The following publication T. Liu et al., "The Role of the Hercules Autonomous Vehicle During the COVID-19 Pandemic: An Autonomous Logistic Vehicle for Contactless Goods Transportation," in IEEE Robotics & Automation Magazine, vol. 28, no. 1, pp. 48-58, March 2021 is available at https://doi.org/10.1109/MRA.2020.3045040.

The Role of the Hercules Autonomous Vehicle During the COVID-19 Pandemic: An Autonomous Logistic Vehicle for Contactless Goods Transportation

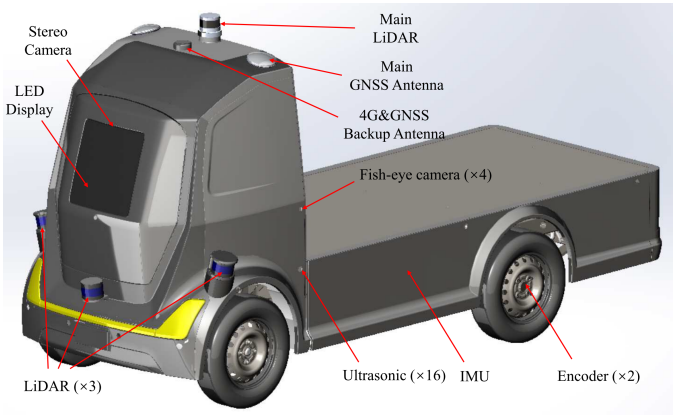
Tianyu Liu*, Qinghai Liao*, Lu Gan, Fulong Ma, Jie Cheng, Xupeng Xie, Zhe Wang, Yingbing Chen, Yilong Zhu, Shuyang Zhang, Zhengyong Chen, Yang Liu, Meng Xie, Yang Yu, Zitong Guo, Guang Li, Peidong Yuan, Dong Han, Yuying Chen, Haoyang Ye, Jianhao Jiao, Peng Yun, Zhenhua Xu, Hengli Wang, Huaiyang Huang, Sukai Wang, Peide Cai, Yuxiang Sun, Yandong Liu, Lujia Wang, and Ming Liu

Fig. 1. A worker dressed in a protective suit is collecting goods from our Hercules logistic autonomous vehicle. There is no person-to-person contact during the process of goods transportation. This photo was taken in Shenzhen, Guangdong, China, February 2020 during the COVID-19 pandemic.

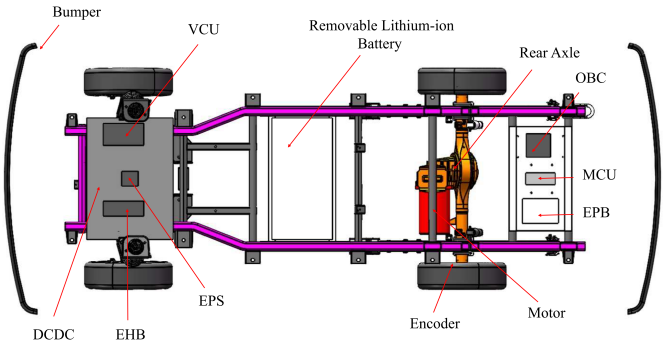
Since early 2020, the coronavirus disease 2019 (COVID-19) has spread rapidly across the world. As at the date of writing this article, the disease has been globally reported in 223 countries and regions, infected over 108 million people and caused over 2.4 million deaths[1]. Avoiding person-to-person transmission is an effective approach to control and prevent the pandemic. However, many daily activities, such as transporting goods in our daily life, inevitably involve person-to-person contact. Using an autonomous logistic vehicle to achieve contact-less goods transportation could alleviate this issue. For example, it can reduce the risk of virus transmission between the driver and customers. Moreover, many countries have imposed tough lockdown measures to reduce the virus transmission (e.g., retail, catering) during the pandemic, which causes inconveniences for human daily life. Autonomous vehicle can deliver the goods bought by humans, so that humans can get the goods without going out. These demands motivate us to develop an autonomous vehicle, named as Hercules, for contact-less goods transportation during the COVID-19 pandemic. The vehicle is evaluated through real-world delivering tasks under various traffic conditions.

* Equal Contribution

[1]https://covid19.who.int/ (accessed on Feb. 17, 2021)



(a) The sensors on the vehicle (with the cargo box removed).



(b) The sensors and control units on the mobile base (chassis).

Fig. 2. The sensors used in our vehicle and the modules in the chassis. Note that the cargo box is replaceable. It is not shown in the figure.

There exist many studies related to autonomous vehicles, however, most of these works focus on the specific modules of autonomous driving systems. For example, Sadigh *et al.* [1] developed a planning method that models the interaction with other vehicles. Koopman *et al.* [2] presented a testing paradigm for autonomous vehicles. Some researchers have tried to construct the complete autonomous driving systems [3]. Compared with these studies, we build a complete system and add several new modules, such as the cloud server module. We also make some adjustments, such as considering novel dynamic constraints, in our solution to make the vehicle more suitable for the contact-less good transportation. In the follow-

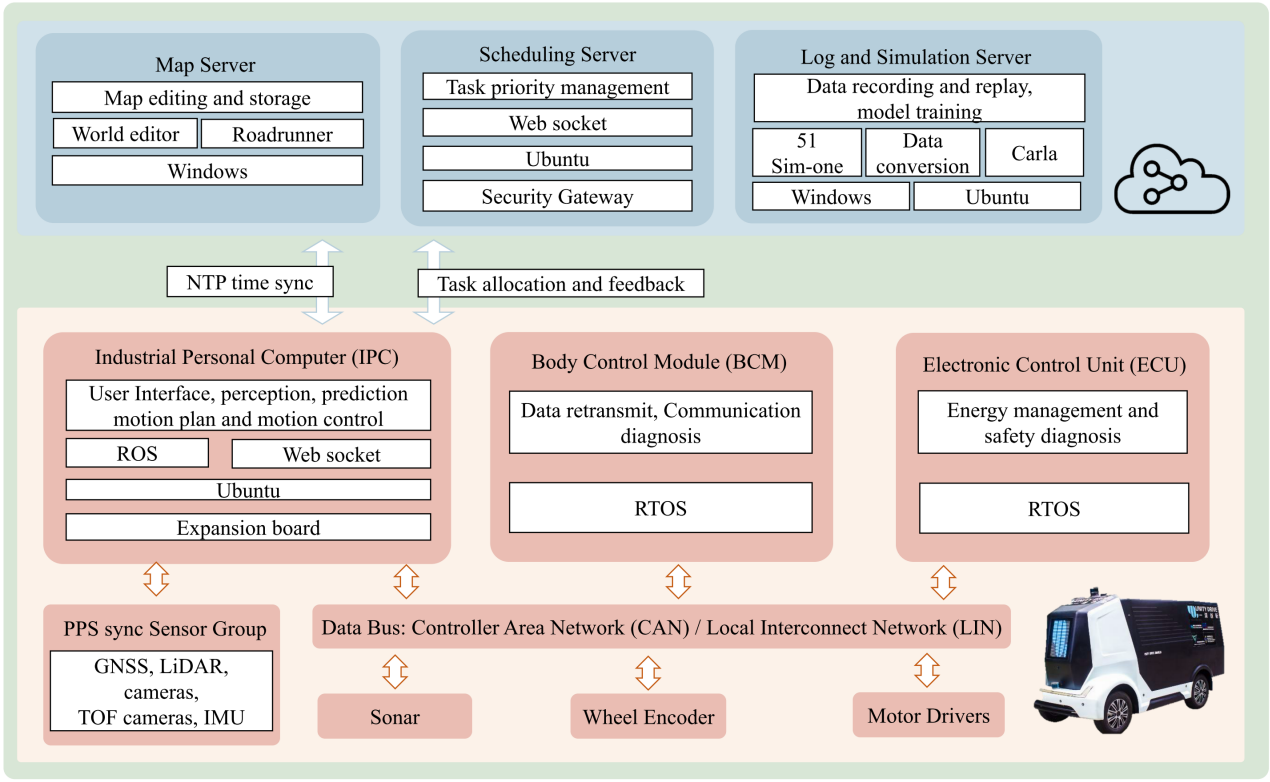


Fig. 3. The software architecture of our vehicle. The part shown in the yellow box is running on the vehicle, and the part shown in the blue box is running on the cloud. The figure is best viewed in color.

ing sections, we provide details on the hardware, software, as well as the algorithms to achieve autonomous navigation including perception, planning and control. This paper is accompanied by a demonstration video and a dataset, which are available at https://sites.google.com/view/hercules-vehicle.

I. Hardware System

The hardware system of our vehicle mainly consists of a fully functional Drive-by-Wire (DBW) chassis and autonomous driving-related devices. Fig. 2 shows the sensors and the 3-D model of our vehicle.

A. Fully Functional DBW Chassis

To achieve autonomous driving, the first task is to equip a vehicle with the full Drive-by-Wire (DBW) capability. Our DBW chassis can be divided into four parts: 1) Motion control. This module includes Motor Control Unit (MCU), Electric Power Steering (EPS), Electro-Hydraulic Brake (EHB) and Electronic Parking Brake (EPB). MCU supports both the speed control and torque control. EPS controls the steering angle and speed of the the vehicle. The combination of MCU and EPS controls the longitudinal and lateral motions. EHB controls the brake. EPB controls the parking brake; 2) Electronic accessories control. A vehicle has some basic electronic accessories such as lights and horns. They are controlled by the Body Control Module (BCM); 3) Basic sensors. Our chassis are equipped with some basic sensors, such as bumper, wheel encoder and Tire Pressure Monitoring System (TPMS). The bumper and TPMS are both safety-critical sensors. Specifically, bumper is used to detect collisions and is the last defence to prevent further damage when accident occurs; 4) System control. The chassis system is controlled and managed via Vehicle Control Unit (VCU) which is responsible for coordinating each module. It keeps communicating with the Industrial Personal Computer (IPC), performing parameter checking and sending commands to other modules. In addition, VCU is responsible for critical safety functions, such as the stopping signal from the emergency button. In our chassis, VCU and BCM are implemented on one device.

There are two batteries in our vehicle. A 12 V Lead-acid starter battery and a 72 V removable lithium-ion battery, which can support the maximum 80 Km running distance. The lithium-ion battery powers the chassis, IPC, sensors and accessories. It has a built-in Battery Management System (BMS) to monitor and manage the battery. The removable design allows the vehicle to operate at 24 hours a day without stopping for a recharge. An On-Board Charger (OBC) with a Direct Current Direct Current (DCDC) converter takes about 5 hours to fully charge the battery.

B. Autonomous Driving-related Devices

The devices related to autonomous driving are: 1) Computation platform. Our vehicle is equipped with an IPC which has an Intel i7-8700 CPU with 6 cores and 12 threads, 32 GB memory, and a 1050Ti NVIDIA Graphics card. It is able to run deep learning-based algorithms; 2) Sensors. As shown in Fig. 2(a), our vehicle is equipped with four 16-beam LiDAR,

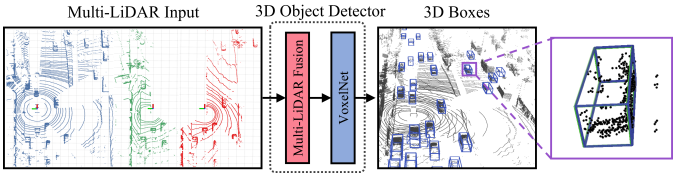


Fig. 4. Overview of the 3-D object detection module. The inputs are multiple point clouds captured by synchronized and well-calibrated LiDARs. We use an early-fusion scheme to fuse the data from multiple calibrated LiDARs, and adopt the VoxelNet [4] to detect 3-D objects from the fusion results.

one MEMS short-range LiDAR, four fish-eye cameras, 4×4 ultrasonic radars, one IMU and one high-precision GNSS system which supports RTK and heading vector; 3) Auxiliary devices. We have 4G/5G Data Transfer Unit (DTU), Human Machine Interface (HMI), LED display, remote controller. The DTU allows the IPC to be connected to our cloud management platform via the Internet. The LED display is programmable and can be controlled by the IPC. Hence, it can interact with other traffic participants like pedestrians and human drivers. Also, it can be used for advertisement. The remote controller is necessary in the current stage to ensure safety.

II. Software System

Fig. 3 shows the software architecture of our autonomous vehicle. It can be generally divided into two parts: software system running on the vehicle, and software system running on the cloud.

A. Software System on the Vehicle

There are three main computing platforms on the vehicle: the IPC, Electronic Control Unit (ECU) and BCM. The IPC is used to run algorithms for autonomous navigation. The ECU is used to ensure the safety of the vehicle by energy management and safety diagnosis. The applications on ECU run on the Real-Time Operating System (RTOS), which satisfies the real-time requirements. The BCM connects the IPC and ECU. It also runs on the RTOS, which meets the real-time requirements. It detects the communication between the nodes of the CAN network by heartbeat protocols. When major nodes of this network experience an outage or crash, the BCM stops transmitting high-level CAN signals from the IPC to the VCU and waits for human interventions.

The sensors on the vehicle are synchronized by a 1 Hz Pulse per Second (PPS) signal from the external GNSS receiver. The IPC receives data from the sensors and uses them at different frequencies. For example, the state estimation module updates at 100 Hz, providing real-time enough position feedback for the control system. This is achieved by fusing LiDAR measurements with other high-frequency sensors, e.g., IMU or wheeled odometer. The LiDAR object detection module runs at 10 Hz according to the refresh rate of the LiDAR. All the modules are developed on the Robot Operating System (ROS) to facilitate data communication.

B. Software System on the Cloud

The software system on the cloud mainly includes the map server, the scheduling server and the log and simulation server.

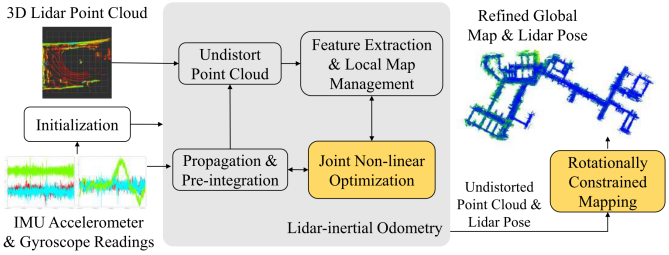


Fig. 5. The schematic diagram of our 3-D point-cloud mapping system. After the initialization, the system will estimate the states and refine the global map and Lidar poses in the odometry and mapping sub-modules, respectively.

The map server stores pre-built maps. The scheduling server performs the task allocations and collects the status of every registered running vehicle. It also plays the role of accessing the map data for routing, transmitting sensor data into the map server, recording the key information into the log server, and replaying the data recorded for a good trace-back. The log and simulation server run the end-to-end simulator Carla and 51Sim-One. The clock synchronization between the platforms on the vehicle and cloud is manipulated based on the network time through the Network Time Protocol (NTP).

III. Perception

Perception serves as the fundamental component of autonomous navigation. It provides necessary information for planning and control. This section describes two key perception technologies used in our vehicle.

A. Multiple Lidar-based 3-D Object Detection

The 3-D object detection aims to recognize and classify objects, as well as estimate their poses with respect to a specific coordinate system. We use multiple Lidars for object detection. The first step is to calibrate the Lidars. In this work, we propose a marker-based approach [5] for automatic calibration without any additional sensors and human intervention. We assume that three linearly independent planar surfaces forming a wall corner shape are provided as the calibration targets, ensuring that the geometric constraints are sufficient to calibrate each pair of Lidars. After matching the corresponding planar surfaces, our method can successfully recover the unknown extrinsic parameters with two steps: a closed-form solution for initialization based on the Kabsch algorithm [6] and a plane-to-plane iterative closest point (ICP) for refinement. The overview of our 3-D object detection is shown in Fig. 4. The inputs to our approach are multiple point clouds captured by different Lidars. We adopt an early-fusion scheme to fuse the data from multiple calibrated Lidars at the input stage. With the assumption that the Lidars are synchronized, we transform the raw point clouds captured by all the Lidars into the base frame, and then feed the fused point clouds into the 3-D object detector [4]. The final output is a series of 3-D bounding boxes.

B. 3-D Point-cloud Mapping

The 3-D point-cloud mapping aims to build the 3-D map of the traversed environments. Fig. 5 shows the diagram of

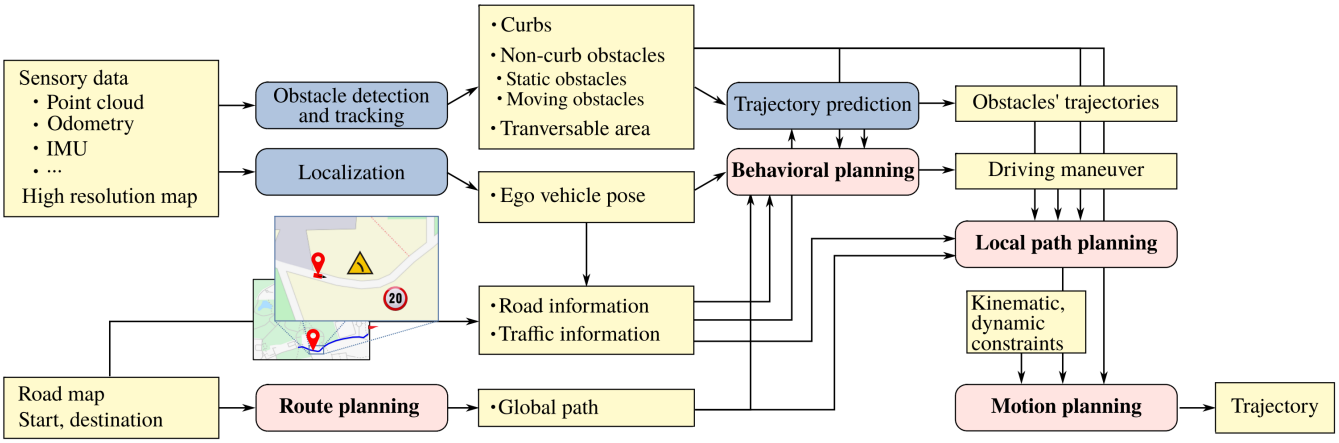


Fig. 6. The schematic diagram of planning for our autonomous vehicle. The planning process consists of four layers: route planning, behavioral planning, path planning and motion planning. The four layers are coloured in pink.

our mapping system. The inputs to the system are the raw data from the IMU and 3-D Lidar (i.e., accelerometer and gyroscope readings from the IMU and point clouds from the 3-D Lidar). The system starts with an adapted initialization procedure, followed by two major sub-modules: the Lidar-inertial odometry and rotationally constrained mapping. Since the vehicle usually remains still at the beginning of mapping, we do not need to excite the IMU to initialize the module as described in [7], which is more suitable for hand-held applications. With the stationary IMU readings, the initial orientation for the first body frame can be obtained by aligning the average of the IMU accelerations to the opposite of the gravity direction in the world frame. The initial velocity, and IMU biases are set to zero. Then, the Lidar-inertial odometry optimally fuses Lidar and IMU measurements in a local window. The mapping with rotational constraints further refines the Lidar poses and the point-cloud map.

IV. Planning

Planning enables autonomous vehicles to acquire future paths or motions towards the destination. The planning for autonomous driving is challenging, because traffic environments are usually with dynamic objects, bringing about risks and uncertainties [8]. Autonomous vehicles are required to interact with various road participants, including cars, motorcycles, bicycles, pedestrians, etc. The planner needs to meet the vital requirements of safety, and the kinematic and dynamic constraints of vehicles, as well as the traffic rules. To satisfy these requirements, our planning is hierarchically divided into four layers: route planning, behavioral planning, local path planning, and motion planning. Fig. 6 shows the four-layer planning process.

A. Route Planning

The route planning aims at finding the global path from the global map. For autonomous driving, the route planner typically plans a route given a road network. For structured environments with clear road maps, we use path planning algorithm A* to find the route by establishing the topological graph. However, driveways in industrial parks or residential areas are often not registered in the road net. Furthermore, some of the traversable areas in these places are unstructured and not clearly defined. We employ experienced drivers as teachers to demonstrate reference routes in these places. Fig.8 shows the global routes in the road network with arrows indicating the forward directions.

B. Behavioral Planning

Behavioral planning decides the manoeuvres for local navigation. It is a high-level representation of a sequence of vehicle motions. Typical manoeuvres are lane keeping and overtaking. This layer receives information from the global maps and finds the category of the local area to give specifications on path planning. For example, unstructured environments, like parking lots, have different requirements on planning. Given the road map and the localization of the ego-vehicle, features of the local area can be obtained. As shown in Fig. 6, road information that indicates the road environment classification of the global path segments is helpful for behavioral planning. Furthermore, traffic information from traffic signs helps in making decisions. The road and traffic information together with the estimation of other moving agents allows the behavioral planner to follow or overtake the front car, or pull over the ego-vehicle.

C. Local Path Planning

The local path planning generates a geometric path from the starting pose to the goal pose for the vehicle to follow. The time complexity of this process increases with increased path length, so it is often limited to a local range to ensure real-time planning. The local path planner needs to tackle motion constraints of the vehicle to generate collision-free paths that conform to the lane boundaries and traffic rules. Fig. 6 shows the online local path planning for driving on standard roads. Here we plan the path in the Frenet coordinate system. With the global path as the reference path, it defines the lateral shift

to the path and the distance travelled along the path from the start position. We drew multiple samples with different speeds and lateral offsets. Then a graph search method is adopted to search the path with the minimum cost. To define the cost of the coordinates of each curve, we take into consideration the quality of the curve, ending offset to the global path, and other factors (e.g., the potential trajectories of other agents).

D. Motion Planning

Given the planned path, motion planning is the final layer which optimizes the trajectory with dynamic constraints from the vehicle, the requirements for comfort and energy consumption. The planned trajectory specifies the velocity and acceleration of the vehicle at different timestamps, so it is also called trajectory planning. Though the path planned in the Frenet frame contains speed information, the dynamic constraint of the vehicle is not yet considered. Besides this, the local planning process is time-consuming and has a low update rate, which is inadequate to handle dynamic obstacles and emergency cases. The motion planner optimizes the trajectory given the information of obstacles, the constraints from the vehicle, and the path from the local path planner. It outputs the final trajectories for the controller to follow at a much higher updating rate to ensure safety.

V. CONTROL

The main task of vehicle control is to track the planned trajectory. In the past decade, many trajectory tracking controllers have been developed, among which the Model Predictive Controller (MPC) [9] is the most popular one. The schematic diagram of our controller is shown in Fig. 7.

As we can see, there are two inputs to the trajectory tracking controller. One is the trajectory $s(t)$, which includes the information (e.g., desired coordinates, curvatures, speed) from the motion planner, the other is the feedback information $x'(t)$ from the state estimator. Sometimes, sensor feedback from the chassis cannot be directly sent to the controller, or more feedback quantities are required by the controller, which is difficult to obtain from sensors. In such cases, a state feedback estimator is required but not a must. In Fig. 7, the output of the trajectory tracking controller $u(t)$ is sent to the chassis after being processed by a lower controller. The lower controller can work for many purposes. For example, our autonomous vehicle can work in both the autonomous-driving mode and the parallel-driving model (i.e., the remote control mode). The trajectory tracking controller only functions in the autonomous-driving mode, which means that only in this mode the lower controller takes as input $u(t)$.

The vehicle control can be divided into the lateral control, which controls steer angles, and the longitudinal control, which controls the car speed. There are two types of MPCs in the area of the autonomous vehicle. One is kinematics-based while the other is dynamics-based. The kinematics-based MPC is a combined controller that integrates the lateral control and longitudinal control. Therefore, the longitudinal PID controller highlighted in the dashed box in Fig. 7 may be not required. The vector of two control quantities $u(t)$ (i.e., steer angle and speed), will be directly given by the MPC. However, the dynamics-based MPC is a standalone lateral controller of which the output is a control quantity of the steering angle. In such a case, a longitudinal PID controller that outputs the speed control quantity will be required. And the outputs of these two controllers constitute $u(t)$.

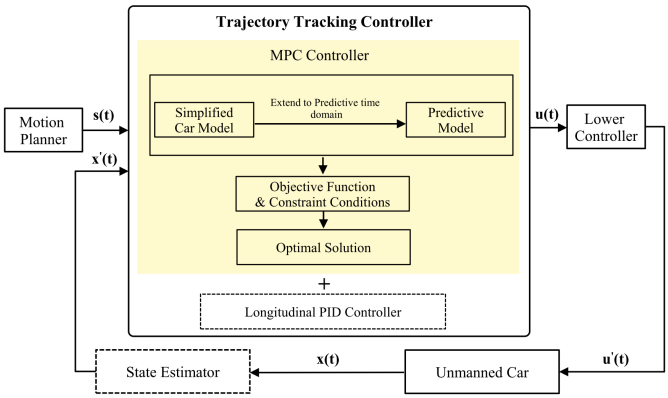


Fig. 7. The schematic diagram of our controller. The main component is the trajectory tracking controller.

VI. EVALUATION

This section describes the real tasks of contact-less goods transportation using our vehicle during the COVID-19 pandemic in China. From Feb. 2, 2020 to May. 27, 2020, we have deployed 25 vehicles in 3 different cities (Zibo, Shandong; Suzhou, Jiangsu; Shenzhen, Guangdong) in China. Our current server can handle 200 vehicles simultaneously. The total running distances of each vehicle reached 2,500 Km. The details of the transportation tasks are summarized in Tab. I. Selected demonstration photos during the tasks are displayed in Fig. 8. Note that in case of failures during the autonomous navigation, such as unavoidable accidents and system errors, we build a parallel driving system to back up our vehicle control. The parallel driving system is a remote control system based on 4G/5G technology. When using 4G with good signal, the latency is usually between 30ms and 60ms. We set the system to automatically adjust the bit rate to ensure that the vehicle is not out of line. When using 5G, the latency can be less than 20ms. If the vehicle is out of line, it will stop immediately. We have tested the function with several vehicles on several real road environments, and the experimental results show that the function works well. The vehicle control is immediately taken over by a human driver in case of any failure for the autonomous navigation. We expect less human intervention during our goods transportation tasks. The performance is evaluated by the number of occurrences of human interventions.

VII. LESSONS LEARNED AND CONCLUSIONS

For object detection, we found that real-time performance deserves much more attention than accuracy in practice. The perception algorithms should be efficient since they lie in the front-end of the whole autonomous navigation system. Therefore, we replaced the dense convolution layers with

Fig. 8. Selected task routes and demonstration photos. The first column includes three representative routes: (a) A 9.6 Km route in Zibo, Shandong for vegetable delivery; (f) A 1.2 Km route in Suzhou, Jiangsu for lunch meal delivery; (i) A 1.6 Km route in Shenzhen, Guangdong for lunch meal delivery. Photos taken in Zibo, Shandong: (b) Starting from the logistics company; (c) Crossing the gate of the logistics company; (d) and (e) Urban main roads. Photos taken in Suzhou, Jiangsu: (g) Left turn; (h) Spraying disinfectant in a residential area. Photos taken in Shenzhen, Guangdong: (j) Heavy traffic environment; (k) Surrounded with road users and meeting with oncoming cars; (l) U-turn in a narrow road; (m) Traffic light at night; (n) Contact-less picking up meal.

TABLE I
THE DETAILS OF THE CONTACT-LESS GOODS TRANSPORTATION TASKS DURING THE COVID-19 PANDEMIC IN CHINA.

City	Task	Distance	Time Duration	Payload	Environments	Characteristics
Zibo, Shandong	Vegetable Delivery	9.6 Km	75 min	600 KG	Urban Main Road	Light Traffic Heavy Payload Slow Speed
	Vegetable Delivery	5.4 Km	50 min	960 KG	Urban Main Road Residential Area	Light Traffic Heavy Payload Slow Speed
Suzhou, Jiangsu	Meal Delivery	1.2 Km	30 min	80 KG (100 boxes of meals)	Urban Main Road	Medium Traffic Left-turn U-turn
	Road Disinfection	0.6 Km	10 min	-	Residential Area	Narrow Road Crowded Obstacle Slow Speed
Shenzhen, Guangdong	Meal Delivery	1.6 Km	20 min	64 KG (80 boxes of meals)	Urban Main Road Residential Area	Heavy Traffic Narrow Road U-turn
	Meal Delivery	4.0 Km	40 min	96 KG (120 boxes of meals)	Urban Main Road Residential Area	Heavy Traffic Narrow Road U-turn

spatially sparse convolution in our 3-D object detection module. As a result, the inference time is boosted from 250 ms approximately to 56 ms.

For the point-cloud mapping, we found that our system was capable of dealing with the rapid motion of the vehicle and short-term point-cloud occlusions. Since most of the Lidars are mounted parallel to the ground and the vehicles always move along the ground, the typical ring structure of the point clouds makes the system difficult to observe in terms of translational movements vertical to the ground plane. Drift in this direction is inevitable during long-term operations. In practice, we learnt that the GPS localization results signalled the potential loop, leading to a consistent larger-region map for Lidar-based localization. In very crowded dynamic traffic environments, the system could be degraded by the disturbances from moving objects [10]. To tackle this issue, we use semantic segmentation to remove movable objects to get clear point-cloud data.

For the planning part, we found that the four-layer hierarchical planning is necessary and effective. The working environments of our vehicle are complex in terms of road structures, traffic conditions, driving etiquette, etc. Layer-wise planning makes the system extensible for multiple environments and various requirements. Furthermore, it is essential to attach importance to the uncertainty from the perception modules. The uncertainty comes from the limited accuracy of the perception system as well as the time delay in processing. For the planning, we avoid hitting the critical conditions and leave safe distances for the planned trajectory.

For the control part, we found that the greatest advantage of using an MPC can be gained by adding multiple constraints in the control process. When the vehicle operates at low speed, the kinematic constraints restrain the vehicle motion planning and control. But with the increment of speed, dynamic characteristics become more influential. As aforementioned, the dynamic-based MPC is much more accurate than the kinematic-based MPC since the predictive model is more accurate. However, we found that the complex model prediction is not the best option. With regards to low- and middle-speed operation environments, a simplified predictive model with multiple constraints would be sufficient.

From the real-life operations, we found that more conservative settings for obstacle detection could lead to more false positives. This would decrease the speed or even freeze the vehicle, and hence cause traffic jams. On some roads, the minimum allowed speed is not indicated, we need to keep the vehicle speed not too slow. Otherwise, it would cause annoyance for other vehicle drivers. Clearly and easily identified human-machine interface is also important. It can be used to inform other vehicle drivers in advance what the autonomous vehicle will do. Otherwise, the other vehicle drivers would feel frightened because they could not anticipate the behaviours of the autonomous vehicle. For example, our vehicle often frightens other vehicle drivers when it is making a reverse even the reversing light is flashing. Using a screen to notify the reversing behaviour could alleviate the issue. In some cases, not strictly obeying the traffic rules would be good for the autonomous navigation. For example, it would be wise to change the lane when there happens a traffic accident ahead of the ego-lane, even the lane changing behaviour is not allowed according to the traffic rules. Otherwise, the vehicle could not get over.

The successful daily operations demonstrated that using our autonomous logistic vehicle could effectively avoid virus spread due to human contact. It effectively builds a virtual wall between the recipient and sender during the goods transportation. For quantitative measures, we can compute from Tab. I that the average distance for each task per vehicle is $(9.6 + 5.4 + 1.2 + 0.6 + 1.6 + 4.0)/6 \approx 3.7$ Km. As the total running distance is 2,500 Km, the number of tasks is $2,500/3.7 \approx 676$. According to our observation, there are usually 4 times of person-to-person contacts in each task of the traditional goods transportation. So the number of avoided contacts would be $4 \times 676 = 2,704$. As we have

25 running vehicles, the total number of avoided contacts would be $25 \times 2,704 = 67,600$. Currently, there is a huge demand for contact-less goods transportation in many infected areas. We believe that continuous long-term operations could extensively improve our vehicle and enhance the maturity of our technologies.

VIII. Acknowledgements

This work was supported by the Hong Kong RGC Project No. 11210017, Guangdong Science and Technology Plan Guangdong-Hong Kong Cooperation Innovation Platform Project No. 2018B050502009, Shenzhen Science and Technology Innovation Commission Project No. JCYJ2017081853518789, Macao Science and Technology Development Fund Project No. 0015/2019/AKP.

References

[1] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions." in *Robotics: Science and Systems*, vol. 2. Ann Arbor, MI, USA, 2016.

[2] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[3] M. R. Endsley, "Autonomous driving systems: A preliminary naturalistic study of the tesla model s," *Journal of Cognitive Engineering and Decision Making*, vol. 11, no. 3, pp. 225–238, 2017.

[4] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[5] J. Jiao, Q. Liao, Y. Zhu, T. Liu, Y. Yu, R. Fan, L. Wang, and M. Liu, "A novel dual-lidar calibration algorithm using planar surfaces," in *2019 IEEE IV*. IEEE, 2019, pp. 1499–1504.

[6] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, vol. 34, no. 5, pp. 827–828, 1978.

[7] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3d lidar inertial odometry and mapping," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[8] M. Liu, "Robotic online path planning on point cloud," *IEEE transactions on cybernetics*, vol. 46, no. 5, pp. 1217–1228, 2015.

[9] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[10] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110 – 122, 2017.

Tianyu Liu Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: liutianyu@unity-drive.com.

Qinghai Liao The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: qinghai.liao@connect.ust.hk.

Lu Gan The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: lganaa@connect.ust.hk.

Fulong Ma The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: fmaaf@connect.ust.hk.

Jie Cheng The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: jchengai@connect.ust.hk.

Xupeng Xie The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: xxieak@connect.ust.hk.

Zhe Wang Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: wangzhe@unity-drive.com.

Yingbing Chen The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: ychengz@connect.ust.hk.

Yilong Zhu The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: yzhubr@connect.ust.hk.

Shuyang Zhang Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: shuyang.zhang@unity-drive.com.

Zhengyong Chen Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: chenzhengyong@unity-drive.com.

Yang Liu The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: yang.liu@connect.ust.hk.

Meng Xie Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: xiemeng@unity-drive.com.

Yang Yu The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: yyubj@connect.ust.hk.

Zitong Guo Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: guozitong@unity-drive.com.

Guang Li Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: liguang@unity-drive.com.

Peidong Yuan Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, China. Email: yuanpeidong@unity-drive.com.

Dong Han Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Email: dong.han@siat.ac.cn.

Yuying Chen The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: ychenco@connect.ust.hk.

Haoyang Ye The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: hy.ye@connect.ust.hk.

Jianhao Jiao The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: jjiao@connect.ust.hk.

Peng Yun The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: pyun@connect.ust.hk.

Zhenhua Xu The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: zxubg@connect.ust.hk.

Hengli Wang The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: hwangdf@connect.ust.hk.

Huaiyang Huang The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: hhuangat@connect.ust.hk.

Sukai Wang The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: swangcy@connect.ust.hk.

Peide Cai The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: peide.cai@connect.ust.hk.

Yuxiang Sun The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. Email: yx.sun@polyu.edu.hk.

Yandong Liu Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Email: yd.liu@siat.ac.cn.

Lujia Wang Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Email: lj.wang1@siat.ac.cn.

Ming Liu The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China. Email: eelium@ust.hk.