Collaborative Navigation Strategies Based on Vehicles Features in Connected Environments

Charlotte Beaune, Elwan Héry, Vincent Frémont, Antonio Sgorbissa, Carmine Recchiuto

Collaborative Navigation Strategies Based on Vehicles Features in Connected Environments

Charlotte Beaune[1] , Elwan Héry[1], Vincent Frémont[1]
LS2N (UMR CNRS 6004)
École Centrale de Nantes, France

Antonio Sgorbissa[2] , Carmine Recchiuto [3]
DIBRIS
Università degli studi di Genova, Italia

***Abstract*—In the intelligent vehicles community, exploring and navigating through urban environments represent one of the greatest challenges to overcome. This challenge can be addressed using Vehicle-To-Everything technologies which enable connected vehicles to communicate with each other and with other actors in the scenario such as smart infrastructures and pedestrians. In this work, we propose a communication strategy among actors in urban scenarios to exchange useful data to improve their performance in achieving their respective goals. Our approach is inspired by communities of interest in human groups, where members share data on a common topic. As actors are organized as communities, the data transmitted among actors is more efficient and thus reduces the overall communication effort. To demonstrate the effectiveness of this communication strategy, we propose a case study to explore its influence on the navigation behaviours of a fleet of autonomous vehicles and the decrease of the total bandwidth, especially in dynamic areas with traffic jams. We show that, when performing navigation in simulated environments, it reduces by seven times the average number of messages exchanged among actors while maintaining the same efficiency in navigation toward vehicles' respective goals. Our method is also validated in real-world experiments, in an unstructured environment with a fleet of mobile robots. For more information, the video of the experiments is available at https://youtu.be/Dp4LxykEGAY**

***Index Terms*—Collaborative Navigation, Vehicle-To-Everything Communication, Autonomous Vehicles, Vehicular Ad-hoc Networks, Conditional Transmissions**

I. Introduction

Urban scenarios are a challenge for autonomous vehicles (AVs) navigation because of the mixed-traffic situations in the form of shared areas between AVs and human drivers, pedestrians and other actors [1]. This is one of the reasons why, using the SAE (Society of Automotive Engineers) 6-level scale of automation [2], [3], [4], transitions from vehicles led by human (level 0 to 3) to led by machine only (level 4 and 5) is challenging to realize.

However, a number of prior works explore the hypothesis that some of these difficult situations, such as platooning or navigating with intersections and roundabouts, can be overcome with the help of collaborative navigation [5], [6]. This connectivity between vehicles is increasingly conceivable, because the urban environments tend to be more and more connected thanks to the wide coverage of mobile communication such as WiFi and 4G/5G. For this use the IEEE Intelligent Transportation System Society conceived standard messages [7] to better organize these communications between AVs.

Contributions of this paper include a collaborative high-level navigation strategy relying on Vehicle To Everything (V2X) communication network. We worked with a focus on building a VANET (Vehicular Ad-Hoc Network) [8] relying on these V2X communications, efficient yet simple to scale up on other driving scenarios, based on similarity features among vehicles. We used a centralized communication framework, based on the REST (Representational State Transfert) [9] to send data to a centralized server and retrieve alert messages and useful data within features-based groups of vehicles. Our method is tested in both simulated urban world, a dense, structured and dynamic environment with traffic jams avoidance and rerouting scenarios. This paper also includes technical contribution on building a network strategy, in real-world experiments in an open environment, dealing with real-world communication and navigation challenges.

The rest of the paper is organized as follows: Section III provides background knowledge to build a collaborative communication method based on features of fleet of vehicles, along with a case study related to navigation and path planning in a dynamic environment. Section IV presents details on the implementation of the method with the help of the *flask* library [10], that allows us to develop a RESTful [9] server in a Python-based framework. We will also discuss the results of our method in a simulated CARLA (Car Learning to Act) environment [11], a dense, dynamic and structured simulation environment used for navigation tasks. Section V presents the implementation of the method on ground mobile robots and the results of the experiments.

II. Related works

Many studies considered collaborative strategies in intelligent transportation systems as a must-have, all the more because smart cities can afford fast, reliable network connection [5]. Relying on mobile communications processes (4G/5G), communication architectures and standard messages (namely IEEE 802.11p [12] [7]), VANETs are used to broadcast alert messages or useful data among fleets of actors and build efficient, collaborative navigation strategies for AVs.

[1]firstname.lastname@ls2n.fr
[2]antonio.sgorbissa@unige.it
[3]carmine.recchiuto@dibris.unige.it

In [13], the authors solely focus on disseminating data to all vehicles within a restricted area, without taking into account data flooding that may occur in dense and dynamic environments. This will be our baseline when conducting simulation experiments. Exploiting different communication architectures and data dissemination strategy, depending on the context is the key to out-perform the baseline and build more efficient collaborative strategies.

As in [14], our method focus on alert-dependent area of communication. In terms of network resources, the less data is sent, the better the communication system will be to handle multiple actors. Main focus is to minimize the bandwidth usage when dealing with an obstacle in urban scenarios. Authors propose a simulated environment/real experiments with 10 vehicles in an uncongested area. In our simulated scenario, we present urban environment experiments involving 30 vehicles in a crowded space, in order to emulate traffic jam conditions. We also rely on a centralized data server to gather useful information and then dispatch the processed alert signals to the concerned groups of vehicles, thus reducing the bandwidth usage.

Similarly, as in [15], we design a simulation scenario where vehicles must go to a certain goal location, in a crowded area with traffic jams.The goal is to send to the concerned vehicles ad-hoc messages in order for them to re-route and thus avoiding obstructed lanes. In this work fleet topology relies only on geometrical features and also works in a shrunken area, with all vehicles having the same goal. In our work, we try to achieve the same goal using the CARLA simulator, building a more complex environment, thus requiring a more complex strategy.

Our approach is more similar to [16], where the authors propose a fleet creation topology based on opportunistic similarities among actors, mainly geographic proximity based metrics as well as contact duration and future driving lane indicator. As their method is based on vehicle-to-vehicle communication, authors design a leader-follower scheme to process and send alert messages to the rest of the fleet. For our work, we design a vehicle-to-everything communication model that is able to deal with different fleet creation metrics. Our metrics include geometrical but also goal-related metrics or future motion prediction similarities, that we called membership conditions, allowing us to build *communities* of driving actors.

III. Building a collaborative communication method based on vehicles features

A. Communities of Driving Actors

We define communities of vehicles as a group of vehicles that share a common state, goal, or intentions at a given time. For examples, all vehicles moving along the same lane and that might be stuck in the same traffic jam cluster or cars that are looking for a parking place in the same area. Based on the way communities are created, they can overlap with each other depending on the rules and conditions defining the membership of a given vehicle to a given community. Relevant information will be shared only among members of same communities. For example, if two vehicles are not close to each other, and therefore belong to different communities, they will not exchange local information such as local obstacles in the way. But the idea does not exclude AV from one community (e.g. the "non-parking" one) to communicate with another community to exchange relevant information. For instance, a vehicle that finds a free parking place may share the information in the "looking for a parking place" community, even if it does not belong to this one. As opposed to [14], our method can deal with various types of vehicular groups and not only distance-based groups, allowing heterogeneity of supported actions The manner in which data is exchanged within and between communities also depends on the driving approach we aim to implement. Collaborative driving can be more or less "ethical" (users exchange useful information as a sign of courtesy [17]) depending on how much drivers are willing to exchange information. In this paper, the focus is set on only exchanging useful data, also known as conditional transmission [18]. This approach will prevent the overflow of communication channels with unnecessary data among all AVs in a fleet.

Thus, given a fleet of vehicles V which is defined as:

$$V = [v_1, \ldots, v_n] \tag{1}$$

With vehicle v_i detecting m features:

$$\bar{f}_i = [f_{1,i}, \ldots, f_{m,i}] \tag{2}$$

We can define the membership of the vehicle in the community G_j as:

$$G_j = \{v_i \in V, g_j(\bar{f}_i) \leq 0\} \tag{3}$$

Here, function $g_j(\bar{f}_i)$ defines the membership condition for a vehicle i to be part of the community G_j. As an example, when defining the community $G_{ego}^{vicinity}$ of vehicles in the vicinity of ego-vehicle v_{ego}, the membership condition is:

$$g_e = d_{v_{ego}}(\bar{f}_i) = \|X_{ego} - X_i\|_2 - R_{max} \tag{4}$$

with $X_{ego} = [x_{ego}\ y_{ego}\ z_{ego}]^T \in \bar{f_{ego}}$, $X_i = [x_i\ y_i\ z_i]^T \in \bar{f}_i$, positional features respectively of the ego-vehicle v_{ego} and vehicle v_i, $\|.\|_2$ the L2-norm and R_{max} the maximum distance from ego-vehicle we check for members.

B. Communication Strategy

In this work, we focus on the architecture of the communication strategy, thus we do not discuss the choice of data or exchange frequency.

A decentralized approach for communication of AV is based on the exchange of data without the help of a global storage of information. Each vehicle is considered as independent in terms of data management, it can send and receive information to all other actors but this ad-hoc approach demands additional computation from all the vehicles. Methods have been determined to reduce the computational cost but the results are not always satisfying [19]. However decentralized approach can deal with communication breaks as it relies on multiple sources of information.

A centralized approach for communication between vehicles, where data flow through a central repository - a server on the Cloud for example - can handle more data from more vehicles. In fact, the global monitor can separate all data based on their provenance and timestamp. By gathering information from vehicles, it becomes easier to reconstitute the global environment. This architecture can also handle heterogeneous vehicles and intelligent infrastructures. The drawback is that information flow to the master needs to be fully available in comparison with the decentralized approach.

In this work, we assume that a wireless communication is always available and that load of messages sent and received by the server is the same for all vehicles. Therefore, we choose to use a centralized way of exchanging messages. This enables processing numerous communities of vehicles as we can use several criteria to cluster them. The overall communication strategy is depicted in Fig. 1.

Accordingly to the REST API web service approach, the server is only considered as a container for the data: information exchanges will be triggered by actors that will send a proper request to the server through REST APIs. Information is accessible via a Uniform Resource Identifier (URI) provided by the *flask* environment. Messages from vehicles are sent to their reserved URIs (defined by their identifiers), while the server gathers all data from the fleet and groups it based on this information. Each community is represented as a list of identifiers, corresponding to the vehicles that belong to the community. Therefore, if a vehicle sends a warning message or navigation message, the server stores it to topics reserved to the according communities and then, members vehicles access it. Depending on messages received vehicles can then adapt their navigation behaviour.

With the community-based strategy, we can determine if a vehicle v_i needs to get the message from our ego-vehicle. A Boolean function triggers message sending. This function $f_{aff}(v)$ equals 1 when v_i needs to receive data from v_{ego}, and 0 otherwise :

$$f_{aff}(v_i) = \begin{cases} 1, & v_i \in \bar{G_{ego}} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

with,

$$\bar{G_{ego}} = \bigcap_{v_{ego} \in G_j} G_j \tag{6}$$

IV. Experiments in a Simulated Environment

In dynamic urban environments, we try to understand how collaborative navigation can change the behaviour of our fleet of vehicles in the presence of uncertainties such as a traffic jam or a stopped vehicle in the way. Our hypothesis is that our collaborative strategy between vehicles will reduce the average number of requests to the server while keeping an acceptable execution time to reach pre-defined goals.

A. A Case Study: Re-planning a path in presence of traffic

Several situations can benefit from the collaborative communication system we propose. For example, Fig. 2 represents a simple situation of crossing roads. The red car is considered to be our ego-vehicle v_{ego} that is blocked by an obstacle on the lane. This car belongs to two different communities: Vehicles that have their position in the vicinity of the green circle (range around the ego-vehicle), represented in green, and the community of vehicles that want to go in the lane blocked. Sending information that the lane is blocked is not relevant for all the vehicles in the fleet, only the vehicles that belong to the intersection of these two communities will receive the warning message.

Following this, a case study is navigation in dynamic environments where traffic jams can appear and affect the traffic flow. Most navigation systems are based on the exploration and navigation of graphs with possible positions in the environment are represented as nodes of an oriented graph. The fastest path in a graph can be computed using algorithms such as Dijkstra [20] or A* algorithm [21]. Usually, graph relies on weighted edges, that correspond to the distance between two nodes. This weight then allows the A* algorithm to find the closest path between two nodes according to a specified metric. When a lane is occluded by an obstacle and without a method or a possibility to perform lane changes, the autonomous vehicle stops in front of the obstacle. Exchanging information before sensing the obstacle in connected environments allows vehicles in a fleet not to be blocked behind it and to change their plan without encountering the blocked lane. Therefore, time spent in traffic jams for example can be reduced and allows the traffic to be smoother in crowded and dynamic environments. In order to force other agents to

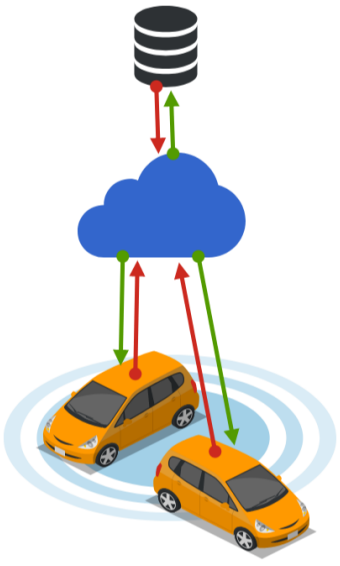

Fig. 1. Overall centralized communication strategy with a cloud-like server. Red arrows represent communication from server to actors and master and green ones from actors and master to server.

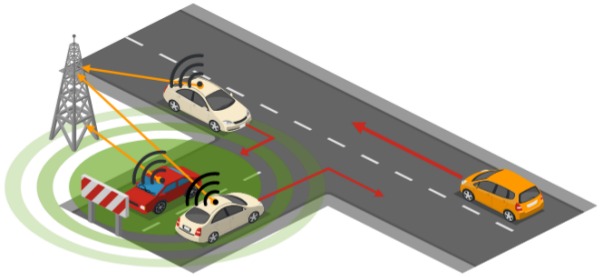

Fig. 2. Use case for collaborative navigation in a dynamic environment: Defining the different communities of vehicle where belongs the ego-vehicle (in red).

change their trajectory, we can perform changes within the graph of the possible waypoints. In our example, we use the defined community "same path" to define vehicles that share common waypoints in their path. For that, we use the Longest Common String (LCS) algorithm [22] to determine if two vehicles belong in the same community. To each actor is attached a path in the form of a list of registred waypoints and the LCS algorithm returns the size of the longest sub-string shared by the two of them. Defining a threshold for the number of shared waypoints n then allows us to parametrize the community. In our implementation we use $n = 15$ with a distance between waypoints corresponding to $d = 0.2$ m. When the ego-vehicle needs to send a warning message, the vehicles that belong in the G_{ego} receive it, with:

$$G_{ego} = G_{ego}^{LCS} \bigcap G_{ego}^{vicinity} \tag{7}$$

$$G_{ego}^{LCS} = \{v_i \in V, g_{ego}^{LCS}(v_i) \leq 0\} \tag{8}$$

$$g_{ego}^{LCS}(v_i) = n - LCS(path_{ego}, path_{v_i}) \tag{9}$$

As mentioned in [12], in theory the WiFi 802.11p range (mostly used for V2V communication) is 2 km and can widely vary depending on the conditions. Practically, in ideal condition (no blocked signals, no canyon effect), the range is up to 500 m in cities. In [23], authors have chosen to limit the range to 200 m when navigating in urban environments. In our method, as we exchanged information about obstructed lanes, that our city map is smaller than the one used in [23], and the vehicle's speed is limited to 30 km/h, we decided to limit the range of communication to 20 m, with the vicinity community membership defined in (4). As the environment is crowded, and streets are small, when testing with large fleet of vehicles, this range is sufficient to access concerned vehicles when sending emergency messages.

B. Simulation Framework

We assume that vehicles are not allowed to perform lane changes when encountering an obstacle. A* algorithm performs the global planning for our vehicles, assigning to each vehicle the shortest path (with euclidean heuristics) from start

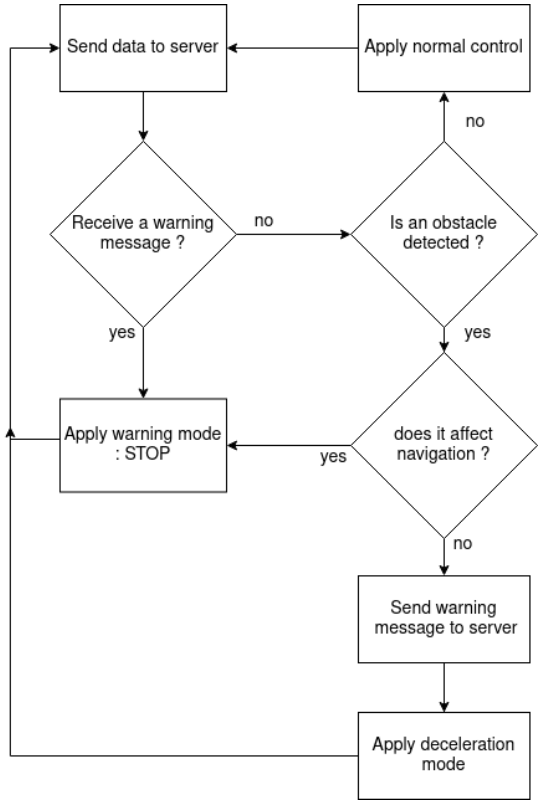

Fig. 3. Behaviour for the ego-vehicle

to goal pose. Each vehicle's controller returns a set of (throttle, steer, brake) controls. This set is directly managed by the CARLA Python API to simulate the behavior of the vehicle following a local plan. We add a set of self-designed rules to the local planner (see Fig.3):

1) stop when encountering an obstacle within a 3-meter range;
2) if a vehicle is stuck in the same location for more than 120 seconds: We consider that it has failed to reach its goal, therefore its navigation is stopped.

Moreover, in order to simulate actors that do not have direct access to data of other actors in the fleet, we used threads for each vehicle. The server is build with the Flask library developed in Python [10]. Each vehicle thread manages the lifecycle of the actor in the simulated environment and the connectivity to the server. A master thread builds the server and manages the master client (see Fig 1).

We perform a test campaign to test our hypothesis that collaborative navigation may reduce the number of requests to shared server while maintaining an efficient navigation. For each simulation run, we evaluate the time spent by each of our fleet of vehicles to succeed in its mission or, if it is blocked for more than 180 seconds, we consider that the mission has failed. We also evaluate the number of requests to the server, reach-ability rate for the fleet and number of collision per vehicle. This framework will be repeated 50 times (with different configurations and number of vehicles in the

fleet each time) in order to perform a statistical analysis of results aimed to compare our collaborative navigation strategy with a centralized approach where messages from the master server is sent to every actor of the fleet. Testbeds are designed with multiple scenarios (with fleet of vehicles from 10 to 50 vehicles in an urban area) and run with the two communication approaches.

C. Simulation Results

The proposed tests aim to see if the proposed method presents a benefit for navigation, by evaluating the impact of collaborative strategies on the average time taken by vehicles in a fleet to reach their goal. They were realized with the CARLA Simulator [11] to simulate fleet of vehicles navigating in urban environment (here Town 10). The simulations were conducted with Python3 language on a Ubuntu 20.04 laptop with processor Intel® Core™ i7-11850H.

Fig. 4 shows a box plot graph with the average travel time for each of the tests performed along with the number of requests to server and average success rate, that allows us to identify the tests in which our collaborative strategy helped more. Especially in tests with few numbers of vehicles that do not require lots of data exchanged if not needed. Figures on the right column show the results for our method while figures on the left column show results for a naive communication strategy where every vehicle can communicate with each other. Our method has meaningfully decreased the number of requests to the server while maintaining a reasonable time of execution for vehicles to reach their respective goals: the overall success rate in mission is still acceptable for large fleets of vehicles (with a number of vehicles in the fleet superior to 30) and even outperforms the naive communication method when the fleet is small. For large fleets of vehicles, execution time per vehicle stays globally the same as the naive communication method. In average, the number of requests to server has been decreased by more than 20 times with our method (6950.66 requests for the regular communication method for one test scenario compared to 288.66 for our method), which was to be expected. This result gives an insight of the communication bandwidth reduction that can be expected from such a communication method.

D. Conclusions on the simulation results

Simulated environment results shows the proof-of-concept of our method in a dense and dynamic environment. Some of the results of our method have been mitigated because of the power resources the simulated environment took, freezing the simulation and affecting controllers processes of the actors, thus the overall success rate and average time for missions. In situations where the time taken for both the communication and non-communication methods is identical, or where the designed method requires more time, two explanations are possible: Firstly, when vehicles do not encounter obstacles, collaborative strategies become redundant as there are no interruptions or need for rerouting. Secondly, sometimes bugs in the low-level car controller result in improper command execution, causing vehicles to get stuck. Moreover, limits of the simulated environment (not possible to park for example) led us to consider real-world use-cases in order to better experiment our method with other vehicles features. A fairly simple simulation with our method can be seen on: https://youtu.be/Dp4LxykEGAY

V. Experiments with a fleet of mobile robots

A. Considered Framework

For real-world experiments, we used a fleet of ROSBots mobile robots designed by Husarion[1] that we can control as one fleet of vehicles. To manage communication between vehicles, a ROS master is running in a decentralized machine and launches the different nodes, publishing and subscribing to the useful topics for collaborative navigation method. Useful data is then sent to the server, that performs the creation of communities and the creation of warning messages. We designed a simple graph of the environment for our 3 mobile robots to navigate within (see Fig. 5). An obstacle is placed among the edge $[(1\ m, 0\ m), (3\ m, 0\ m)]$. The robots 1 and 2 have the same goal $(x_{g1}, y_{g1}) = (3\ m, 0\ m)$ and, the robot 3 have the goal $(x_{g2}, y_{g2}) = (1\ m, 1\ m)$. We designed a simple navigation method that runs on each navigation node, following:

1) get the pose and the range sensor data from the mobile robot and find the closest graph node to pose according to the graph created with the NetworkX [24] Python library;
2) compute shortest path from the current node to the goal node with A*algorithm and euclidean distance as the heuristic;
3) compute commands to send to the mobile robot with a pure pursuit controller;
4) stop robot navigation when encountering an obstacle (i.e. the sensor range data detects an object at a defined distance in the front)

For our communication strategy, we add the following steps: (v) send pose, intention, goal and identifier of each AV to the FLASK server; (vi) create distance-based and intention-based communities w.r.t. poses and goal intentions of the mobile robot available on the server; (vii) send a warning message from master process to the according communities when a mobile robot encounters an obstacle; (viii) recompute path when receiving a warning message and avoid the obstructed lane.

B. Experimental results

With the designed method, we can observe the behaviour of the fleet of vehicles when encountering an obstacle on the way. As explained in the previous section, the server sends a warning message to the concerned vehicles in order for them to change their path according to the lane obstructed. In a qualitative way, these straight-forward tests proved the

[1] https://husarion.com/manuals/rosbot/

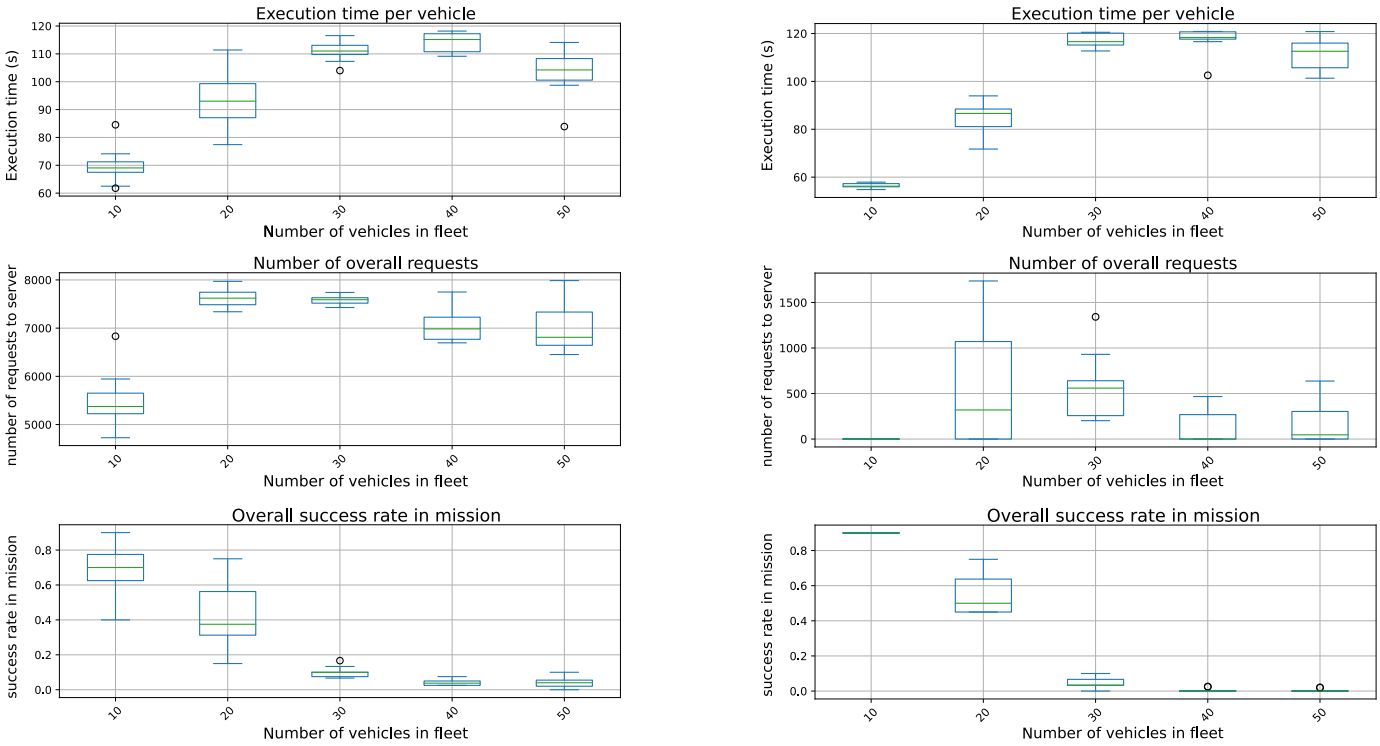

Fig. 4. Simulation results for the two communication strategies: naive communication to all vehicles of the fleet (left column) and our method - community-based communications - (right column). The number of overall request corresponds to the average number of tops received by a vehicle in the fleet from the centralized server.

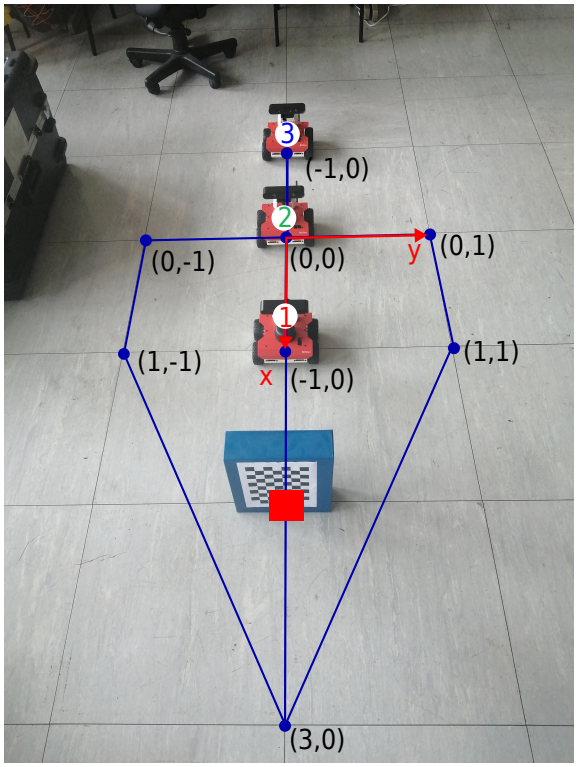

Fig. 5. Framework for the experiments, red frame is the world frame. Blue dots represent nodes of the graph, red square is the obstacle. Robots 1 and 3 are heading to point $(x, y) = (3m, 0m)$ and robot 2 to point $(x, y) = (0m, 1m)$

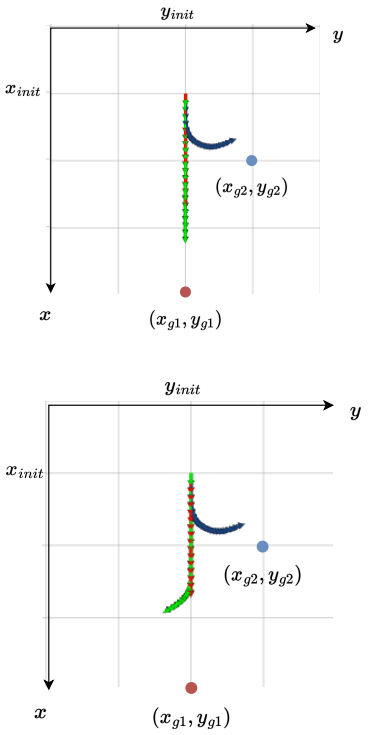

Fig. 6. RViz visualization of the path of the mobile robots (robot 1 in red, robot 2 in green and robot 3 in blue) without communication method (top) and our designed method (down) with an obstacle at $(x, y) = (2.5\ m, 0\ m)$.

feasibility of the method on a fleet of mobile robots with real-time capabilities and inherent flaws of real-world experiments (data loss and latency for e.g.). The trajectories of the fleet of vehicles with and without the method can be seen in Rviz visualization in Fig. 6. Results of the experimentation can be seen on: https://youtu.be/Dp4LxykEGAY

VI. Conclusions

In this paper, we designed a method of communication between vehicles based on their position in the environment and their driving intentions. The latter is based on path planning of our fleet of collaborative vehicles and relies on only range sensor information and the initial plan of the vehicles. These ones can communicate with each others through a centralized server and exchange helpful data to adapt their path to the dynamic environment they navigate. We performed campaign of tests with our communication strategy to prove the efficiency of it, comparing it with a communication method where all vehicles send all messages to each other. We show that our method significantly reduce globally the number of requests to the server by more than 7 times, while also ensuring that the vehicles respect their predefined global plan in an acceptable time. In addition, we implemented the method on both simulated environment (CARLA Simulator) and real-world (ROSbots) to visualize effect of our method on the fleet of mobile robots. Perspectives on the subject include exploring different case studies that may involve multiple and/or different intentions (e.g., finding a parking place in a crowded neighbourhood, passing a roundabout, managing unsignalized intersection) as well as add more data from different sensors (camera image, LIDAR data, etc.). Resiliency of our method to data loss and latency is also a topic that need to be investigated, especially with real-world experiments. Moreover, our communication method can be extended not only to a bigger fleet of vehicles, but also to intelligent infrastructure, e.g. Road-Side Units (RSU). This will improve the coverage of the environment and, with the help of Vehicle-To-All (V2X) communications, enhance collaborative navigation in urban environments.

ACKNOWLEDGMENT

This work was carried out in the framework of the NExT Senior Talent Chair DeepCoSLAM, which were funded by the French Government, through the program Investments for the Future managed by the National Agency for Research (ANR-16-IDEX-0007), and with the support of Région Pays de la Loire and Nantes Métropole.

References

[1] I. Kaparias, M. G. H. Bell, W. Dong, A. Sastrawinata, A. Singh, X. Wang, and B. Mount, "Analysis of pedestrian–vehicle traffic conflicts in street designs with elements of shared space," *Transportation Research Record*, vol. 2393, no. 1, pp. 21–30, 2013.

[2] Y. Jiang and T. Hsiao, "Deep learning in perception of autonomous vehicles," in *Proceedings of the 2021 International Conference on Public Art and Human Development (ICPAHD 2021)*, pp. 561–565, Atlantis Press, 2022.

[3] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–23, 05 2019.

[4] Y. Ma, Z. Wang, H. Yang, and L. Yang, "Artificial intelligence applications in the development of autonomous vehicles: A survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, pp. 315–329, 03 2020.

[5] C. Lim, K.-J. Kim, and P. Maglio, "Smart cities with big data: Reference models, challenges, and considerations," *Cities*, vol. 82, 05 2018.

[6] S. Masi, P. Xu, and P. Bonnifait, "Adapting the Virtual Platooning Concept to Roundabout Crossing," in *29th IEEE Intelligent Vehicles Symposium (IV 2018)*, pp. 1366–1372, June 2018.

[7] "Intelligent transport systems (its), vehicular communications; geonetworking; part 4: Geographical addressing and forwarding for point-to-point and point- to-multipoint communications; sub-part 1: Media-independent functionality," tech. rep., ETSI, August 2017.

[8] H. Shahwani, S. Shah, M. Ashraf, M. Akram, J. Jeong, and J. Shin, "A comprehensive survey on data dissemination in vehicular ad hoc networks," *Vehicular Communications*, vol. 34, p. 100420, 10 2021.

[9] R. T. Fielding, *REST: Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine, 2000.

[10] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, Inc., 1st ed., 2014.

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, vol. 78 of *Proceedings of Machine Learning Research*, pp. 1–16, 2017.

[12] S. Demmel, G. Larue, D. Gruyer, and A. Rakotonirainy, "An ieee 802.11p empirical performance model for cooperative systems applications," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 10 2013.

[13] Y. Mylonas, M. Lestas, A. Pitsillides, and P. Ioannou, "Speed adaptive probabilistic flooding for vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 64, pp. 719–723, 09 2011.

[14] B. Ducourthial, V. Cherfaoui, T. Fuhrmann, and S. Bonnet, "Experimentation of a road hazard anticipation system based on vehicle cooperation," *Vehicular Communications*, vol. 36, p. 100486, Aug. 2022.

[15] L. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," *Journal of Physical Agents*, vol. 2, 01 2008.

[16] Y. Zeng, D. Li, and A. V. Vasilakos, "Opportunistic fleets for road event detection in vehicular sensor networks," *Wireless Networks*, vol. 22, pp. 503–521, June 2015.

[17] G. Contissa and F. Lagioia, "The ethical knob: Ethically-customisable automated vehicles and the law," *Sistemi Intelligenti*, vol. 29, pp. 601–614, 12 2017.

[18] B. Ducourthial, Y. Khaled, and M. Shawky, "Conditional transmissions: Performance study of a new communication strategy in vanet," *Vehicular Technology, IEEE Transactions on*, vol. 56, pp. 3348 – 3357, 12 2007.

[19] N. Mahdoui, V. Fremont, and E. Natalizio, "Communicating multi-uav system for cooperative slam-based exploration," *Journal of Intelligent & Robotic Systems*, vol. 98, 05 2020.

[20] E. W. Dijkstra, L. Beauguitte, and M. Maisonobe, "A Note on Two Problems in Connexion with Graphs. Numerische Mathematik 1, p. 269271 Version bilingue et commentée," 2021.

[21] D. Foead, A. Ghifari, M. Kusuma, N. Hanafiah, and E. Gunawan, "A systematic literature review of a* pathfinding," *Procedia Computer Science*, vol. 179, pp. 507–514, 01 2021.

[22] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

[23] R. Guyard and V. Cherfaoui, "VANET distributed data fusion for traffic management," in *22nd IEEE International Conference on Intelligent Transportation Systems (ITSC 2019)*, (Auckland, New Zealand), pp. 1851–1856, Oct. 2019.

[24] A. Hagberg, P. Swart, and D. Chult, "Exploring network structure, dynamics, and function using networkx," *Proceedings of the 7th Python in Science Conference*, 01 2008.