2022 IEEE Intelligent Vehicles Symposium (IV)
June 5-9, 2022. Aachen, Germany

Cloud Assisted Connected and Automated Mobility System Architecture Design and Experimental Verification: The 5G-MOBIX Autonomous Truck Routing Use Case

Tahir Sari[1], Mert Sever[1], Arda Taha Candan[2], Gülsüm Tuba Çibuk Girgin[2], Emre Girgin[2], and, Mehmet Haklıdır[2]

2022 IEEE Intelligent Vehicles Symposium (IV) | 978-1-6654-8821-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IV51971.2022.9827077

Abstract— **In this study, potential usage of 5G networks for cloud assisted connected and autonomous mobility solution is demonstrated with preliminary field tests of Autonomous Truck Routing use case in H2020-ICT18 5G-MOBIX project. The preliminary tests are performed at Ford Otosan test track in Eskişehir, Turkey. The SAFIR cloud server developed by TUBITAK BILGEM is used over 5G network. The 5G-MOBIX Autonomous Truck Routing system architecture is composed of an autonomous truck, cloud and smart infrastructure having three road side units with LIDAR sensors. Navigation software stack is operated on cloud according to data received from autonomous truck and road side units. On-board unit performs required data transmission between cloud and truck. Motion control algorithms are executed to follow reference way points with a rapid prototyping controller in autonomous truck. Feasibility of the proposed connected and automated mobility system architecture is verified.**

I. INTRODUCTION

It is reported that 92% of the road crashes are mainly caused by human perception (driver inattention, distraction etc.) and decision making (driving too fast, adjusting inter vehicular distance wrong etc.) errors [1]. Moreover, poor driving styles have a negative impact on fuel economy [2]. Autonomous vehicles are expected to enhance road safety and fuel economy at the same time thanks to precise control algorithms and advanced perception, decision making algorithms. Capabilities of autonomous vehicles can be significantly improved by wireless technologies enabling vehicle-to-everything (V2X) communications [3]. Connecting autonomous vehicles with roadside infrastructures and networks, provides two major enhancement opportunity. Firstly, sensors and human driver have very similar field of views. Communicating with sensors over smart infrastructure networks provides significantly increased environmental awareness beyond on vehicle sensors line of sight [4], [5]. European Telecommunications Standards Institute (ETSI) has developed an ITS station architecture [6]. According to the ITS station architecture, the cooperative awareness is the most fundamental feature [7]. Secondly, the autonomous driving systems are computationally very demanding. An autonomous vehicle system may need to process a huge amount of data (as high as 2 GB/s)[8], since perception, decision making, planning and control algorithms deal with larger amount of sensor data and executes matrix-vector operations, numerical optimization algorithms in real time. Hence, the distributed computing architectures can be developed to offload on board vehicle processing unis [9], [10], [11], [12]. In this study, the focus is on Connected and Automated Mobility (CAM) system architecture design and its experimental verification with 5G network. The work carried out at the Ford Otosan Inonu Test Track in the scope of 5G-MOBIX, which is project funded by the H2020 research and innovation program.

A. Related Work

Kumar et al.[4], developed "Carcel" which is a cloud-assisted autonomous vehicle system. The Carcel utilizes sensor data from both vehicle and roadside infrastructure to detect objects beyond the vehicle line of sight. Cloud also receives planned path from vehicle to determine critical regions requires more or less resolution of 3D-point cloud sensor data. Hence, the limited bandwidth problem is tackled by request based communication strategy according to obstruction of planned path. Therefore, a communication system, CarSpeak, is introduced by Kumar et al. [5]. The content-centric design is adopted rather than message source/ID based approach. CarSpeak's information objects are regions around the ego vehicle. Hence, ego vehicle can prioritize according to its driving zone among real time stream of 3D-point cloud data along the whole road as proposed in Carcel [4]. Sasaki et al. [9] proposed infrastructure based autonomous vehicle control system having nodes at cloud server and mobile edge computing server. They aimed to free vehicles from costly sensors and prevent from deadlock situations in multi vehicle scenarios. The vehicle sensor data which is composed of position, velocity, yaw and ID directly sent to edge. The edge controller is able to achieve stable operation by PID based computing and forwarding vehicle control data (steering, braking and accelerating) directly to ego vehicle. If the network latency is lower than a threshold, the edge controller prefers to forward vehicle sensor data to

[1] Tahir Sari and Mert Sever are with Research and Development Center, Autonomous Driving Systems, Ford Otosan, 34885 Istanbul, Turkey tsari@ford.com.tr, msever1@ford.com.tr

[2] Arda Taha Candan, Gülsüm Tuba Çibuk Girgin, Emre Girgin and Mehmet Haklıdır are with the Robotics and Autonomous Systems Laboratory, TUBITAK BILGEM, 41470 Kocaeli, Turkey arda.candan@tubitak.gov.tr, tuba.girgin-a@tubitak.gov.tr, emre.girgin@tubitak.gov.tr, mehmet.haklidir@tubitak.gov.tr

cloud server. Then cloud server computes required vehicle control data to offload edge server. However, the cloud server fail to achieve stable operation when latency increased above threshold temporarily. Therefore, an automatic switching algorithm based on network latency is used to balance computational load between cloud and edge servers without unstable operation. In Sasaki et al. [10], the previous work is extended by analysing the effect of actual delay of internet with realistic fluctuation. In [9] and [10], remote control is considered without autonomous mode of each ego vehicle. Collapse of these solutions is inevitable when burst packet loss occurs. Hence, coordination of these remote control architectures with autonomous mode of each ego vehicle is proposed in Sasaki et al. [11]. The switching algorithm is developed for arbitration between remote and autonomous control modes according to network conditions in terms of latency and packet loss. Sasaki et al.[12] developed a cloud-assisted autonomous vehicle system under assumption of 5G networks. Self-localization with NDT matching, occupancy grid construction and path planning with A^* are performed in cloud. Receiving Lidar point cloud data, ground removal on point cloud and pure pursuit path following control tasks are implemented in vehicle side. ROS [13] and Autoware [14] are highly utilized in their study.

In the light of aforementioned studies, it is apparently seen that autonomous vehicles can benefit from cloud with several ways such as offloading by distributed computing, efficient platooning with smart infrastructure, cooperative perception in urban crossroads, cooperative planning in urban crossroads, simulation, HD map production and model training [15].

B. Motivation

In this study, cloud-assisted connected and autonomous mobility system architecture is proposed and experimentally verified. The 5GMOBIX Autonomous Truck Routing (ATR) system architecture is composed of an autonomous truck, road side units, 5G network and a cloud server. The autonomous truck is equipped with rapid prototyping controller unt (RPCU) and On-Board Unit (OBU). The on-board unit receives vehicle data (such as speed, longitude latitude, altitude and azimuth) and forwards to Cloud. Road-side units forward raw lidar data to the cloud. The cloud receives ego vehicle information and road-side unit information to perform navigation tasks such as object detection, localization and global/local planning. The RPCU performs lateral and longitudinal motion control functionalities. Through the study, as a cloud service it is highly benefited from the Safir Cloud service which is a next-generation cloud infrastructure and developed by TÜBİTAK BİLGEM. Safir Cloud's fault tolerant and elastic architecture enhancing its scalability allows it to be exploited for cloud based navigation applications.

The 5GMOBIX ATR system architecture combines some previous solutions from cloud-assisted autonomous mobilitiy literature. Firstly, the autonomous truck is freed from costly sensors such as lidar [9] by replacing lidar sensors at road-side units as proposed in [16], [17], [18]. Secondly, the RPCU is offloaded from computationally heavy navigation tasks such as object detection, occupancy grid construction, global and local path planning [12].

C. Rest Of the Paper

Rest of the paper is structured as follows. The system architecture is presented in Section II. Navigation stack with cloud computing is proposed in Section III. Motion control with on board computer is introduced in Section IV. Section V illustrates the filed testing results. Conclusions are given in Section VI.

II. System Architecture

The 5G-Mobix ATR system communication architecture is mainly composed of on-board unit, road side units and cloud layers. The architecture is depicted in Figure 1.

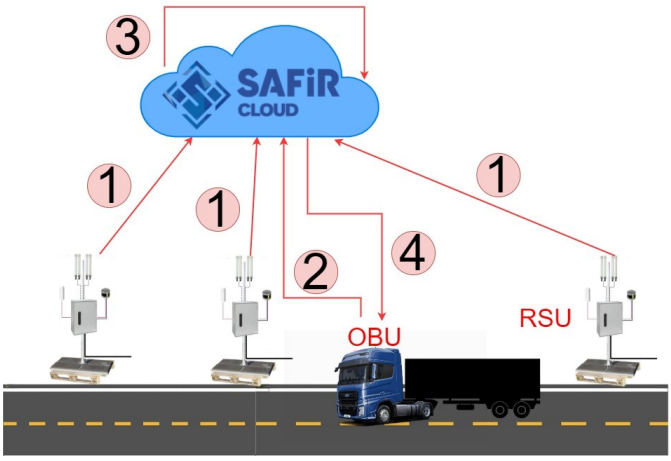

Fig. 1. Autonomous Truck Routing communication architecture

For this study, three Road-Side Units (RSU) that are developed by IMEC company are deployed on the field. Each RSU contains all the V2X communication hardware that is needed to foresee the connectivity over 5G (Quectel RM500Q) and C-V2X PC5 (Cohda MK6c). A 4G modem is included for remote configuration and monitoring. Time synchronization purposes a Global Navigation Satellite System (GNSS) in integrated. Additionally, each RSU has PCEngines APU3D processing unit and Velodyne VLP-32 LIDAR.

The cloud based data acquisition is performed in four main steps. Firstly, collected raw data from LIDARs is sent to Safir Cloud as pointed out with data flow arrow (1) in Figure 1. Meanwhile, OBU in vehicle collects vehicle position, speed and heading data from dSPACE MicroAutoBox II rapid prototyping control unit and sends it to Safir Cloud as denoted with (2) in Figure 1. Then, Safir Cloud listens specified UDP ports, sync. messages, creates point cloud and calculates required reference way points for truck as shown by (3) in Figure 1. Finally, calculated reference way points are sent to truck (4). The 5G-Mobix Autonomous Truck Routing system performs with full functionality when

the autonomous truck follows received reference way points with longitudinal and lateral motion control algorithms. Functionality of the 5G-Mobix Autonomous Truck Routing is provided by cloud and edge sensing based ultra-fast and ultra-reliable communication over 5G. Hardware architecture of the autonomous truck is shown by Figure 2.

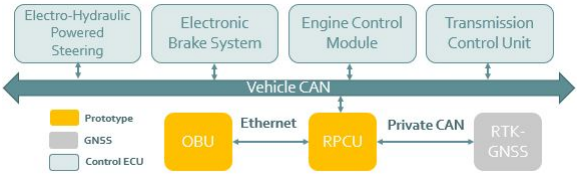

Fig. 2. Vehicle subsystem and hardware architecture

Truck is equipped with electro-hydraulic powered steering system, Novatel PwrPak7D-E1 RTK-GNSS for 1 cm precise positioning, dSPACE MicroAutoBox II rapid prototyping controller unit (RPCU) for motion control algorithms (longitudinal and later controllers developed by Ford Otosan engineers) and UDP communication functionalities, and 5G on-board unit.

The RPCU collects all related data from vehicle ECUs (such as electronic brake system, engine control module and transmission control units). Then it performs as a gateway unit to exchange messages between on board unit. The OBU forwards the vehicle data from RPCU to Safir Cloud and forward required navigation data from Safir Cloud to RPCU. The on-board unit (OBU) is developed by IMEC company [19]. OBU has similar features as IMEC – RSUs. The OBU unit is installed inside of the vehicle. All the antennas of the 5G, 4G and the C-V2X PC5 modules have standard SMA connections and magnetic mounts so that they can be installed outside the vehicle for optimal transmission and reception.

The message content of the communication between vehicle and cloud is given in Table I. Here, truck routing status and system state messages are published to initiate 5G-Mobix ATR functionality and to observe autonomous truck status. Emergency Stop Status is triggered to indicate sudden braking. Reference way points and reference speed are computed by navigation stack in cloud. Longitude, latitude, altitude (LLA) and azimuth values belong to origin are transmitted for the required transformations of Local Coordinate Frame (LCF). Vehicle speed, vehicle position (LLA) and vehicle heading information are published to be utilized in localization functionality of the navigation stack. The message counters are monitored to detect any freeze/timeout issue.

TABLE I
COMMUNICATION MESSAGE CONTENT

Vehicle to Cloud	Cloud to Vehicle
System State	Truck Routing Status
Time	Time
Emergency Stop Status	Reference Waypoints [X,Y]
Vehicle Speed	Reference Speed
Vehicle Position (LLA)	LLA of LCF Origin
Heading	Azimuth of LCF Origin
Message Counter	Message Counter

III. NAVIGATION

In this section, localization and path planning algorithms running on the Safir Cloud are described. The the cloud application uses global path planning, local path planning and object detection algorithms to generate the waypoints required for autonomous driving of the vehicle by combining the data from the vehicle and road-side units in the ROS environment. The navigation architecture of ATR is shown in the Figure 3.

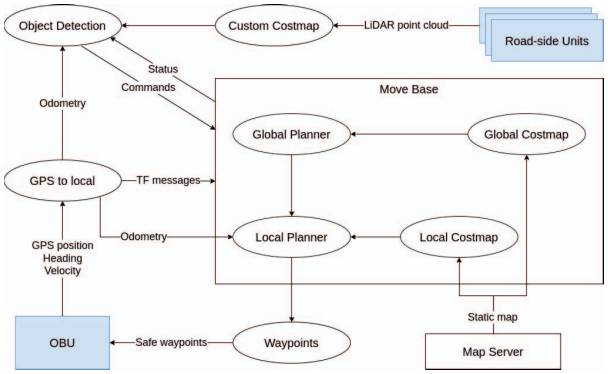

Fig. 3. Navigation architecture of ATR

A. Localization

Localization is a critical perception task that involves estimating the position of the vehicle in relation to a reference coordinate system. A high level of localization accuracy allows an autonomous vehicle to understand its surroundings and establish a sense of the road and lane structures. The fact that the sensors are located on the road-side rather than on the vehicle for the purpose of this study has brought a challenge in this manner. To solve this, a GPS position is selected as a reference point. In order to convert all GPS points to local coordinates in the cloud, the geodesic between this reference point and the desired point is calculated and converted to east north up (ENU) local points as x, y. Finally, the local points published in the ROS environment. In this way, the vehicle odometer and sensor data are fused in the ROS environment on the local map.

B. Global Planner

The global path planning is conducted on a static global map which is a 2D grid-like map whose cells have higher values for the stationary obstacles like walls and trees and lower values for the collision-free regions. Based on this global cost map, a graph structure can be formed to exploit traditional shortest path algorithms on it. The path found in this undirected graph acts as global guidance to the agent.

In our study, for the path planning part, the widely used ROS package *move_base* [20] is utilized for the navigation stack. Its global planning part, the package named *navfn*,

takes the static cost map and computes a path to be fed to the local planner. The well-known Dijkstra's shortest path algorithm [21] is adopted to create a global path by *navfn*. However, note that the edge construction process of the graph is not that trivial because of the nonholonomic constraints of the vehicle. For instance, trucks can not perform lateral movements and their turning radius strictly depends on the wheelbase and track width of it. The *move_base* package takes this metrics into account as well. However, the global path planner is not robust to the dynamic objects that were not originally on the static map and may lead to navigation problems such as freezing robots.

C. Local Planner

Local planner is an algorithm that aims to optimize the global plan for each cycle according to the vehicle's dynamics and sensor data. In our study, *teb_local_planner* [22], a ROS plugin which is compatible with vehicles with differential drive, is used.

teb_local_planner is an algorithm based on a method called "time elastic band" which is rests upon the augmentation of classic "elastic band" theory [23]. The elastic band deforms the global path with respect to the shortest path and object avoidance. But while doing this, it does not take into account any dynamic constraints of the robot (in this case the vehicle). *teb_local_planner*, on the other hand, considers dynamic constraints such as the limited velocity and acceleration of the robot, ensuring that the target is reached in a minimum time. With the extension [24], the algorithm is also optimized for car-like robots. Since the dynamic constraints of the trucks are challenging, the use of *teb_local_planner* was found appropriate in this study.

The $25m$ long road resulting from the local planner is converted into 250 (x, y) waypoints with $0.1m$ intervals. The calculated velocity coming from *move_base* is also added to the waypoints information and sent to the OBU continuously, enabling autonomous driving for obstacle-free situation.

D. Object Detection

The ROS package *move_base* is used for path planning and navigation. The modules in the *move_base* package are designed in a way that the agent never stops until it reaches the goal. Furthermore, the agent is programmed to avoid dynamic obstacles and find other feasible paths continually. After finding a feasible way, the agent continues on its new path without cease. However, this solution is not applicable to the problem depicted in this paper. Therefore, a custom ROS package named as *obstacle_detection* is implemented to detect obstacles and provide solutions.

Obstacle detection module gets the locations of the agent and the obstacle points. Then a 2-D imaginary cone is created, starting from the base of the agent. The angle and the length of the cone are provided by related configuration files. After that, the obstacle points inside the imaginary cone are traversed. The module adjusts the speed of the agent according to the distance between the base of the agent and the nearest obstacle point. If the distance is shorter than the minimum permitted distance for routing, then an emergency stop signal is sent to the *move_base* controller. The agent waits until the obstacle point moves away. Then the agent continues routing on its path to the goal location.

The slow threshold and stop threshold are chosen as 12 meters and 7 meters, respectively. Angle of the 2-D cone is set as 40 degrees.

E. Unit Testing

Unit tests of the navigation stack were performed by simulating the movement of the vehicle. During the tests, velocity and path followed by vehicle were observed while making a U-turn on an obstacle-free road. Maximum velocity that the vehicle can reach is determined as $10km/h$. Results are shown in the Figure 4. In the top plot, the path followed by the vehicle during the tests is shown. In the bottom plot, the planned velocity for the vehicle along the path is shown. It is seen that the velocity of the vehicle varies between $7km/h$ and $10km/h$ when it is going straight. In addition, at two inflection points of the turn it is seen that the average velocity decreases depending on the dynamic constraints of the vehicle.

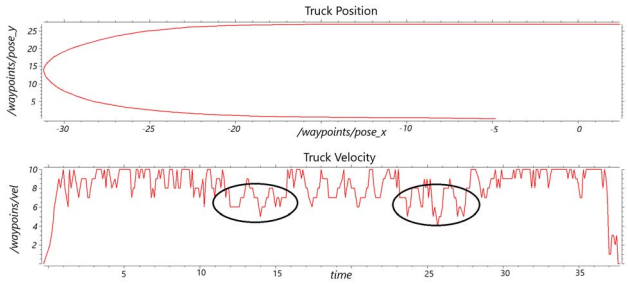

Fig. 4. Unit test results of cloud based path planning

IV. Motion Control

In this section, lateral and longitudinal control algorithms implemented in test truck are described. Thereafter, unit testing results for velocity tracking and path following are presented.

A. Lateral Controller

The border side corridor crossing problem is a low speed use case. Hence, the pure pursuit [25] algorithm is preferred for lateral control due to its simplicity and its well-known path following accuracy at low speeds. The algorithm is based on geometric interpretation of a circular arc which connects the rear axle center and an intersection point (look ahead point) on reference way points. The look ahead point is determined by searching for a reference waypoint at look ahead distance, L_d, away of rear axle center as illustrated in Figure 5.

Then, the vehicle's steering angle δ can be determined using only the look ahead point location and the angle α between the vehicle's heading vector and the look-ahead

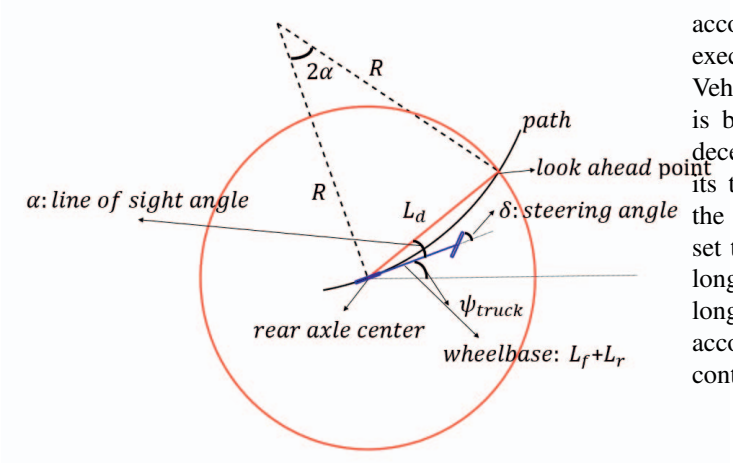

Fig. 5. Geometry of pure pursuit

vector. Applying the law of sines to Figure 5 results in

$$\frac{L_d}{sin(2\alpha)} = \frac{R}{sin(\frac{\pi}{2}-\alpha)} \tag{1}$$

Using the simple geometric bicycle model of an Ackermann steered vehicle, the steering angle can be written as

$$\delta = atan\left(\frac{2(L_f+L_r)sin(\alpha)}{L_d}\right) \tag{2}$$

where L_d is the look ahead distance, L is the wheelbase value and α is the angle between truck heading and look ahead vector heading.

B. Longitudinal Controller

The longitudinal controller is designed to track reference speed commands received from cloud. The reference speed tracking control is performed by switching between torque and deceleration control modes. The switching logic is shown by Figure 6. The reference speed tracking task is performed

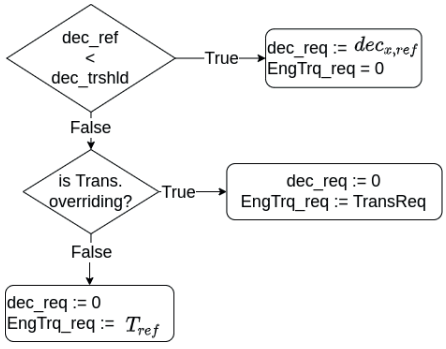

Fig. 6. Flow chart of switching between torque and deceleration modes.

by collaboration of three different subsystem which are engine control module (ECM), transmission control unit (TCU) and electronic brake system (EBS) in subject vehicle. Torque mode is executed by requesting Engine Torque from ECM over vehicle CAN line. The requested engine torque value can be set to longitudinal controller output or TCU output according to above mentioned logic. Deceleration mode is executed by requesting deceleration value from EBS over Vehicle CAN line. When the deceleration reference value is below its threshold value, EBS is requested to perform deceleration control. If the reference deceleration is above its threshold, TCU overriding status must be monitored. If the TCU is overrinding Engine Torque, requested torque is set to TCU output. Otherwise, the requested torque is set to longitudinal controller output. In the torque control mode, the longitudinal controller reference torque value is computed according to reference acceleration value computed by PID controller which is given by:

$$acc_{x,ref} = K_p e_v + K_d \dot{e}_v + K_i \int e_v \tag{3}$$

where $e_v = v_{x,req} - v_x$ is the velocity tracking error. The reference acceleration value is utilized to calculate reference engine torque value as follows:

$$T_{ref} = T_{ff} + T_{elc} + T_{gspid} \tag{4}$$

where T_{ff} denotes the feedforward term, T_{elc} presents engine loss compensation term and T_{gspid} stands for gear scheduled pid term. The feedforward torque term is nothing but longitudinal dynamics calculations which are commonly used in literature [26]. Engine loss term, $T_{e,loss}$, is modeled by a 1-D look-up table. Here, EL1Dtable stands for the 1-D look-up table. The only input is ω_e which denotes engine speed in rpm. The gear scheduled PI term is $T_{gspid} = K_p e_a + K_i \int_{reset} e_a$. where $e_a = acc_{x,ref} - a_x$ is the acceleration tracking error. K_p and K_i are proportional and integral terms of gear scheduled PI controller. Both terms are in the form of 1D look-up table with current gear as breakpoints. The reference deceleration value is computed by $dec_{x,ref} = \text{VelErr2Dec1Dtable}(e_v)$.

C. Unit Testing

In this section, both lateral and longitudinal controllers are verified by unit testing. The unit testing studies are conducted at Ford Otosan Eskişehir Test Track which is shown in Figure 9. Path following performance of the pure pursuit lateral controller is shown in Figure 7. Here, the reference way points are generated according to actual geometry of Greece-Turkey border side roads. At the bottom plot, black dots represent reference way points blue dots represent reference way point detected at look ahead distance and red line show actual truck position in local cartesian coordinates. At the top plot, red line denotes path following error in meters, dashed blue line stands for the error bound with 0.50 m, dashed black line shows the error bound with 0.25 m. As can be observed from Figure 7, reference way points are followed accurately. The algorithm detects the corresponding way point at a look ahead distance away from rear axle center, then the steering wheel angle is computed according to equation (2). Since the truck is initiated outside of reference way points, path following error starts at 5.303 m. However, it is apparently seen that path following error is reduced below 0.50 and 0.25 meter bounds, respectively. Common width of any lane is 3.5 meter. The test truck

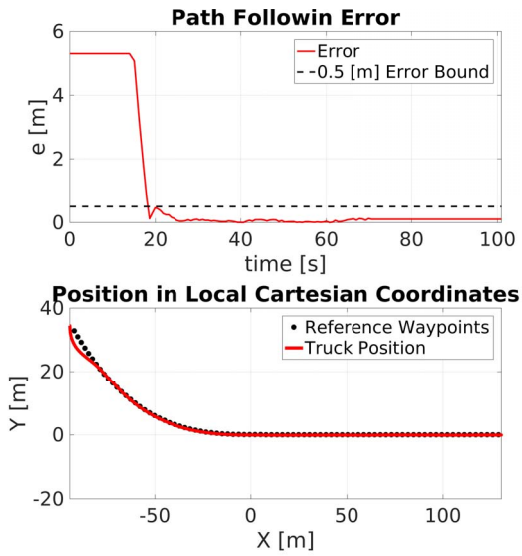

Fig. 7. Lateral controller performance in terms of path following accuracy

has 2.5 meter width. Therefore, 0.50 meter is maximum available path following error without violating its lane. The lateral controller satisfies the requirement with keeping path following error below 0.25 meter along the reference way points. Speed tracking performance of the cascaded longitudinal controller is shown by Figure 8.

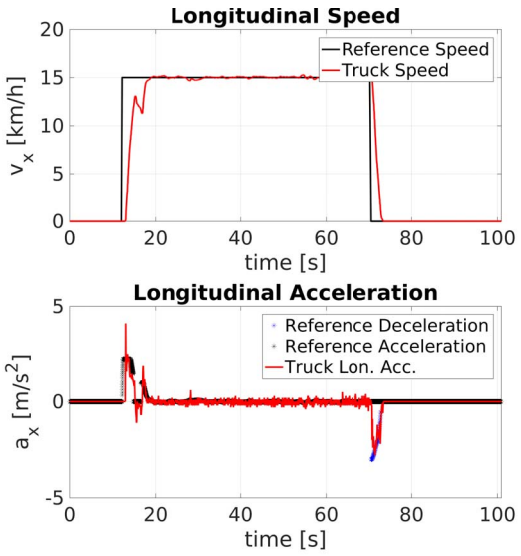

Fig. 8. Longitudinal controller performance in terms of reference speed tracking and acceleration/deceleration tracking

At the top plot, black line represents reference speed value, red line represents actual truck speed. At the bottom plot, black dots denote reference acceleration, blue dots denote reference deceleration and red line stands for actual longitudinal acceleration of the Truck. It is apparently seen that reference speed value 15 $[kph]$ is successfully reached by cascaded longitudinal controller. There is a small temporarily oscillated response around 11 $[kph]$ due to gear shift process. The small transient oscillation in the truck speed response is occurred due to the rapid decrease in reference acceleration value shown in bottom plot. The reference acceleration value is set to zero during gear shift, since acceleration tracking performance of any longitudinal controller can not be guaranteed during gear shift. When the reference speed value suddenly decreased to zero as shown in top plot, it is observed that reference deceleration value is reached approximately -3 $[m/s^2]$. Tracking reference deceleration value is achieved by EBS system as can be seen from bottom plot. Then the truck speed is quickly converged to zero and standstill state is reached.

V. Field Testing

In this section, integrated testing of motion control with navigation stack on Safir Cloud is presented. Unit testing of the navigation stack is previously presented in Figure 4. Unit testing results of motion control are presented in Figure 7.and Figure 8. The integrated testing studies are carried out at Ford Otosan Eskişehir Test Track which is shown by Figure 9.

Fig. 9. Ford Otosan Eskişehir test track.

Performance of the Autonomous Truck Routing system in terms of path following and reference speed tracking can be seen from Figure 10.

As can be seen from Figure 10, integrated testing results are compatible with unit testing studies. The maximum allowable path following error bound 0.50 meters is violated at the beginning. Thereafter, path following error is always kept under 0.50 meters. The reference speed tracking is also successfully achieved. Here reference speed and reference way points are generated by global and local planners in navigation stack. The Autonomous Truck Routing system architecture of 5G-Mobix project is experimentally verified by these preliminary test results before actual border crossing tests between Turkey and Greece.

VI. Conclusions

In this study, feasibility of the proposed connected and automated mobility system architecture is demonstrated. However, there are some further enhancement opportunities. The path planning approach is based on a graph structure with static objects and collision-free regions. When the unit testing and integrating testing results are compared,

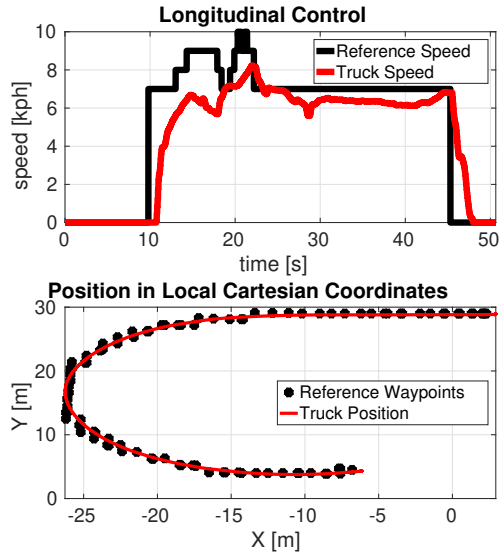

Fig. 10. Integrated test results of lateral controller with cloud based path planning

it is shown that current path planner results discontinuous curvature profiles. HD map like strategy having smooth lane center line and related attributes might be beneficial. 5G-MOBIX project aims to demonstrate potential of different 5G features on real European roads and highways and create and use sustainable business models to develop 5G corridors. The preliminary testing studies must be repeated at Ipsala(Turkey) border crossing point between Greece and Turkey. Hence, performance of the whole system architecture in terms of connectivity and motion control can be further investigated. In addition, motion control algorithms are not tested with semitrailer case.

ACKNOWLEDGMENT

The work presented in this paper is part of the European Project 5G-MOBIX. This project is funded by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825496. The content reflects only the authors' view and the European Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] NHTSA, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," *National Highway Traffic Safety Administration, Washington, DC: US Department of Transportation*, 2015.

[2] J. C. Ferreira, J. de Almeida, and A. R. da Silva, "The impact of driving styles on fuel consumption: A data-warehouse-and-data-mining-based discovery process," *IEEE Transactions on intelligent transportation systems*, vol. 16, no. 5, pp. 2653–2662, 2015.

[3] U. Montanaro, S. Dixit, S. Fallah, M. Dianati, A. Stevens, D. Oxtoby, and A. Mouzakitis, "Towards connected autonomous driving: review of use-cases," *Vehicle system dynamics*, vol. 57, no. 6, pp. 779–814, 2019.

[4] S. Kumar, S. Gollakota, and D. Katabi, "A cloud-assisted design for autonomous driving," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 41–46.

[5] S. Kumar, L. Shi, N. Ahmed, S. Gil, D. Katabi, and D. Rus, "Carspeak: a content-centric network for autonomous driving," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 259–270, 2012.

[6] E. T. S. Institute, "Intelligent Transport Systems (ITS); Communications Architecture," *European Standard (Telecommunications Series),Sophia-Antipolis, France*, 2010.

[7] ——, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," *European Standard (Telecommunications Series),Sophia-Antipolis, France*, 2019.

[8] S. Liu, J. Tang, Z. Zhang, and J.-L. Gaudiot, "Computer architectures for autonomous driving," *Computer*, vol. 50, no. 8, pp. 18–25, 2017.

[9] K. Sasaki, N. Suzuki, S. Makido, and A. Nakao, "Vehicle control system coordinated between cloud and mobile edge computing," in *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*. IEEE, 2016, pp. 1122–1127.

[10] ——, "Layered vehicle control system coordinated between multiple edge servers," in *2017 IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 2017, pp. 1–5.

[11] K. Sasaki, S. Makido, and A. Nakao, "Vehicle control system for cooperative driving coordinated multi-layered edge servers," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*. IEEE, 2018, pp. 1–7.

[12] Y. Sasaki, T. Sato, H. Chishiro, T. Ishigooka, S. Otsuka, K. Yoshimura, and S. Kato, "An edge-cloud computing model for autonomous vehicles," in *In 11th IROS Workshop on Planning, Perception, Navigation for Intelligent Vehicle*, 2019.

[13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[14] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–68, 2015.

[15] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1697–1716, 2019.

[16] J. Wu, H. Xu, and J. Zheng, "Automatic background filtering and lane identification with roadside lidar data," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.

[17] J. Zhao, H. Xu, H. Liu, J. Wu, Y. Zheng, and D. Wu, "Detection and tracking of pedestrians and vehicles using roadside lidar sensors," *Transportation research part C: emerging technologies*, vol. 100, pp. 68–87, 2019.

[18] J. Zhao, H. Xu, J. Wu, Y. Zheng, and H. Liu, "Trajectory tracking and prediction of pedestrian's crossing intention using roadside lidar," *IET Intelligent Transport Systems*, vol. 13, no. 5, pp. 789–795, 2019.

[19] D. Naudts, V. Maglogiannis, S. Hadiwardoyo, D. Van Den Akker, S. Vanneste, S. Mercelis, P. Hellinckx, B. Lannoo, J. Marquez-Barja, and I. Moerman, "Vehicular communication management framework: A flexible hybrid connectivity platform for ccam services," *Future Internet*, vol. 13, no. 3, p. 81, 2021.

[20] S. Pütz, J. S. Simón, and J. Hertzberg, "Move base flex a highly flexible navigation framework for mobile robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3416–3421.

[21] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[22] C. Roesmann, W. Feiten, T. Woesch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*, 2012, pp. 1–6.

[23] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 1993, pp. 802–807 vol.2.

[24] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5681–5686.

[25] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, Tech. Rep., 1992.

[26] U. Kiencke and L. Nielsen, *Automotive control systems: for engine, driveline, and vehicle*. IOP Publishing, 2000.