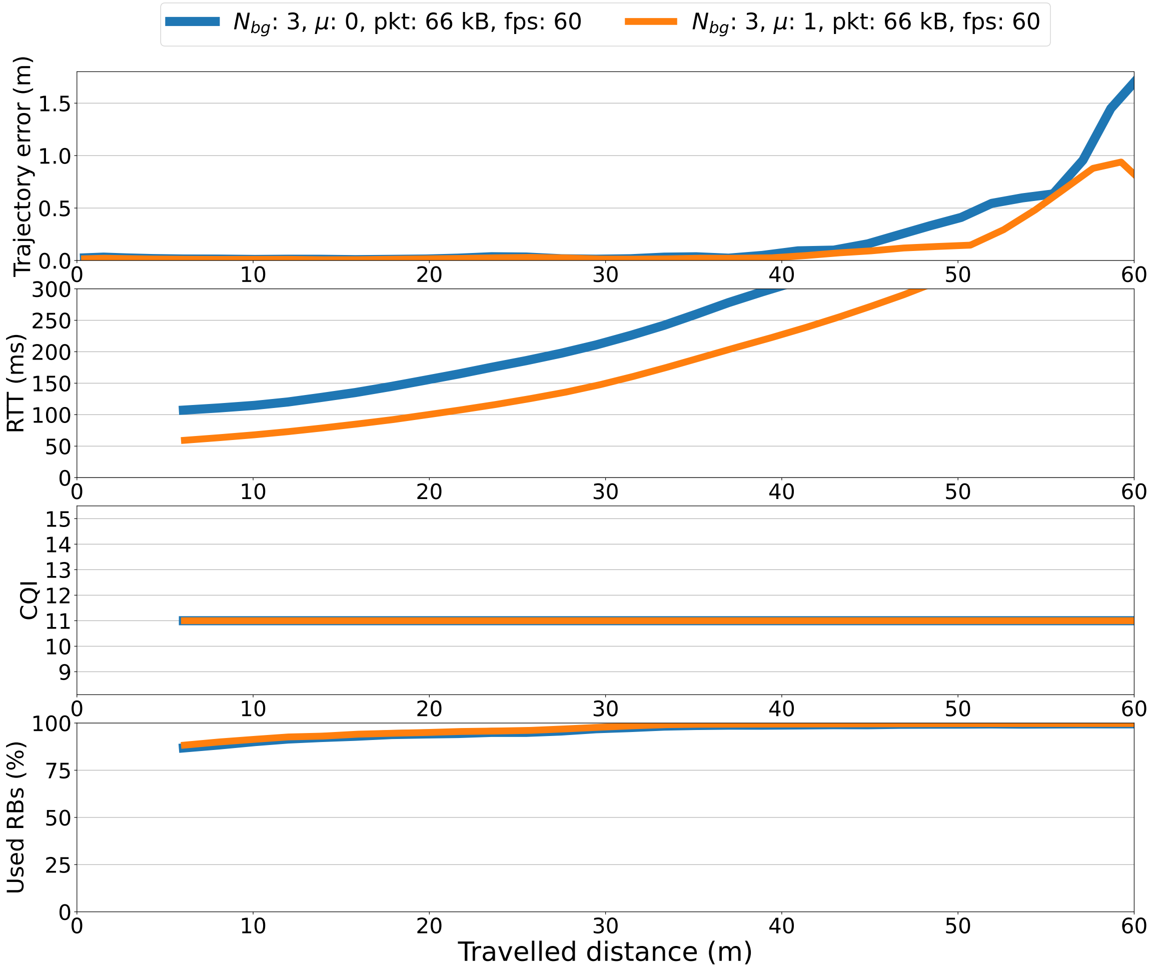


Figure 19: *City trip with moving background failed runs - Trajectory error, RTT, CQI, and allocated resource blocks over distance from origin*

Figure 19 traces the trajectory error, RTT, CQI and used RBs for the scenarios where the simulations failed. As was the case before, these configurations are the most demanding ones in terms of data rate and QoS, hence the result is the same: the amount of used RBs goes to 100% from the start and the operator loses control right away.

4.3 Reaction to obstacles

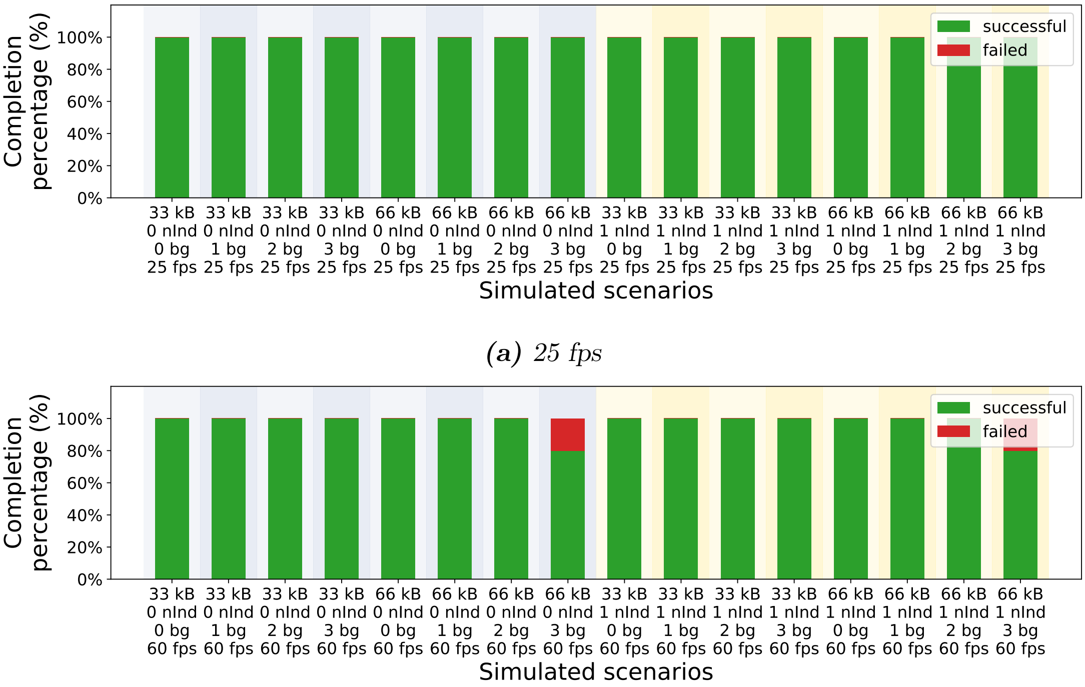
This section contains the results of the experiments conducted in the presence of obstacles. In the case of a city context, both obstacles present from the start and suddenly appearing were tested. For the highway scenarios, only the latter option was chosen.

As was the case for the city trip results, the completion percentage for each scenario will be plotted as bars colored in green for the successful share of runs and red for the share of failed runs.

The following 4-subplot configuration (e.g. in Figure 21) resembles the one described for the city trips, the only difference being that since the trajectory error is not meaningful in such scenarios as the ones involving obstacles, the first subplot instead traces the distance to the obstacle location (when present in the world).

The distance to the obstacle, RTT, CQI and RB utilization percentage will be traced as subplots sharing the same x-axis corresponding to the distance from the origin point. In some cases, the plots will be trimmed not to start from the exact origin in order to better emphasize the important section which is towards the end, where the obstacle is.

4.3.1 Static obstacle



(a) 25 fps

(b) 60 fps

Figure 20: *City static obstacle - Simulations completion percentage*

Figure 20 shows an overall look at the completion percentage across the 10 runs executed over each of the 32 considered scenarios. The reason why this test was not repeated for the highway case is that, as the results show, there is almost no situation where a collision cannot be avoided. There were a couple of cases where a collision indeed occurred and they were all a small number of runs for the scenarios with the most load on the network which would make even a simple turn in the road impossible while using ToD.

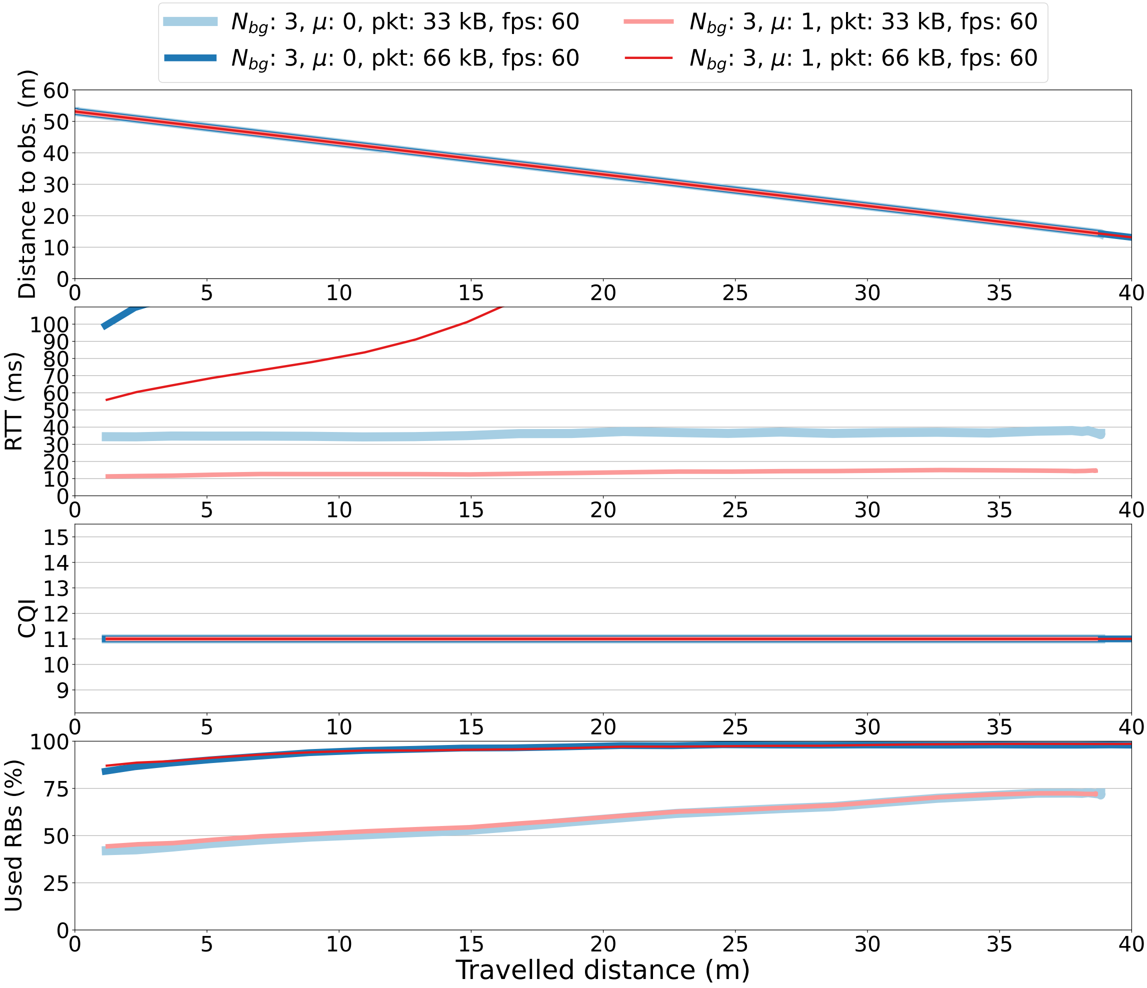


Figure 21: *City static obstacle with 3 background users - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 60 fps*

Figure 21 traces the distance to obstacle, RTT, CQI and used RBs in the presence of 3 background users per base station for scenarios with packet size of 33kB and 66kB, numerology of 0 and 1 and 60fps, i.e. the most demanding scenarios, for runs that did not fail. The scenarios with the bigger packet size saturate the available resource blocks and the latency goes off the charts. Nevertheless, at least some instructions go through and a collision is avoided.

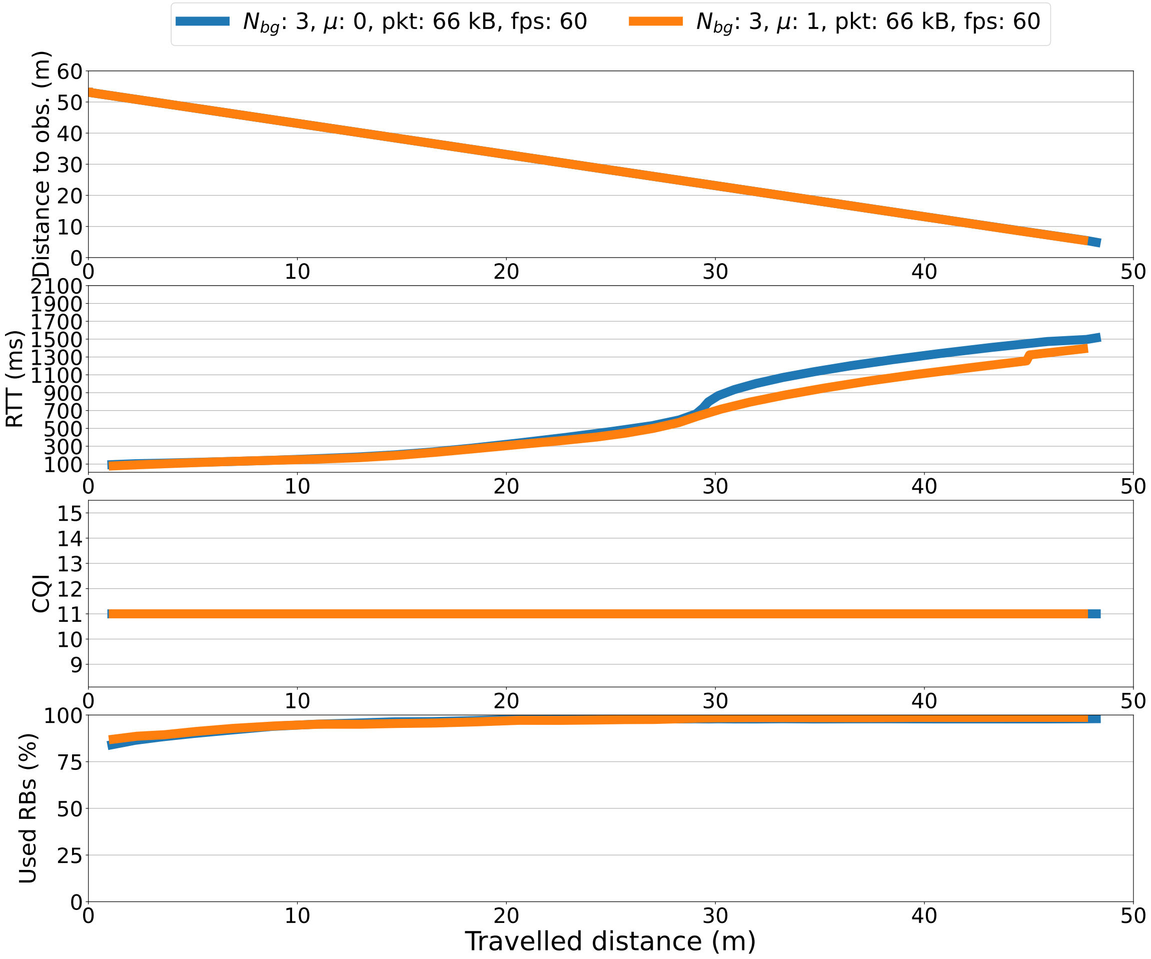
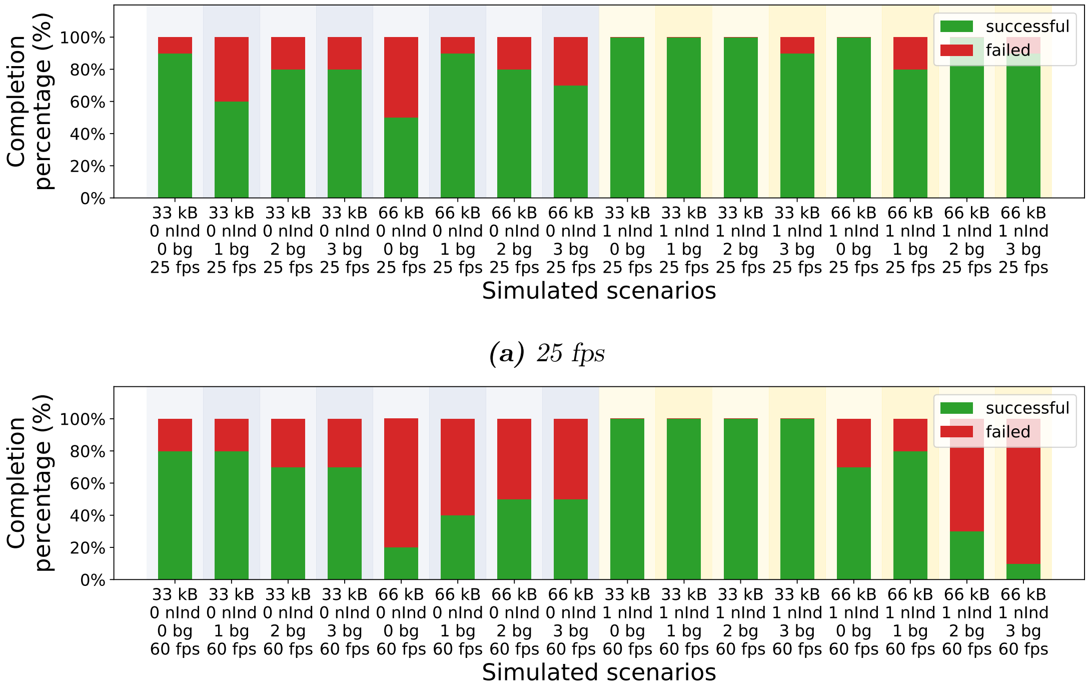


Figure 22: *City static obstacle failed runs - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin*

Figure 22 shows the same metrics for the very few failed runs. These are the only cases where, even when the obstacle is observed from afar, the strain on the network is so much that the ToD operator is unable to stop the vehicle.

4.3.2 City sudden obstacle

The sudden obstacle scenarios tell a much more interesting story.



(a) *25 fps*

(b) *60 fps*

Figure 23: *City sudden obstacle - Simulations completion percentage*

Figure 23 illustrates the completion rate for each scenario and the most noticeable pattern is that the right side of both subfigures is much greener overall than the left, the difference being the numerology index μ, which is 0 on the left and 1 on the right. The only exception is for those scenarios where the bandwidth is saturated.

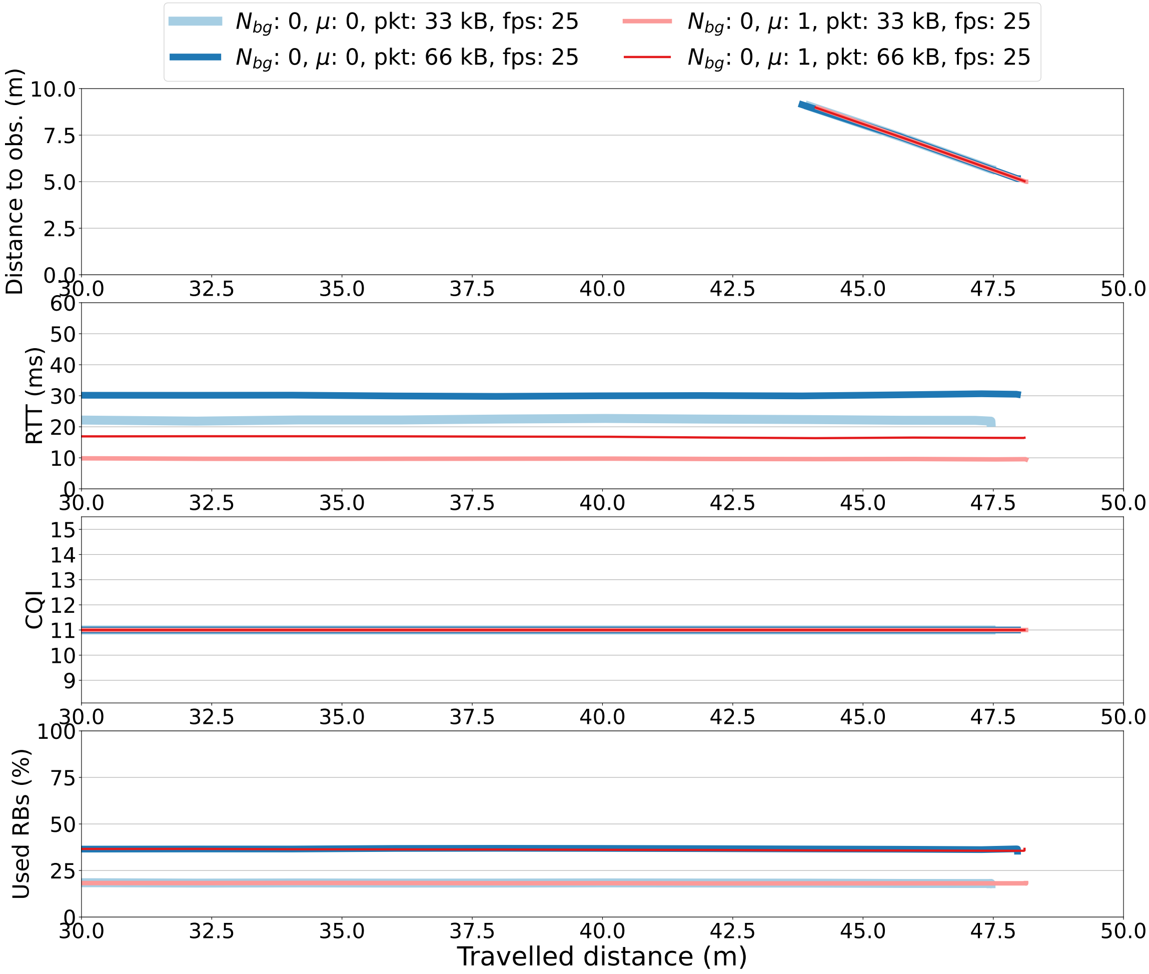


Figure 24: *City sudden obstacle with no background - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 24 traces the distance to obstacle, RTT, CQI and used RBs in the absence of background for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps, each averaged over the respective successful runs.

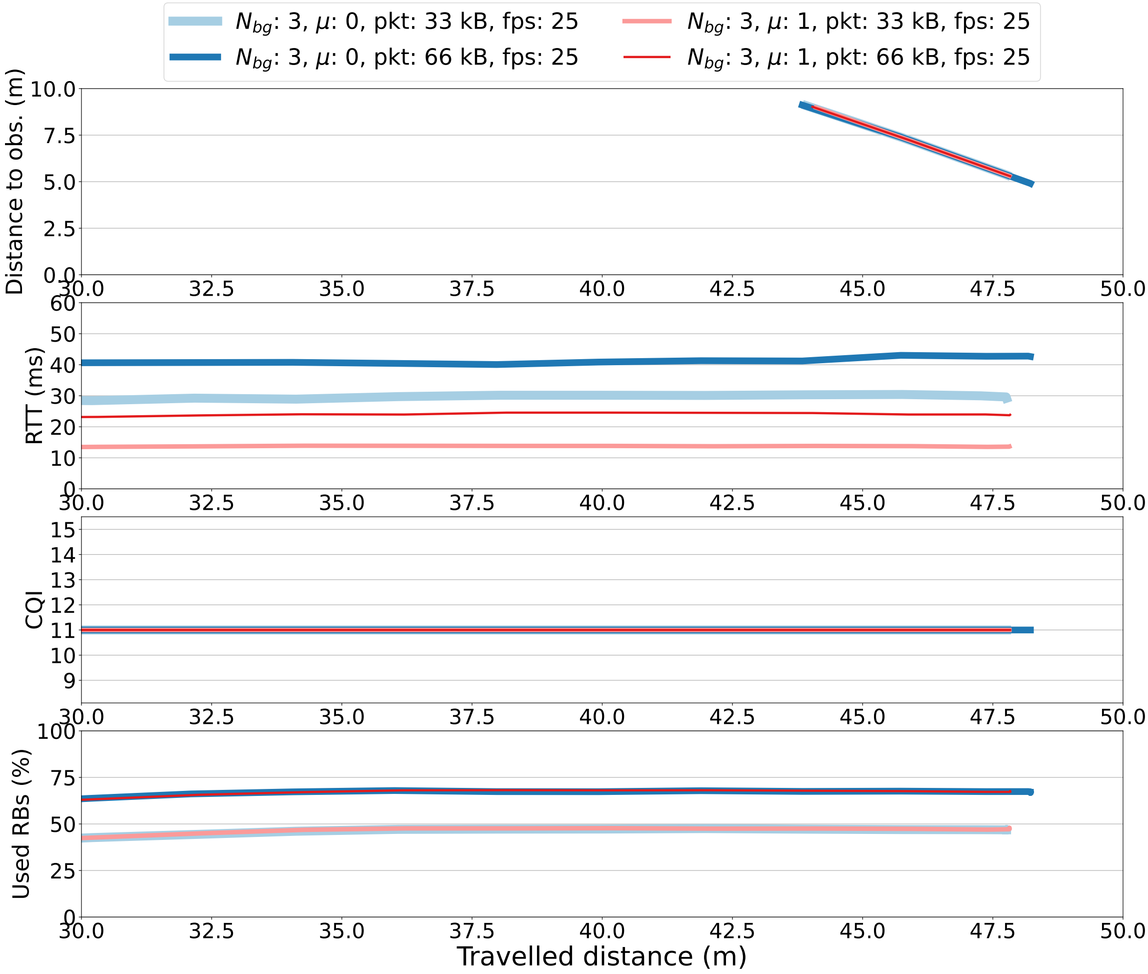


Figure 25: *City sudden obstacle with 3 background users - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 25 traces the distance to obstacle, RTT, CQI and used RBs in the presence of 3 background users for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps.

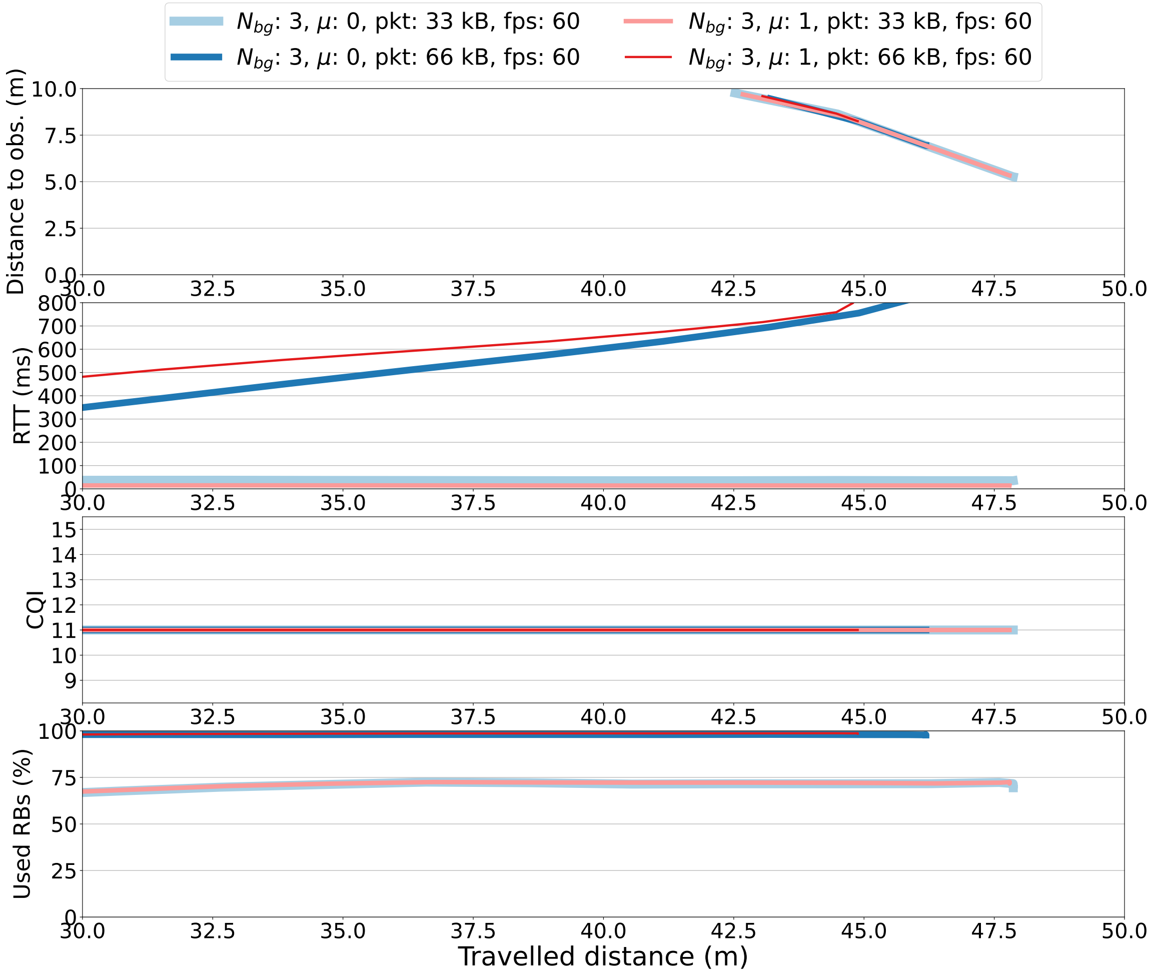


Figure 26: *City sudden obstacle with 3 background users - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 60 fps*

Figure 26 traces the distance to obstacle, RTT, CQI and used RBs in the presence of 3 background users for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 60fps. It can be noticed that for the 66kB scenarios the absence of a crash was a stroke of luck, as the resource block utilization is pegged at 100% and the delay is completely out of range for safe operation. In fact, the successful runs make up a minority of the 10 runs for each scenario.

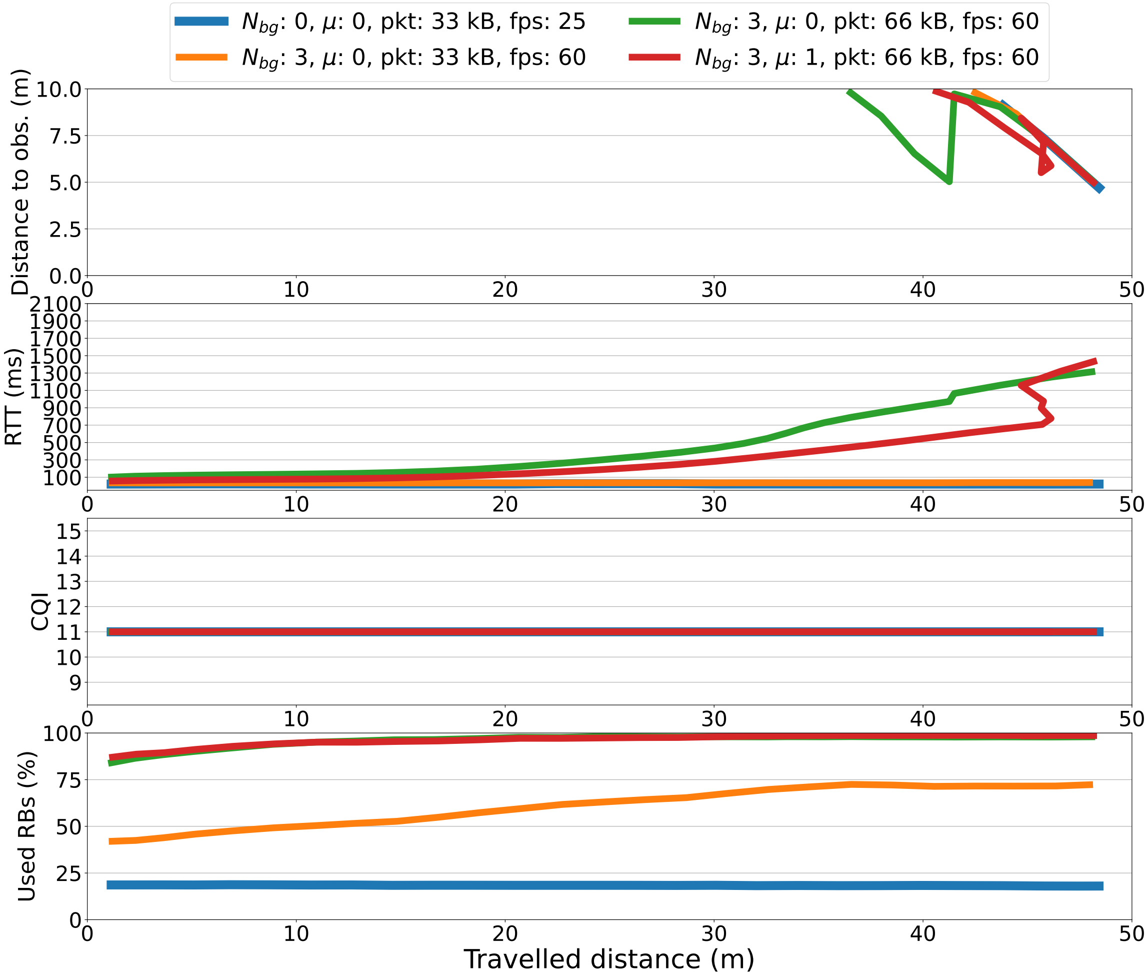
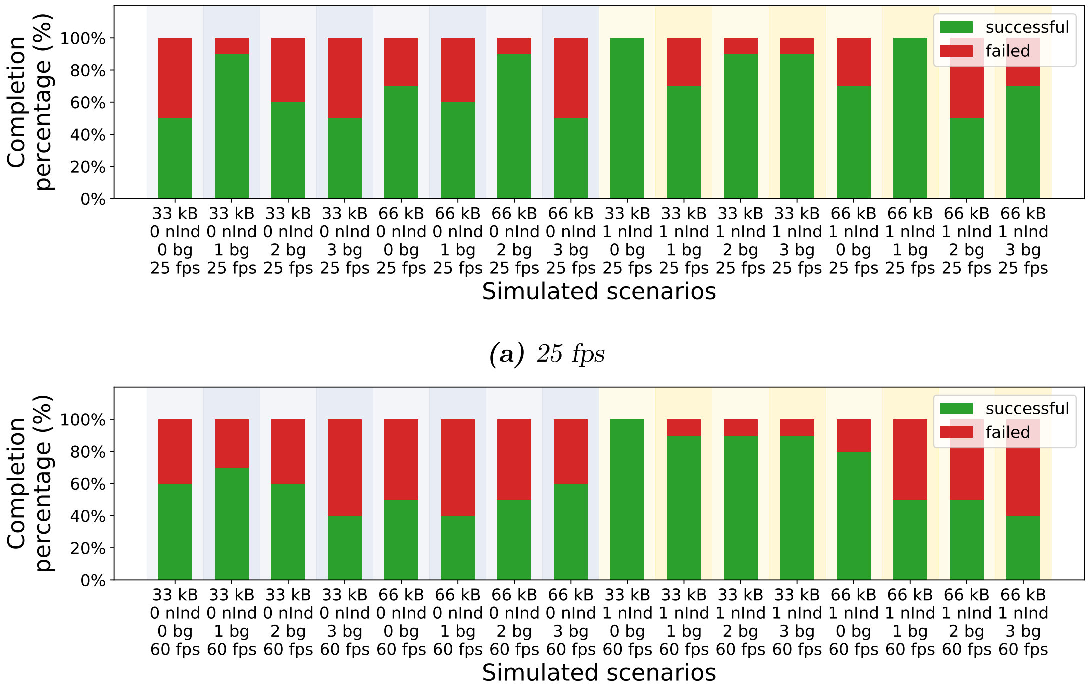


Figure 27: *City sudden obstacle failed runs - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin*

Figure 27 traces the distance to obstacle, RTT, CQI and used RBs for a selection of the failed runs. For cases where the resource block utilization and RTT are within regular margin, the failures have to do with the slight variance of network conditions across different random seeds in a situation where timing is very tight. The trend is apparent in the overall success/failure rates. For the rest of the scenarios, it is evident that since the channel is saturated and packets pile up, it is impossible for data to be exchanged with the vehicle and stop it in time.

4.3.3 Highway sudden obstacle



(a) *25 fps*

(b) *60 fps*

Figure 28: *Highway sudden obstacle - Simulations completion percentage*

Figure 28 illustrates the completion rate for each scenario and the same considerations as the city context hold true: scenarios with numerology index μ of 1 have a higher success rate even as the available bandwidth stays the same because of the overall lower RTT.

It is worth noting that the selected spawn distance of 44 meters was purposely chosen as an *edge case*, which shows that in tight circumstances even a few milliseconds of added or reduced delay will make a difference. However, the same tests repeated using a distance of 45 meters yielded an almost completely successful set of simulations.

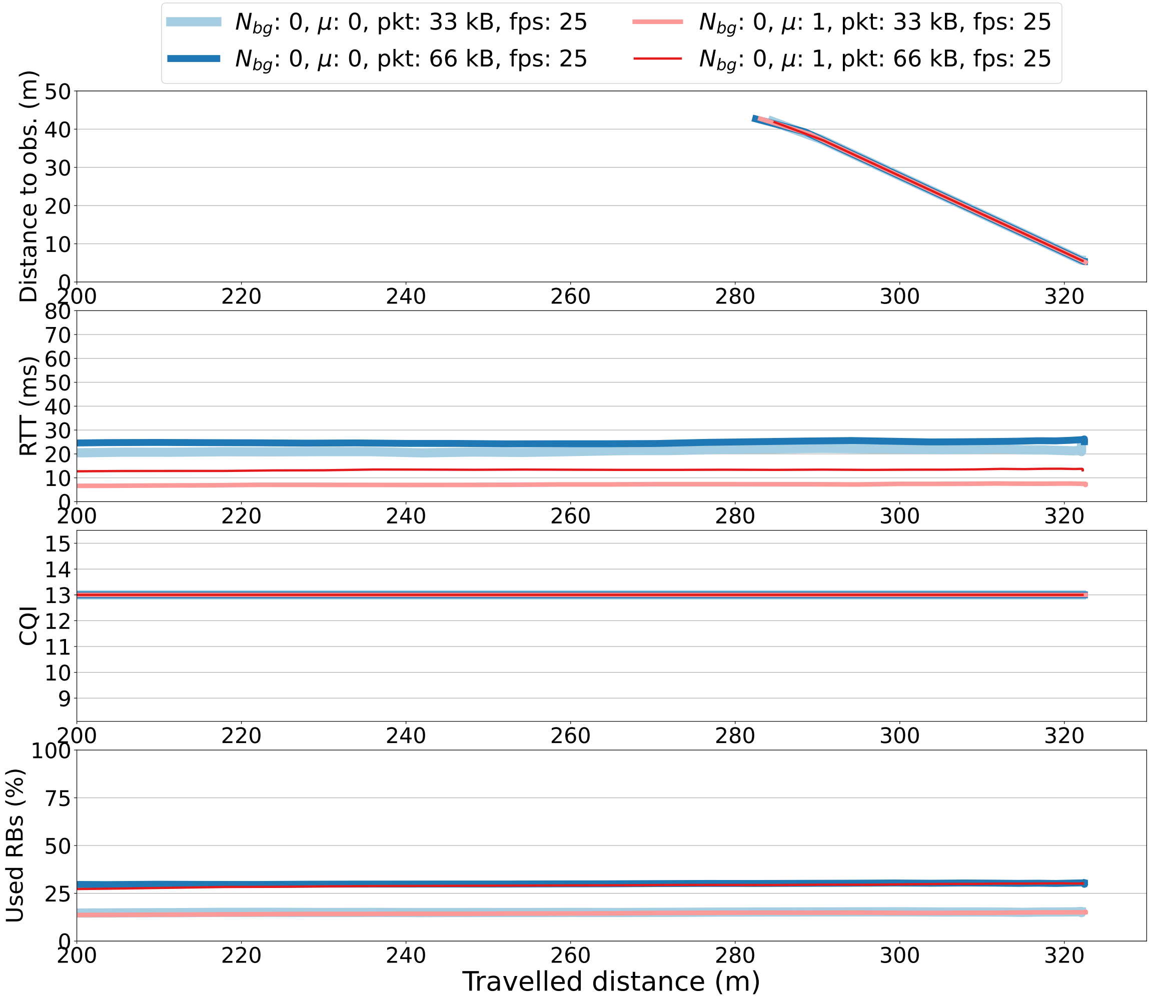


Figure 29: *Highway sudden obstacle with no background - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 29 traces the distance to obstacle, RTT, CQI and used RBs in the absence of background for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps, each averaged over the respective successful runs.

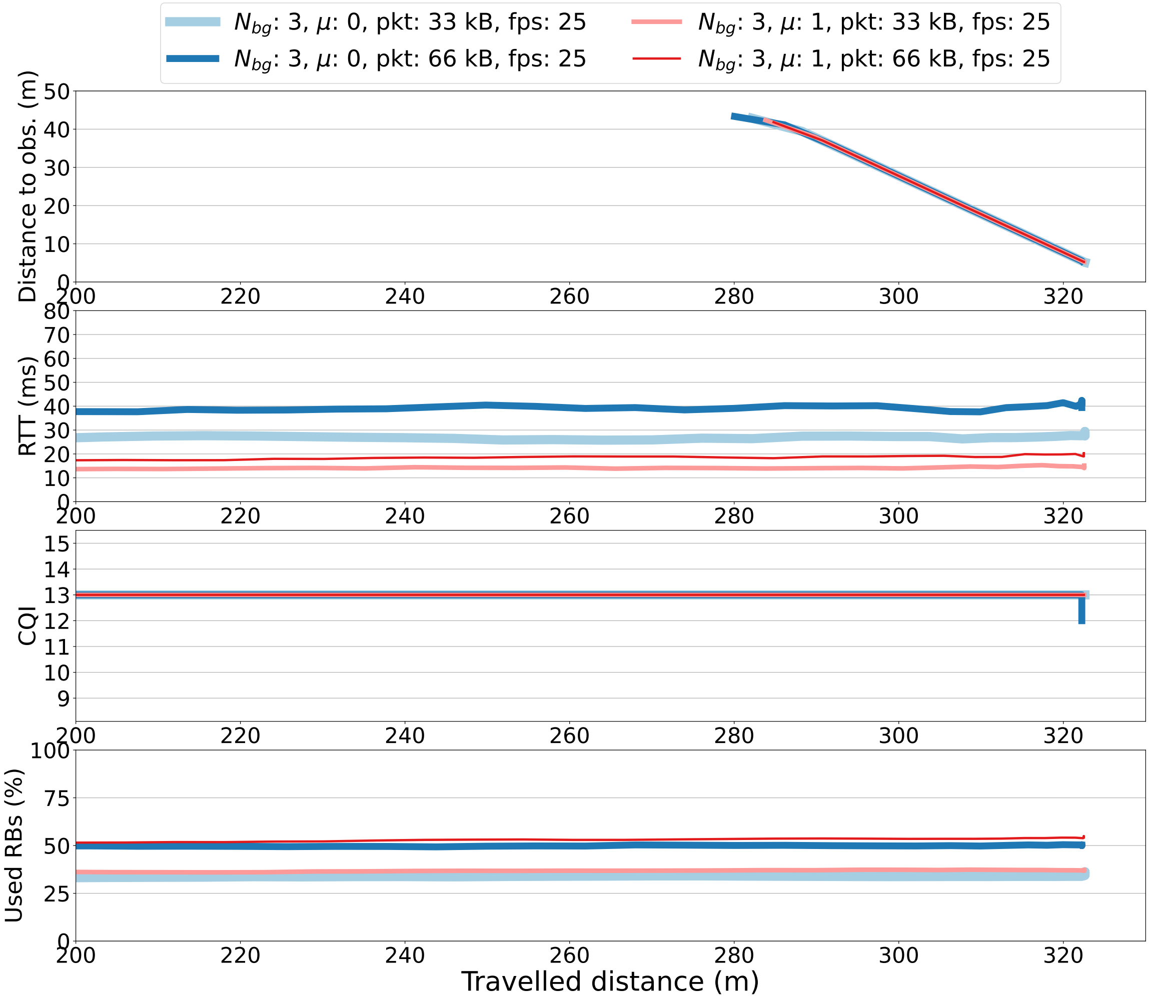


Figure 30: *Highway sudden obstacle with 3 background users - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 25 fps*

Figure 30 traces the distance to obstacle, RTT, CQI and used RBs in the presence of 3 background users for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 25fps.

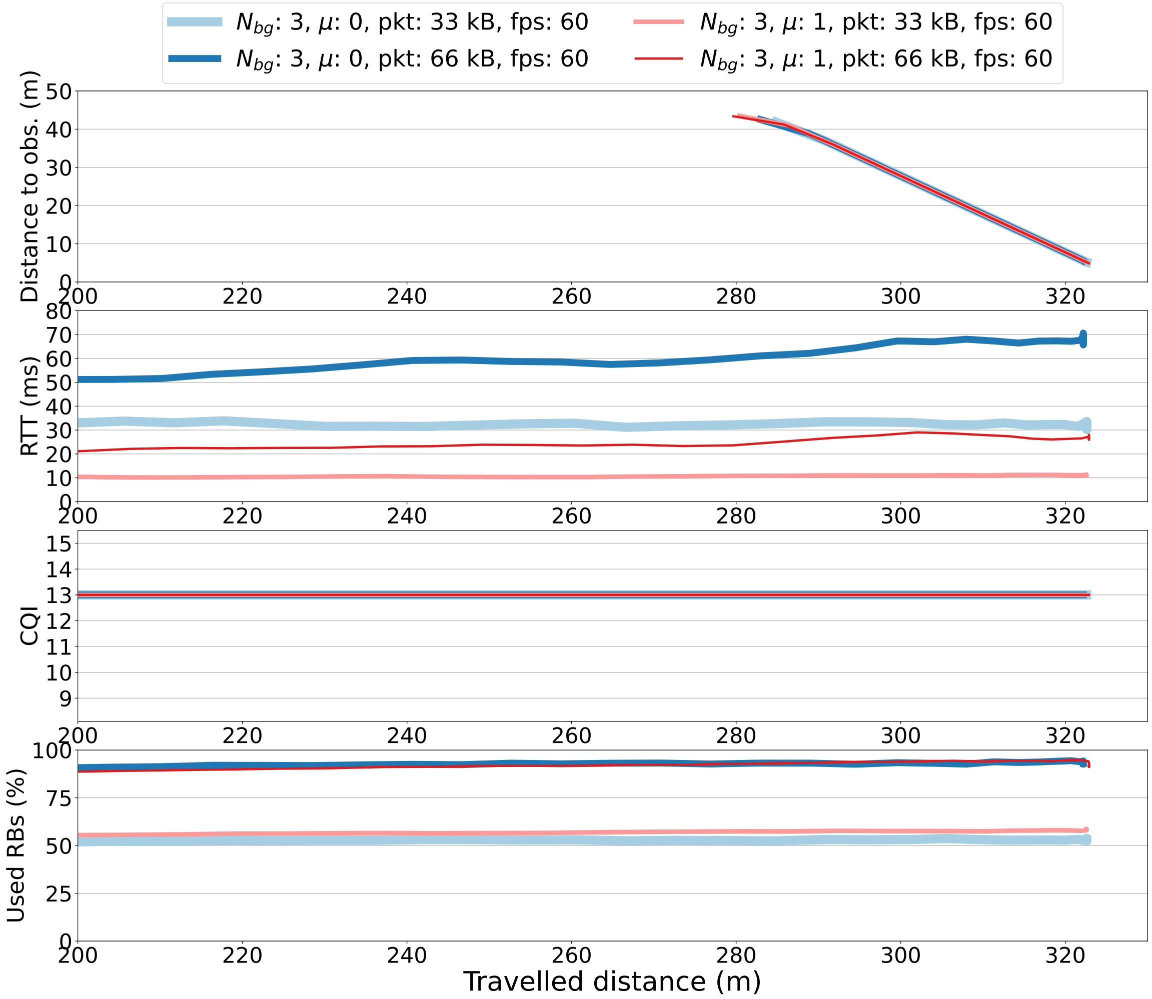


Figure 31: *Highway sudden obstacle with 3 background users - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin, 60 fps*

Figure 31 traces the distance to obstacle, RTT, CQI and used RBs in the presence of 3 background users for scenarios with packet size of 33 and 66kB, numerology of 0 and 1 and 60fps. Differently from the previous context, the resource block utilization on average doesn't tend to saturate just as much, likely because of the slightly higher channel quality, but still it's not at a level that would tolerate much variation in the network conditions.

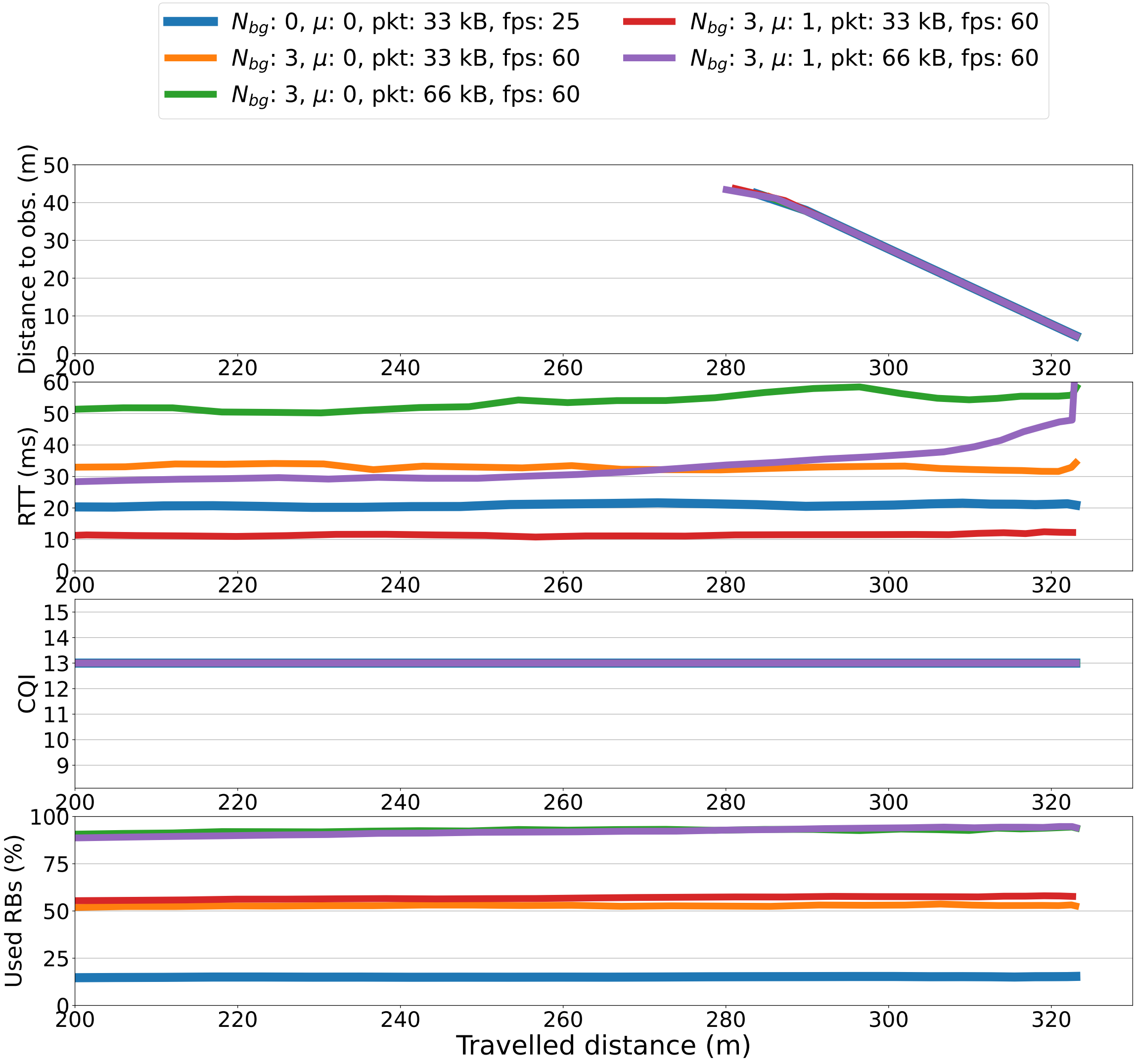


Figure 32: *Highway sudden obstacle failed runs - Distance to obstacle, RTT, CQI, and allocated resource blocks over distance from origin*

Figure 32 traces the distance to obstacle, RTT, CQI and used RBs for a selection of the failed runs. It shows that there were failures for all levels of network load and delay, once again influenced by randomness.

4.4 Discussion

The simulation results have shown that Tele-Operated Driving is a possibility under the right circumstances and with the right setup. With a 20MHz bandwidth available and especially with other users impacting the shared medium, it is important to mind the data rates of the video feeds and the refresh rate of the information sent to the remote ToD operator.

The most impactful parameter has been the Round-Trip Time calculated as the time between the detection of the environment by the sensors on the HV and the corresponding instruction issued back by the ToD operator based on that data. The more it increases the more disconnected the Operator is from reality time-wise and the harder it is for them to keep a steady control of the vehicle, avoiding drifting too much from the predetermined trajectory and eventually crashing into a wall or not stopping in time for a sudden emergency.

The RTT is greatly influenced by the load posed on the network by the HV itself and by other users. The channel quality also has an impact in the choice of more or less efficient modulation schemes thus impacting the effective maximum throughput at any given time. The varying distance from the base station and subsequent handover to a closer one when moving around the world are also critical. An exhaustion of the available network resources must be avoided as continuous communication is vital for a successful execution of the remote driving tasks.

Overall, configurations with a numerology index μ of 1 have proven to be more reliable, because of its generally lower delay in exchanging data and instructions, especially in the sudden obstacle scenarios where even a reduction in latency of a few milliseconds makes a big difference.

Conclusion

The work presented in this thesis has conducted a performance evaluation of a simulated ToD service over a 5G network in a multitude of different scenarios.

An overview and definition of a Tele-Operated Driving service were provided, as described by the 5G Automotive Association, together with its technical and functional requirements and a discussion of some possible use cases.

In order to conduct the performance evaluation, a software basis made up of CARLA driving simulator and OMNeT++ network simulator, mated through a message-oriented middleware that enables co-simulation, was employed. It was expanded to allow for the testing of specific scenarios involving more than one vehicle and special spawning conditions for obstacles. More specifically, a long route spanning highway and urban sections of the chosen map was evaluated using increasing levels of network background generated in two different ways. A couple of scenarios involving obstacles on the vehicle's path were also devised, both with the obstacle in a fixed position from the start in an urban environment and suddenly appearing in a city and highway context. Each simulated scenario was evaluated over a set of parameter configurations that condition the bandwidth needed by the Host Vehicle and the amount of background network traffic.

An analysis of the recorded simulation data was provided for relevant parameter configurations of each considered scenario, and it was shown that Tele-Operated Driving is certainly a possibility under the right conditions and with the correct setup. It is possible to achieve safe operation within the specifications and requirements of the 5GAA even with a basic 5G network configuration.

The most impactful parameter in the success or failure of this type of service is the overall latency, or RTT, which was generally lower with a numerology index μ of 1 and influenced by channel quality and network load.

Future work might include some experimentation with Carrier Aggregation for the networking side and tests in situations with external vehicle traffic using different agents as ToD operators.

Achieving real-time operation of the simulation would allow for testing with human operators and verifying the effect the network delay has on their ability to keep control of the vehicle.

An interesting expansion of the simulation framework would be towards being able to gather additional data from the infrastructure itself and leveraging multi-access edge computing (MEC) [18], where collisions can be more easily avoided by assessing and predicting movements made by other road users.

Bibliography

[1] 5GAA, “Tele-Operated Driving (ToD): Use Cases and Technical Requirements,” 5G Automotive Association, Tech. Rep., 2021. [Online]. Available: https://5gaa.org/content/uploads/2021/08/ToD_D1.1-Use-Cases-and-Technical-Requirements.pdf

[2] Fifth Generation Cross-Border Control (5GCroCo), “Test Case Definition and Test Site Description for Second Round Tests and Trials,” https://5gcroco.eu/images/templates/rsvario/images/5GCroCo_D2_2.pdf, 2021.

[3] C. Bachhuber and E. Steinbach, “A system for high precision glass-to-glass delay measurements in video communication,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 2132–2136.

[4] V. Cislaghi, C. Quadri, V. Mancuso, and M. A. Marsan, “Simulation of Tele-Operated Driving over 5G Using CARLA and OMNeT++,” in *2023 IEEE Vehicular Networking Conference (VNC)*, 2023, pp. 81–88.

[5] CARLA Documentation contributors. (2024) CARLA Documentation. [Online]. Available: https://carla.readthedocs.io/en/latest/

[6] R. Tavenard. (2021) An introduction to dynamic time warping. [Online]. Available: https://rtavenar.github.io/blog/dtw.html

[7] R. Liu, D. Kwak, S. Devarakonda, K. Bekris, and L. Iftode, "Investigating Remote Driving over the LTE Network," in *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, ser. AutomotiveUI '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 264–269. [Online]. Available: https://doi.org/10.1145/3122986.3123008

[8] SAE International Automotive and ISO, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," International Organization for Standardization, Standard, 2021. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:iso-sae:pas:22736:ed-1:v1:en

[9] 5GAA, "Tele-Operated Driving (ToD): System Requirements Analysis and Architecture," 5G Automotive Association, Tech. Rep., 2021. [Online]. Available: https://5gaa.org/content/uploads/2021/09/5GAA_ToD_System_Requirements_Architecture_TR.pdf

[10] S. Neumeier, P. Wintersberger, A.-K. Frison, A. Becher, C. Facchi, and A. Riener, "Teleoperation: The Holy Grail to Solve Problems of Automated Driving? Sure, but Latency Matters," in *Proceedings of the 11th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, ser. AutomotiveUI '19. Association for Computing Machinery, 2019, p. 186–197. [Online]. Available: https://doi.org/10.1145/3342197.3344534

[11] A. Varga, *OMNeT++*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–59. [Online]. Available: https://doi.org/10.1007/978-3-642-12331-3_3

[12] INET Framework contributors. (2024) INET Framework - An open-source OMNeT++ model suite for wired, wireless and mobile networks. [Online]. Available: https://inet.omnetpp.org/

[13] G. Nardini, G. Stea, and A. Virdis. (2024) Simu5G: simulator for 5G New Radio networks. [Online]. Available: https://inet.omnetpp.org/

[14] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G–An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks," *IEEE Access*, vol. 8, pp. 181 176–181 191, 2020.

[15] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 1–16. [Online]. Available: https://proceedings.mlr.press/v78/dosovitskiy17a.html

[16] ETSI, "3GPP TS 38.300 version 15.16.0 Release 15," European Telecommunications Standards Institute, Sophia-Antipolis, France, Standard, Feb. 2024.

[17] R. S. D. Sousa, A. Boukerche, and A. A. F. Loureiro, "Vehicle Trajectory Similarity: Models, Methods, and Applications," *ACM Comput. Surv.*, vol. 53, no. 5, sep 2020. [Online]. Available: https://doi.org/10.1145/3406096

[18] M. Malinverno, G. Avino, C. Casetti, C. F. Chiasserini, F. Malandrino, and S. Scarpina, "Edge-Based Collision Avoidance for Vehicles and Vulnerable Users: An Architecture Based on MEC," *IEEE Vehicular Technology Magazine*, vol. 15, no. 1, pp. 27–35, 2020.

Acknowledgements

Quando ho iniziato il mio percorso universitario, ormai quasi sei anni fa, questo momento appariva lontanissimo, eppure eccoci qui!

Un sentito ringraziamento va in primis al mio relatore, Christian Quadri, per avermi supportato nell'inizio molto travagliato del mio percorso di tesi magistrale e per avermi affiancato durante tutta la sua durata. Il suo aiuto è stato fondamentale e non penso ci sarebbe stata una tesi da discutere oggi senza le sue preziose dritte.

Ringrazio infinitamente i miei genitori, i miei nonni e tutta la mia famiglia per avermi permesso di studiare ciò che desideravo ed essermi stati vicino, supportandomi in ogni momento di difficoltà. Sono sempre stati e sempre saranno un punto di riferimento per me.

Ringrazio Marisa, Elisa e Silvano per avermi accolto in casa loro durante il mio percorso di studi e avermi fatto sentire parte della famiglia.

Ringrazio Paolo per avermi dato il consiglio giusto al momento giusto per la scelta dell'università, senza di lui non sarei qui adesso.

Ringrazio i tanti amici acquisiti durante questi anni di università per aver reso l'intera esperienza memorabile, per avermi accompagnato in tutte le fasi serene e turbolente che si sono susseguite e per avermi regalato legami che dureranno nel tempo. Non sarebbe stato lo stesso senza di loro.

I would like to thank all the wonderful people I met during my time in Zürich. They helped me grow up and mature as a person at a faster rate than ever, and I will always be grateful for that. In particular, I would like to thank Zoelle for being a great flatmate and for the company we've kept each other during the making of our theses, even from halfway across the globe.