Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/XXXX.2022.1234567

Towards Optimal Placement & Runtime Migration of Time-Sensitive Services of Connected and Automated Vehicles

Osama Elgarhy[1], Yannick Le Moullec[1], Luca Reggiani[2], Muhammad Moazam Azeem[3], Member, IEEE, Tarik Taleb [4], Senior Member, IEEE, and Muhammad Mahtab Alam[1], Senior Member, IEEE

[1]Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, Estonia
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy
[3]Department of Computer Science and Engineering Qatar University, Doha, Qatar
[4]Faculty of Electrical Engineering and Information Technology, Ruhr University Bochum, Bochum, Germany

Corresponding author: Osama Elgarhy (email: osama.elgarhy@taltech.ee).

This work has received funding partly from the European Union's Horizon 2020 Research and Innovation Program under Grant 951867 (5G-ROUTES project) and Horizon Europe Research Program under grant agreement No. 101058505 and from the Estonian Education and Youth Board OUF11 "AIoT*5G - Artificial ¨ intelligence, edge computing and IoT solutions in distributed systems". In addition, it was conducted in part at ICTFICIAL Oy and received funding in part from the European Union's HE Research and Innovation Program HORIZON-JUSNS-2023 through the 6G-Path Project under Grant 101139172. This material reflects only the authors view and the EC Research Executive Agency and the Commission are not responsible for any use that may be made of the information it contains.
This article has supplementary downloadable material available at https://doi.org/10.1109/OJVT.2024.3496583, provided by the authors.

ABSTRACT In this paper, the goal is to reduce the time needed for the placement and migration of services of Connected Automated Vehicles (CAV) using precise hybrid positioning method. First, to place a service in a Multi-access Edge Computing (MEC) node, there should be sufficient resources in the served MEC node; otherwise, the service would be placed on the neighboring MEC node or even on the core node, resulting in higher delays. We start by modeling our problem with the aid of traffic theory to analytically obtain the necessary number of resources for achieving the desired delay. Second, to reduce the migration process delay, the migration should begin before the vehicle reaches the MEC node. Thus, an AI lane-based scheme is proposed to predict candidate nodes for migration based on precise positioning. Precise positioning data is acquired from a Real-Time Kinematic Global Navigation Satellite System (RTK-GNSS) measurement campaign. The obtained imbalanced raw data is treated and used in the prediction scheme, and the resulting prediction accuracy achieves 99.3%. Finally, we formulate a service placement and migration delay optimization problem and propose an algorithm to solve it. The algorithm shows a latency reduction of approximately 50% compared to the core placement and up to 29% compared to the benchmark prediction algorithm. Moreover, the simulation results for the proposed service placement and migration algorithm show that in case the MEC resource calculations are not used, the delay is 2.2 times greater than when they are used.

INDEX TERMS 5G, Beyond 5G, MEC, RTK, Service Migration, Position prediction, Service placement, and Vehicular Communication.

I. Introduction

DELAY sensitive services are of great importance for applications in verticals, such as Intelligent Transportation Systems (ITS), especially for Connected Automated Vehicles (CAV) [1], [2]. One of the main requirements for CAV use cases is to move such services closer to the vehicle by utilizing Multi-access Edge Computing (MEC) [3], [4]. To avoid excessive delay in a mobile environment, the serving node should be changed (when necessary and if possible) to maintain the expected service quality level and minimize the delay. Therefore, services must be flexibly deployed and seamlessly migrated as close to the vehicle as possible to minimize service delays [5]- [6].

In general terms, *migration* can be described as the process of transferring the state, memory, and storage used by a service from one physical machine to another. There are three types of migration: cold, warm, and hot migration. While warm and hot are known as live, cold is known as non live migration [7]. In this paper, we are interested in delay-sensitive CAV applications and, consequently, we focus on live migration.

As for service placement and migration in the case of cellular networks (e.g., 5G and 6G), specifically for autonomous driving, the typical latency requirements are mostly between 10 and 100 ms [8], [9]. For example, in [9], the Virtual reality and online gaming use cases are associated to a maximum tolerable latency of 20 ms. For these latency-sensitive scenarios, the cellular network should include MEC nodes in addition to the core node to reduce the delay.

In this paper, we consider a cellular network composed of core and MEC nodes. Vehicles within the MEC service area either request services or require service migrations. In the case when a service is migrated, there is a delay resulting from the migration process. Moreover, the delay is strongly affected by the amount of MEC resources, such as CPU and memory. The goal is to minimize the service placement and migration delay by optimizing the amount of MEC resources, minimizing the migration process delay, and optimizing the service placement, that is, choosing the core node or one of the MEC nodes that minimizes the delay.

A. State of the art of service placement and migration

In this section, we review the most relevant research carried out on service placement and migration. As explained, the migration process causes a delay; therefore, position prediction is highly beneficial in reducing it. However, many studies in the literature do not consider position prediction. In the following, we start with papers that do not consider position prediction and then later present papers that use position prediction in their service placement and migration research.

A review of live migration can be found in [10]. The authors start by explaining live migration; then, a review of the literature on service migration in MEC is carried out. The research work in [11] focused on service placement in ITS. The authors focused on minimizing the delay in service placement for network nodes with a specific amount of resources. An optimization problem to minimize the delay was formulated, and then the authors proposed a low-complexity service placement algorithm to solve it. The performance of the proposed algorithm is similar to that achieved by the optimal solution of the optimization problem. The authors of [12] and [13] proposed an algorithm for service placement. The authors formulated an optimization problem to minimize the total delay considering several causes of delay in the 5G cellular network. The authors provided the optimal solution and proposed a heuristic algorithm, which was compared to the optimal integer linear programming solution; the runtime of the heuristic algorithm is in seconds compared to several hours for the case of the optimal solution. However, service migration is not considered; only service placement is considered. In [14], the authors formulated a resource allocation multiconstraint minimization optimization problem by considering i) end-to-end latency and ii) limited resources (e.g., CPU and RAM). Performance was evaluated using computer simulations for several latency requirements. Finally, the authors present the results in terms of number of allocated resources, execution time, number of used edge clouds, and average service per cloud. In [5], the authors studied service migration in fog computing. Given that a moving vehicle should trigger the need for service migration, three schemes are discussed: no migration, migration based on changing cell, or migration based on QoS (based on the delay increase). The schemes were compared in terms of frequency of migration, migration delay, and reliability. Each of the schemes shows some advantages over the others, that is, the first scheme is more suitable when there is sufficient backhaul capacity to handle multiple hops and when reliable communication is needed. The second scheme is more suited for one-hop access and requires a lower latency. The third scheme is a trade-off between the first and second schemes. Therefore, the authors concluded that each of the schemes is suitable for some cases depending on the scenario and the desired performance metrics. The authors of [15] surveyed the resources and resource issues of MEC in the literature. It covers topics such as resource type, request scheduling, migration, and service placement. In a heterogeneous network non-CAV scenario, and for K (single and several) edge nodes, the authors of [16] formulated an optimization problem and then proposed an algorithm to find the suitable location of edge nodes in the network with minimized delay and energy consumption.

Regarding the papers that considered position prediction, the authors in [6] utilized a Genetic Algorithm (GA) and Convolutional Neural Network (CNN) algorithm in their proposed solution to perform service placement and migration. The authors used a CNN to predict the future location of self-driving vehicles for the next instance. Subsequently, using this information, a GA is used to determine the service placement for all vehicles in a region. The authors assessed the performance of the proposed algorithm in the case when both the CNN and GA were used versus when only the CNN was used. The algorithms' performance in terms of rejection rate was 3.9% and 20.2%, respectively; whereby the rejection rate indicates the percentage of services violating the QoS requirements. The authors in [17] performed migration utilizing a prediction method based on Lyapunov optimization and a deep-learning algorithm. The goal is to minimize migration latency and operational cost (consuming bandwidth and energy). A mathematical analysis to determine the theoretical bounds for the latency and operational cost trade-off is presented. The authors showed that their algorithm achieved low operational cost, small queue size, and low latency. For example, their algorithm achieved up to 83% less latency compared to the case when migration was not considered. It is worth noting that, in this work, the core node is not considered as one of the migration options, and the work is focused only on MEC nodes. For MEC container migration, the authors of [18] used a Recurrent Neural Network (RNN) to predict a vehicle's trajectory. Throughout consecutive time slots, the RNN algorithm used the position information of the vehicle (that is, longitude and latitude) for trajectory prediction. The highest accuracy achieved was 97%. However, neither the core nor unbalanced dataset problem was considered. In [19], the authors performed service migration targeting to reduce energy consumption. The authors made position predictions based on the angle of arrival. Simulations were performed to assess the overall performance of the migration and position prediction schemes, and the achieved accuracy was found to be 94%. However, the core node was not presented in the model. In [20] on service migration, the authors formulated an optimization problem to improve energy consumption and reliability. The authors proposed a position prediction algorithm that utilizes three neural networks applied to traffic and driving data for prediction. Furthermore, the authors calculated the dropping probabilities of users. The accuracy of the position-prediction scheme was approximately 87%. However, the core was not included in this analysis. In the context of smart cities, the author of [21] propose an optimization problem that minimizes energy consumption and latency. As part of their algorithm, they utilized Transformers [22] and proposed a Transformer-based Mobility Prediction (TMP) to predict the location of users in the near future. The input of the model is the MEC placement historical data, that is, the number of mobile devices per edge node placement history. In [23] the authors formulated a Mixed-Integer Programming problem to optimize MEC off-loading and radio resource allocation. The authors considered the users' mobility and utilized the mobility context that allows the prediction of the future location of the user based on the speed and direction of the user.

To sum up, the state-of-the-art research does not consider the effect of MEC resource calculation on delays. Moreover, in the literature, either node prediction is not performed, the core node is not included in the analysis, or several cellular network-specific delay components are not included. By node prediction, we mean that the vehicle's position is not predicted to identify the node to which the service will migrate. In this paper, we use a prediction scheme and illustrate its importance. We also include the core node in our analysis for a more realistic scenario.

B. Motivations and contributions

The main goal of introducing MEC nodes is to reduce delay by placing the services closer to the user. The amount of MEC resources is critical for a seamless service experience. In the literature, it is widely assumed that the availability of MEC resources is always granted without identifying the actual amount of required MEC resources. For illustrating this aspect, let us assume that only a few MEC resources are available in a given service area. If this area has many users requesting service placement or migration, then limited services will be placed on the MEC nodes, whereas most of the services will run from the core node. In this case, the average delay experienced will be high. Thus, one major gap in the reported research is the lack of identification of the amount of required MEC resources. Allocating too many resources is a waste, whereas too few resources result in longer delays. Therefore, to fill this gap in the literature, we provide an analytical formulation to optimize the amount of MEC resources and achieve the expected delay value and the quality of service requirements.

Second, to reduce the migration delay, an Artificial Neural Network (ANN)-based vehicle position prediction scheme is proposed to help identify the future serving (i.e., target) MEC node and start the migration process in advance to reduce the delay. The prediction scheme is based solely on information related to the vehicle, that is, its position and speed. Precise positioning information is important for the implementation of prediction schemes. We show that the accuracy of the prediction decreases when precise positioning information is not used, which leads to an increase in delay. Real-Time Kinematic Global Navigation Satellite System (RTK-GNSS) devices are used to obtain precise positioning data for the ANN algorithm.

Third, to optimize the placement of the different services in the MECs and core nodes, we consider various delay components of the cellular systems and formulate an optimization problem that is solved optimally by Integer Linear Programming Solvers (ILPS). Then, we propose an algorithm to perform service placement and migration to avoid the long computation time stemming from the combinatorial nature of the problem. It is worth noting that throughout this paper, we assume, without loss of generality, that each vehicle or user is requesting a unique service that requires placement or migration and more resource usage.

Our novel contributions can be summarized as follows:

- We find the necessary amount of MEC resources to achieve the desired expected delay value and QoS. The analytical solution is obtained by modeling the service placement problem using traffic theory. The proposed model is verified by comparison with computer simulations. After verifying the proposed model, the worst observed error is less than 1%. Moreover, the model is tested for different arrival rates, coverage distances, user speeds, resource requirement distributions, and departure rate cases.
- We propose a vehicle position prediction ANN scheme based on lane information and precise vehicle position. This scheme allows us to predict the future coarse position of the vehicle, thus starting the migration

process in advance. The accuracy of the prediction scheme improves by 15% when precise positioning is used in the one-lane scenario and by up to 20% when lane information and precise positioning are used in the multi-lane scenario.

- We developed an integrated RTK-GNSS testbed positioning terminal device. Integrating the RTK-GNSS into the network has the advantage of securely exchanging correction data through the dedicated protocol, Secure User Plane Location (SUPL), which was not used previously. The development includes the use of LPP to RTCM converter to convert 3GPP's LPP data to RTK-GNSS's RTCM.
- We collect an actual dataset of integrated RTK-GNSS from a measurement campaign and use it to train the ANN algorithm to predict the vehicle position in a one-lane scenario. The imbalanced dataset is treated using different resampling techniques. Finally, the achieved accuracy of the future position prediction is 99.3%.
- We propose a service placement and migration algorithm that uses vehicle position prediction. The proposed algorithm achieves 29% shorter latency than a benchmark prediction scheme.

Each of these contributions is part of a framework for reducing delays. The migration algorithm migrates the service to the node that reduces the delay; the prediction algorithm predicts the candidate nodes in advance to reduce the delay of the migration process itself; the RTK-GNSS accurate positioning provides the necessary data for position prediction; finally, the resource calculation guarantees that the nodes have sufficient resources to host the services; otherwise, all the previous steps will have little to no effect on the delay minimization and optimization, that is, it provides the lower limit on the delay value.

Figure 1 shows the general paper structure while connecting it to the contributions. The proposed framework consists of three components: MEC resource calculation, position prediction, and the service placement and migration algorithm. We begin by introducing and evaluating each component separately, in its own section, and then integrate them to complete the framework. The remainder of this paper is organized as follows. Section II presents a mathematical analysis model for the required MEC resources, and the calculation of expected delays, followed by an evaluation of the proposed model. Position prediction to reduce the migration delay is presented in Section III. Additionally, measurement camping to obtain the necessary real-world dataset is presented, followed by an evaluation of the proposed prediction scheme. In Section IV, the 5G network-related delay components are presented, and an optimization problem followed by an algorithm to minimize the service placement and migration delay based on these components is provided. The proposed framework is then completed by integrating the MEC resource calculation and position prediction parts (which are presented in Sections II, and III,

TABLE 1: Summary of notations.

Notation	Definition
Ct	The average crossing time of the MEC service area.
Ct_c	Average crossing time of users leaving the service area before terminating their services
Ct_t	Average service time of users terminating their services before exiting the service area
D_n^u	The delay that user u experiences being served by MEC node n
D_c^u	The delay user u experiences being served by the core node
d_{th}^u	The delay threshold of user u
ER	The vehicles' entering rate to the service area of the MEC
L	Length of road segment [Km]
M	Maximum amount of available MEC resources
MaR	The maximum amount of resources a user can request
miR	The minimum amount of resources a user can request
P_s	The probability of having s users requesting services within the service area
Pu_t	The probability that a user terminates his/her service before leaving the service area
Pu_c	The probability that a user crosses that service area before the service is terminated
Re	Variable indicating the number of requested resources
R_n	The available amount of resources in MEC node n
SP	The saturation probability
Tr	The traffic within the service area
V	vehicle travel speed [Km/s]
x_n^u	Binary placement variable for the service of user u at MEC node n
x_c^u	Binary placement variable for the service of user u at the core node c

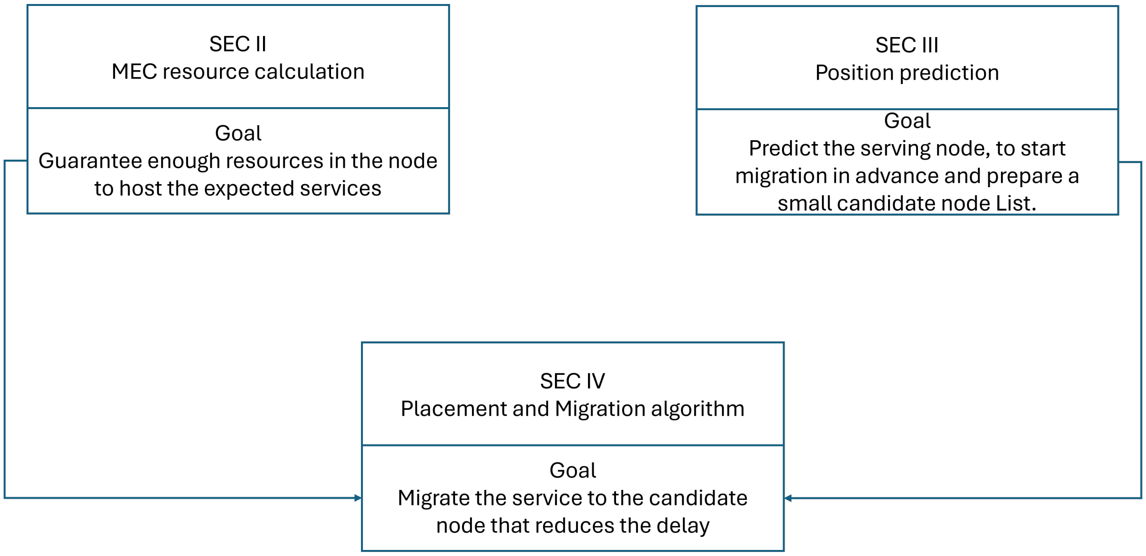

SEC II
MEC resource calculation
Goal
Guarantee enough resources in the node to host the expected services
SEC III
Position prediction
Goal
Predict the serving node, to start migration in advance and prepare a small candidate node List.
SEC IV
Placement and Migration algorithm
Goal
Migrate the service to the candidate node that reduces the delay

FIGURE 1: The general paper structure.

TABLE 2: Summary of Abbreviation.

Abbreviation	
ANN	Artificial Neural Network
CAV	Connected Automated Vehicles
COG	Course Over Ground
COGD	COG Difference
GMLC	Gateway Mobile Location Center
HTTP	Hypertext Transfer Protocol
IDS	Imbalanced Data Set
ILPS	Integer Linear Programming Solvers
ITS	Intelligent Transportation Systems
KPIs	Key Performance Indicators
LB	Lane Based position prediction scheme
LPP	LTE Position Protocol
MEC	Multi-access Edge Computing
MLP	Mobile Location Protocol
NLG	Network Location Gateway
OMA	Open Mobile Alliance
PSPMA	Proposed Service Placement and Migration Algorithm
RTK-GNSS	Real-Time Kinematic Global Navigation Satellite System
SB	Speed Based position prediction scheme
SLP	SUPL Location Platform
SMOTE	Synthetic Minority Over-sampling
SUPL	Secure User Plane Location
SVMSMOTE	Support Vector Machine SMOTE

respectively). Then, the evaluation results of the entire frame with and without the use of the MEC resource calculation (Section II) are presented. Finally, conclusions, limitations, and future work are presented in Section V. The main notations used in this article are listed in Table 1 and the list of abbreviations is reported in Table 2.

II. Analysis of expected delay and MEC resources

It can be assumed that the delay experienced by a user when its service is placed in a MEC node is shorter than when the service is placed in the core node [24], [25]. Therefore, to reduce the delay, a delay-sensitive service should not be placed in the core node unless there are not enough resources (e.g., storage, CPU, or memory) at nearby MEC nodes. Two of the main factors that determine whether a service will be placed in the MEC or the core node are the amount of available resources in the MEC and the number of users requesting services within the service area. Because the resources in the MEC are limited and several users are requesting the service, as long as they are within its service area, it may happen that some of these users will not find enough resources. This behavior can be modeled using traffic theory to calculate the probability that MEC resources become saturated. We define the saturation probability as the probability that all resources are used. Once the MEC resources are saturated, the user service is placed in the core node rather than in the MEC. This section will allow us to calculate i) the amount of MEC resources to serve a certain area, and ii) the expected delay for users within the area.

First, we describe the envisioned scenario. Let us assume that there is a MEC serving a given segment of a road or highway of length $L[Km]$. Once a vehicle travel with a speed of $V[Km/s]$ within the range of the MEC, the service/application instance should be placed in it. Once the

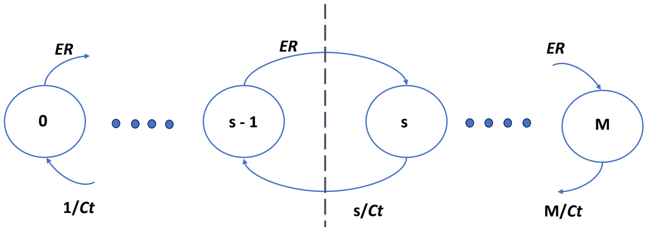


FIGURE 2: The change in the number of served users in the service area can be represented as a transition between *states* in a *process*, where each state represents the number of served users at a given instance. Each circle represents a state of the service area (s means there are s served users in the service area). From a given state s, the number of users increases with a rate of *ER* and decreases with a rate of *s/CT*, where *ER* is the entering rate and *CT* is the average crossing time.

vehicle exits the service area, that is, after traveling a distance L, at time $t = L/V[s]$, the service is no longer needed and should be de-associated from the MEC, and the allocated resources are accordingly released. Because the speed of the vehicles is random, with a given mean, the crossing time is also random. Thus, we use the average crossing time Ct rather than t. We assume that the vehicles requesting the service enter the service area at a rate $ER[s^{-1}]$. Moreover, considering other traffic-based systems, we assume that the entry process follows a Poisson process, similar to [26]. We choose a Markovian blocking system to model our system. The details of choosing this system can be found in Appendix A in the supplementary material/media.

We denote the number of users that request a service in the MEC within its service area as traffic, and use the conventional traffic equation [27], the traffic Tr within the service area can be written as

$$Tr = ER \cdot Ct, \tag{1}$$

We denote the probability of having s users requesting the service within the service area as P_s. While the entering rate ER does not depend on the number of users within the service area, the leaving rate should depend on the number of users within the service area s in addition to the average crossing time Ct. In other words, if the crossing time is short and the number of users is high, the leaving rate is high. In Figure 2, we define the state of the system as an instance in which a specific number of served users exist in the service area. The flow represents the transition probability between the states, that is, the probabilities of moving from one state to the other. Assuming stationary conditions, the flow should be balanced: by making a cut between two states, the flow going into the cut is equal to the flow coming out of the cut. Therefore, we can write a balance equation between the entering and leaving rates as follows:

$$ER \cdot P_{s-1} = \frac{s}{Ct} P_s. \tag{2}$$

Given that the maximum number of available MEC resources M is limited, not all users are served by the MEC. Once the resources are saturated, some users get served by the core node. To calculate the expected delay and required MEC resources, we need to write the saturation probability.

The saturation probability SP is determined from the case wherein the system is full, that is, the number of occupied resources is M [27], in other words $s = M$, then the SP can be calculated as (for interested readers the proof can be found in Appendix B in the supplementary material/media):

$$SP = \frac{(Tr)^M}{M!} \frac{1}{\sum_{w=0}^{M} \frac{(Tr)^w}{w!}}. \tag{3}$$

Equation (3) is known as the Erlang B formula and there are tables, recursive equations, and algorithms for this formula that can be used to calculate the SP given M or M given SP. The saturation probability indicates the percentage of users served by the core node (SP) and the percentage of users served by the MEC ($1 - SP$). Therefore, for a given area, the expected delay is calculated as:

$$\textit{Expected delay} = \textit{Core delay} \times SP + \textit{MEC delay} \times (1 - SP) \tag{4}$$

The use of this result is two-fold:

- The required amount of MEC resources can be calculated according to the desired SP and expected delay from (3) and (4). Thus, by using these equations, we can determine the amount of MEC resources to be deployed in a given area to achieve a certain QoS.
- For any service area and given amount of resources, the expected delay value can be calculated using (4).

It is worth noting that in Section IV, we present the necessary delay components for calculating the *Core delay* and *MEC delay*. Furthermore, it is worth mentioning the following points:

- Since services can consume different types of resources, we use the term ”resource unit” to represent any type of resource. For example, (3) should be used with each type of resource to perform the necessary resource calculations for that specific type of resource, as will be illustrated later.
- Another aspect is the CPU resource; the CPU can be shared among services to accommodate a large number of services; that is, the total requested resources are greater than the capacity of the CPU resources. However, this approach leads to extra delay and possible blocking of services [20]. Therefore, we conjecture that services with stringent delay requirements, such as the CAV studied herein, should have dedicated and scheduled resources. Therefore, in our model, each service is guaranteed CPU resources and is scheduled to ensure deterministic behavior. Note that similar approaches are adopted in real-time operating systems (RTOS) [28].

TABLE 3: Traffic model parameters' values

Traffic model parameters	
$L[Km]$	0.5, 1, 2, 5
$V[Km/h]$	Normal distribution with mean: 60, 80, 100
$ER[s^{-1}]$	1:10
M	350, 750
Number of vehicles crossing the coverage area	110,000
Pu_t	0.1, 0.5, 0.8
MaR	5,10,15
miR	1
NStd (std of normal distribution)	0.5, 1, 2.5, $\frac{MaR-miR}{2}$
NMean (mean of normal distribution)	$\frac{MaR-miR}{2}$

- In (4), *Core delay* and *MEC delay* are the average delay values the service will experience when placed in the core node and the MEC node, respectively. These values include cellular network delay components, such as transmission and processing delays. The delay components details are presented in Section IV. As for the case when there are different services with different delays, the *Core delay* and *MEC delay* will have to be calculated considering the probabilities of the services. For example, the *MEC delay* can be calculated as:

  $$MEC\ delay = P_{ser1}Del_{ser1} + P_{ser2}Del_{ser2} + ...$$

  where P_{ser1} is the probability of service 1 (where the probability of service 1 is the likelihood of having services 1 in the service area) and Del_{ser1} is the delay that service 1 experiences as per the calculations presented in Section IV.

A. Model verification

We made several assumptions to model the system and obtain (3), such as the Poisson arrivals one. To verify our results, we perform computer simulations in which vehicles travel at a random speed, crossing a service area of a MEC node, and requesting a service. We then count the number of requests that are denied/blocked and use it to calculate the saturation probability. Simulations do not use nor depend on the proposed model or (3). The simulation is performed for different values of available resources, and the corresponding saturation probability $SP[\%]$ is obtained. The saturation probability is then calculated from (3) for different values of available resources. The used parameters' values are listed in Table 3. The results of the simulation analysis and the model are shown in Figure 3 to be compared with each other. We can make the following remarks:

- The values on the X-axis represent the number of resource units. We assume that each service requires one unit of resources. Note that in the case of different services with different resource requirements, each service can require more than one resource unit, as we will show later.
- Each line represents a different value of entering rate.
- The saturation probability value indicates the expected percentage of services to be placed in the core node. For example, in the first line from the left, deploying a MEC with 50 resource units in a given area leads to an approximately 20% saturation probability, which means that 20% of the services will be placed in the core node.
- The results of the simulation and the model almost overlap.

The next step is to assess the performance of the model for all parameters that affect the saturation probability $SP[\%]$, and we compare different values of V, L, *ERs* and M. Table 4 shows the values for the difference in $SP[\%]$ between the numerical results and the model. The difference is calculated for random speeds with different average values of V, different lengths L of the service area, and different entrance rates *ERs*. For each of these parameters, $SP[\%]$ is calculated and simulated for a range of maximum available resources, M. Specifically, the values shown in Table 3, that is, four coverage distances, three average velocities of vehicles and ten different entrance rates, over the range M. Then, the mean, Standard Deviation (STD), and median error are calculated over this range of available resources. Moreover, instead of having a table with more than 100 entries, we do not report all the means for all the *ERs*, L and V; rather we take only the worst (i.e., the maximum mean of difference) among the *ERs* and report it in Table 4 for different V and L. Thus, these results show the worst performance difference. By investigating the cases that are reported in Table 4, it is observed that 75% of these cases have an error (i.e., the maximum difference in SP between the model and the simulation) of less than or equal to 0.5%, and the maximum error is approximately 1%. Although the perceived error is small, as can be seen from Table 4, a small extra number of resources can be used as a guard margin.

Next, we verify the model for two additional scenarios. The first is when users' services can be terminated before leaving the service area. We name this test case "Within coverage termination." The second is when there are different services, where each can require a different number of resources; that is, users do not request just one resource unit; rather, they request a random number of resources within a range. We call this test case "Different services requesting different resources." Before starting to analyze the test cases, we note the following about Equation (3); the equation is provided for an average arrival rate, average departure rate, and normalized value of requested resources.

In the first test case ("Within coverage termination"), users stop using or terminate the service before crossing the service area. The point within the service area where the

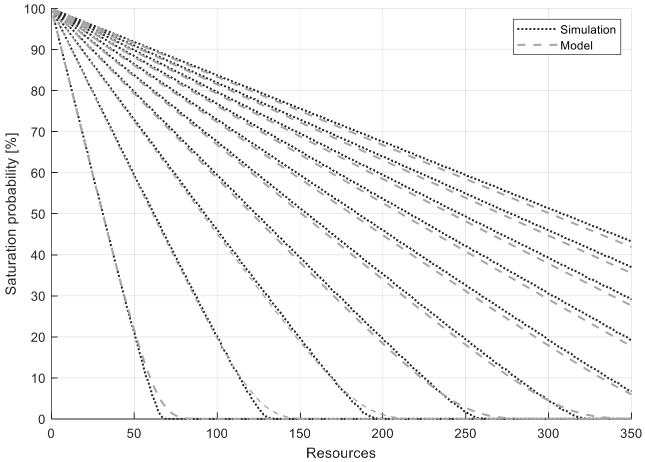


FIGURE 3: Saturation probability for different number of maximum available resources for different values of enterance rate. The entrance rate values range from 1 to 10 vehicles per second. The x-axis represents the number of available MEC resources.

TABLE 4: The difference in $SP[\%]$ between the simulation and the model.

V [Km/h]	L [Km]	Max Mean of difference	Standard Deviation	Max Median of difference
60	0.5	0.3118	0.5577	0
	1	0.8686	0.6847	0.9154
	2	1.0735	0.6022	1.1051
	5	0.8523	0.5578	0.8144
80	0.5	0.1525	0.7144	0
	1	0.1622	0.3879	0.1133
	2	0.5029	0.2713	0.5411
	5	0.3734	0.2593	0.3499
100	0.5	0.1853	0.7225	0
	1	0.1728	0.6780	0
	2	0.1628	0.6484	0.2104
	5	0.1671	0.1322	0.1427

user stops using the service is random, leading to a random service time for these users. We denote the average service time of these users as Ct_t, and the average service time for users who cross the service area before terminating their service as Ct_c. We calculate Ct as:

$$Ct = Ct_t Pu_t + Ct_c Pu_c, \tag{5}$$

where Pu_t is the probability that a user terminates his/her service before leaving the service area and Pu_c is the probability that a user crosses that service area before the service is terminated, $Pu_c = 1 - Pu_t$. Equation (5) is used in (1) and (3) to calculate the traffic and saturation probabilities, respectively. To verify whether the traffic model is suitable for this case, we repeat the same model verification steps used to obtain the results in Table 4. The results are presented in Table 5. Note that the error is very small, similarly to Table 4. Therefore, the model can be used to predict well the MEC resources in this scenario. This test is repeated for different values of Pu_t, as shown in Table 3. Regarding Ct_t, the test is performed for three scenarios: i) service termination occurs at the start of the service area, ii) service termination occurs in the middle of the service area, and iii) service termination occurs at the end of the service area. The results are similar to those reported in Table 5. The worst performance is found to be 1.1%, which confirms that the model can be used to predict MEC resources in this scenario. Note that in case of two or more average arrival rates (e.g., when there is traffic originating within the service area in addition to vehicular traffic), the global arrival rate can be calculated in a similar way to Ct in (5) given the probability of each arrival rate.

TABLE 5: The difference in $SP[\%]$ between the simulation and the model - Case: Within coverage termination.

	60 [Km/h]	80 [Km/h]	100 [Km/h]
L [Km] = 0.5	0.30	0.15	0.18
L [Km] = 1	0.84	0.15	0.17
L [Km] = 2	1.08	0.49	0.16
L [Km] = 5	0.87	0.37	0.17

In the second test case ("Different services requesting different resources"), the users' resource requests follow a random distribution. Two distributions are studied, uniform and normal. First, we study the uniform distribution case, in which the average number of requested resources per user belongs to a uniform distribution. Assume that the minimum number of resources a user can request is miR, and the maximum number is MaR , while the variable Re indicates the possible values of resources. A uniform distribution is defined as follows:

$$f(Re) = \begin{cases} \frac{1}{MaR - miR}, & \text{if } miR \leq Re \leq \ MaR, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

The average value is $\frac{MaR+miR}{2}$.

Because all the resource requests have the same weight in the uniform distribution, the wider the range of resources (that is, $MaR - miR$), the less relevant the average value in the resource calculations. In this case, the selection of MEC resources depends on the design goals. If Mobile Network Operators (MNO) aim for high QoS, they should use the maximum resources, that is, MaR resources. For a more conservative approach, they can select an average value.

The second distribution is normally distributed. A truncated version is used, i.e., between MaR and miR. The standard deviation is denoted as NStd. The Performance is affected by the variance and range.

The same model verification steps used to obtain the results in Table 4 are repeated for the two distributions, the uniform and normal. The results are presented in Table 6. It can be seen from the table that Uniform distribution has the worst performance for the reasons mentioned before.

TABLE 6: The difference in $SP[\%]$ between the simulation and the model - Case: Different services requesting different resources.

V [Km/h]	L [Km]	Uniform distribution	Normal distribution
60	0.5	2.65	0.39
	1	3.91	0.36
	2	4.83	0.50
	5	5.34	0.72
80	0.5	2.81	0.68
	1	3.87	0.74
	2	4.90	0.80
	5	5.64	0.95
100	0.5	2.61	0.88
	1	3.72	1.00
	2	4.80	1.0
	5	5.56	1.06

TABLE 7: The difference in $SP[\%]$ between the simulation and the model. Normal distribution - different values of *MaR* and standard deviations.

Case	SP [%] difference
NStd = 0.5, *MaR* = 10	0.9
NStd = 1, *MaR* = 10	1.47
NStd = 2.5, *MaR* = 10	5.5
NStd = $\frac{MaR-miR}{2}$, *MaR* = 10	7.35
NStd = 0.5, *MaR* = 15	1.1

Moreover, when *MaR* is increased to ten, the maximum error (i.e., the maximum difference in SP between the model and the simulation) becomes 7.9%. On the other hand, the Normal distribution results in error comparable to the original *same service* case (i.e., Table 4). To investigate the Normal distribution case further, the model verification steps are repeated for different values of *MaR* and STD, and the worst cases are reported in Table 7. It is clear that as the STD increases, the error increases. In the case of a high STD value, the Normal distribution approaches the Uniform distribution; therefore, their performance are similar. In the case of a low STD value, the Normal distribution approaches the single/*same service* case and the error turns out to be small.

B. Performance evaluation of the MEC resources analysis

This section shows the importance of MEC resource calculations. We use the simulation parameters described in section A, and we set ER to 3 vehicles per second. Note that changing ER does not affect the fundamental observations obtained from the results. We simulate a range of available MEC resources and show the results of three strategies: i) MEC, ii) Core, and iii) Random. In the MEC strategy, services are placed in the MEC nodes as long as there are sufficient resources. In the Core strategy, the services are placed in the Core nodes, and in the Random strategy, the services are placed randomly in the core nodes or MEC nodes, provided that the MEC has sufficient resources.

The performance of the three strategies is shown in Figure 4. The investigation is conducted for two cases: the first is when the amount of available MEC resources is small, and the second is when the amount of MEC resources is high. Both cases are presented in Figure 4, where the upper zoomed-in part presents the results for a small number of resource cases. We begin with a small number of resource cases. Because the goal of deploying MEC nodes is to reduce delay, the MEC strategy is expected to achieve the lowest delay. However, from the upper zoomed part of Figure 4 (i.e., when a few number of resources are used), we can see that the MEC strategy and the Random strategy achieve the same performance. Moreover, the gain compared with the Core strategy is negligible. These results imply that simply deploying MEC nodes is not sufficient to reduce the delay, and the necessary amount of MEC resources to serve the traffic needs to be calculated.

A greater range of available resources is shown in Figure 4. We note the following: i) once the amount of MEC resources increases, the gain of the MEC strategy over the Core one becomes more evident; ii) for a large portion of the curves, the Random and MEC strategies achieve the same performance, and within this portion, the MEC strategy does not achieve its full potential; iii) for the MEC strategy, at a certain point, the average delay becomes constant, and the amount of MEC resources at this point can be calculated from (3). This observation shows the importance of finding the optimal amount of MEC resources using (3), since using any amount of resources larger than this optimal amount will lead to waste of physical resources and high cost for deployment.

Regarding the behaviour of the Random and MEC strategies, the following observations are made:

1) In the MEC strategy, the services are placed in the core node once there are insufficient resources in the MEC. Similarly, also in the Random strategy, the services are placed in the core node once there are insufficient resources in the MEC.
   - Thus, in the first part of Figure 4, that is, the decreasing linear part, the results/behaviors of the two strategies are identical.
2) The Random strategy is realized using a uniform random variable that determines whether the service will be placed in the MEC or core node. Asymptotically, the probability of placing the service in the MEC node is equal to the probability of placing the service in the core node, which is equal to 50%.
   - Thus, in the second part of Figure 4, that is, the almost constant horizontal segment, as there

are enough MEC resources, the almost constant average delay of the Random strategy can be calculated as

$$\textit{Average delay} \approx \frac{1}{2}\textit{Average Core delay} + \frac{1}{2}\textit{Average MEC delay} \tag{7}$$

where the *Average Core delay* is the average delay the user's service will experience if it is placed in the Core node, and the *Average MEC delay* is the average delay if it is placed in the MEC node. Without loss of generality, in this specific simulation, their values are 28 ms and 11 ms, respectively; hence, the *almost* constant value for the Random strategy is ≈ 19.5 ms.

3) From point 2), it can be also understood that half of the traffic is placed in the MEC, and the other half in the core node.

- The optimal amount of resources for the MEC strategy at a given ER can be calculated using (3) to accommodate all the traffic. However, because the traffic is halved for the Random strategy, approximately half of this optimal amount is required, and any additional resources are simply wasted.

From this section and the above results, we conclude that if (3) is not used, either the delay is high or the resource waste is high. Moreover, we illustrated that MNOs can use a practical equation similar to those used in networks, call centers, and servers to efficiently calculate the required amount of MEC resources to optimize resources and reduce delays.

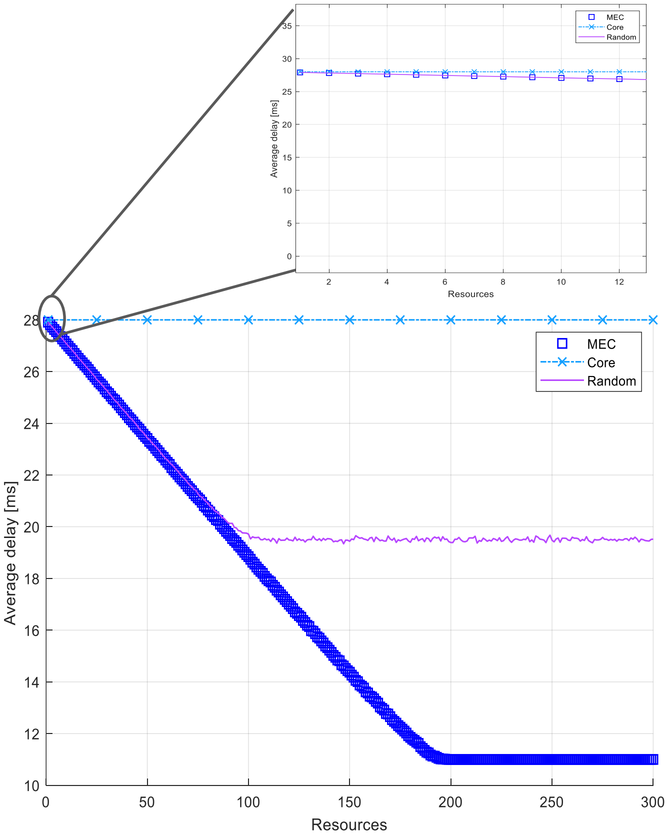


FIGURE 4: Average delay for a range of available amount of MEC resources. The x-axis represents the number of available MEC resources.

III. Proposed vehicle position prediction based on precise positioning

The migration process itself contributes to the delay. As mentioned previously, one way to reduce the migration process delay is to start the migration in advance; thus, the migration destination should be predicted [29], [30]. Furthermore, one way of preparing and shortlisting candidate nodes is to predict the future vehicle position. In this section, we introduce a position prediction method based on the position of the vehicle within the road, that is, the lane or the position within the lane. Therefore, accurate positioning is required to achieve this prediction. Therefore, RTK-GNSS will be used as an enabler of this prediction method. The RTK-GNSS is known to provide high-accuracy centimeter-level positioning.

A. Prediction of the vehicle future position

As explained previously (Section I), predicting the vehicle position plays an important role in delay reduction. However, there is no need to predict the position of the vehicle with high accuracy. It is sufficient to know which MEC nodes will be close to the vehicle in the near future. Depending on the scenario and road layout, this can sometimes be decided in an easy manner. For example, in Figure 5, we can predict the future nodes from the heading/course of the vehicle.

There are also more complex scenarios in which we need to know whether the vehicle will turn or continue straight ahead. For example, in Figure 6, assuming that services for both the yellow (top car) and red (bottom car) cars are placed in MEC 1, we need to know if the services should be migrated to MEC 2 or to MEC 3. Assuming that the top car did not change its speed or lane, it will continue ahead with a high probability, and its service should be migrated to MEC 2. However, because the bottom car is in the right lane, given that the car is moving at a low speed or decelerating, we can assume with a degree of confidence that the bottom car will take the U-turn and will be served by MEC 3.

From these scenarios, we can conclude that the future serving MEC node and location of the vehicle can be predicted in a practical way. The main idea is to use basic information, such as lane, speed, and acceleration. Furthermore, maps and traffic information can be used.

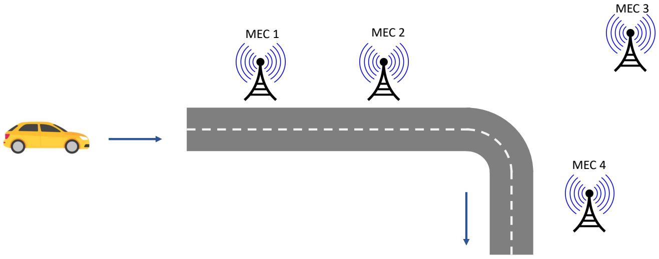


FIGURE 5: Road scenario 1 - the vehicle's trajectory is known.

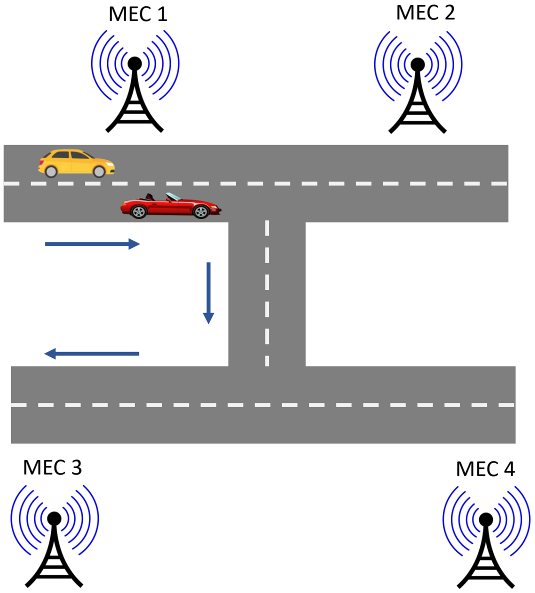


FIGURE 6: Road scenario 2 - the vehicle will either turn or continue straight.

TABLE 8: Simulation parameters.

ANN	
Number of epochs	100
K-fold number of splits	10
Activation functions	rectifier, sigmoid
Optimizer	adam
Number of layers	3
Input layer neurons	5, 10, 25
Number of synthetic dataset instances	1500
Number of real dataset instances	1455
Loss function (cost function)	binary cross entropy
Service placement and migration parameters [31] [11]	
TTI/sub-carrier spacing	1 [ms] / 15 [KHz]
Transmission time	8-83 [ms]

TABLE 9: Mean, Median, and standard deviation of the accuracy for the turning and intersection scenarios.

	Mean	Median	STD
Turning scenario	91.7 %	92 %	2.28 %
Intersection scenario	75.32 %	75.5 %	2.59 %

B. Performance evaluation of the proposed position prediction scheme

In this section we present the results for the proposed lane-based positioning prediction scheme in which an ANN is used. We begin by showing the simulation results for the multi-lane scenario, followed by the results for the one-lane scenario. Finally, we present the results obtained using a dataset from trials using RTK-GNSS. The simulation parameters are presented in Table 8. For interested readers, the synthetic and the real datasets and a realization of the ANN are uploaded to data repository[1].

An ANN is developed using Python for the prediction scheme. Three features are used as inputs for the ANN: lane information, speed, and acceleration. K-fold Cross-Validation (CV) [32] is used for validation and to reduce bias (overfitting). For the dataset, we first present the results of a synthetic dataset that contains random parts to exhibit real-life behavior. We assess the performance of the prediction scheme by obtaining its accuracy in two scenarios: i) a turning scenario and ii) an intersection scenario. In the turning scenario, the goal is to predict whether a vehicle will turn or continue traveling forward, as shown in Figure 6. In the second scenario, there is a three-lane cross-leg intersection, and the prediction scheme must predict whether the vehicle will turn right, left, or move forward. The results for both the scenarios are presented in Table 9. The accuracy for the first scenario is 91.7 % and it decreases to 75.32 % for the intersection scenario owing to higher uncertainty.

It is worth noting that the accuracy of the scheme depends on the different actions the vehicle can take. For a straight road, as shown in Figure 5, the vehicle can only move forward; thus, there is no uncertainty and the accuracy is the highest. For the turning scenario, as shown in Figure 6, the vehicle can take two actions: moving forward or turning. Thus, a degree of uncertainty is added, and the accuracy of the scheme decreases. Finally, for the intersection scenario, the vehicle can take more actions, that is, a right turn, a left turn, or move forward. Thus, the accuracy is the lowest compared with the other scenarios.

To demonstrate the importance of lane information, we introduce another version of the position prediction scheme. In this version, we drop lane information and use only speed and acceleration information. We refer to this version as the Speed Based (SB) scheme and the one with lane information as the Lane Based (LB) scheme. We then compare the

[1] https://doi.org/10.48726/zbapt-3hz53

TABLE 10: Mean, Median, and standard deviation of the accuracy for one-lane intersection scenario.

	Mean	Median	STD
Expected RTK synthetic dataset	66.7 %	66.25 %	3.3 %
Speed only (SB)	51.45 %	52 %	3.45 %

performances of the two schemes. First, we compare the two schemes on a high-speed road, for example, a highway. The difference in the achieved accuracy between the two schemes is approximately 3%. Second, we compare the two schemes on a low-speed road, for example, a school zone. The difference in the achieved accuracy between the two schemes is approximately 20%, in favor of LB. It can be concluded that the accuracy of the two schemes is affected by the speed limit and range.

One lane intersection: If the road does not have multiple lanes but only one lane, this becomes a limitation to the position prediction scheme. To overcome this limitation, we use RTK-GNSS. The centimeter-level accuracy of RTK-GNSS allows the mimicking of multiple lanes within a single lane. The conjecture is that the driver will drive their vehicle in the lane according to the action they are planning to take; for example, a driver making a right turn will move the vehicle slightly to the right. We then compare this version of the scheme with the SB scheme in the intersection scenario. The results are presented in Table 10. It can be seen that there is an expected decrease in accuracy compared to the multiple-lane scenario, as presented in Table 9. Nonetheless, using the RTK results in a gain of approximately 15% compared to not using the RTK, i.e., using only speed information. It is worth noting that the results are obtained using a synthetic dataset mimicking RTK performance. However, in the next subsection, a real dataset from the RTK measurements is used.

C. RTK-GNSS measurements campaign

In this section, we assess the performance of the proposed prediction scheme based on a real dataset acquired from an RTK-GNSS measurement campaign. The measurement campaign is conducted for the one-lane scenario. The RTK device is mounted on a car moving within the Tallinn University of Technology (TALTECH) campus and the neighboring streets for several laps. For interested readers, the collected real dataset are uploaded to data repository[2]. The position, longitude and latitude, speed over ground, and Course Over Ground (COG) are stored to create the dataset. COG represents the direction (heading) of travel. The COG Difference (COGD) is also calculated. COGD represents the difference in the angle between the current COG and the previous COG. A basic if-else algorithm can successfully detect that a vehicle is turning once the COGD reaches approximately 10°. We denote this Moment Of Detection, that is, the ten degrees COGD turning moment, as MOD. The basic if-else algorithm cannot make a decision based on smaller COGDs, such as 5°or less, because they might result from a driving maneuver. For the field trials, a rover and base station are used. The base station and its antenna are placed on a roof for maximum exposure to the GNSS signal. The rover is placed inside the vehicle and connected to a laptop to store measurements. The antenna of the rover is placed on top of the vehicle connected to the rover. Several devices have been tested and used for the measurement campaign, an example of a GNSS-RTK device used as the rover and base station is shown in Figure 7a. The used antenna is a Ublox Multi-band Active GNSS Antenna. The GNSS-RTK device integrated with the cellular network consists of three main components: Raspberry Pi 4 (RPi4), 5G HAT (includes Quectel 5G Standalone (SA) modem) and Septentrio MosaicHAT (includes Mosaic-X5 RTK receiver) with GNSS-RTK receiver [3]. The three RPi4, 5G HAT, and MosaicHAT can be considered as a single smart device with 5G network support and centimeter-accuracy GNSS receiver. In Figure 7b, the received correction data are shown, and in Figure 7c, the dashboard is presented, where it can be seen that the accuracy is in the order of centimeters.

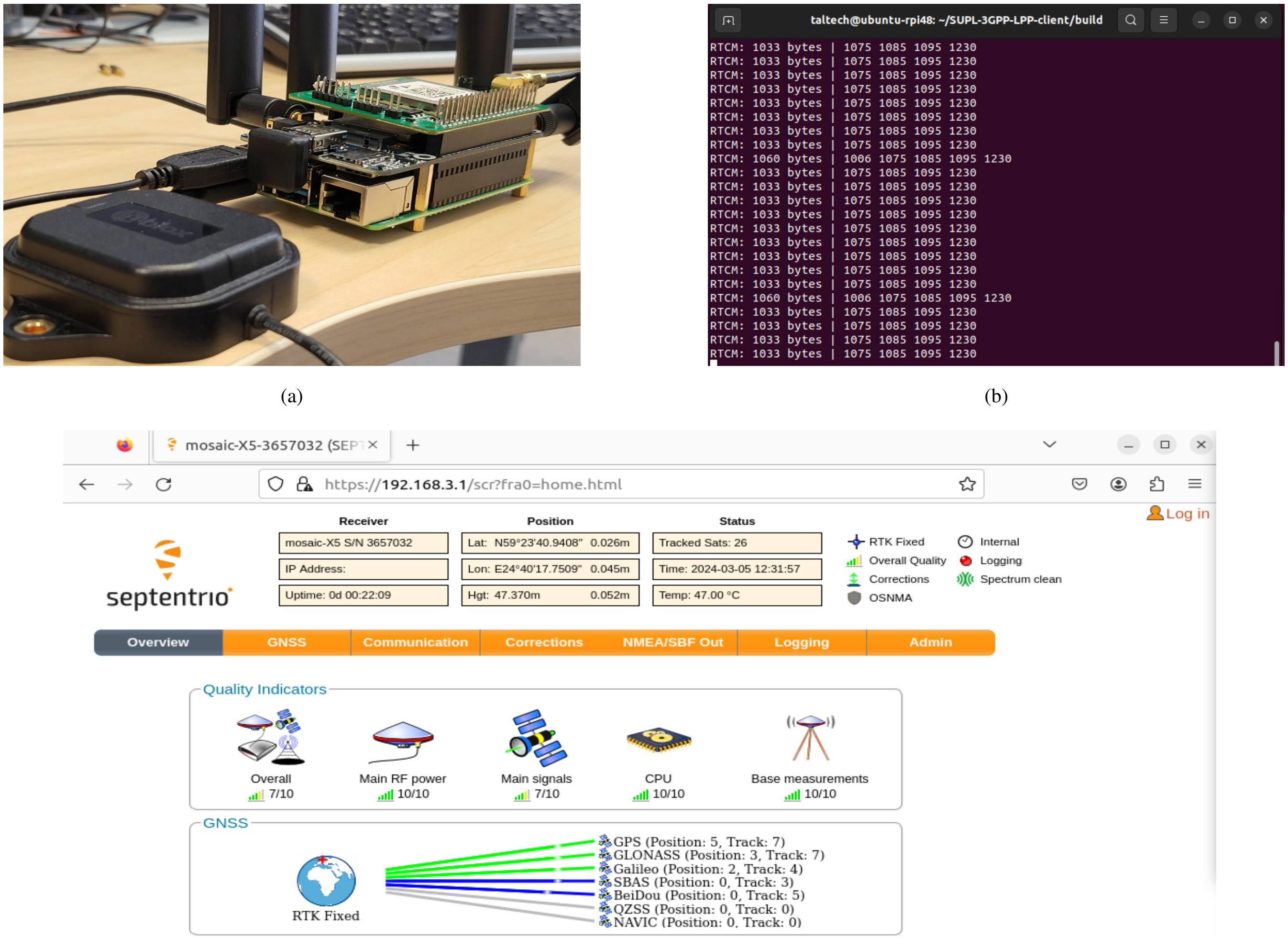
1) Results of the neural network

The measurements and the corresponding dataset are used to train the same ANN used in Section B (Table 8). Because the basic if-else algorithm can predict turning at MOD, we calculate the accuracy of the ANN at MOD. The input features of the ANN are the COG, COGD, and stored COGs for the preceding four seconds. Finally, the dataset is used to train the ANN, and the prediction accuracy of the ANN at the MOD is found to be 95.94%. It is important to note that the if-else algorithm cannot perform the prediction before MOD. Thus, the real strength of the ANN is performing the prediction seconds before reaching the MOD. Hence, the accuracy of the ANN is calculated when the prediction is performed 1 second, 2 seconds, and 3 seconds before the MOD. We name these prediction moments as $MOD-1$, $MOD-2$, and $MOD-3$, respectively. However, we found that the accuracy does not change with the changing of the prediction moment, that is, the ANN achieved the same accuracy of 95.94% for all of them.

To understand these results, we must describe the data set. The dataset is a matrix whereby each row represents an input features instance, and for each instance, there is a corresponding output. The output indicates whether the vehicle turns or not. Vehicle turning is represented by one and not turning is represented by zero. Upon inspection, it is observed that the output of the dataset is mostly zero. This is because most of the time the vehicle is not turning, but it is moving along the road. Note that this is the case

[2] https://doi.org/10.48726/zbapt-3hz53

[3] Other setup/devices such as ArdusimpleRTK2B V3 equipment with a Ublox ZED-F9P RTK-GNSS module, were tested/used as well


(a)

(b)

```
taltech@ubuntu-rpi48: ~/SUPL-3GPP-LPP-client/build
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1060 bytes | 1006 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1060 bytes | 1006 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
RTCM: 1033 bytes | 1075 1085 1095 1230
```

mosaic-X5-3657032 (SEPT

https://192.168.3.1/scr?fra0=home.html

Receiver	Position	Status
mosaic-X5 S/N 3657032	Lat: N59°23'40.9408" 0.026m	Tracked Sats: 26
IP Address:	Lon: E24°40'17.7509" 0.045m	Time: 2024-03-05 12:31:57
Uptime: 0d 00:22:09	Hgt: 47.370m 0.052m	Temp: 47.00 °C

Quality Indicators

Overall	Main RF power	Main signals	CPU	Base measurements
7/10	10/10	7/10	10/10	10/10

GNSS

RTK Fixed

GPS (Position: 5, Track: 7)
GLONASS (Position: 3, Track: 7)
Galileo (Position: 2, Track: 4)
SBAS (Position: 0, Track: 3)
BeiDou (Position: 0, Track: 5)
QZSS (Position: 0, Track: 0)
NAVIC (Position: 0, Track: 0)

(c)

FIGURE 7: Integrated GNSS-RTK measurement campaign (a) GNSS-RTK device. (b) Incoming correction data. (c) The dashboard shows the location, the accuracy, the connected satellites and quality indicators.

for any of the CAV environments, for example, highways and urban. Furthermore, because the output of the dataset is mostly zero, the ANN merely predicts zero, which means that the true positive is zero, and the sensitivity is zero. The true positive is the number of correctly predicted ones, and sensitivity is equal to the true positive divided by the actual number of ones. Therefore, this dataset can be described as imbalanced.

2) Imbalanced dataset and treatment thereof

Some classification problems comprise a majority class and a minority class, that is, one class is much more prevalent than the other. In this case, the output of the corresponding dataset is rarely the minority class. This type of dataset is known as an Imbalanced Data Set (IDS). The problem with an IDS is that the classifier output is only the majority class. The solutions for IDS are mainly based on resampling the dataset [33]–[35]. Resampling means upsampling the minority class, that is, increasing the number of data points, or downsampling the majority class. One of the most common resampling techniques is the Synthetic Minority Oversampling Technique (SMOTE), which uses the K-nearest neighbor to create synthetic instances within the minority class.

In the following, we present the results of the ANN after resampling. First, upsampling is used on the dataset, and then the ANN is trained in the same manner as in Section III.C.1. The achieved accuracy, after Cross Validation (CV), is 66.52%. To improve the performance, we include additional features such as longitude, latitude, and speed. The ANN achieves an accuracy of 98.82%. Furthermore, at $MOD-1$, $MOD-2$, and $MOD-3$, the accuracies were 96.38%, 93.55%, and 92.69%, respectively.

The predictions performed at $MOD-sec$ can play an important role in the migration process, where sec is

TABLE 11: Achieved accuracy using different resampling techniques to counter IDS. The best accuracy is achieved by the Upsampler and is highlighted in the table.

Resampling technique	Upsampling (P) or undersampling (D)	Accuracy
Upsampler	p	98.82%
SMOTE	p	92.94%
SVMSMOTE	P	98.71%
BorderlineSMOTE	P	98.64%
ADASYN	P	98.18%
NeighbourhoodCleaningRule	D	96.44%
TomekLinks	D	96.41%
CondensedNearestNeighbour	D	75.16%
NearMiss	D	80.53%

TABLE 12: Parameters used for optimization using Grid Search.

Parameter	Values
Batch size	5, 25, 30
Number of epochs	100, 200, 500, 800, 1000, 1500, 2000
Optimizer	Adam, rmsprop, Adamax, SGD

TABLE 13: Results from MATLAB's classification learner. The best result is achieved by the Ensemble classifier and is highlighted.

Classification family	Best classifier	Accuracy of best classifier
Tree	Coarse tree	98.5%
Naive Bayes (NB)	Kernel NB	95.2%
SVM	Linear SVM	95.9%
KNN	Weighted KNN	97.0%
Ensemble	RUSBoosted Trees	98.8%
Neural Network (NN)	Bilayered NN	96.2%
Kernel	SVM Kernel	97.9%

a generic number of seconds before MOD, for example $MOD-3$. Because there is a possibility that not all MECs will have sufficient resources for migration once the vehicle reaches them, the early predictions at $MOD-sec$ can be used to reserve the resources for the service at the predicted MEC node without actually starting the migration. Moreover, if a service requires a long migration time, depending on how important it is, migration can be initiated as early as possible. A flow diagram of this operation is shown in Figure 8. Depending on the service, a suitable sec is chosen to perform the prediction, that is, at $MOD-sec$, and reserve resources. Moreover, if the service has a high priority or a long migration time, it is considered a special service. For special services, rather than just reserving resources, live migration can start because of the prediction at $MOD-sec$.

Next, we attempt other resampling methods to handle the IDS. The imblearn package of MIT [36] is used and the results are presented in Table 11. First, it is observed that the upsampling methods achieve better performance than the downsampling methods. For the upsampling methods, while SMOTE achieves an accuracy of 92.94% and the accuracy of Support Vector Machine SMOTE (SVMSMOTE) is 98.71%, the best accuracy is 98.82% and is achieved by the upsampler, which simply duplicates the minority class data points.

Finally, we determine the best parameters for the ANN using the grid search function. Different numbers of epochs, patch sizes, and optimizers are used to tune the ANN to achieve the best accuracy. ANN tuning is performed on the data modified by two resamplers: the upsampler and SVMSMOTE. After using the Grid search optimization parameters given in Table 12, it is found that the best accuracy is 99.3%, which is achieved by using the upsampler, batch size $= 25$, number of epochs $= 2000$, and *Adam* optimize. Note that in this case, the F1 score, which can be considered an average of the sensitivity and the precision, is found to be also 99.3%. Moreover, ANN is tuned for $MOD-1$, $MOD-2$, and $MOD-3$. The prediction accuracies, after tuning, are 97.67%, 96.3%, and 95.7% for $MOD-1$, $MOD-2$, and $MOD-3$, respectively.

To further assess the performance of our ANN, we use MATLAB's "Classification Learner app". We select all the classifiers offered by MATLAB. The most important results are listed in Table 13. The ensemble classifier achieves the best performance. The ensemble classifier uses several classifiers to achieve better performance than any one of them. The ensemble classifier shown in the table is based on a family of classifiers called "Trees" classifiers. Moreover, it uses the Random undersampling boosting (RUSBoost) method to handle the IDS. In Table 13, we show only the results of the best classifier for each classification family.

3) The hybrid 5G integrated RTK-GNSS positioning architecture

For the sake of completeness of this RTK-GNSS part, in this section, we propose how to integrate it into the 5G network.

To achieve the high positioning accuracy for 5G, hybrid methods are one of the main enablers [38]. Hybrid methods include combination of 3GPP and non-3GPP positioning technologies, such as RTK-GNSS. As a prospect, the hybrid RTK-GNSS should be integrated into the 5G network. Thus, it can securely provide the necessary precise positioning information over the 5G network. In this section, we propose a hybrid 5G integrated RTK-GNSS positioning architecture.

Figure 9 shows the transmission of RTK-GNSS assistance data via the 3GPP unicast, where the main data channel is the User Plane (U-Plane). U-Plane protocols carry the location messaging over the data connection to the user equipment (NG-UE). The main component here is the Network Location Gateway (NLG), which includes the Gateway

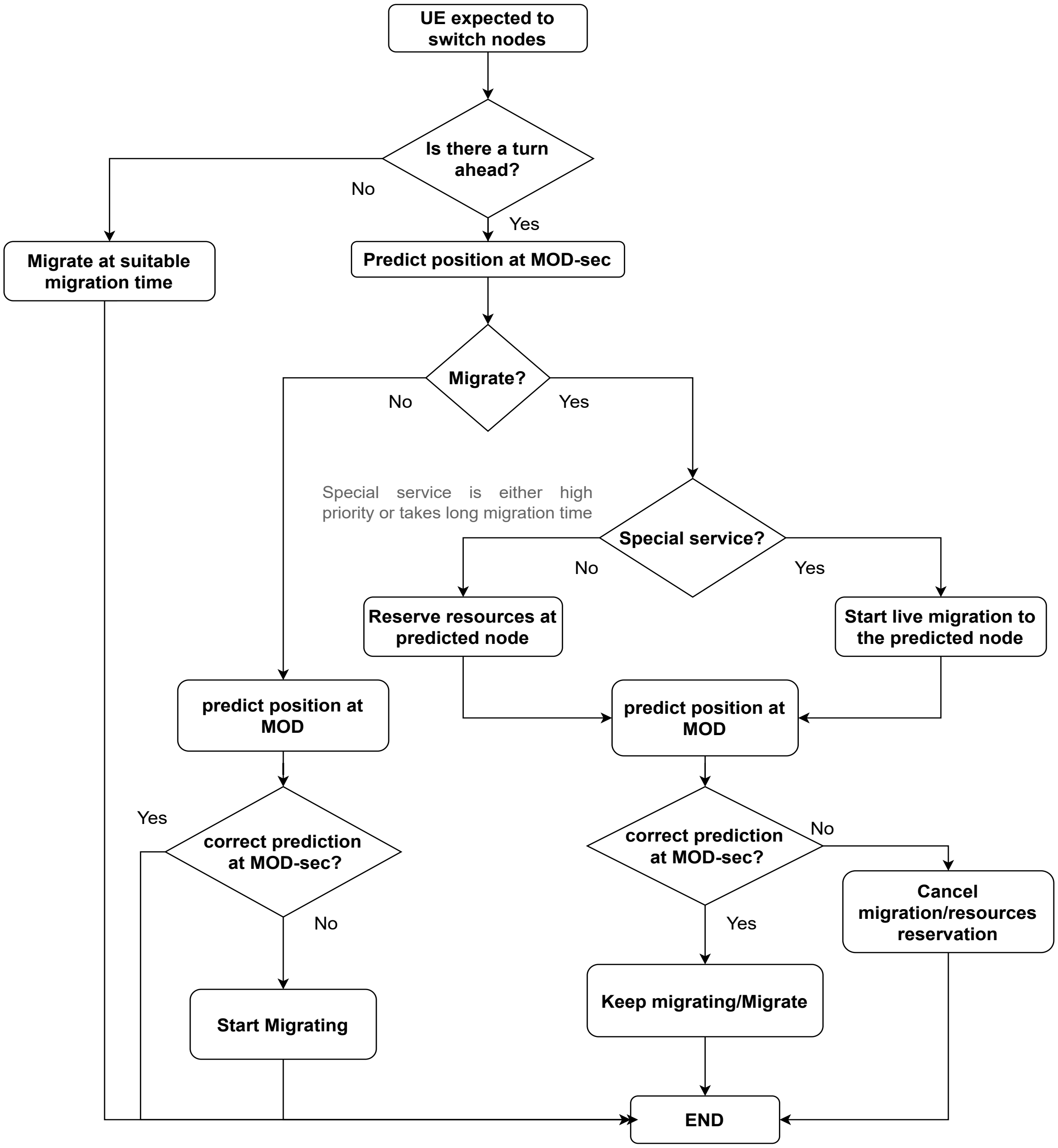

UE expected to switch nodes
Is there a turn ahead?
No
Yes
Migrate at suitable migration time
Predict position at MOD-sec
Migrate?
No
Yes
Special service is either high priority or takes long migration time
Special service?
No
Yes
Reserve resources at predicted node
Start live migration to the predicted node
predict position at MOD
predict position at MOD
Yes
correct prediction at MOD-sec?
No
correct prediction at MOD-sec?
No
Yes
Cancel migration/resources reservation
Start Migrating
Keep migrating/Migrate
END

FIGURE 8: Flow diagram of utilizing $MOD - sec$. The prediction can be performed seconds before MOD. The accuracy of the prediction algorithm is lower; however, the prediction algorithm can be utilized with high-priority services to reduce the migration delay.

Mobile Location Center (GMLC) node. The GMLC offers a standardized and secured interface (the Le interface) for external location applications for example, Mobile Location Protocol (MLP), this is denoted as Le:MLP, as seen in Figure 9. MLP messages are typically transported with Hypertext Transfer Protocol (HTTP) and SUPL Location Platform (SLP) functionality, where SUPL stands for Secure User Plane Location. MLP can be used to initiate the positioning request of users in the 5G network. The Open Mobile Alliance (OMA) developed a SUPL standard to enable U-Plane positioning in a secure way. The SUPL standard can be used in 5G networks, with the ability to provide IP connections. The location protocols are exchanged between the SUPL client (SET) in the NG-UE and the SLP in the network (referred to as SUPL Server). The RTK-GNSS server sends GNSS assisted data to the SLP using the RTCM v3 protocol over the NTRIP connection. SLP then rebroadcasts assisted data via LTE Position Protocol (LPP) protocol over a U-Plane using a SUPL 2.0 connection. LPP

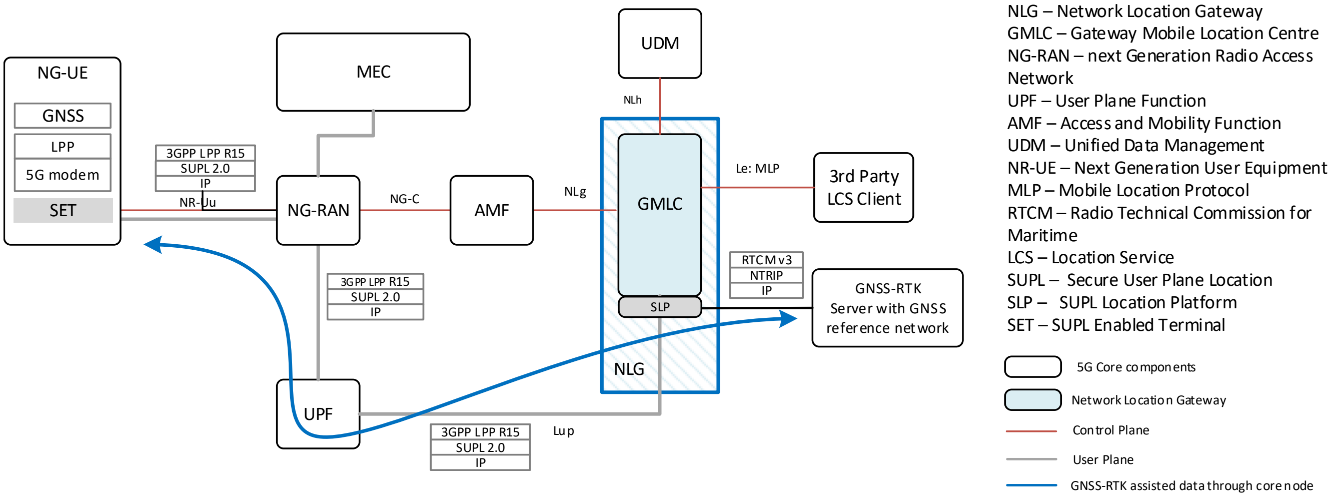

NG-UE
GNSS
LPP
5G modem
SET
3GPP LPP R15
SUPL 2.0
IP
NR-Uu
MEC
NG-RAN
NG-C
AMF
NLg
UDM
NLh
GMLC
SLP
NLG
Le: MLP
3rd Party LCS Client
RTCM v3
NTRIP
IP
GNSS-RTK Server with GNSS reference network
3GPP LPP R15
SUPL 2.0
IP
UPF
3GPP LPP R15
SUPL 2.0
IP
Lup
NLG – Network Location Gateway
GMLC – Gateway Mobile Location Centre
NG-RAN – next Generation Radio Access Network
UPF – User Plane Function
AMF – Access and Mobility Function
UDM – Unified Data Management
NR-UE – Next Generation User Equipment
MLP – Mobile Location Protocol
RTCM – Radio Technical Commission for Maritime
LCS – Location Service
SUPL – Secure User Plane Location
SLP – SUPL Location Platform
SET – SUPL Enabled Terminal
5G Core components
Network Location Gateway
Control Plane
User Plane
GNSS-RTK assisted data through core node

FIGURE 9: 5G integrated RTK-GNSS hybrid positioning system architecture. The architecture is partially developed from [37].

is a point-to-point protocol that allows multiple connections to different devices.

Note that the architecture in Figure 9 is not intended to present how the services (i.e., the services to be placed or migrated) are placed in the core or MEC nodes. This architecture is intended to show how RTK-GNSS is integrated into a 5G network. In the figure, the UDM, NLG, AMF, and UPF are the core network elements.

D. Position prediction benchmark

The benchmark is the last piece of information needed in the position prediction section. Therefore, in this section, we introduce the benchmark position prediction method that will be used in the performance evaluation in Section IV.D. This benchmark, used in [19], is named Markov Chain (MC) method [39] and is based on the Markov progress of the user's movement. The main idea is that the next serving node can be predicted based on the past visited cells/MEC nodes. For example, knowing that a user has visited specific k cells/nodes, the next node that the user will visit is predictable. The probability that user u will visit node n at time instant i is $P(n_i|n_{i-1}, n_{i-2}...n_{i-k})$. In [39], $k = 2$ was chosen, whereas in [19] different values of k were tested, and $k = 2$ was found to be the best. Therefore, we use with $k = 2$. Given a dataset of the history of the nodes visited by users in a given area, the transition probability can be calculated as:

$$P(n_i|n_{i-1}, n_{i-2}) = \frac{N_n(n_{i-1}, n_{i-2})}{TN(n_{i-1}, n_{i-2})} \quad (8)$$

where N_n is the number of visits to node n given an interval of time in the past. TN is the total number of visits to all feasible nodes given the same interval of time. The MC can be considered a road-specific average estimation, i.e., how often, on average, vehicles take a specific turn on a specific road.

IV. Service placement and migration delay minimization - Optimization problem formulation and solution

In this section, we first introduce a general service placement and migration optimization problem, and then include position prediction and resource calculation. We start by listing the different delay components in the MEC and core nodes. Based on these components, an optimization problem is formulated to determine the optimal service placement that minimizes delay. We then explain how position prediction and resource calculation affect this problem. A service placement and migration algorithm that utilizes the position prediction from Section III is proposed. Finally, we show i) the simulation results for the algorithm, ii) the importance of the resource calculations part from Section II, and iii) the results that include the resource calculation.

A. Delay components

As reported in [12], there are three main components of delay:

1) Radio delay, which includes

- Transmission delay (Transmission time): this depends on the communication, that is, the packet size or data, to be transmitted as part of the service, and on the Transmission Time Interval (TTI). TTI depends on Sub-Carrier Spacing (SCS). The 15KHz, 30 kHz, 60 kHz, and 120 kHz SCS correspond to 1 ms, 0.5 ms, 0.25 ms, and 0.125 ms, respectively.
- Propagation delay: signal propagation in the channel.

- Processing delay: channel estimation, encoding and decoding time.
- Queuing delay: the total time until all packets are transmitted (it depends on the number of available resource blocks and number of users to be scheduled).
- Hybrid Automatic Repeat Request (HARQ).
- Round-Trip Time (RTT).

2) Backhaul delay: Transmission time over the Xn interface (backhaul between different gNBs) or NG (backhaul between gNB and the core node or an aggregation point) propagation time over the Xn or NG interfaces.
3) Service computational delay: The computational time of the service, which depends on the computational capabilities of the node.

The main reason for the difference between the core node and MEC delays is the backhaul delay because of the distance between the users and the core node. It is worth noting that backhaul delay is often added to the processing delay of the core node. With advances in CPU design, the computational capabilities in the core and MEC node should not cause a significant delay difference.

B. Delay optimization problem

Let us consider a scenario in which there are multiple users, each requesting a unique service/application, and multiple MEC nodes. We define D_n^u as the delay that user u experiences when served by MEC node n, and D_c^u as the delay that user u experiences when served by the the core node. We define a binary placement variable x_n^u, which is equal to one if the service of user u is placed at node n, and zero otherwise, and x_c^u, which is equal to one if the service of user u is placed at the core node c, and zero otherwise.

This problem can be viewed as an assignment problem in which users are assigned to nodes. We can illustrate this by presenting the cost assignment matrix. The goal is to minimize the total cost, which is the delay in our problem. It is worth noting that the rows represent the users, and the columns represent the nodes:

$$\begin{array}{cc} & \begin{array}{cccc} 1^{st}\text{ node} & 2^{nd}\text{ node} & . & \text{core node} \end{array} \\ \begin{array}{c} 1^{st}\text{ user} \\ 2^{nd}\text{ user} \\ . \\ \text{last user} \end{array} & \begin{bmatrix} x_1^1D_1^1 & x_2^1D_2^1 & . & x_c^1D_c^1 \\ x_1^2D_1^2 & x_2^2D_2^2 & . & x_c^2D_c^2 \\ . & . & . & . \\ x_1^lD_1^l & x_2^lD_2^l & . & x_c^lD_c^l \end{bmatrix} \end{array}$$

The objective function is to minimize the sum delay experienced by all users, that is,

$$\min f_d\left(x_n^u, x_c^u\right) = \sum_{n\in N}\sum_{u\in U} x_n^u D_n^u + \sum_{u\in U} x_c^u D_c^u \tag{9}$$

Subject to

$$\sum_{n\in N} x_n^u + x_c^u = 1, \forall u \in U, \tag{10}$$

$$\sum_{u\in U} x_n^u Q^u \leq R_n, \forall n \in N, \tag{11}$$

$$\sum_{n\in N} x_n^u D_n^u + x_c^u D_c^u \leq d_{th}^u, \forall u \in U. \tag{12}$$

The constraint (10) guarantees that the service of a user can be placed at only one node at a time. Constraint (11) indicates that the required resources Q^u for all services placed in node n must be less than the available resource R_n of that node. The analogue $\sum_{u\in U} x_c^u Q^u \leq R_c$ is not required, assuming that the core node can accommodate any number of requests. Finally, the delay constraint (12) guarantees that the delay experienced by the user is less than the delay threshold d_{th}^u of the user. Note that, because there is a unique service for each user, we can state that the service of the user is placed in a node or just the user is placed in a node.

To solve the optimization problem, we must first identify certain characteristics. The objective function and constraints of the problem are linear and the variables are binary. Thus, this is a binary integer programming problem, which can be solved by computer solvers, for example, MATLAB or CPLEX, or by methods such as branch and bound [40]. Finally, the optimization problem is combinatorial because of its binary nature, and is characterized by a long computational time in the case of a large number of users [11], [12]. For interested readers, the method for mapping the optimization problem to MATLAB's ILPS function *intlinprog* can be found in Appendix C in the supplementary material/media.

Given that the service can request different types of resources, for example, CPU, memory, and storage, the optimization problem should be modified, by simply replicating the resource constraint, (11), for each type of resources. Otherwise, (11) can be rewritten as $\sum_{u\in U} x_n^u Q^{u,rt} \leq R_n^{rt}, \forall n \in N, \forall rt \in RT$, where rt is the resource type index that belongs to the set of available resource types RT. In case the position prediction from Section III is used, the optimization problem should be reduced to only one core node and one MEC node, in case the service area is served by one MEC. Moreover, for each user, it reduces to the problem of choosing between two nodes given their delay. If the resource calculation step from Section II is used, and the saturation probability, SP, was chosen to be equal to zero, that is, the MEC node has sufficient resources to host all the services, then the resource constraint, (11), shall be removed. Next, we propose a service placement and migration algorithm that utilizes the position prediction to prepare a list of candidate nodes.

C. Proposed service placement and migration algorithm

In this section, we propose an algorithm to perform service placement and migration by utilizing position prediction. Figure 10 shows a flow chart of the service placement algorithm. Once a user enters the network and requests a

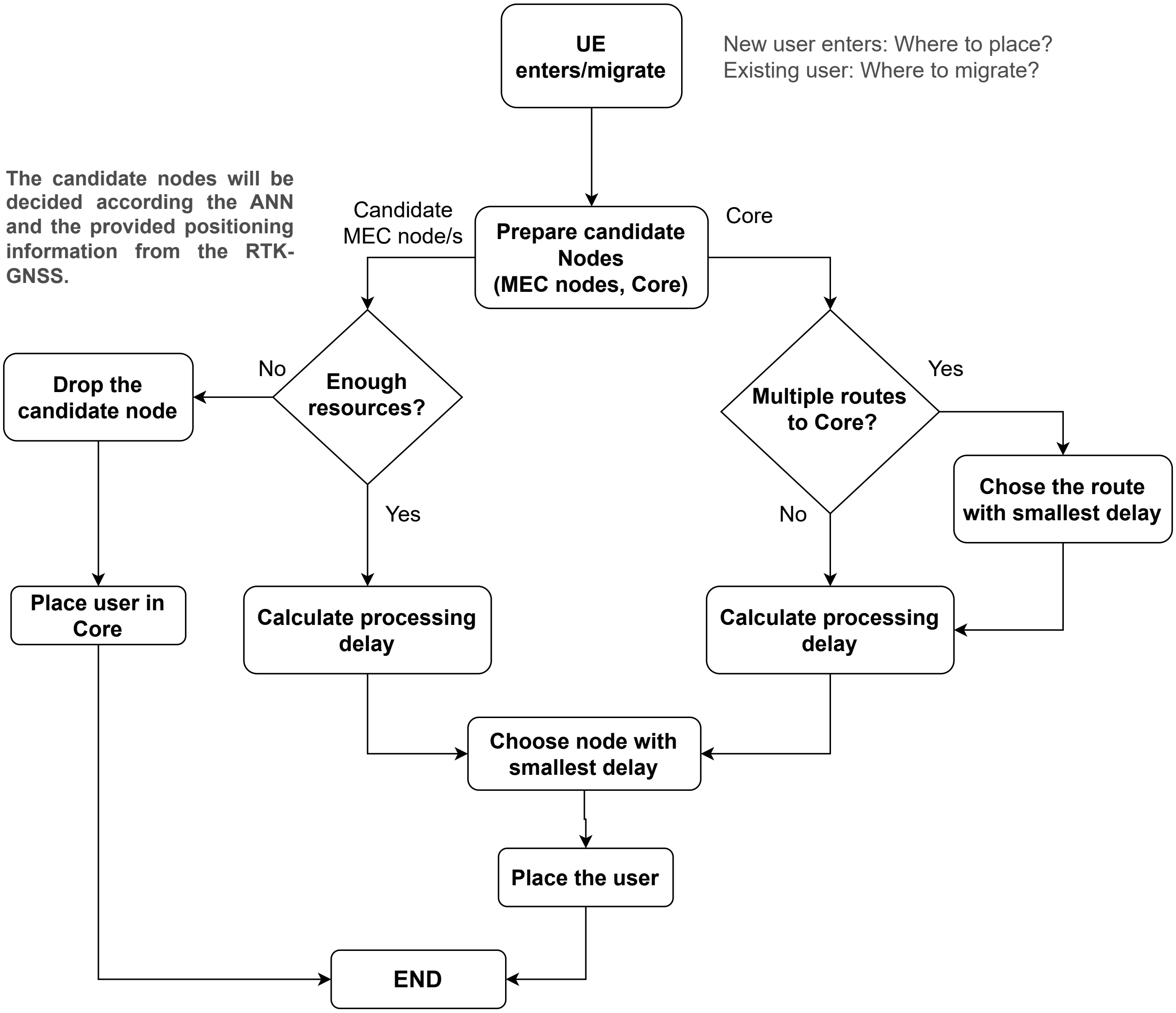

UE enters/migrate
New user enters: Where to place?
Existing user: Where to migrate?
The candidate nodes will be decided according the ANN and the provided positioning information from the RTK-GNSS.
Candidate MEC node/s
Prepare candidate Nodes (MEC nodes, Core)
Core
No
Enough resources?
Drop the candidate node
Yes
Multiple routes to Core?
Yes
Chose the route with smallest delay
No
Place user in Core
Calculate processing delay
Calculate processing delay
Choose node with smallest delay
Place the user
END

FIGURE 10: Flow chart of the proposed service placement and migration scheme. The migration algorithm utilizes the position prediction scheme to prepare the candidate MEC node/s list.

service, that is, upon the initial placement for a service or migration, we need to place the service in the most suitable node. Candidate nodes are identified for the placement. The propagation delays in the routes to the core node are calculated, and the route with the shortest delay is selected. Finally, the processing time for the core node is computed. For the candidate MEC nodes, the available resources are checked, and if there are insufficient resources to host the user's service, then the core node hosts the service. Otherwise, the processing delay of the MEC is added and the overall delay is calculated. Finally, the delay between the MEC and core nodes is compared, and the service is placed in that node accordingly. The candidate node list is obtained by predicting the future vehicle position, as described in Section III. The complexity analysis of the algorithm, and other real-world considerations and challenges, can be found in Appendix D in the supplementary material/media.

It is worth noting that to compare the delay experienced by a user at different nodes, we should consider only the delay terms relevant to the comparison. For example, the radio delay term is the same in the case of the user-MEC route or user-MEC-Core route; therefore, the propagation term is not important for the comparison. Hence, we limit the comparison between the core node and MEC to the processing delay, including backhaul delay. The backhaul propagation delay depends on the medium, either wired, such as fiber, or wireless, for example, microwave or mmWave.

Another aspect is the route to the core node. The user can reach the core node through different access points in the cellular network. In this case, only one route is selected, which is the shortest route. For multiple available MECs, there are two criteria: i) the MEC must have available resources, and ii) the closest MEC, that is, the MEC with the minimum propagation delay, should be chosen.

Although the MEC node should introduce a lower latency than the Core node, the algorithm accounts for any cases in which that might not be true. If the position prediction is not available, the algorithm can still be used as an alternative to the optimization problem to avoid long computational time for a large number of users. However, the algorithm must calculate the delay of all surrounding MEC nodes because there is no predicted MEC node list.

D. Performance evaluation of the proposed service placement algorithm

This Section presents the simulation results for the Proposed Service Placement and Migration Algorithm (PSPMA). The algorithm using the proposed prediction scheme LB is compared to when the benchmark MC is used, as described in Section III.B. Then, we show the impact of having the optimal amount of resources, through (3) in Section II, on the algorithm.

1) Proposed algorithm without the MEC resource calculation

First, we evaluate the performance of PSPMA when the proposed prediction schemes are used (without the use of the MEC resource calculations - Section II). Three algorithms are compared. The first is Core placement, which means that all the services will be placed in the core node. This algorithm is denoted as ”C algorithm” in the figures. The second is the benchmark algorithm, that is, the MC algorithm. The last one is the LB algorithm, which is based on the LB prediction. The aim of this section is: to show the importance of the proposed position prediction (Section III) by comparing it to a benchmark, to show the effect of adding MEC nodes to the network, and to to demonstrate the need to apply the MEC resource calculation Section II. All simulations are performed for 1000 iterations. Finally, the average delay is calculated. An incorrect prediction results in more than one migration, which causes more delay, that is, a penalty in the total delay.

In addition to the simulation parameters of Table 8, some assumptions and parameters are applied exclusively to this analysis (Figures 11 and 12) and are not used in the previous or later analyses and results.

- We assume that a dedicated MEC is used, such as in [41].
- We assume that all MEC nodes have the same computational capabilities and serve the same service area, that is, they are grouped.

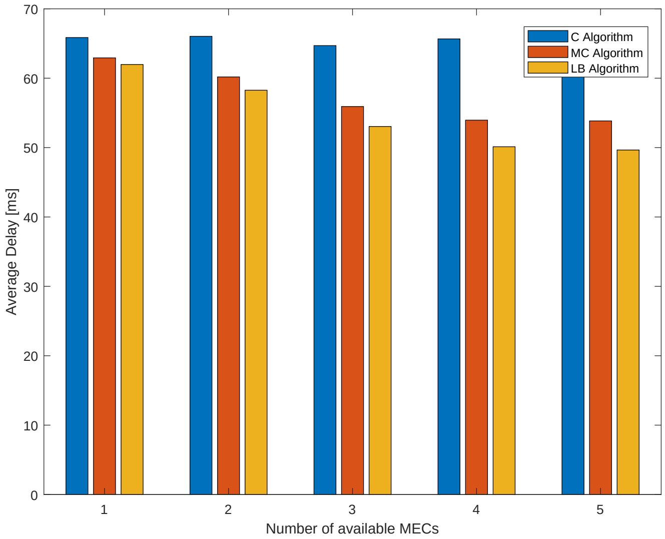


FIGURE 11: Average delay for different number of MECs.

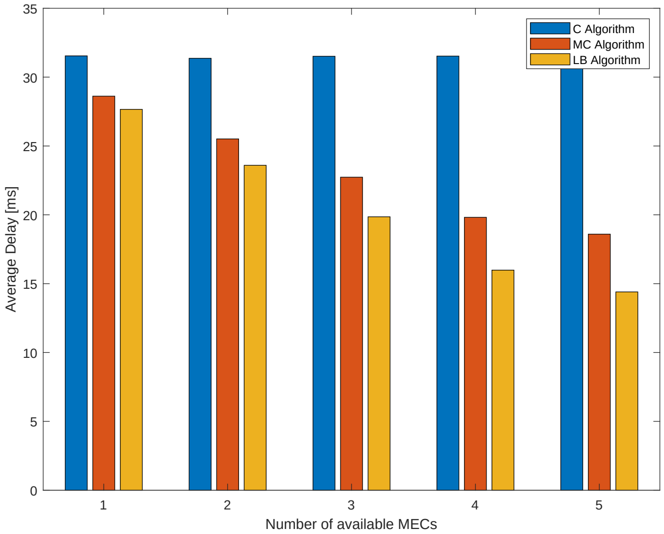


FIGURE 12: Average delay for different number of MECs, with a reduced average transmission time.

- Number of users [5-35], number of MEC nodes [1-5], and the number of CPU cores per MEC is 8.
- We did not perform the MEC resource calculation (Section II). Hence, we do not know the required amount of resources.

In Figures 11 and 12, the effect of adding more MECs can be observed. As the number of available MECs increases, the average delay decreases. As for the use of the MC and LB and their impact on performance, the MC based placement algorithm results in more errors than the LB algorithm. More errors lead to migration to the incorrect node. For example, considering Figure 6, if the algorithm incorrectly

TABLE 14: Effect of resource calculation using (3) on the average delay of the PSPMA for different ER values. The reported average delay values are in milliseconds.

	$ER = 1$	$ER = 2$	$ER = 3$	$ER = 4$	$ER = 5$
PSPMA with (3)	11.0948	11.0955	11.0948	11.0956	11.0951
PSPMA without (3)	12.2666	19.8237	22.5953	23.9554	24.7695
Random without (3)	19.6892	20.3420	22.6789	23.9759	24.7864

starts migrating the service to MEC 2 rather than MEC 3, then the service should be remigrated to MEC 3. This extra migration increases the service migration delay, that is, it incurs a penalty owing to the wrong decision. In addition to the delay penalty, live migration is costly and consumes additional network resources. In Figure 11, we can observe that even though LB outperforms MC, there is no significant difference between them. A key factor is the transmission time, which is the largest delay component in this simulation and overshadows the other delay components, including the incorrect migration penalty. Once the transmission time range is reduced, the difference between the two algorithms becomes more apparent, as presented in Figure 12. The MC's latency can be up to 29% greater than that of the LB. Nonetheless, these milliseconds of difference can be critical in safety and automated mobility applications.

Some important information is obtained by observing the case when one MEC is used, as presented in Figures 11 and 12. Even though the MEC is installed, the gain compared with the core placement, that is, the ”C algorithm”, is not that high. However, as we increase the number of MEC nodes, that is, the number of available resources, the gain increases. This highlights the strong need for the MEC resource calculations (Section II).

2) Proposed algorithm with the MEC resource calculation

In this part, we use the MEC resource calculation (Section II) (traffic analysis), and demonstrate its positive impact on the performance of the PSPMA. As shown in Section II, not calculating the correct number of resources affects the delay significantly. In this part, we assess the PSPMA when the MEC resource calculation, from Section II and equation (3), is used. The PSPMA with (3) is compared to the PSPMA without (3) and a Random placement strategy.

We use the simulation parameters provided in Table 3 and Table 8, the results are reported in Table 14. It can be seen from the results in Table 14 that as the ER increases, the performance of PSPMA with (3) is stable because the resources are adapted to traffic. However, the performance of PSPMA without (3) deteriorates until the difference between it and the Random strategy becomes negligible. These results further demonstrate that optimizing the service placement without optimizing the number of resources is insufficient, as anticipated in the contributions of this work with respect to the approaches presented in the literature.

Finally, we note that even when the cellular network has a very limited number of MEC nodes (even one node) to cover hot or important areas, the results of this work are still valid. In terms of resource calculations, it is essential to perform MEC node resource calculations because of the importance of the areas that need to be served by the MEC node/s. For the service placement and prediction algorithm, it is still necessary to decide whether the vehicle will be served by this MEC node, so reducing the service migration downtime.

V. Conclusions and future work

In this paper, we investigated service placement and migration delay and proposed different delay minimization mechanisms. First, we modeled the problem using traffic theory, which allowed us to determine the necessary amount of MEC resources and to calculate the expected delay for a given service area. We verified our model and found that the worst error is mostly less than 1%.

Then, an ANN lane-based future vehicle coarse positioning algorithm was presented, demonstrating the advantage of adding the lane in which the vehicle is traveling as one of the main features for predicting vehicle mobility. Moreover, we demonstrated that using such a feature improves the accuracy. The ANN uses precise positioning information from RTK-GNSS. Using computer simulations, we showed that the accuracy of the prediction algorithm decreased by approximately 15% if precise positioning information is not used. Real RTK-GNSS positioning measurements were obtained and used to train the ANN algorithm in a one-lane scenario. The data acquired were imbalanced. Therefore, we treated it using resampling techniques. The ANN was optimized to improve the results and was validated using cross-validation, achieving an accuracy up to 99.3%. Furthermore, we compared our ANN with the classifiers offered in MATLAB. In conclusion, the various results demonstrated the advantages of using hybrid RTK-GNSS positioning.

Then we formulated an optimization problem considering the different delay components for the MEC and core node, and we demonstrated that the problem could be solved optimally using the ILPS. Then, an algorithm for service placement and migration utilizing position prediction was presented. The algorithm uses the RTK-GNSS positioning, and reduces latency by 29% over the benchmark. Finally, we demonstrated the importance of performing the MEC resources calculations. Future work will be conducted to address the current limitations, in particular, investigating the efficiency of the scheme in a dense urban environment with additional features (e.g., lane changing), investigating the migration delay and migration penalty, and including optimization goals in addition to delay minimization, such as cost (because multiple migrations incur high costs).

ACKNOWLEDGMENT

This work has received funding partly from the European Union's Horizon 2020 Research and Innovation Program under Grant 951867 (5G-ROUTES project) and Horizon Europe Research Program under grant agreement No. 101058505 and from the Estonian Education and Youth Board OUF11 "AIoT*5G - Artificial ¨ intelligence, edge computing and IoT solutions in distributed systems". In addition, it was conducted in part at ICTFICIAL Oy and received funding in part from the European Union's HE Research and Innovation Program HORIZON-JUSNS-2023 through the 6G-Path Project under Grant 101139172. This material reflects only the authors view and the EC Research Executive Agency and the Commission are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] 5GPPP, "5G trials for cooperative, connected and automated mobility along european 5G cross-border corridors - challenges and opportunities," *5GPPP white paper, 1.0., 5GPPP project collaboration*, vol. 1, 2020.

[2] J. He, Z. Tang, X. Fu, S. Leng, F. Wu, K. Huang, J. Huang, J. Zhang, Y. Zhang, A. Radford, L. Li, and Z. Xiong, "Cooperative connected autonomous vehicles (cav): Research, applications and challenges," in *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, 2019, pp. 1–6.

[3] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 369–382, 2016.

[4] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, pp. 12–19, 2013.

[5] J. Li, X. Shen, L. Chen, D. P. Van, J. Ou, L. Wosinska, and J. Chen, "Service migration in fog computing enabled cellular networks to support real-time vehicular communications," *IEEE Access*, vol. 7, pp. 13 704–13 714, 2019.

[6] A. Dalgkitsis, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Data driven service orchestration for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[7] K. Alizadeh Noghani, "Service migration in virtualized data centers," Ph.D. dissertation, Karlstads universitet, 2020.

[8] 3GPP, "TS 22.186 Service requirements for enhanced V2X scenarios (Release 17)," April 2022.

[9] 5GAA, "C-V2X use cases and service level requirements volume II," *5GAA Automotive Association Technical Report, Ver. 2.0.*, 2023.

[10] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, pp. 23 511–23 528, 2018.

[11] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, "Edge-enabled v2x service placement for intelligent transportation systems," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1380–1392, 2020.

[12] T. Subramanya, D. Harutyunyan, and R. Riggio, "Machine learning-driven service function chain placement and scaling in mec-enabled 5g networks," *Computer Networks*, vol. 166, p. 106980, 2020.

[13] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency-aware service function chain placement in 5g mobile networks," in *2019 IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 2019, pp. 133–141.

[14] B. E. Mada, M. Bagaa, T. Taleb, and H. Flinck, "Latency-aware service placement and live migrations in 5g and beyond mobile systems," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[15] A. Sarah, G. Nencioni, and M. M. I. Khan, "Resource allocation in multi-access edge computing for 5g-and-beyond networks," *Computer Networks*, vol. 227, p. 109720, 2023.

[16] B. Li, P. Hou, H. Wu, R. Qian, and H. Ding, "Placement of edge server based on task overhead in mobile edge computing environment," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 9, p. e4196, 2021.

[17] H. Ma, Z. Zhou, and X. Chen, "Leveraging the power of prediction: Predictive service placement for latency-sensitive mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6454–6468, 2020.

[18] W. Zhang, J. Luo, L. Chen, and J. Liu, "A trajectory prediction-based and dependency-aware container migration for mobile edge computing," *IEEE Transactions on Services Computing*, 2023.

[19] I. Labriji, F. Meneghello, D. Cecchinato, S. Sesia, E. Perraud, E. C. Strinati, and M. Rossi, "Mobility aware and dynamic migration of mec services for the internet of vehicles," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 570–584, 2021.

[20] C. Wang, J. Peng, L. Cai, H. Peng, W. Liu, X. Gu, and Z. Huang, "Ai-enabled spatial-temporal mobility awareness service migration for connected vehicles," *IEEE Transactions on Mobile Computing*, 2023.

[21] H. Huang, W. Zhan, G. Min, Z. Duan, and K. Peng, "Mobility-aware computation offloading with load balancing in smart city networks using mec federation," *IEEE Transactions on Mobile Computing*, 2024.

[22] O. Stenhammar, G. Fodor, and C. Fischione, "A comparison of neural networks for wireless channel prediction," *IEEE Wireless Communications*, pp. 1–7, 2024.

[23] L. Huang and Q. Yu, "Mobility-aware and energy-efficient offloading for mobile edge computing in cellular networks," *Ad Hoc Networks*, vol. 158, p. 103472, 2024.

[24] K. Jiang, H. Zhou, X. Chen, and H. Zhang, "Mobile edge computing for ultra-reliable and low-latency communications," *IEEE Communications Standards Magazine*, vol. 5, no. 2, pp. 68–75, 2021.

[25] S. Kekki, W. Featherstone, Y. Fang, P. Kuure, A. Li, A. Ranjan, D. Purkayastha, F. Jiangping, D. Frydman, G. Verin *et al.*, "MEC in 5G networks," *ETSI white paper*, vol. 28, no. 2018, pp. 1–28, 2018.

[26] D. C. Gazis, *Traffic theory*. Springer Science & Business Media, 2006, vol. 50.

[27] V. B. Iversen, "Teletraffic engineering and network planning," *Technical University of Denmark*, p. 270, 2010.

[28] Q. Li and C. Yao, "Real-time concepts for embedded systems," *Computing Reviews*, vol. 45, no. 5, p. 268, 2004.

[29] I. Farris, T. Taleb, A. Iera, and H. Flinck, "Lightweight service replication for ultra-short latency applications in mobile edge networks," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[30] I. Farris, T. Taleb, H. Flinck, and A. Iera, "Providing ultra-short latency to user-centric 5g applications at the mobile network edge," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 4, p. e3169, 2018.

[31] K. Lee, J. Kim, Y. Park, H. Wang, and D. Hong, "Latency of cellular-based v2x: Perspectives on tti-proportional latency and tti-independent latency," *IEEE Access*, vol. 5, pp. 15 800–15 809, 2017.

[32] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[33] N. N. Chiplunkar and T. Fukao, *Advances in Artificial Intelligence and Data Engineering: Select Proceedings of AIDE 2019*. Springer Nature, 2020, vol. 1133.

[34] P. Bruce, A. Bruce, and P. Gedeck, *Practical statistics for data scientists: 50+ essential concepts using R and Python*. O'Reilly Media, 2020.

[35] K.-K. R. Choo and A. Dehghantanha, *Handbook of Big Data Analytics and Forensics*. Springer, 2021.

[36] "Imbalanced-learn (imported as imblearn), MIT-licensed library," https://imbalanced-learn.org/, accessed: 2024-04-11.

[37] 3GPP, "TS 138.305 Functional specification of User Equipment (UE) positioning in NG-RAN (Release 16)," October 2021.

[38] ——, "TS 22.261 Service requirements for the 5G system (Release 17)," October 2022.

[39] A. Hadachi, O. Batrashev, A. Lind, G. Singer, and E. Vainikko, "Cell phone subscribers mobility prediction using enhanced markov chain algorithm," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 1049–1054.

[40] J. W. Chinneck *et al.*, "Practical optimization: a gentle introduction," *Systems and Computer Engineering, Carleton University, Ottawa.*, vol. 11, 2006, http://www.optimization101.org.

[41] "Local Packet Gateway (LPG)," https://www.ericsson.com/en/portfolio/cloud-software-and-services/cloud-core/packet-core/cloud-packet-core/local-packet-gateway, accessed: 2023-11-25.

Osama Elgarhy received the B.Sc of electrical and communication engineering in 2009. From 2009 until 2013 he worked as a teaching assistant in Pyramids higher institute of engineering, Cairo, Egypt. He received the M.Sc. and Ph.D. degrees in Telecommunication engineering from Politecnico Di Milano, Milan, Italy, in 2016 and 2020, respectively. In 2016 he spent a research period in Azcom Technology, Milan, Italy. He is currently a Post-Doctoral Researcher with the Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, Tallinn, Estonia. His current research interests include resource allocation for cellular systems, NB-IoT, and device to device communications.

Yannick Le Moullec received the M.Sc. degree in electrical engineering from Université de Rennes I, Rennes, France, in 1999 and the PhD degree in Electrical Engineering from Université de Bretagne Sud, Lorient, France, in 2003. From 2003 to 2013, he successively held Postdoc, Assistant, and Associate Professor positions at Aalborg University, Aalborg, Denmark. He then joined Tallinn University of Technology, Tallinn, Estonia as a Senior Researcher and then on a professorship of Cognitive Electronics. He has supervised 15 PhD theses and 60 MSc theses. He is and has been PI, co-PI, and WP leaders in several European and national projects. His research interests include cognitive electronics, embedded systems, IoT, and wireless communication.

Luca Reggiani received the PhD in Electronics and Communications Engineering in 2001 from Politecnico di Milano (Italy). He has collaborated with several industries and Universities in the field of wireless communications and magnetic recording, as a consultant or within Italian and European research programs. He is a co-founder of two startup companies in the ICT field and his research interests include mobile systems, wireless sensor networks, UAV applications.

M. Moazam Azeem (Member, IEEE) received his Master's degree in system on chip design from KTH, Stockholm, Sweden, in 2010, and the Ph.D. degree in Radiocommunications from France-Telecom in collaboration with CNAM, Paris, France, in 2014. He is currently working as Assistant Professor with the College of Engineering, Department of Computer science and Engineering, Qatar University, Doha, Qatar. He was a Researcher with the School of Computing and Digital Technology, Birmingham City University, Birmingham, U.K., in 2021. From 2018–2020, he was a System Architect with Department of Wireless Communication, Continental Automotive, France. From 2015–2018, he was a Research Associate with Sorbonne University, Paris, France. He is also the inventor of Patent in reliable data transmission for multiple users in cellular communication. His main research interests include wireless communicaton, spectrum sensing, opportunistic spectrum access, erasure correcting codes, dependable fault tolerant computing and Multiple FPGA Boards. He Contributed to FP7 European project that was started by European Commission with budget of 8.1€ billion. He also contributed to Industrial project Flex at Orange Labs, WinNoCod and WasgaServe projects at Sorbonne University.

Tarik Taleb (Senior Member, IEEE) received the B.E. degree (with distinction) in information engineering and the M.Sc. and Ph.D. degrees in information sciences from Tohoku University, Sendai, Japan, in 2001, 2003, and 2005, respectively. He is currently a Full Professor at Ruhr University Bochum, Germany. He was a Professor with the Center of Wireless Communications, University of Oulu, Oulu, Finland. He is the founder of ICTFICIAL Oy, and the founder and the Director of the MOSA!C Lab, Espoo, Finland. From October 2014 to December 2021, he was an Associate Professor with the School of Electrical Engineering, Aalto University, Espoo, Finland. Prior to that, he was working as a Senior Researcher and a 3GPP Standards Expert with NEC Europe Ltd., Heidelberg, Germany. Before joining NEC and till March 2009, he worked as an Assistant Professor with the Graduate School of Information Sciences, Tohoku University, in a lab fully funded by KDDI. From 2005 to 2006, he was a Research Fellow with the Intelligent Cosmos Research Institute, Sendai. Taleb has been directly engaged in the development and standardization of the Evolved Packet System as a member of the 3GPP System Architecture Working Group. His current research interests include AI-based network management, architectural enhancements to mobile core networks, network softwarization and slicing, mobile cloud networking, network function virtualization, software-defined networking, software-defined security, and mobile multimedia streaming.

Muhammad Mahtab Alam (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from Aalborg University, Denmark, in 2007, and the Ph.D. degree in signal processing and telecommunication from the University of Rennes1 France (INRIA Research Center), in 2013. He did his postdoctoral research (2014–2016) at the Qatar Mobility Innovation Center, Qatar. In 2016, he joined as the European Research Area Chair and as an Associate Professor with the Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, where he was elected as a Professor in 2018 and Tenured Full Professor in 2021. Since 2019, he has been the Communication Systems Research Group Leader. His research focuses on the fields of wireless communications–connectivity, mobile positioning, 5G/6G services and applications. He has over 15 years of combined academic and industrial multinational experiences while working in Denmark, Belgium, France, Qatar, and Estonia. He has several leading roles as PI in multimillion Euros international projects funded by European Commission (Horizon Europe LATEST-5GS, 5G-TIMBER, H2020 5G-ROUTES, NATOSPS (G5482), Estonian Research Council (PRG424), Telia Industrial Grant etc. He is an author and co-author of more than 150 research publications. He is actively supervising a number of Ph.D. and Postdoc Researchers. He is also a contributor in two standardization bodies (ETSI SmartBAN, IEEE-GeeenICT-EECH), including "Rapporteur" of work item: DTR/ SmartBAN-0014".