On the Feasibility of Using 5G Enabled Smartphones to Improve Safety of Vulnerable Road Users

Joel Puga
Urban and Mobile Computing department
Centro de Computação Gráfica (CCG)
Guimarães, Portugal
joel.puga@ccg.pt

Filipe Meneses
Centro Algoritmi / Centro de Computação Gráfica
Universidade do Minho
Guimarães, Portugal
meneses@dsi.uminho.pt

Adriano Moreira
Centro Algoritmi / Centro de Computação Gráfica
Universidade do Minho
Guimarães, Portugal
adriano@dsi.uminho.pt

2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring) | 979-8-3503-1114-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/VTC2023-Spring57618.2023.10199493

***Abstract* — Autonomous vehicles require sophisticated sensors to safely and timely detect the conditions of the world around them and act accordingly. However, just like human senses, sensors have limitations and blind spots. Vulnerable Road Users, such as pedestrians and cyclists, are at particular risk since they can quickly come to the road from a blind spot too late for the vehicle to react. This article evaluates the use of a commercial 5G-enabled smartphone and a purpose-designed app to advertise a Vulnerable Road User position to a connected autonomous vehicle using European Telecommunications Standards Institute standard messages. The results obtained in a real testbed proved that a 5G network is capable of supporting the low latency required for this use case, even though the usefulness of the positioning data transmitted was limited by the accuracy of the GPS embedded in commercial smartphones.**

Keywords—5G networks, Vulnerable Road Users, Latency

I. Introduction

5G-MOBIX[1] is a European Union (EU) funded project aiming to develop and evaluate automated vehicle functionalities using 5G core technological innovations. One of the concerns addressed in this project was the detection of Vulnerable Road Users (VRUs), like pedestrians or cyclists, that can interfere with an autonomous vehicle but are located behind obstacles that block the reach of vehicle sensors.

Two approaches were studied during the 5G-MOBIX trials to address this concern: placing Road Side Units (RSUs) on potential sensor blind spots that alert the autonomous vehicle of a VRU presence; and creating an app for Android smartphones that periodically advertise the VRU location to the autonomous vehicle. This article describes the experimental setup, trials and results for the second approach.

Cooperative Connected and Automated Mobility (CCAM) applications require a near real-time delivery of messages between the different actors. As such, the most critical measurement in our case is end-to-end (e2e) latency. Therefore, latency was the main metric used in this work to evaluate the feasibility of the approach identified above.

The testbed and experiments were conducted on the particular case of a cross-border scenario, where additional difficulties have to be tackled for the multiple use cases addressed by the 5G-MOBIX project.

In this paper, many implementation details are provided to enable the reproducibility of this work. Nevertheless, recreating the used testbed requires access to a properly configured 5G network.

II. Related Work

The main challenges for CCAM services in a cross-border and multi-MNO (Mobile Network Operators) environment have been identified in [2] and [3], including the scenarios of having Anticipated Cooperative Collision Avoidance systems. In addition, low latency was recognised as one of several key technical issues to be addressed in a cross-border and multi-domain environment.

Latency can be influenced by several factors, including the MNO MEC (Multi-access Edge Computing) implementation that can be done on the network core (worst from a latency perspective) or, on the other end, at the regional point-of-presence or the radio cell level. In [4], the authors demonstrated that the influence of MEC services on network latencies is not significant, with a measured mean value of 99 ms.

Today, there are few places where all conditions can be met to conduct a cross-border study using 5G networks for autonomous driving. For example, in [5], the authors simulated cross-border operations by switching carrier networks in vehicle-to-road collaboration to assist advanced driving. One of the advantages of this work is that the experiments were conducted over a pre-commercial network in a real cross-border scenario.

III. Approach and System Architecture

The Spanish-Portuguese testbed of 5G-MOBIX includes two Multi-access Edge Computing (MEC) deployments, one on each side of the border. Each MEC has an MQ Telemetry Transport (MQTT) broker (Mosquitto[6]) responsible for routing messages from and to the vehicles and other devices. For the experiments described in the paper, a specially crafted Android App was used to send messages from the VRU to the MEC. Since all the tests with the VRU App were made on the Portuguese side, it was connected only with the Portuguese MEC's MQTT broker. Another component of the MEC takes the messages sent to the broker and re-sends them to topics that the vehicles are subscribing to. The MQTT brokers of the two MECs are also interconnected, so messages sent to one MEC are replicated to the other and then resent to vehicles connected to the latter.

A simplified version of the testbed architecture is shown in Fig. 1, with just the components necessary for testing the VRU App. Since the latency measurements relevant to this

study were on the communications between the smartphone and the Portuguese-side broker, the Spanish-side MEC has been omitted from the diagram. However, the shuttle was connected to the broker on the Spanish MEC.

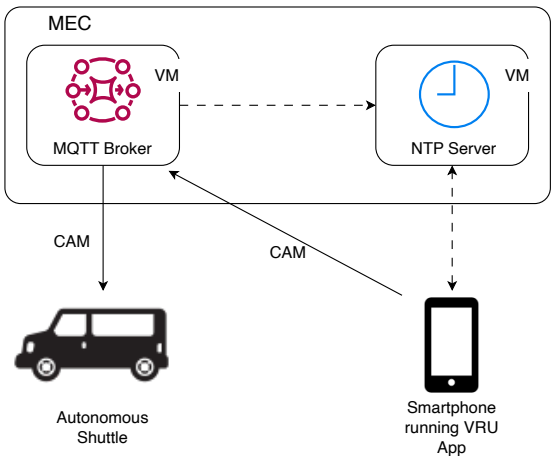

Fig. 1. Simplified architecture of the system with a single MEC

Latency measurements require comparing the smartphone and MEC timestamps. For the comparison to be accurate, a Network Time Protocol (NTP) server was installed and configured on the MEC, and all the MEC components and the smartphone were synchronized with it. To synchronize the smartphone with the server, the App ClockSync[7] was used.

The App sends periodic CAM messages with a payload made of the geographic position of the VRU as obtained from the smartphone GPS through the native Android API. The main goal of the performed tests was to evaluate if the VRU is able to advertise her position to the autonomous shuttle fast enough to prevent an accident.

IV. Experimental Setup

The trials were executed in the Old International Bridge between the Portuguese city of Valença and the Spanish city of Tui. This bridge has a road platform under a train platform. The stone columns needed to support the latter (see Fig. 2) create a blind spot for any autonomous vehicle driving on the road platform where a VRU can stand undetected. Therefore, the trials with the VRU App were done on the Portuguese side of the bridge.

The smartphone used for the trials was a Xiaomi Mi 10T Pro 5G. Sniffing network traffic on the Android OS requires root access, as, for security reasons, an App cannot access the IP traffic of other Apps and/or the system. For this same reason, it is also not possible for an application to set system time without root access. Therefore, as time synchronization and network traffic sniffing were essential to measure the performance of 5G communications, the smartphone had to be rooted.

The smartphone was connected to a 5G small cell installed on the bridge on the Portuguese side (point 1 in Fig. 2).

The Android App was developed using React Native[8], a framework for multi-platform development using Javascript. Because European Telecommunications Standards Institute's (ETSI) Intelligent Transportation System (ITS) standard messages [15] need to be encoded using ASN.1 UPER encoding [16], and no React Native library was available, at the time of development, that could do it; so the encoding was performed by a Python script running on the Termux[9] terminal emulator for Android. To avoid delays with messages going back and forth between the App and the Python script, the Python script is also responsible for sending the Cooperative Awareness Messages (CAM) [17] after encoding them. Communication between the App and the Python script is done through a web service developed using the Flask library.

Fig. 2. Trial Location showing the shuttle about to leave the bridge

Since communications between the MEC and the vehicle have been thoroughly tested and documented by other project partners [10][11][12], we decided to focus our efforts on the communication between the smartphones and the MEC.

The Android App logged all the transmitted messages with a time stamp to obtain the required latency measurements. The same was done on the MEC side with received messages. Both logs were then fed to a custom build script that compared them and obtained the latency measurements.

Besides the application-level measurements, we also obtained network-level measurements, using tcpump[13] to create pcap files on both the smartphone and the MEC. Access level data (i.e. information about the mobile network, like current node, signal strength and other network properties) were also captured using G-NetTrack Pro[14]. Both tcpdump and G-NetTrack Pro were running simultaneously with the App to capture data in real-time and allow cross-referencing between the logged data.

In order to evaluate the system performance under different network conditions, test runs were performed with the network under artificial stress and without it. When under no stress, there were only communications between the smartphone and the MEC and between the MEC and the autonomous vehicle (and vice versa in this last case) on the network. For the stressed network runs, stress was created by a partner using a vehicle with 3 OBUs with three modems each. IP traffic was generated using Iperf, to a maximum of 1.2 Gpbs.

V. Trial Execution

The trial occurred on the Portuguese side of the bridge. With the test Smartphone running the VRU App, a pedestrian crossed the bridge from the right-hand side of the road (point A in Fig. 2) to the left-hand side (point B in Fig. 2). The autonomous vehicle crossed the bridge in the Spain-Portugal direction (Fig. 3 gives a birds-eye view of the VRU and autonomous vehicle trajectories). All sensors and other data inputs to the vehicle were turned off to validate the App

functionality without external interference. A human operator was in place and prepared to act in case of failure or another emergency. The VRU crossed the street ahead of the autonomous vehicle with a safety gap to allow reaction time in case of any issues.

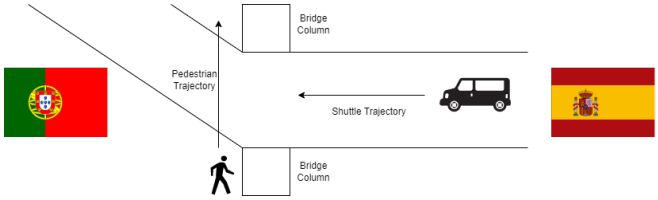

Fig. 3. Trial Plan

VI. Results

A. Application-level latency measurements

The application logged every message sent at the application level, while the MEC kept its own log of messages received in the MQTT broker. We sent 3847 CAM messages divided by six continuous runs (three with light network load and three with heavy network load). A total of 2002 messages were sent while the network was unencumbered, while 1845 were sent with the network artificially stressed. These data was aggregated in one-second intervals, resulting in 3021 data points, 1511 with artificial network stress and 1510 without. Due to limitations and battery-saving features of Android, it was not possible to get GPS readings for the CAM messages at will or force a constant rate. As such, the number of messages sent per second varied from 1 to 3. As mentioned before, all messages were encoded using ASN.1 UPER encoding.

For the application-level, our target value for end-to-end latency was 200ms, within the parameters recommended in [19][20][21]. Latency was defined as the difference between the sending time on the Smartphone and the receiving time on the MEC's broker. Other factors, like packet loss, were not taken into account.

The median of all our readings for message end-to-end latency (Fig. 4) was 56ms, well below our pre-determined ceiling of 200ms. However, as seen on the graph in Fig. 4, there are a few outliers, with the maximum latency obtained being 6021ms and the minimum being 5ms. Furthermore, significant differences exist between the maximum latency obtained on test runs done over a network not stressed and those done with a network stressed (Fig. 4). The runs with the stressed network had a maximum latency of 6021ms, well above the maximum for runs without network stressing (259ms). On the other hand, the minimum and median latency of tests done with network stressing (5 and 54 ms, respectively) were slightly lower than the ones without stressing (7 and 63 ms, respectively).

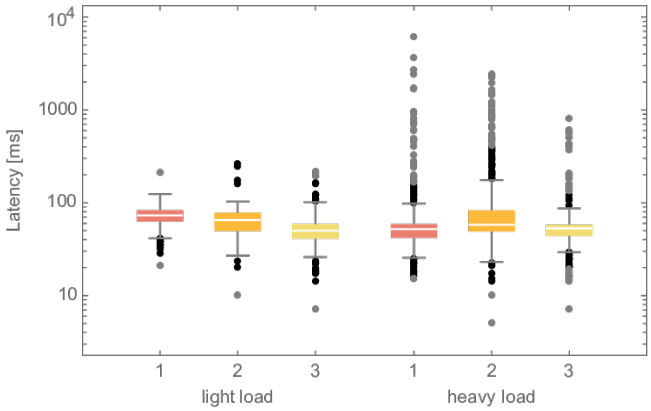

Fig.4. Box Plot of Application-Level Latency Readings

On the network-level, we used tcpdump to capture traffic on both the smartphone and the MEC. We captured a total of 8308 MQTT sent packets, 4268 with network stress and 4040 without. Once more, we aggregated the results in one-second intervals, giving us 3090 data points, 1543 with network stress and 1547 without. The network-level data was captured concurrently with the application-level data.

B. Network-level latency measurements

For the network-level (like for the application-level), our target value for e2e latency was 200ms, within the parameters recommended in [19]20][21].

The median end-to-end latency was 24ms, with the maximum being 651ms and the minimum 1ms (readings were collected in milliseconds and rounded up, so 1ms readings may include latencies under 1ms). Again, we see significant differences between the maximum e2e latency obtained on test runs done without network stressing and those done with network stressing (Fig. 5). The runs with network stressing had a maximum latency of 651ms. The minimum and median latency for readings done with network stressing were 1 and 651 ms, respectively. For readings done without network stress, these same values were 1 and 391ms.

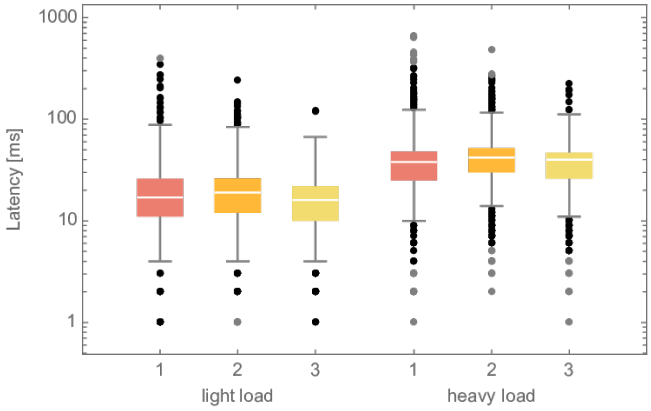

Fig. 5. Box Plot of Network-Level Latency

C. Packet loss

Our target value for maximum acceptable packet loss was 10%. Due to the low speeds of VRUs, it was deemed as an acceptable target.

The overall mean packet loss rate was just under 6,22%, with some significant outliers (Fig. 6). Curiously, the mean packet loss rate for runs done with network stressing was only 5,12%, lower than the overall mean, and the mean for runs done without network stressing was 7,32%. On the other hand, among the samples where packet loss occurred (i.e. it was

higher than 0%), the minimum packet loss rate for readings with network stressing was 7,69%, substantially higher than the same value for readings without stressing (3,45%).

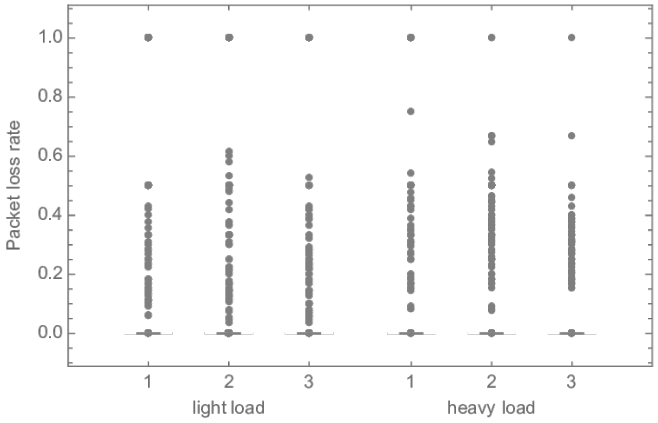

Fig. 6. Box Plot of Packet Loss Rate

D. Other Concerns

The usefulness of the data transmitted by the VRU App was limited by the lack of accuracy of the GPS embedded in the smartphone. The median accuracy estimated by Android was 4.02m, which is a significant error for the envisioned application. As a result, during the trials, the pedestrian often had to advance several meters from his destination for the autonomous vehicle to assume a clear road and resume its operation. Fig. 7 shows a histogram of estimated accuracy, as provided by the Android API, for all samples where the issue is clearly visible. The tests were done in the lower tier of a two-tiered bridge, with the upper deck used for a railroad track, which might have exacerbated the lack of precision. Nevertheless, on test readings realised outside the bridge, but in the same general area, the best readings gave us an average accuracyof just under 3m, which indicates that part of the precision problem is due to the Android phone GPS hardware and/or software.

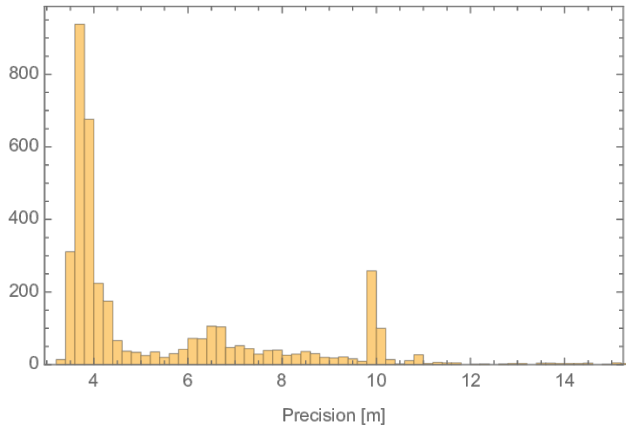

Fig. 7. Histogram of Accuracy Readings

We also detected a significant delay between the actual position of the VRU and the position reported by the smartphone. We believe that this may be due to Android optimizations implemented to conserve battery or a side effect of the previously mentioned lack of accuracy and other GPS limitations, but to confirm this and measure the delay will require further trials, as it is necessary additional equipment and development beyond the scope of 5G-MOBIX.

VII. Conclusions

The results obtained through these experiments showed that a 5G enabled smartphone can achieve the latency required for CCAM uses cases, with and without network stressing. However, there are a few outliers.

The device used for testing also showed itself capable of obtaining GPS readings and creating, encoding and sending CAMs fast enough to achieve a rate high enough for pedestrian use.

The main issues encountered were the lack of accuracy of commercial smartphone GPS receivers and the delay in updating the smartphone position. It is expected, nevertheless, that as technology evolves and becomes cheaper and battery capacity increases, these issues will eventually be solved. Another possibility is applying post-processing to position data to correct the error. We are currently considering an approach that uses Kalman filtering.

The smartphone used in the tests was a high-end model since, at the time of acquisition, only high-end models had 5G capabilities. Further testing must be done on a more diverse set of devices, especially low-end ones, to verify that they achieve acceptable performance for CCAM use cases.

Another possibility considered and partially developed but not tested because of time constraints was using smartphones to receive Decentralized Environmental Notification Messages (DENM) [18] and alert the user to possible dangerous situations. Implementation and testing of this feature could help verify if smartphones can be used to improve VRU safety further.

Acknowledgment

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

References

[1] 5G-MOBIX project website. Available at https://www.5g-mobix.com/

[2] Kousaridas A, Schimpe A, Euler S, Vilajosana X, Fallgren M, Landi G, Moscatelli F, Barmpounakis S, Vázquez-Gallego F, Sedar R, Silva R, Dizambourg L, Wendt S, Muehleisen M, Eckert K, Härri J, Alonso-Zarate J. 5G Cross-Border Operation for Connected and Automated Mobility: Challenges and Solutions. Future Internet. 2020; 12(1):5. https://doi.org/10.3390/fi12010005

[3] D. Hetzer, M. Muehleisen, A. Kousaridas and J. Alonso-Zarate, "5G Connected and Automated Driving: Use Cases and Technologies in Cross-border Environments," 2019 European Conference on Networks and Communications (EuCNC), Valencia, Spain, 2019, pp. 78-82, doi: 10.1109/EuCNC.2019.8801993.

[4] M. Kutila, K. Kauvo, P. Aalto, V. G. Martinez, M. Niemi and Y. Zheng, "5G Network Performance Experiments for Automated Car Functions," 2020 IEEE 3rd 5G World Forum (5GWF), Bangalore, India, 2020, pp. 366-371, doi: 10.1109/5GWF49715.2020.9221295.

[5] C. Zhang, S. Ning, L. Yu and J. Chu, "Design of continuity services for 5G vehicle-road collaboration in cross-border scenarios," 2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Dali, China, 2022, pp. 122-126, doi: 10.1109/ICCASIT55263.2022.9986634.

[6] Eclipse Mosquitto website. Available at https://mosquitto.org

[7] Clock Sync App by Seiko Time Creation Inc. Available at https://play.google.com/store/apps/details?id=com.useinc.ss501k.clocksync

[8] React Native website. Available at https://reactnative.dev/

[9] Termux website. Available at https://termux.dev/en/

[10] Cimoli, Bruno, Haitao Xing, Victor Ho, Igor Passchier, Geerd Kakes, Simon Rommel, Henk Nijmeijer, and Idelfonso Tafur Monroy. 2022. "Practical Challenges in Hybrid Communication Ecosystems Based on

Its-G5 and Lte for Cacc and Glosa." Frontiers in Future Transportation 3.

[11] Ouden, Jos den, Victor Ho, Tijs van der Smagt, Geerd Kakes, Simon Rommel, Igor Passchier, Jakub Juza, and Idelfonso Tafur Monroy. 2022. "Design and Evaluation of Remote Driving Architecture on 4G and 5G Mobile Networks" Frontiers in Future Transportation 2.

[12] Hosseini, Seyed, Mohannad Jooriah, David Rocha, João Almeida, Paulo Bartolomeu, Joaquim Ferreira, Carlos Rosales, and Marta Miranda. 2022. "CCAM Service Continuity in a Cross-Border Mec Federation Scenario." Frontiers in Future Transportation 3.

[13] TCPDUMP website. Available at https://www.tcpdump.org

[14] G-NetTrack website. Available at https://gyokovsolutions.com/g-nettrack/

[15] Intelligent Transport Systems (Its); Users and Applications Requirements; Part 2: Applications and Facilities Layer Common Data Dictionary. 2014. ETSI.

[16] Information Technology Asn.1 Encoding Rules: Specification of Packed Encoding Rules (Per). 2002. ITU-T.

[17] Intelligent Transport Systems (Its); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. 2014. ETSI.

[18] Intelligent Transport Systems (Its); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service. 2014. ETSI.

[19] 5G-PPP White Paper on Automotive Vertical Sector, October 2015

[20] ETSI TR 102 638 V1.1.1 (2009-06) Technical Report Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions

[21] ETSI TS 101 539-1 V1.1.1 (2013-08) Intelligent Transport Systems (ITS); V2X Applications; Part 1: Road Hazard Signalling (RHS) application requirements specification