On enhancing transmission performance for iov based on improved greedy algorithm

Longqi Wei[1,4], **Junyuan Feng**[2,5] **and Yancong Deng**[3,6]

[1]Tokyo Institute of Technology
[2]University of California, Santa Barbara
[3]University of California San Diego

[4]wei.l.ac@m.titech.ac.jp
[5]junyuanfeng@umail.ucsb.edu
[6]Yad002@eng.ucsd.edu

Abstract. Nowadays, the development of 5G and IoT technology can be observed from abundance of their applications, such as VR/AR/MR, communication, healthcare, transportation and so on. Some of these applications are in need of quality improvement like delay and resource consumption decrease since the need of users and devices about the relatively developed technique gradually increase as well. About this kind of QoS improvement in the field of VANET, this paper first has a brief review on this concept and related key word and makes use of the concept of Edge Computing, eventually provides information about the research that focuses on the optimization of Task Offloading function by proposing a new algorithm. The new algorithm consists of 3 C++ programs and determines the relatively effective task offloading strategy by taking both calculating resource consumption and communication latency into consideration. In order to prove the developed performance of the proposed algorithm, paper uses other two kinds of algorithms to have comparison, finally reaches the conclusion that proposed algorithm has the best system cost performance among the comparison objects.

Keywords: task offloading, VANET, greedy algorithm.

1. Introduction

1.1. VANET

VANET, the abbreviation of Vehicular ad-hoc network, is a system that is mainly developed/active on vehicles and able to combine vehicles and road units together [1]. In details, when vehicles move and cross together, they can upload and download tasks and information from each other and units set beside roads, calculate, compute and make use of all available resource with the help of VANET system. This system is invented in order to combine the vehicle network together and make full use of the giant potential power of it to achieve further goals such as self-driving, traffic controlling, road safety monitoring and other highly developed techniques that can make people's life more convenient than it was before the rapid improving 5G era nowadays. As the mobile large calculating resource inside the car network can be effectively utilized by VANET with few or even no "transfer stations", the total abilities, such as computing rate and transferring latency, are highly evaluated through many existing related researches and experiments at past. The model of VANET is also a very important part of its relative studies and researches, but since it is hardly related to our research, we won't discuss much about this topic here.

There are abundant existing communication technologies that can help build, contribute to and optimize a VANET system on its advantages mentioned above such as latency and communication efficiency nowadays, and in this research, we make use of one of these methods, which is called Mobile Edge Computing.

1.2. Mobile edge computing

MEC, Mobile Edge Computing and also known as Multi-access Edge Computing, is a kind of method that harvest the large amount of available computing power and storage space (mainly concentrate on edge servers) to deal with the provided mission[2], which helps decrease the latency when doing observation around the surrounding environment, communication between mobile and non-mobile equipment and calculation about the tactic during the whole computing process, make full use of the determined most appropriate strategy, reduce the system cost and eventually develop the whole performance of the system. Having so many advantages described above, there is no doubt that MEC will be one of the most focused technologies in this computer-ruled era.

This approach is beneficial for Augmented Reality (AR), 5G, Internet of Things (IoT), self-driving car and many other technologies nowadays. For example, mobile AR technology needs computer vision computing and real-time display that should be dealt with under situations having abundant storage and calculation source, which cannot be provided by one simple mobile device[3]. Another application is on vehicle and self-driving, which uses cars linked in the vehicular network as edge servers and storage sources to calculate and relates to devices/equipment set on the two sides of roads and IoT field and therefore is related to VANET network and its applications. Also, it has already been under research and utilized by a large number of papers, which will be later discussed in part 2: Related works, companies like Huawei, Apple, Lenovo, Equinix and other leading technology-based organizations around the whole developed world. Such academic studies, experiments and commercial utilizations, plans and their similar ones can be found all over the internet, showing that this technology is valuable in the both fields and can contribute a lot to the society from abundance of ways. This is the reason why we begin the research focusing on this topic.

With this kind of broad utilization, we are going to discuss and focus on one of its applications on task offloading division on moving vehicle mentioned above, which use these moving cars as edge servers in the MEC network in order to apply the advantages of MEC into self-driving car and VANET system as we mentioned in part 1.1. Additionally, combining the advantages of VANET and MEC, the final goal of the research is to achieve improvement on the task offloading function.

1.3. Task offloading

Task offloading is a kind of function that devices offload tasks to servers at the edges of the network[4], which are close to devices and therefore have advantages like 1.low latency: costing little time to receive data from device and send information to device; 2. High privacy: nearby server (for example, roadside device[5]) and thus no need to upload to large and far ones that are monitored by far more people; and so on. It is also a popular topic nowadays and is discussed in many papers about its optimization, application and other related fields.

In this paper, we will main talk about its detailed application with MEC in VANET.

2. Related work

As mentioned in Introduction part, there is a large number of academic papers in which application and optimization about MEC and VANET are under discussion. In this part, we are going to list some of these helpful and valuable previous articles which influence or contribute to the accomplishment of this paper:

When first focusing on the research of this paper, we had an investigation on the features and existing problems of VANET, MEC and other relative technologies in the related fields at the beginning, mainly combining some related key words such as Augmented Reality, 5G and the development and control of self-driving vehicle system.

During this investigation, we found the connection between the application of MEC, 5G and AR, which may also help the optimization of our research and experiment through "Context-Based MEC Platform for Augmented-Reality Services in 5G Networks" [6] at first. In this paper writers discuss about the utilization of MEC platform for AR in 5G environment, combing these three key words together partly to solve the high latency problem, which is the result of letting only one single device to deal with tasks, with the help of edge server platform in detail. This is also a great example in which researchers use MEC and the combing-together technique to solve system cost or service latency problem and may be utilized on our algorithm development.

Additionally, we found "A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects"[2], realizing that one of the factors that should be improved during the process of system optimization is the communication latency by reading this paper: In this paper the authors talk about the application of 5G Mobile Edge Computing on Mobile Augmented Reality since one of the advantages of MEC is the ability to reduce the communication latency with the utilization of edge servers and is needed for Augmented Reality, because delay serves as an important factor for the performance evaluation of AR systems[7] and also an important part of our research.

After the broad investigation in order to find the feature that we should concentrate on in the following experiment session, we decide to have deeper and further understanding of the VANET and the detailed use of its categories through some essays. As the result, "V2V COMMUNICATION SURVEY - WIRELESS TECHNOLOGY"[8], on behalf of other papers that we read during this period, truly help us deepen the understanding of us: In this paper, the writers talk about the concept and structure of Vehicle-to-Vehicle (V2V) technology[9], which is included in Machine-to-Machine (M2M) and is a relatively popular part of IoT, able to contribute to the connection between vehicles and vehicles together to share information, offload tasks and therefore both related to and benefit from MEC; These papers are relatively highly related to our research and showed us the possibility to effectively combine the mentioned techniques together and achieve optimization.

Also, we read "Traffic and Computation Co-Offloading With Reinforcement Learning in Fog Computing for Industrial Applications"[10] and other kinds of similar papers in order to find the classic approach that can help proof the advantage of our algorithm by the means of comparison through experimental group and control groups: authors evaluate the offloading performance by simulation, focusing on metrics, including "energy cost caused by mobile devices (Vehicle/Device in V2V/D2D)" and "delay spending on data transmission and computation", of their scheme (DRLS and DDS, dynamic RL scheduling algorithm and deep dynamic scheduling algorithm) and other three schemes: NO, DDO and NDO in this paper. As the result, authors' scheme is better in energy cost but has higher service delay than other three because writers put a greater weight on energy cost rather than latency. On the other hand, we put the same weight on both of these two factors in our experiment.

3. Method

Since there are high performance requirements for the communication/computing latency and computing resource consumption for existing related technologies, such as IoT, 5G and self-driving car that are related to the main topic of this paper, in order to achieve optimization on these requirements in this promising field, after a relatively long period of efforts, we manage to build an algorithm which is based on non-cooperative game theory and invented to create the appropriate task offloading strategy about which server a specific task will be sent to, local server having lower latency and edge/cloud servers which have higher available computing resource (when there isn't enough resource in edge server, this algorithm decides to upload task to cloud server), and finally achieve the "system cost minimization" goal, which means that the algorithm manages to have high performance on both resource consumption of servers and latency produced during computation/communication between units and servers (This kind of situation/strategy distribution is partly referred from [11] and similar papers).

3.1. Information notification in advance

At first, units and devices in IoT will "tell" the edge servers the detailed information of the service that is related and going to be deal with in the coming task for each individual task.

3.2. Popularity evaluation

Then, the edge server will have evaluation on the popularity of services by the frequency of the utilization of each service (how many times each service is called and used) and decide whether the service should be saved in the edge server or not mainly based on the popularity. In other words, the more one service is requested by tasks, the more likely system will decide to save it in the local server in order to save time and reduce latency which may be produced during the service delivering process.

3.3. Service offloading

For those service which are used fewer and therefore have less popularity, since the storage of edge servers is limited, these will not be saved in the server and be recorded in the cloud server instead. Then, if some tasks request the less-used services, the system will download these ones from the cloud server. Considering the server storage and service popularity, the distribution strategy will be designed in order to minimize the total delay time during the unavoidable transmission processes.

3.4. Task procession

For the services that are already saved in the edge server, the requesting tasks will be dealt with and executed in the edge server as well; On the other hand, when some tasks request the less popular services, system needs to make a decision about the place where the tasks are going to be offloaded and measured, either cloud or edge server, in order to save time and decrease time lag.*A subsubsection.* The paragraph text follows on from the subsubsection heading but should not be in italic.

These steps are performed in a series of program (in detail, 3 programs: Task Offloading program to make the decision about the task offloading according to non-cooperative game theory; Dynamic Service Caching program to achieve dynamic service cache by maximizing popularity of service and tusing 0-1 backpack algorithm; Adjustment of Task Offloading program to reach the final task offloading decision by combing the results of the two programs above together) written in C++ language. In the program, there are 5 IoT devices, each of which has a task to be dealt with. These tasks are going to be dealt by one edge server and a cloud server. To represent the three methods used during the whole process (as we mentioned above, local/cloud/edge server), the algorithm uses 0,-1 and 1.

For the detailed information, we first input the information about CPU, IoT device and tasks into Task Offloading program and decide how many times the program should iterate, then get the initial distribution strategy; Secondly, we activate Dynamic Service Caching program to get the cache distribution strategy in the edge server (whether there is cache in the server or not) for every service requested by tasks; After getting all the strategies produced by the above steps, we input this information into Adjustment of Task Offloading file, eventually reach the final offloading decision. These steps are displayed in figure 1 as well.

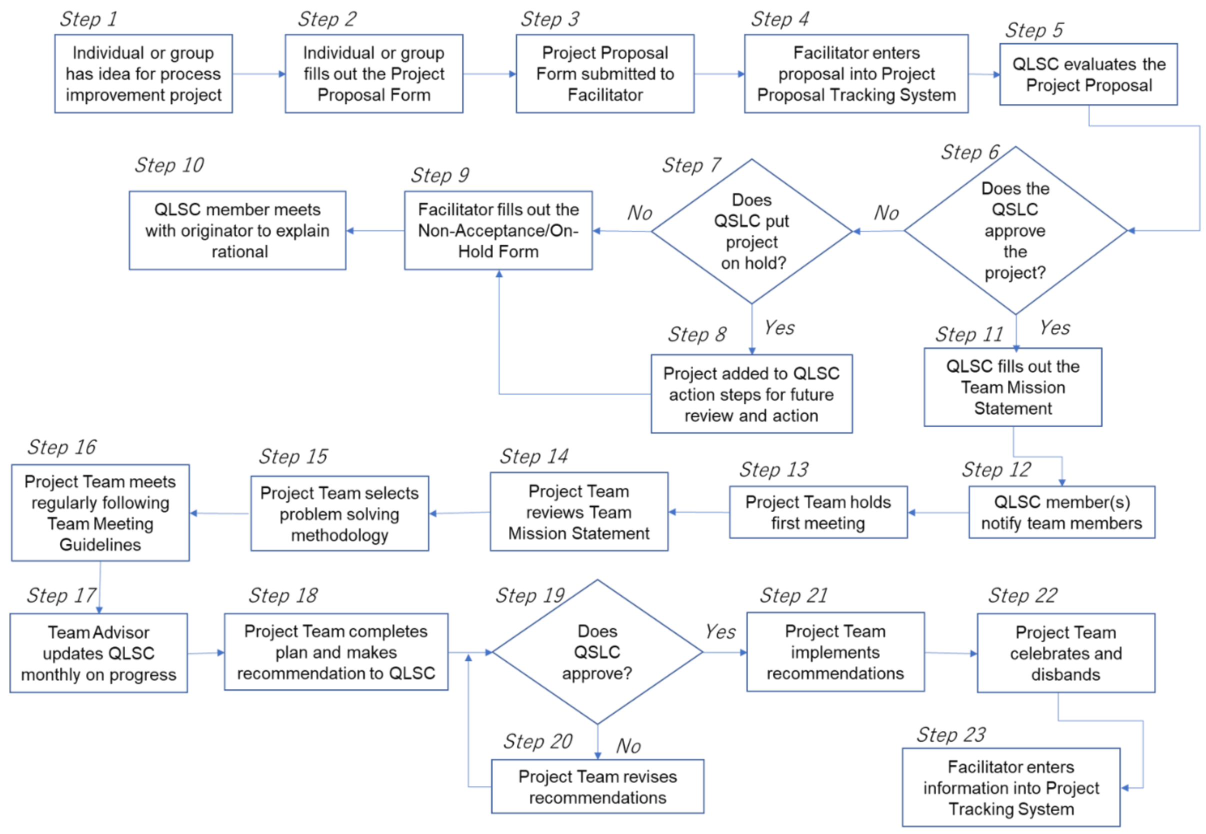
Step 1
Individual or group has idea for process improvement project
Step 2
Individual or group fills out the Project Proposal Form
Step 3
Project Proposal Form submitted to Facilitator
Step 4
Facilitator enters proposal into Project Proposal Tracking System
Step 5
QLSC evaluates the Project Proposal
Step 6
Does the QSLC approve the project?
No
Step 7
Does QSLC put project on hold?
No
Step 9
Facilitator fills out the Non-Acceptance/On-Hold Form
Step 10
QLSC member meets with originator to explain rational
Yes
Step 8
Project added to QLSC action steps for future review and action
Yes
Step 11
QLSC fills out the Team Mission Statement
Step 12
QLSC member(s) notify team members
Step 13
Project Team holds first meeting
Step 14
Project Team reviews Team Mission Statement
Step 15
Project Team selects problem solving methodology
Step 16
Project Team meets regularly following Team Meeting Guidelines
Step 17
Team Advisor updates QLSC monthly on progress
Step 18
Project Team completes plan and makes recommendation to QLSC
Step 19
Does QSLC approve?
No
Step 20
Project Team revises recommendations
Yes
Step 21
Project Team implements recommendations
Step 22
Project Team celebrates and disbands
Step 23
Facilitator enters information into Project Tracking System

Figure 1. The procedure of the whole proposed algorithm program.

After these steps, all of the tasks will be offloaded to the most appropriate places in which they can be most effectively dealt with and transported. Additionally, if the appropriate distribution strategy is correctly executed, the whole system cost, the combination of low latency and resource usage, will be minimized, which means the surplus available system resource can be made use of to achieve other goals such as the development of system ability and more computing/transportation tasks, leading to the optimization of the whole system as expected in the future.

4. Experiment and analysis

In this part, we will discuss about the experiment, which is made in order to test the performance of the algorithm by changing the parameters of the situations in which it is applied and comparing the output with other results from other two algorithms, and perform analysis on the total result of the calculation and comparison in a simple term.

4.1. Experiment

As for this technology, we conducted a series of tests on the developed task offloading algorithmic program, which is mentioned in the Method section, by changing the situation the algorithm is facing for several time and comparing the output system cost with two other chosen algorithms that are used under the same condition (parameters). According to these tests, we got the result showing that when having the same set of CPU frequencies and edge server storage space, this program relatively significantly has lower system cost and thus outperformed the two existing algorithms, TOCS (Task Offloading and Service Cache, an algorithm without the help of cloud server and thus has a shortage in storage resource) and TO (Task Offloading, an algorithm that only focus on task offloading and doesn't take cache of service into account like the other two programs: doesn't calculate about the popularity of each service and thus doesn't have measurement in advance), when they were used for comparison in terms of task offloading, and thus this algorithm has a certain degree of advantage in terms of task offloading mission, which can be beneficial to the operation of MEC.

4.2. Result and analysis

The specific results of our evaluation according to the experiments are shown as graph 2, 3, 4 and 5, and in the following part we are going to explain about the three graphs in detail:

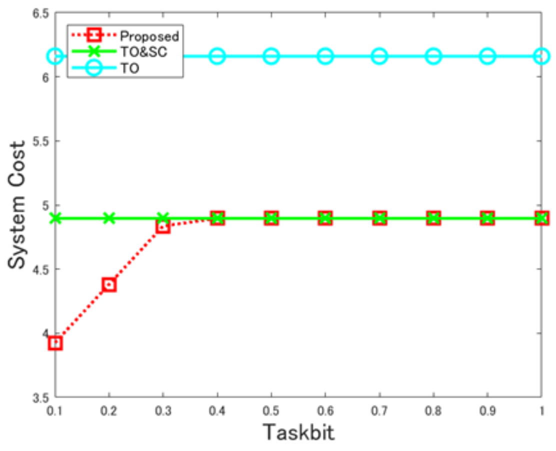

Figure 2. The change of system cost when task bit changes.

In the experiment of figure 2, we plan to change task bit of the given system from 0.1 to 1 without changing any other parameters in order to test the performance by the mean of control variates. As the result, apart from the TO algorithm which constantly has higher system cost than the other two algorithm and therefore has a worse performance, from 0.1-0.3, proposed algorithm uses less resources of the system than TO&SC algorithm, therefore acts better than these two programs which simply consider offloading and system cost.

In the experiments of figure 3 and 4, we have a test on the effect to the system cost produced by CPU frequency, mainly by changing the frequency of the fifth server in the simulating program. This test is divided into two sections (former is shown in graph 3, and latter is shown in graph 4), the relatively low frequency band (0.01 to 0.1 GHz) and the relatively high frequency band (0.1 to 1 GHz) in order to more obviously show the change and difference among the three algorithms. The results are shown in the two following figures.

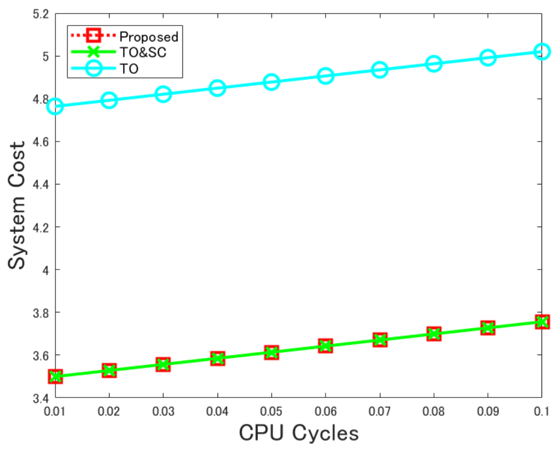

Figure 3. The change of system cost when CPU usage changes (low frequency).

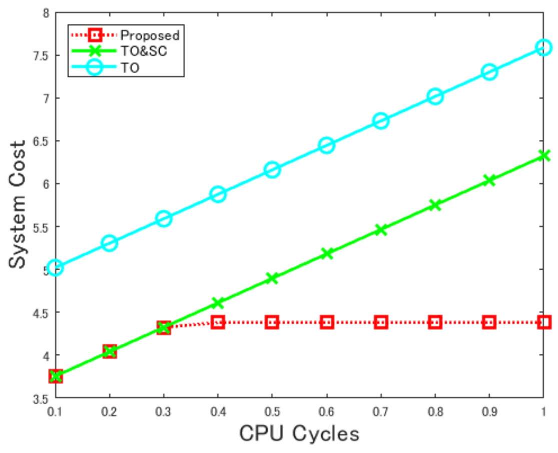

Figure 4. The change of system cost when CPU usage changes (high frequency).

Figure 3 shows the result of separated test part between 0.01 and 0.1 GHz for the fifth server's CPU frequency. In this experiment session, the TO&SC algorithm has the same performance as proposed algorithm, while TO program has higher system cost value than the others, eventually showing that the proposed algorithm performs better than the two existing programs in the field of system cost.

As for the experiment in the high frequency band, we find that from the result shown in figure 4, the system cost value of proposed program becomes different with TO&SC when the frequency reaches 0.3 GHz and consistently has lower cost value than TO&SC, meaning that in the range of 0.3 – 1 GHz in this experiment, the difference between proposed algorithm and TO&SC (the utilization of local server and device in this situation: TO&SC only considers the use of edge and cloud server, while our proposed program effectively combines local, edge and cloud server/device together in order to decrease the latency and minimize the consumed calculating resource) gradually plays a more important role in the whole computing process of system cost value.

On the other hand, we also test the influence of forth server on behalf of other ones as the experiments mentioned above all focus on fifth device only. In this experiment, we change CPU frequency in the range of 0.1 – 1 GHz, 1.4 GHz is also taken into consideration because the highest frequency in the limitation of 4th server in this experiment is 1.4 GHz). The result is shown in figure 5:

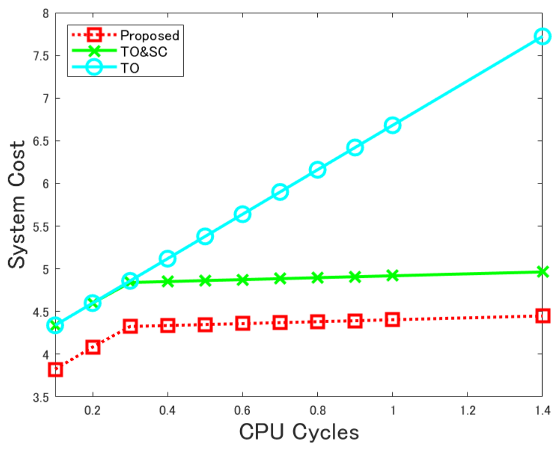

Figure 5. The change of system cost when CPU usage changes (high frequency, 4th server).

Different from the experiment on 5th server, TO&SC algorithm has worse performance between 0.1 – 1 GHz, compared with proposed program. This result tells us that the function of TO&SC isn't always the same as our algorithm (Nearly the same in the range of 0.01 – 0.3 GHz in 5th device; Constantly higher system cost in 4th device. Additionally, the same changing tendency in 4th device, while constantly increase in 5th device). We also test the situation when the CPU frequency is set as 1.4GHz, the maximum of the IoT device setting.

Additionally, we can also find that the performance of the system, displayed by the system cost value, hardly changes after the CPU frequency value reaches 0.3 GHz and becomes larger gradually in the two algorithms which take the service cache into account, meaning that the cost-consuming effect of task offloading function may be negated by the influence produced by the service cache.

5. Conclusion

From the comparison between algorithm TOSC, TO and the proposed task offloading algorithm on the aspect of system cost with the different server frequency and taskbit set, it can be seen that the proposed one performs better than the other two programs, showing that this proposed algorithm can contribute to the VANET field with the reduced total system cost in several same provided experiment situation. Additionally, further optimization of the program is also expected in the aspect of calculating efficiency and communicating latency by more development of it.

References

[1] Muhammad Nadeem Majeed, Dr. Shahbaz Pervez Chattha, Dr. Adeel Akram, Dr. Mohammad Zafrullah, "VEHICULAR ADHOC NETWORKS HISTORY AND FUTURE DEVELOPMENT ARENAS", Journal of Electrical Engineering Volume3(Issue1) : pp1-6, 2013

[2] Y. Mao, C. You, J. Zhang, K. Huang and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," in IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2322-2358, Fourthquarter 2017

[3] R. Cai. "A study on MEC-based mobile augmented reality applications", Beijing University of Posts and Telecommunications, 2017.

[4] Firdose Saeik, Marios Avgeris, Dimitrios Spatharakis, Nina Santi, Dimitrios Dechouniotis, John Violos, Aris Leivadeas, Nikolaos Athanasopoulos, Nathalie Mitton, Symeon Papavassiliou, "Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions," Computer Networks, Volume 195, 2021, 108177, ISSN 1389-1286.

[5] K. Abdukodir, A. Muthanna, I. Ibodullokhodzha, A. A. Ateya and A. Koucheryavy, "Development of Edge Computing Distribution Method in VANET Based Real-Time Systems," 2019 IEEE International Conference on Electrical Engineering and Photonics (EExPolytech), 2019, pp. 120-123.

[6] Y. Wang, T. Yu and K. Sakaguchi, "Context-Based MEC Platform for Augmented-Reality Services in 5G Networks," 2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall), 2021, pp. 1-5

[7] Kjell Brunnström, Elijs Dima, Tahir Qureshi, Mathias Johanson, Mattias Andersson, Mårten Sjöström, "Latency impact on Quality of Experience in a virtual reality simulator for remote control of machines", Signal Processing: Image Communication, Volume 89, 2020, 116005, ISSN 0923-5965

[8] Khairnar, Dr. Vaishali & Pradhan, Shrikant, "V2V COMMUNICATION SURVEY - WIRELESS TECHNOLOGY" (2014). V2V communication survey wireless technology. Int.J.Computer Technology & Applications (IJCTA). 3.

[9] Xinzhe Wang, Yancong Deng, Tianshu Wang, Yiwei Zhang, and Hongxing Jiang. 2021. "A Hybrid Vehicle-to-Vehicle Transmission Model for Vehicular Networks". In Proceedings of the 11th International Conference on Information Communication and Management (ICICM '21). Association for Computing Machinery, New York, NY, USA, 34–40.

[10] Y. Wang, K. Wang, H. Huang, T. Miyazaki and S. Guo, "Traffic and Computation Co-Offloading

with Reinforcement Learning in Fog Computing for Industrial Applications," in IEEE Transactions on Industrial Informatics, vol. 15, no. 2, pp. 976-986, Feb. 2019, doi: 10.1109/TII.2018.2883991.

[11] C. -M. Huang, Z. -Y. Wu and S. -Y. Lin, "The Mobile Edge Computing (MEC)-Based VANET Data Offloading Using the Staying-Time-Oriented k-Hop Away Offloading Agent", 2019 International Conference on Information Networking (ICOIN), 2019, pp. 357-362.