The Role of the Hercules Autonomous Vehicle During the COVID-19 Pandemic

Use Cases for an Autonomous Logistic Vehicle for Contactless Goods Transportation

By 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

Since early 2020, COVID-19 has spread rapidly across the world. At the date of this writing, the disease has been globally reported in 220 countries and regions, infected more than 68 million people, and caused more than 1.5 million deaths (see https://covid19.who.int/) for up-to-date statistics. 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 contactless 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 virus transmission (e.g., retail and catering) during the pandemic, which causes inconveniences for daily human life. Autonomous vehicles can deliver the goods purchased by humans so that they can receive these without going out. Such demands motivated us to develop an autonomous vehicle, Hercules, for contactless goods transportation during the COVID-19 pandemic. The vehicle is evaluated through real-world delivery tasks under various traffic conditions.

There exist many studies related to autonomous vehicles; however, most 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 complete autonomous driving systems [3]. Compared with these studies, we built a complete system and added several new modules, such as the

Digital Object Identifier 10.1109/MRA.2020.3045040
Date of current version: 29 January 2021

1070-9932/21©2021IEEE

cloud server module. We also made some adjustments in our solution, such as considering novel dynamic constraints, to make the vehicle more suitable for contactless goods transportation (Figure 1). In this article, we provide details on the hardware and software as well as the algorithms to achieve autonomous navigation, including perception, planning, and control. This article is accompanied by a demonstration video and a data set, which are available at https://sites.google.com/view/hercules-vehicle.

The Hardware System

The hardware system of our vehicle mainly consists of a fully functional drive-by-wire (DBW) chassis and autonomous driving-related devices. Figure 2 depicts the sensors and the 3D model of our vehicle.

A Fully Functional DBW Chassis

To achieve autonomous driving, the first task is to equip a vehicle with full DBW capability. Our DBW chassis can be divided into four parts: 1) motion control, 2) electronic accessories control, 3) basic sensors, and 4) system control. The motion control module includes the motor control unit (MCU), electric power steering (EPS), electrohydraulic brake (EHB), and electronic parking brake (EPB). The MCU supports both the speed control and torque control. The EPS controls the steering angle and speed of the vehicle. The combination of the MCU and EPS controls the longitudinal and lateral motions, while the EHB controls the brake, and the EPB controls the parking brake.

The electronic accessories control module includes some basic electronic accessories such as the vehicle's lights and horns, which are controlled by the body control module (BCM). In the basic sensors module, our chassis is equipped with some basic sensors, such as a bumper, wheel encoder, and tire pressure monitoring system (TPMS). The bumper and TPMS are both safety-critical sensors. Specifically, the bumper is used to detect collisions and is the last defense to prevent further damage when accidents occur. In the system control module, the chassis system is controlled and managed via the vehicle control unit (VCU), which is responsible for coordinating each module. This unit keeps communicating with the industrial personal computer (IPC), performing parameter checking and sending commands to other modules. In addition, the VCU is responsible for critical safety functions, such as the stopping signal from the emergency button. In our chassis, the 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 a maximum 80-km running distance. The lithium-ion battery powers the chassis, IPC, sensors, and accessories. It has a built-in battery management system to monitor and manage the battery. The removable design allows the vehicle to operate 24 h/day without stopping for a recharge. An onboard charger with a dc–dc converter takes about 5 h to fully charge the battery.

Autonomous Driving-Related Devices

The devices related to autonomous driving include the 1) computation platform, 2) sensors, and 3) auxiliary devices. For the computation platform, our vehicle is equipped with an IPC that has an Intel i7-8700 CPU with six cores and 12 threads, 32-GB memory, and a 1050Ti NVIDIA graphics card. It is able to run deep learning-based algorithms. In terms of sensors, as illustrated in Figure 2(a), our vehicle is equipped with four 16-beam lidars, one microelectromechanical system short-range lidar, four fish-eye cameras, 16 ultrasonic radars, one inertial measurement unit (IMU), and one high-precision global navigation satellite system (GNSS) that supports real-time kinematic **positioning** and heading vector positioning. The auxiliary devices include a 4G/5G data transfer unit (DTU), a human–machine interface, an LED display, and a 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.

> **Avoiding person-to-person transmission is an effective approach to control and prevent the pandemic.**

The Software System

Figure 3 portrays the software architecture of our autonomous vehicle. It can be generally divided into two parts: the

Figure 1. A worker dressed in a protective suit collects 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.

software system running on the vehicle and the software system running on the cloud.

The 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 through energy management and safety diagnosis. The applications on the 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 controller area network (CAN) by heartbeat protocols. When the 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 intervention.

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., the IMU or wheeled odometer. The lidar object detection module runs at 10 Hz according to the refresh rate of the lidar. All of the modules are developed on the Robot Operating System (ROS) to facilitate data communication.

The 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. The map server stores prebuilt 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 good traceback. The log and simulation server run the end-to-end simulator Carla and the 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).

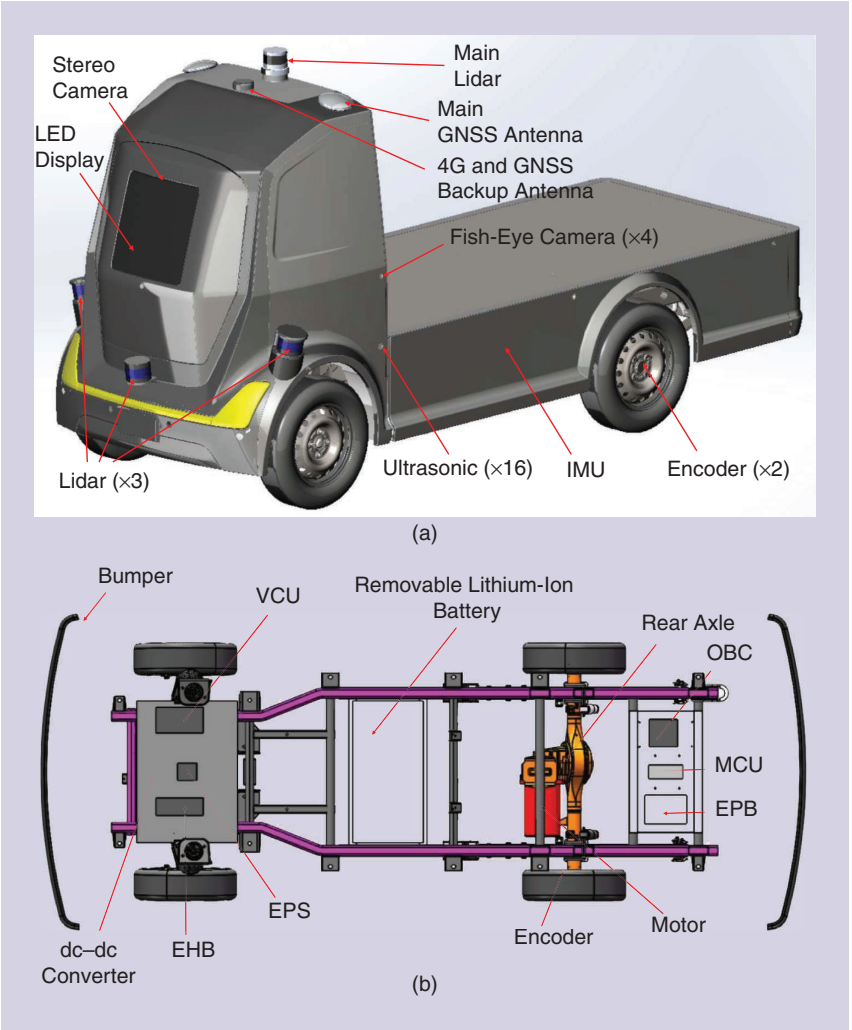

Figure 2. The sensors used in our vehicle and the modules in the chassis. Note that the cargo box is replaceable, but it is not shown in the figure. (a) The sensors on the vehicle (with the cargo box removed). (b) The sensors and control units on the mobile base (chassis). OBC: on-board charger.

Perception

Perception serves as the fundamental component of autonomous navigation. It provides the necessary information for planning and control. This section describes two key perception technologies used in our vehicle: multiple lidar-based 3D object detection and 3D point-cloud mapping.

Multiple Lidar-Based 3D Object Detection

3D object detection aims to recognize and classify objects as well as estimate their poses with respect to a specific coordinate system. We used multiple lidars for object detection. The first step was to calibrate the lidars. In this work, we proposed a marker-based approach [5] for automatic calibration without any additional

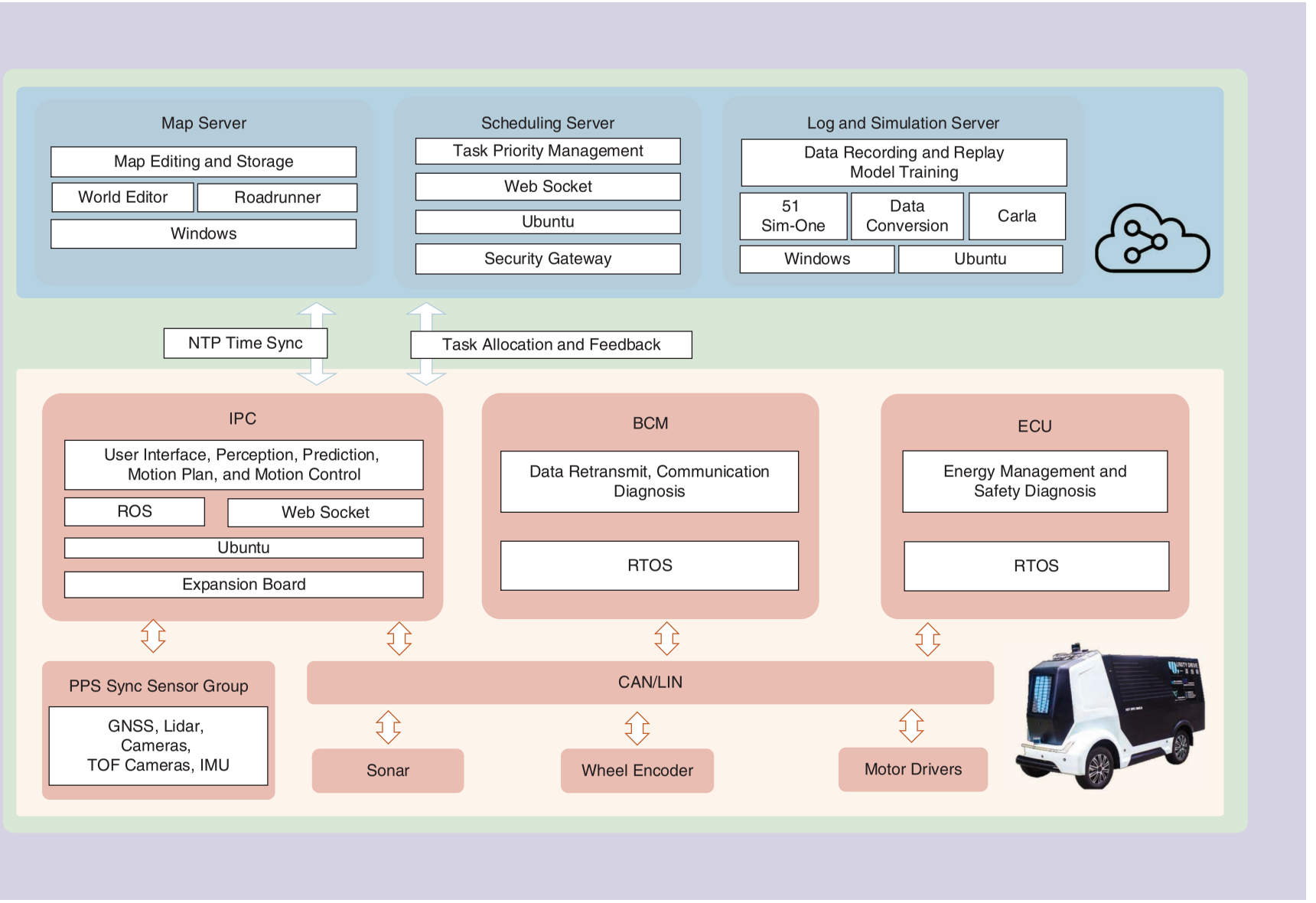

Figure 3. The software architecture of our vehicle. The part in the yellow box is running on the vehicle, and the part in the blue box is running on the cloud. The figure is best viewed in color. LIN: local interconnect network. TOF: time-of-flight.

sensors or human intervention. We assumed 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 successfully recovered 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 for refinement.

The overview of our 3D object detection is depicted in Figure 4. The inputs to our approach are multiple point clouds captured by different lidars. We adopted 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 transformed the raw point clouds captured by all of the lidars into the base frame and then fed the fused point clouds into the 3D object detector [4]. The final output was a series of 3D bounding boxes.

3D Point-Cloud Mapping

3D point-cloud mapping aims to build a 3D map of the traversed environments. Figure 5 illustrates the diagram of our mapping system. The inputs to the system are the raw data from the IMU and 3D lidar (i.e., accelerometer and gyroscope readings from the IMU and point clouds from the 3D lidar). The system starts with an adapted initialization procedure followed by two major submodules: 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 handheld 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 the lidar and IMU measurements in a local window. The mapping with rotational

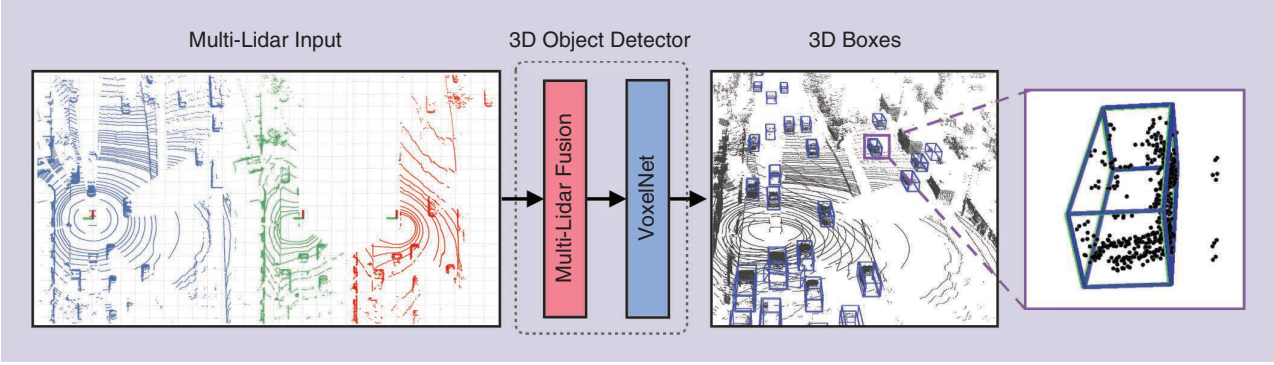

Figure 4. An overview of the 3D object detection module. The inputs are multiple point clouds captured by synchronized and well-calibrated lidars. We used an early fusion scheme to fuse the data from multiple calibrated lidars and adopted the VoxelNet [4] to detect 3D objects from the fusion results.

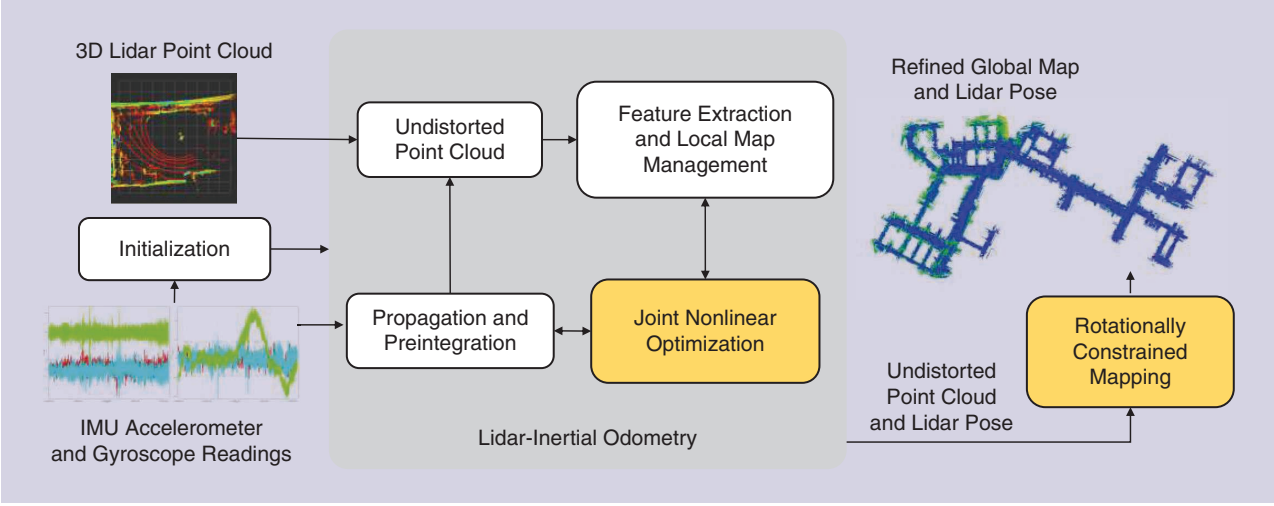

Figure 5. The schematic diagram of our 3D point-cloud mapping system. After the initialization, the system estimates the states and refines the global map and lidar poses in, respectively. The odometry and mapping submodules.

constraints further refines the lidar poses and the point-cloud map.

Planning

Planning enables autonomous vehicles to acquire future paths or motions toward the destination. Planning for autonomous driving is challenging because traffic environments usually include dynamic objects, bringing about risks and uncertainties [8]. Autonomous vehicles are required to interact with various road participants, such as cars, motorcycles, bicycles, pedestrians, and so on. 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. Figure 6 displays the four-layer planning process.

Route Planning

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 used a path planning algorithm 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 employed experienced drivers as teachers to demonstrate reference routes in these places. Figure 7 illustrates the global routes in the road network, with arrows indicating the forward directions.

Behavioral Planning

Behavioral planning determines the maneuvers for local navigation. It is a high-level representation of a sequence of vehicle motions. Typical maneuvers 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 for planning. Given the road map and the localization of the ego vehicle, features of the local area can be obtained. As seen in Figure 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.

Local Path Planning

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 the motion constraints of the vehicle to generate collision-free paths that conform to the lane boundaries and traffic rules. Figure 6 displays the online local path planning for driving on standard roads. Here, we planned 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 traveled along the path from the start position. We drew multiple samples with different speeds and

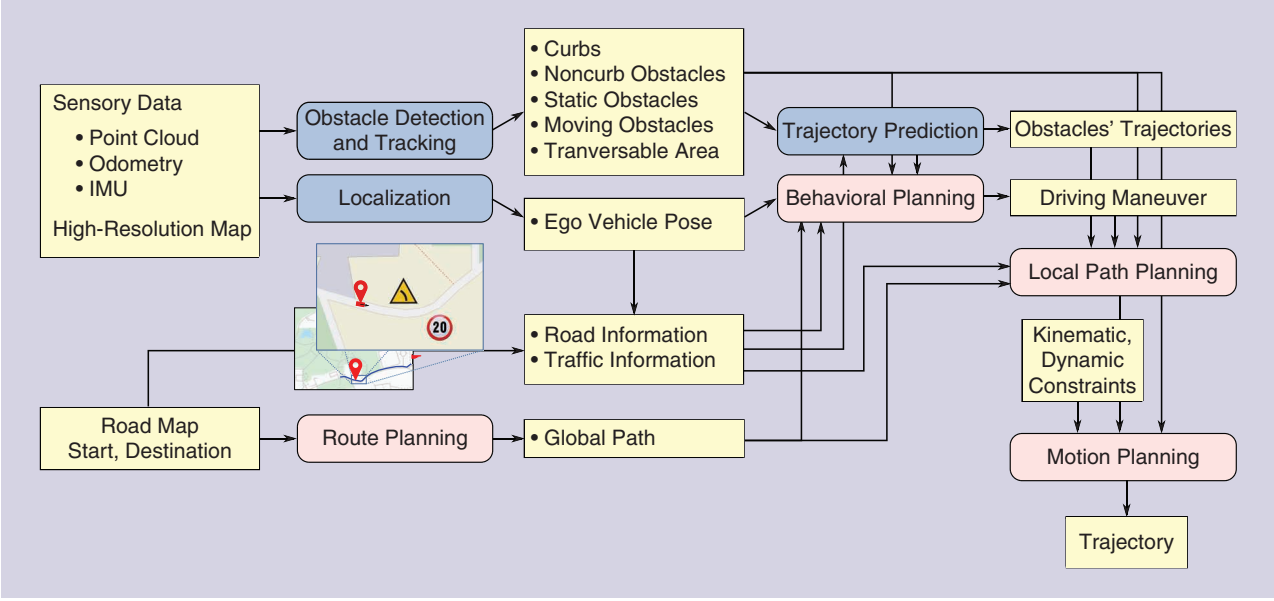

Figure 6. The schematic diagram illustrating the planning for our autonomous vehicle. The planning process consists of four layers: route planning, behavioral planning, path planning, and motion planning. The four layers are colored in pink.

Figure 7. 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; and (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, and (d) and (e) on urban main roads. Photos taken in Suzhou, Jiangsu, (g) a left turn, and (h) spraying disinfectant in a residential area. Photos taken in Shenzhen, Guangdong, (j) a heavy traffic environment, (k) surrounded by road users and meeting with oncoming cars, (l) a U-turn on a narrow road, (m) a traffic light at night, and (n) a contactless interaction while picking up a meal.

lateral offsets. Then a graph search method was adopted to search the path with the minimum cost. To define the cost of the coordinates of each curve, we took into consideration the quality of the curve, ending offset to the global path, and other factors (e.g., the potential trajectories of other agents).

Motion Planning

Given the planned path, motion planning is the final layer that optimizes the trajectory with dynamic constraints from the vehicle (i.e., the requirements for comfort and energy consumption). The planned trajectory specifies the velocity and acceleration of the vehicle at different time stamps, 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.

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. The schematic diagram of our controller is displayed in Figure 8.

As we can see, there are two inputs to the trajectory tracking controller. One is the trajectory $x(t)$, which includes the information (e.g., desired coordinates, curvatures, and 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 are difficult to obtain from sensors. In such cases, a state feedback estimator is required but not a must. In Figure 8, 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 mode (i.e., the remote control mode). The trajectory tracking controller functions only in the autonomous driving mode, which means that only in this mode does the lower controller take $u(t)$ as input.

Vehicle control can be divided into lateral control, which controls steer angles, and 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 proportional-integral-derivative (PID) controller highlighted in a dashed box in Figure 8 may be not required. The vector of two control quantities $u(t)$ (i.e., the steer angle and speed) will be directly given by the MPC. However, the

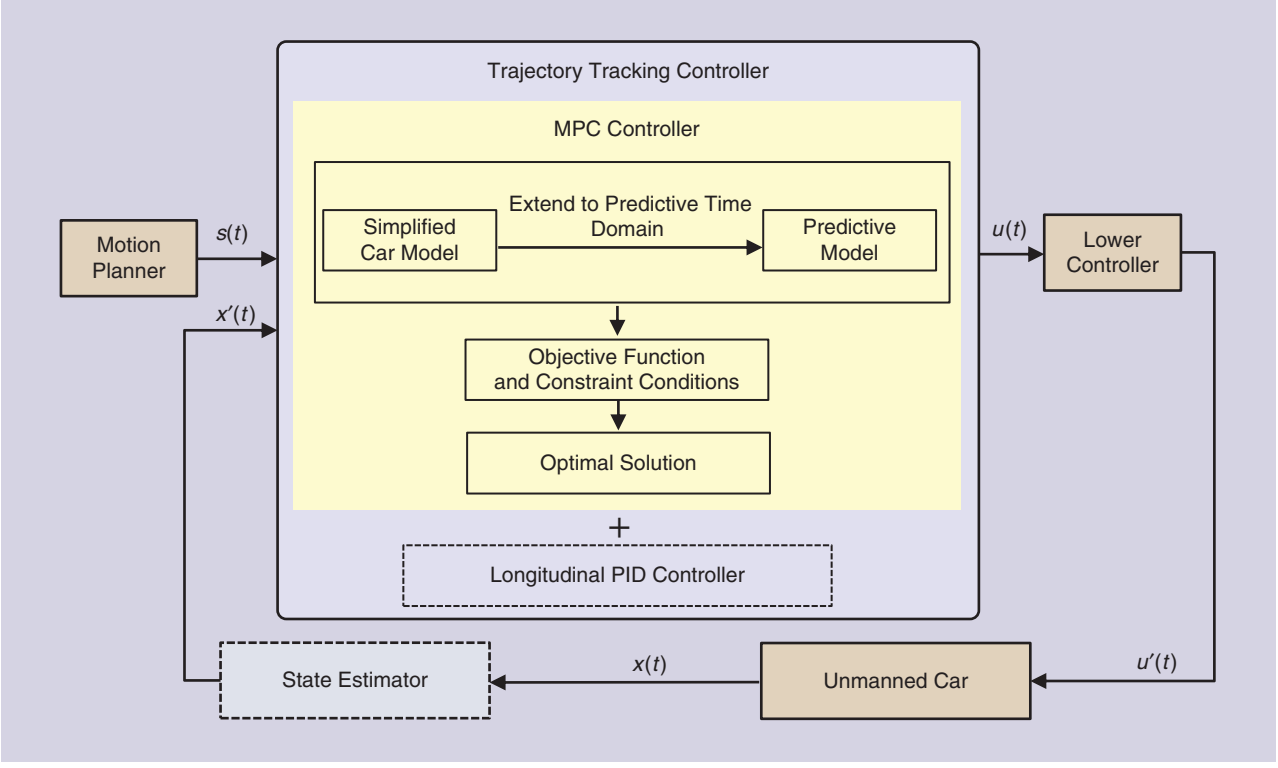

Figure 8. The schematic diagram of our controller. The main component is the trajectory tracking controller.

dynamics-based MPC is a standalone lateral controller, where 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 $u(t)$.

Evaluation

This section describes the real tasks of contactless goods transportation using our vehicle during the COVID-19 pandemic in China. From 2 February to 27 May 2020, we deployed 25 vehicles in three different cities (Zibo, Shandong; Suzhou, Jiangsu; and Shenzhen, Guangdong) in China. Our current server can handle 200 vehicles simultaneously. The total running distance of each vehicle reached 2,500 km. The details of the transportation tasks are summarized in Table 1. Selected demonstration photos of the tasks are displayed in Figure 7.

Our DBW chassis can be divided into four parts: 1) motion control, 2) electronic accessories control, 3) basic sensors, and 4) system control.

Note that, in the case of failure during autonomous navigation, such as unavoidable accidents and system errors, we built 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 a good signal, the latency is usually between 30 and 60 ms. 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 fewer than 20 ms. If the vehicle is out of line, it will stop immediately.

We tested the function with several vehicles on several real road environments, and the experimental results indicate that the function works well. Vehicle control is immediately taken over by a human driver in the case of any failure for autonomous navigation. We expect less human intervention during our goods transportation tasks. The performance was evaluated according to the number of occurrences of human interventions.

Lessons Learned and Conclusions

For object detection, we found that, in practice, real-time performance deserves much more attention than accuracy. 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 spatially sparse convolution in our 3D object detection module. As a result, the inference time was boosted from approximately 250 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 learned that the GPS localization results signaled the potential loop, leading to a consistent larger-region map for the lidar-based localization. In very crowded dynamic traffic environments, the system could be degraded by disturbances from moving objects [10]. To tackle this issue, we used semantic segmentation to remove movable objects to obtain clear point-cloud data.

For the planning part, we found that four-layer hierarchical planning is necessary and effective. The working environments of our vehicles are complex in terms of road structures, traffic conditions, driving etiquette, and so on. Layer-wise planning makes the system extensible for multiple environments and various requirements. Furthermore, it is essential to attach importance to the

Table 1. Details of the contactless 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, and slow speed
	Vegetable delivery	5.4 km	50 min	960 kg	Urban main road and residential area	Light traffic, heavy payload, and slow speed
Suzhou, Jiangsu	Meal delivery	1.2 km	30 min	80 kg (100 boxes of meals)	Urban main road	Medium traffic, left turn, and U-turn
	Road disinfection	0.6 km	10 min	—	Residential area	Narrow road, crowded, obstacle, and slow speed
Shenzhen, Guangdong	Meal delivery	1.6 km	20 min	64 kg (80 boxes of meals)	Urban main road and residential area	Heavy traffic, narrow road, and U-turn
	Meal delivery	4.0 km	40 min	96 kg (120 boxes of meals)	Urban main road and residential area	Heavy traffic, narrow road, and U-turn

uncertainty from the perception modules. This uncertainty comes from the limited accuracy of the perception system as well as the time delay in processing. For the planning, we avoided hitting the critical conditions and left 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 mentioned previously, the dynamic-based MPC is much more accurate than the kinematic-based MPC since the predictive model is more accurate. However, we found that complex model prediction was not the best option. With regards to low- and middle-speed operation environments, a simplified predictive model with multiple constraints would be sufficient.

From our 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, so we need to maintain a vehicle speed that is not too slow in order not to cause annoyance for other vehicle drivers. A 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 could feel frightened because they might not be able to anticipate the behaviors of the autonomous vehicle. For example, our vehicle often startled other vehicle drivers when it was reversing, even when the reversing light was flashing. Using a screen to notify the reversing behavior could alleviate the issue. In some cases, not strictly obeying the traffic rules would be good for autonomous navigation. For example, it would be wise to change lanes when a traffic accident happens ahead in the ego lane, even if the lane changing behavior is not allowed according to the traffic rules. Otherwise, the vehicle would not be able to move forward.

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 goods transportation. For quantitative measures, we can compute from Table 1 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 four instances of person-to-person contact in each task of 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 contactless 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.

Acknowledgments

This work was supported by the National Natural Science Foundation of China, under grant U1713211, Collaborative Research Fund by Research Grants Council Hong Kong, under Project C4063-18G, and HKUST-SJTU Joint Research Collaboration Fund, under project SJTU20EG03, awarded to Prof. Ming Liu.

References

[1] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Proc. Robot., Sci. Syst.* Ann Arbor, MI, 2016, pp. 1–9. doi: 10.15607/RSS.2016.XII.029.

[2] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE Int. J. Transp. Saf.*, vol. 4, no. 1, pp. 15–24, 2016. doi: 10.4271/2016-01-0128.

[3] M. R. Endsley, "Autonomous driving systems: A preliminary naturalistic study of the tesla model S," *J. Cogn. Eng. Decis. Making*, vol. 11, no. 3, pp. 225–238, 2017. doi: 10.1177/1555343417695197.

[4] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 4490–4499. doi: 10.1109/CVPR.2018.00472.

[5] J. Jiao et al., "A novel dual-lidar calibration algorithm using planar surfaces," in *Proc. 2019 IEEE Intell. Veh. Symp. (IV)*, pp. 1499–1504. doi: 10.1109/IVS.2019.8814136.

[6] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Sect. A*, vol. 34, no. 5, pp. 827–828, 1978. doi: 10.1107/S0567739478001680.

[7] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3d lidar inertial odometry and mapping," in *Proc. 2019 IEEE Int. Conf. Robot. Autom. (ICRA)*, pp. 3144–3150. doi: 10.1109/ICRA.2019.8793511.

[8] M. Liu, "Robotic online path planning on point cloud," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1217–1228, 2015. doi: 10.1109/TCYB.2015.2430526.

[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. doi: 10.1016/0005-1098(89)90002-2.

[10] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robot. Auton. Syst.*, vol. 89, pp. 110–122, Mar. 2017. doi: 10.1016/j.robot.2016.11.012.

Tianyu Liu, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: liutianyu@unity-drive.com.

Qinghai Liao, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: qinghai.liao@connect.ust.hk.

Lu Gan, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: lganaa@connect.ust.hk.

Fulong Ma, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: fmaaf@connect.ust.hk.

Jie Cheng, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: jchengai@connect.ust.hk.

Xupeng Xie, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: xxieak@connect.ust.hk.

Zhe Wang, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: wangzhe@unity-drive.com.

Yingbing Chen, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: ychengz@connect.ust.hk.

Yilong Zhu, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: yzhubr@connect.ust.hk.

Shuyang Zhang, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: shuyang.zhang@unity-drive.com.

Zhengyong Chen, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: chenzhengyong@unity-drive.com.

Yang Liu, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: yang.liu@connect.ust.hk.

Meng Xie, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: xiemeng@unity-drive.com.

Yang Yu, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: yyubj@connect.ust.hk.

Zitong Guo, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: guozitong@unity-drive.com.

Guang Li, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: liguang@unity-drive.com.

Peidong Yuan, Shenzhen Unity Drive Innovation Technology Co. Ltd., Shenzhen, 518000, China. Email: yuanpeidong@unity-drive.com.

Dong Han, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518000, China. Email: dong.han@siat.ac.cn.

Yuying Chen, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: ychenco@connect.ust.hk.

Haoyang Ye, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: hy.ye@connect.ust.hk.

Jianhao Jiao, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: jjiao@connect.ust.hk.

Peng Yun, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: pyun@connect.ust.hk.

Zhenhua Xu, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: zxubg@connect.ust.hk.

Hengli Wang, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: hwangdf@connect.ust.hk.

Huaiyang Huang, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: hhuangat@connect.ust.hk.

Sukai Wang, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: swangcy@connect.ust.hk.

Peide Cai, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: peide.cai@connect.ust.hk.

Yuxiang Sun, The Hong Kong Polytechnic University, Hung Hom, Kowloon, 999077, Hong Kong, China. Email: yx.sun@polyu.edu.hk.

Yandong Liu, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518000, China. Email: yd.liu@siat.ac.cn.

Lujia Wang, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518000, China. Email: lj.wang1@siat.ac.cn.

Ming Liu, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, 999077, China. Email: eelium@ust.hk.