

Cloud and Edge Computing for Connected and Automated Vehicles

Other titles in Foundations and Trends® in Electronic Design Automation

From CNN to DNN Hardware Accelerators: A Survey on Design, Exploration, Simulation, and Frameworks

Leonardo Rezende Juracy, Rafael Garibotti and Fernando Gehm Moraes
ISBN: 978-1-63828-162-7

Self-Powered Wearable IoT Devices for Health and Activity Monitoring

Ganapati Bhat, Ujjwal Gupta, Yigit Tuncel, Fatih Karabacak, Sule Ozev and Umit Y. Ogras
ISBN: 978-1-68083-748-3

On-Chip Dynamic Resource Management

Antonio Miele, Anil Kanduri, Kasra Moazzemi, Dávid Juhász, Amir R. Rahmani, Nikil Dutt, Pasi Liljeberg and Axel Jantsch
ISBN: 978-1-68083-578-6

Smart Healthcare

Hongxu Yin, Ayten Ozge Akmandor, Arsalan Mosenia and Niraj K. Jha
ISBN: 978-1-68083-440-6

Contracts for System Design

Albert Benveniste, Benoit Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger and Kim G. Larsen
ISBN: 978-1-68083-402-4

Cloud and Edge Computing for Connected and Automated Vehicles

Qi Zhu

Northwestern University
qzhu@northwestern.edu

Bo Yu

PerceptIn
bo.yu@perceptin.io

Ziran Wang

Purdue University
ryanwang11@hotmail.com

Jie Tang

South China University of Technology
cstangjie@scut.edu.cn

Qi Alfred Chen

University of California, Irvine
alfchen@uci.edu

Zihao Li

Purdue University
zihao@purdue.edu

Xiangguo Liu

Northwestern University
XiangguoLiu2023@u.northwestern.edu

Yunpeng Luo

University of California, Irvine
yunpel3@uci.edu

Lingzi Tu

Southwest Jiaotong University
linzitu@outlook.com

now

the essence of knowledge

Boston — Delft



Documents

Document Sharing Hub

Foundations and Trends® in Electronic Design Automation

Published, sold and distributed by:

now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:

now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

Q. Zhu *et al.*. *Cloud and Edge Computing for Connected and Automated Vehicles*.
Foundations and Trends® in Electronic Design Automation, vol. 14, no. 1-2, pp. 1–170,
2023.

ISBN: 978-1-63828-303-4

© 2024 Q. Zhu *et al.*

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The ‘services’ for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Foundations and Trends® in Electronic Design Automation

Volume 14, Issue 1-2, 2023

Editorial Board

Editor-in-Chief

Axel Jantsch
TU Wien, Austria

Editors

Bashir Al-Hashimi
King's College London

Jason Cong
University of California, Los Angeles

Andreas Gerstlauer
The University of Texas at Austin

Christoph Grimm
TU Kaiserslautern

Ahmed Jerraya
CEA-Leti

Wolfgang Müller
University of Paderborn

Partha Pande
Washington State University

Zebo Peng
Linköping University

Christian Pilato
Politecnico di Milano

Jaan Raik
Tallinn University of Technology

Alberto Sangiovanni-Vincentelli
University of California, Berkeley

Rishad Shafik
Newcastle University

Sandeep Shukla
Indian Institute of Technology, Kanpur

Norbert Wehn
University of Kaiserslautern

Editorial Scope

Topics

Foundations and Trends® in Electronic Design Automation publishes survey and tutorial articles in the following topics:

- System Level Design
- Behavioral Synthesis
- Logic Design
- Verification
- Test
- Physical Design
- Circuit Level Design
- Reconfigurable Systems
- Analog Design
- Embedded software and parallel programming
- Multicore, GPU, FPGA, and heterogeneous systems
- Distributed, networked embedded systems
- Real-time and cyberphysical systems

Information for Librarians

Foundations and Trends® in Electronic Design Automation, 2023, Volume 14, 4 issues. ISSN paper version 1551-3939. ISSN online version 1551-3947. Also available as a combined paper and online subscription.

Contents

1	Introduction	3
2	Mobility Digital Twins with Cloud and Edge Computing	7
2.1	Concept for Mobility Digital Twin Framework	9
2.2	Example Architecture for Mobility Digital Twin	14
2.3	Application: Personalized Adaptive Cruise Control with Human/Vehicle/Traffic Digital Twins	19
2.4	Summary	23
3	Practical Deployments of Edge and Cloud Computing in CAVs	24
3.1	Introduction	24
3.2	Emerging Use Cases of CAVs	26
4	Edge Architecture for CAVs	43
4.1	Computer Architectures for CAVs	43
4.2	Autonomous Driving Computing Acceleration on FPGAs	66
4.3	Software Systems	84
5	Cloud Architecture for CAVs	98
5.1	Distributed Computing for Autonomous Driving	99
5.2	Distributed Mapping	110
5.3	Simulation-Centric Cloud System	118

6	Safety and Security of Cloud- and Edge-enabled CAVs	128
6.1	Addressing Safety for Cloud- and Edge-enabled CAVs . . .	129
6.2	Addressing Security for Cloud- and Edge-enabled CAVs . .	139
	References	146

Cloud and Edge Computing for Connected and Automated Vehicles

Qi Zhu¹, Bo Yu², Ziran Wang³, Jie Tang⁴, Qi Alfred Chen⁵, Zihao Li³, Xiangguo Liu¹, Yunpeng Luo⁵ and Lingzi Tu⁶

¹*Northwestern University, USA; qzhu@northwestern.edu, XiangguoLiu2023@u.northwestern.edu*

²*PerceptIn, USA; bo.yu@perceptin.io*

³*Purdue University, USA; ryanwang11@hotmail.com, zihao@purdue.edu*

⁴*South China University of Technology, China; cstangjie@scut.edu.cn*

⁵*University of California, Irvine, USA; alfchen@uci.edu, yunpel3@uci.edu*

⁶*Southwest Jiaotong University, China; linzitu@outlook.com*

ABSTRACT

The recent development of cloud computing and edge computing shows great promise for the Connected and Automated Vehicle (CAV), by enabling CAVs to offload their massive on-board data and heavy computing tasks. Leveraging the Internet of Things (IoT) technology, different entities in the intelligent transportation system (e.g., vehicles, infrastructure, traffic management centers, etc.) get connected with each other, thus making the entire system smarter, faster, and more efficient. However, these advances also bring significant challenges to public authorities, industry, as well as scientific communities. In terms of system design and control, current cloud and edge architecture of CAVs need to be refined or even redesigned to better function under uncertainties in demand, and to better cooperate with

Qi Zhu, Bo Yu, Ziran Wang, Jie Tang, Qi Alfred Chen, Zihao Li, Xiangguo Liu, Yunpeng Luo and Lingzi Tu (2023), "Cloud and Edge Computing for Connected and Automated Vehicles", Foundations and Trends® in Electronic Design Automation: Vol. 14, No. 1-2, pp 1–170. DOI: 10.1561/10000000058.

existing conventional vehicles and infrastructure. From the performance assessment perspective, models and simulation tools based on artificial intelligence and big data have been widely developed for validation and evaluation of cloud computing and edge computing, but the validity of these models needs to be re-examined with field implementations.

Finally, while the increasing connectivity among vehicles and infrastructures may help improve their perception of the environment and enable coordinated decision making, it also presents new challenges to ensure system safety and security, with inherent disturbances to wireless communication networks and also the inevitably larger attack surface that may be exploited by malicious attacks. In this tutorial, experts from academia and industry will introduce the trends and challenges of applying cloud and edge computing for CAVs, highlight representative works in the literature and discuss their limitations, present new promising solutions, and outline future directions in research and engineering. Particular focus will be given to methodologies and tools for building digital twin frameworks with cloud and edge computing for CAVs, quantitative and formal analysis for ensuring CAV safety under disturbances and uncertainties, system-level CAV security threat landscape and defense solution space, and experiences from practical deployment of cloud and edge computing for CAVs.

1

Introduction

Autonomous driving (AD) has seen tremendous progress over the last two decades, propelled by the advancement of neural network-based machine learning techniques and their usage throughout system stages such as perception (Qian *et al.*, 2022), prediction (Jiao *et al.*, 2022; Jiao *et al.*, 2023a; Jiao *et al.*, 2023b), and planning (Liu *et al.*, 2022f; Liu *et al.*, 2023c; Liu *et al.*, 2023a). Significant progress has been made to improve AD performance in various traffic scenarios, including challenging ones such as unprotected left turns, highway merging, and lane changing. Meanwhile, connected vehicle (CV) technologies using vehicular ad-hoc networks enable information sharing among vehicles and surrounding infrastructures (Kenney, 2011; Zheng *et al.*, 2016b). This provides a great complement to the perception and prediction capabilities of individual vehicles, e.g., by sharing out-of-sight information or future motion plans that cannot be accurately predicted.

However, a number of challenges significantly impede the application of AD and CV techniques in realizing connected and automated vehicles (CAVs) in practice. Some of the major challenges include (1) taking full advantage of these techniques to maximize system performance in various applications and scenarios, (2) reducing additional costs to

deploy these techniques, including software and hardware cost, system maintenance and upgrade (Zheng *et al.*, 2016a; Zhu and Sangiovanni-Vincentelli, 2018), (3) ensuring system safety considering environmental uncertainty and disturbance, as well as the difficulty in analyzing neural networks (Zheng *et al.*, 2017a; Zheng *et al.*, 2019; Zhu *et al.*, 2020; Zhu *et al.*, 2021), and (4) defending possible cyber and physical attacks on CAVs (Liu *et al.*, 2023d).

Addressing these challenges and adopting AD and CV techniques require significant computational resources, which could be challenging to deploy on future production vehicles due to the additional cost, energy overhead, hardware maintenance difficulty, etc. The advancement of cloud and edge computing (Varghese *et al.*, 2021; Sánchez *et al.*, 2022; Gross *et al.*, 2022) provides an appealing way to overcome the computational resource limitations of individual vehicles. CAVs can communicate with edge devices that are roadside units (RSUs) and leverage the RSUs' affiliated local sensors and servers, and may also communicate with cloud devices that are far away and can connect to the internet for more information. Tasks with high resource demands and without real-time requirements, e.g., neural network model training, can be uploaded to the cloud, while tasks with moderate resource demands but requiring real-time communication, e.g., perception of the environment and coordination with other vehicles, can be handled by edge computing. Individual vehicles only need to equip the necessary hardware to maintain normal operation and ensure safety when edge and cloud computing are not available occasionally.

Many works have shown that the AD pipelines may perform poorly in long-tailed traffic scenarios (e.g., extreme weather conditions) and are vulnerable to various input noises and attacks such as blemished traffic signs or dirty patches on the road (Chen *et al.*, 2019b; Sato *et al.*, 2021; Zhang *et al.*, 2022). Some methods and frameworks are proposed to enhance the robustness and security of individual modules (Yang *et al.*, 2020; Jiao *et al.*, 2023b; Jiao *et al.*, 2023a) and the AD pipeline (Jiao *et al.*, 2021) of a single autonomous vehicle.

Edge and cloud computing can significantly improve system performance, safety and security by effective collaboration among vehicles and infrastructures. For instance, with the shared information from CAVs,

we can build a Sybil attack detection system (Luo *et al.*, 2021) based on a credibility-enhanced temporal graph convolutional neural network to defend the transportation system against Sybil attacks. At the same time, a cyber-physical credibility framework based on blockchain technology and vehicles' physical sensing capabilities can be maintained to build trust for connected vehicles (Liu *et al.*, 2021h), enabling quick reaction to attacks in a large-scale vehicular network with low resource overhead. Initial studies of such techniques have demonstrated effectiveness in defense against spoofing attacks, bad-mouthing attacks, and Sybil and voting attacks.

System efficiency can be maximized by allocating different tasks to cloud servers, edge devices, or individual vehicles. However, there are some open design challenges. For example, uploading long-tailed training examples to the cloud can build a robust and general model for vehicles in different areas, which can save effort in modeling training, safety and security verification. However, it leads to performance, privacy and security concerns. A general model does not necessarily perform better than a model trained with local traffic data. Individual vehicles may not want to publicize their daily travel trajectories, and even only a small amount of malicious training data can contaminate the model training in the cloud. Another example is collaborative perception with assistance from edge devices. Real-time local maps can be generated on edge devices by collecting shared information from individual vehicles, which can improve system safety, especially in occluded and complex scenarios. However, sharing raw data from Lidars and cameras costs lots of communication resources. Sharing processed information, e.g., detected objects, will lead to challenges in reaching a consensus on these objects and their positions, and each vehicle needs to equip corresponding computational devices.

In the following sections, we will discuss opportunities and challenges for CAVs with cloud and edge computing, as well as present promising solutions to address these challenges and enhance system performance, efficiency, safety and security. In particular, Section 2 presents a mobility digital twin framework at the transportation level for enabling edge and cloud computing for CAVs. Section 3 discusses insights in practical deployment of edge and cloud computing in CAVs. Sections 4 and 5

present software and hardware architecture designs for edge and cloud computing, respectively, that facilitate CAVs. Finally, Section 6 discusses safety and security challenges for cloud- and edge-enabled CAVs and presents potential solutions.

2

Mobility Digital Twins with Cloud and Edge Computing

The Digital Twin, as an emerging representation of Cyber-Physical Systems (CPS), attracted increasing attention over the past decade (Guo *et al.*, 2017). Based on a market research report, the global Digital Twin market size was valued at USD 5 billion in 2020, and is projected to expand to 86 billion in 2028 at a compound annual growth rate of 42.7% during this period (Grand View Research). It was also pointed out in their report that the automotive and transportation industry takes one of the largest shares in the global Digital Twin market in 2020, among other end-uses like manufacturing, energy, and healthcare.

The emergence of connected vehicle technology introduces another platform to implement the Digital Twin. Since the level of connectivity within our vehicles has greatly improved, these equipped vehicles are able to “talk” with other entities, such as with other connected vehicles through vehicle-to-vehicle (V2V) communications, with traffic infrastructures through vehicle-to-infrastructure (V2I) communications, and with cloud servers through vehicle-to-cloud (V2C) communications (Wang *et al.*, 2020d; Cheng *et al.*, 2020; Ma *et al.*, 2020; Silva *et al.*, 2021). Specifically, V2C communications allow connected vehicles to 1) upload their data to the cloud server, enabling Digital Twins to be

built in the digital (cyber) world based on their counterparts in the physical world; and 2) offload their onboard computations to the cloud server, enabling Digital Twins to build models and calculate guidance information through powerful cloud computing, which can then be fed back to connected vehicles.

In this section, a Mobility Digital Twin (MDT) framework is proposed, which is defined as an Artificial Intelligence (AI)-based data-driven cloud-edge-device framework for mobility services. This framework transforms connected vehicles into Internet of Vehicles (IoV) by leveraging IoT technologies. The cloud-edge architecture is built with Amazon Web Services (AWS) to accommodate the proposed MDT framework and implement its digital functionalities of storage, modeling, learning, simulation, and prediction.

Since traditional mobility system frameworks heavily rely on onboard storage and computing, their functionalities are limited by multiple constraints, such as computing power, accessibility to big data, and easiness of deployments and modifications. On the contrary, the proposed MDT framework in this section addresses these constraints by making the following contributions:

1. **Powerful:** The MDT framework allows users to swiftly adjust cloud resources, ensuring high computing power during peak demand periods.
2. **Shareable:** Data generated by users can be efficiently stored on the cloud/edge, enabling easy retrieval and utilization for seamless operations. It can also be shared with other users for on-demand micro-service utilization.
3. **Manageable:** Users can quickly deploy and activate micro-services on the cloud platform, improving efficiency and reducing setup time. The framework supports OTA updates for easy system upgrades.
4. **Extendable:** The MDT framework enables easy implementation of diverse mobility micro-services with minimal changes to the architecture and data structure.

2.1 Concept for Mobility Digital Twin Framework

There are three layers in the MDT framework proposed in this section: 1) the physical space where human beings, vehicles, and traffic infrastructure reside; 2) the digital space where the digital replicas of those physical entities are located at; and 3) the communication layer which plays a crucial role in facilitating connectivity and allows data streaming between the physical and digital spaces.

Three entities are considered in this MDT framework: *Human*, *Vehicle*, and *Traffic*. Given the existence of the communication layer, each of the entities can be connected to the digital space (e.g., Internet) and exchanges data with each other. Therefore, this MDT framework is a good representation of the IoT, and it allows connected vehicles to act as IoVs with IoT technologies. In this section, we provide a deep dive into this MDT framework regarding all three aforementioned layers, introducing their sections with respect to their concepts and functionalities.

2.1.1 Communication and Data Workflow

The communication layer of this MDT framework sits between the physical space and digital space, and it provides seamless connections between these two spaces. This MDT framework's end-to-end process starts from sampling data in the physical space, where all or part of the data is transmitted upstream to the digital space via the communication layer. Those data will go through one or multiple processes in the digital space internally, including storage, modeling, learning, simulation, and prediction, and the resulting data is transmitted downstream to the physical space via the communication layer. Those data, upon receiving, is applied by the actuators of the physical space to fulfill the end-to-end process.

Since cloud computing is leveraged in this MDT framework, the digital space of the framework is deployed fully or partially on the commercial and/or private cloud. Therefore, the communication module needs to provide access to the cloud for the physical space, which is either direct access or indirect access (via edges). The MDT framework does

not necessarily require any specific wireless communication technology (DSRC (Kenney, 2011), C-V2X (3GPP), or something else in the future) to be served as the communication layer, as long as it can be applied to transmit data between the physical space and the digital space.

2.1.2 Physical Space

If we consider this MDT framework as an end-to-end framework, then the physical space is in charge of both ends of this framework, namely, sampling and actuation. We assume no (or only minimal) computing work needs to be conducted in the physical space, since all (or majority) of that is offloaded to the digital space through the communication layer.

For sampling, sensors in the physical space detect the dynamic status, operating process, or event occurrences, and then aggregate these measurements under various resolutions for their transmission to the digital space. On the other hand, once the processed results are received from the digital space, actuation can be made by physical entities to fulfill this end-to-end framework. Generally, the physical space is defined on a world coordinate, which may contain all the transportation-related physical entities, and can be considered in three aspects: *Human*, *Vehicle*, and *Traffic*.

Human

In this framework, all human beings involved in the transportation system are considered, which include not only drivers but also passengers, pedestrians, cyclists, etc. The sampling process can be accomplished by the human-machine interface in an active manner, or by the in-cabin status sensing, human wellness monitor, and other perception sensors in a passive manner.

The actuation process of the *Human* section in this MDT framework is mainly conducted by drivers. In a transportation environment, if drivers can be provided with additional information from the digital space of this MDT framework, such as adjacent vehicle's lane-change possibility (Liao *et al.*, 2022) or upcoming signal phase and timing

(Wang *et al.*, 2020f), their actuation will be more accurate and in turn benefit other entities in the transportation system.

Vehicle

Vehicle is the core of this MDT framework, as it is the host of drivers and passengers, and also the fundamental component of traffic. Specifically for the *Vehicle* section, the localization module (i.e., GNSS), the perception sensors (i.e., ultrasonic, camera, radar, and/or LiDAR), together with vehicle CAN BUS are in charge of the sampling process. Related data, such as positions, speeds, and accelerations of the ego vehicle and its surrounding vehicles can be sampled from these physical components, and then be propagated to the digital space through communication.

The actuation process of the *Vehicle* section in this MDT framework is conducted by the vehicle steering system, accelerator, and brake. These physical components are able to actuate any lateral or longitudinal control command received from the digital space, and therefore allow the vehicle to achieve its desired motion.

Traffic

The *Traffic* section in the physical space includes various traffic infrastructures, such as traffic signals, roadside units, camera/radar/loop detectors, and electronic traffic signs. These physical components are able to either generate data (e.g., signal phase and timing) by themselves, or measure data (e.g., traffic count and traffic flow) generated by other traffic entities. Such data is sampled and sent to the digital space through the communication layer, benefiting other sections of this MDT framework.

On the other hand, guidance or adjustment received from the digital space can also be actuated by the *Traffic* section to improve the safety and efficiency of the large-scale traffic network. For example, the signal phase and timing of traffic lights can be adjusted to better serve different traffic flows under different situations (Wei *et al.*, 2018). Guidance or warning information can be broadcast to connected vehicles via roadside units, and to all traffic entities via electronic traffic signs.

2.1.3 Digital Space

The aforementioned physical space of this MDT framework handles both ends of this end-to-end framework (i.e., sampling and actuation). On the other hand, the digital space is in charge of the processes between both ends: storage, modeling, learning, simulation, and prediction. One of the biggest strengths of this MDT framework is the data lake, which is a centralized repository that allows structured or unstructured data at any scale to be stored.

Note there exists a misunderstanding about the Digital Twin in the research community, where some simply consider Digital Twin technology as a modeling and simulation technology. In our MDT framework, modeling and simulation are part of the digital processes that are enhanced by the data lake and data sharing in the digital space, where co-simulation platforms can be built to synchronize data from multiple simulators (such as the Unity-SUMO-AWS integrated platform (Zhao *et al.*, 2021c)). However, our MDT framework is more than just modeling and simulation, where other digital processes (i.e., storage, learning, and prediction) play equivalently crucial roles in the digital space. All of the aforementioned digital processes can be applied to mobility micro-services, and they are realized in a more powerful, shareable, manageable, and extendable manner by leveraging cloud computing and edge computing.

Human Digital Twin

Human Digital Twins are digital replicas of real humans in the physical space. This section in the digital space has a human data lake that stores all data sampled from the *Human* section in the physical space, where different humans have their personal databases to be differentiated from others. With real-time data sampling and historical data storage, the *Human Digital Twin* is able to classify drivers into specific driver types by machine learning algorithms like k-nearest neighbors (KNN), and to provide guidance in a customized or personalized manner (Wang *et al.*, 2020e). Taking advantage of the data coming from the *Vehicle* section, the *Human Digital Twin* can also predict future behaviors of drivers (e.g., lane-change intention (Shou *et al.*, 2020)) and detect their

anomalies (Zhang *et al.*, 2017). The results of the aforementioned micro-services can be applied to third parties such as insurance companies, where they can further build a micro-service to set the insurance pricing for different drivers based on their driving behaviors (Weidner *et al.*, 2017).

Vehicle Digital Twin

Vehicle Digital Twins are digital replicas of real vehicles in the physical space. They store data received from connected vehicles, including information about the ego vehicle and its surroundings. This data can be shared with other digital twins for various purposes, such as cooperative localization, perception, planning, and control among connected vehicles. The *Vehicle Digital Twin* also enables predictive maintenance by modeling and simulating time-series vehicle data, allowing for the prediction of potential component failures in the future (Masuda *et al.*, 2019). This information can be used to schedule maintenance before breakdowns occur.

Traffic Digital Twin

Traffic Digital Twins are the digital replicas of traffic infrastructures, which receive data from the *Traffic* section in the physical space. Such sampled data, like signal phase and timing, traffic count, and traffic flow, can be stored in the traffic data lake for future reference. It can also be used for multiple traffic micro-services in real time, such as monitoring the traffic condition (Kumar *et al.*, 2018), variable speed limit (Papageorgiou *et al.*, 2008), routing and navigation (Pan *et al.*, 2020), ridesharing planning (Tripathy *et al.*, 2020), and parking management (Higuchi *et al.*, 2019).

The *Traffic Digital Twin* can be enhanced by the communication among these Digital Twin sections. For example, the micro-service of routing and navigation can be carried out solely by the real-time traffic flow data sampled from camera/radar/loop detectors in the real world. However, they can be further enhanced if behavior preferences are set by the *Human* section and predictions are made by the *Human Digital*

Twin (e.g., a driver/passenger always goes to grocery stores when his/her commute route is highly congested). Additionally, if the *Vehicle* section detects the fuel/battery level is low and sends that to the *Vehicle Digital Twin*, it can also assist the routing and navigation micro-service to find a gas/charging station near a user-preferred grocery store along the original route.

2.2 Example Architecture for Mobility Digital Twin

In this section, we build an example architecture for the proposed MDT concept by leveraging cloud computing and edge computing, enabling both real-time and bulk-batch ingestion, processing, and analytics of mobility data. As shown in Figure 2.1, the architecture can be divided into four layers: 1) The cloud layer, which is built on AWS and its Virtual Private Cloud (VPC); 2) The edge layer, which has a computing component, a communication component, and a storage component; 3) The device layer, which generates data and consumes guidance; 4) The API (Application Programming Interface) layer, which hooks up the cloud layer with external APIs.

The major purpose of designing this architecture is to accommodate the proposed MDT framework shown in Section 2.1, so the Digital Twin does not just remain in the conceptual phase, but can also be deployed in the real world with the help of cloud computing and edge computing. Particularly, the physical space of the MDT framework mentioned in Section 2.1 is represented by the device layer of this example architecture, where mobile apps, simulators, real vehicles and Radio Control (RC) vehicles sample data from *Human*, *Vehicle* and *Traffic*, and also actuate commands received from the edge and cloud layers. The communication level of the MDT framework is positioned within the edge layer's communication component of this example architecture. The digital space of the MDT framework, along with its data lakes and micro-services, spans the whole part of the cloud and API layers, as well as part of the edge layer (except for its communication component).

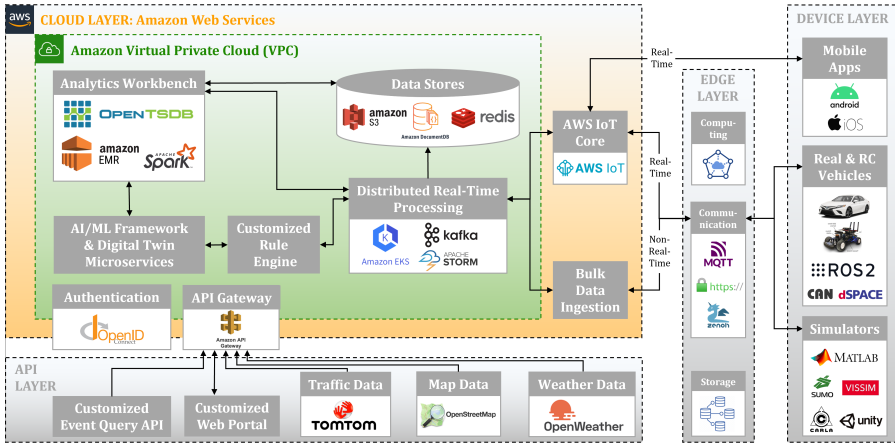


Figure 2.1: Example architecture of the proposed MDT with the cloud layer, edge layer, device layer, and API layer

2.2.1 Cloud Layer

In Amazon VPC of the cloud layer, there are five different modules: Distributed Real-Time Processing, Data Stores, Analytics Workbench, Rule Engine, and AI/ML Framework and Digital Twin micro-services. Specifically, in Distributed Real-time Processing, we leverage existing tools like Amazon EKS, Apache Kafka, and Apache Storm to provide real-time processing and analytics. The Analytics Workbench is the workhorse for big data analytics. It consists of OpenTSDB, which is a distributed, scalable, time-series database built on top of Hadoop and HBase. This supports a writing rate of up to millions of entries per second, supports data storage with millisecond-level precision, and preserves data permanently without sacrificing precision. In addition, Apache Spark, a distributed processing system, is used to conduct predictive analytics using Amazon EMR clusters.

Rule Engine service evaluates the rules configured for entities (e.g., humans and vehicles) on the data received from the Kafka queue, and redirects it to AI/ML Framework and Digital Twin Micro-services based on the rule validation result. AI/ML Framework and Digital Twin

Micro-services are the core of this cloud-edge architecture, where end users are able to implement customized algorithms and applications with various objectives. This module processes time-series data sent from the physical space using statistical techniques, and sends guidance back to the entities in the physical space. The data workflow is triggered via Apache Airflow, an open-source workflow management platform.

Data Stores of our cloud-edge architecture are made up with: 1) Amazon S3, a scalable storage infrastructure to build our Digital Twin data lakes; 2) Amazon DocumentDB (with MongoDB compatibility), a database service that is purpose-built for JSON data to execute flexible, low latency queries to obtain a near real-time record of events in parallel on a massive scale; and 3) Redis, an open-source, highly replicated, non-relational kind of database and caching server.

Outside of Amazon VPC but still inside of the cloud layer sits AWS IoT Core, which enables the connection between IoT devices (such as mobile apps, simulators, real vehicles, and RC vehicles in this study) and AWS cloud without the need to provision or manage servers. It supports various devices and messages, and can process/route those messages to AWS endpoints/devices reliably and securely. A Bulk Data Ingestion module is also developed in this cloud-edge architecture, enabling the ingestion of terabytes of data in batch mode into our data lakes. Some scenarios where this module can be triggered are 1) end-of-vehicle-trip bulk data ingestion; 2) periodic bulk data ingestion; 3) event-triggered data ingestion; 4) in-vehicle data logging. OpenID Connect is a simple identity layer on top of the OAuth 2.0 authorization protocol, which is adopted in this cloud-edge architecture to verify the identity of end users based on the authentication performed by an authorization server, as well as to obtain the basic profile information about end users.

2.2.2 API Layer

External data sources can be integrated into this architecture through the API layer, which enriches the functionalities of digital micro-services. The API layer is connected with the cloud layer of this example architecture through Amazon API Gateway, which is an AWS managed service that can create, publish, maintain, monitor, and secure APIs at any

scale. The API Gateway may act as the “front door” for applications to access data or functionalities from our back-end services.

In the API layer, traffic data (TomTom), map data (OpenStreetMap), and weather data (OpenWeather) can be connected to Amazon API Gateway via HTTPS. With such data, more micro-services can be deployed in the *Traffic Digital Twin* of the proposed MDT framework, and hence provide better guidance towards humans and vehicles in the physical space. Additionally, a customized Event Query API is developed to receive notifications of events and fetch data from the cloud layer. Examples of using this API are event-triggered detection of anomalous driving behavior, and notification of data ingestion in the cloud layer at the end of a trip.

2.2.3 Edge Layer

Although the cloud layer of this example architecture is designed to accommodate most of the digital processes of the proposed MDT framework, this does not necessarily indicate all digital processes mentioned in Section 2.1 must sit on the cloud. In fact, it gets difficult sometimes for connected vehicles to have the cloud access, since they continuously move around and may lose internet connection every now and then. Therefore, a hybrid approach with cloud computing and edge computing can meet the requirements of ultra-low latency for running safety-critical micro-services at the edge (e.g., road-side units), and of extensive resources for running data-driven micro-services on the cloud (Wang *et al.*, 2020a). Edge computing has already been widely researched by various works in the field of connected vehicles (Santa *et al.*, 2020; Santa *et al.*, 2019), and also deployed in the real world in projects like 5G-MOBIX and initiatives like Automotive Edge Computing Consortium (AECC).

In this example architecture, an edge layer is designed with three different modules: computing, communication, and storage.

- The computing module provides a hierarchical computing platform for connected vehicles, where heterogeneous computing nodes (e.g., microprocessor, GPU, and TPU) are responsible for processing a variety of tasks offloaded by vehicles. Sophisticated middleware

mechanisms are designed for determining which type of computing nodes should be selected to handle specific tasks and services.

- The communication module enables real-time data exchange among the cloud, edge, and device layers. The protocols implemented in the communication module are HTTPS and two lightweight publish-subscribe network protocols, MQTT and Zenoh. Zenoh has been designed to address the needs of applications that deal with data in movement, data at rest and computation in a scalable, efficient and location transparent data manner.
- The storage module is capable of caching temporary data. The cached data could be either downloaded from the cloud layer or uploaded from the device layer. Some representative mobility applications that could benefit from the storage module include OTA (i.e., caching the update content distributed by the cloud at the edge) and crowd-sourcing high-definition (HD) map (i.e., caching uploaded sensor data from vehicles to update the local map fragment accordingly at the edge). In our case study shown in Section 2.3, the storage module of the edge layer also plays a crucial role to store the raw CAN BUS data of the vehicle.

2.2.4 Device Layer

The device layer of this example architecture is shown on the right side of Figure 2.1, which stands for the *Human*, *Vehicle* and *Traffic* building blocks in the physical space of the MDT framework. Mobile apps are designed for both Android and iOS, where end users' position and speed data (measured by GPS and gyroscope) can be uploaded to AWS IoT Core directly via MQTT in real time.

Real vehicles and RC vehicles are the major data sources in the device layer of this example architecture. Dynamics data from CAN BUS (or ROS2 for RC vehicles), position data from the localization module, together with perception data from perception sensors (e.g., camera, radar, LIDAR), is sampled in real time and pushed to the edge layer. On the other hand, guidance from the digital space of the MDT

framework is received from the edge layer, and gets actuated on real vehicles and RC vehicles.

External simulators, such as MATLAB, microscopic traffic simulators SUMO and PTV VISSIM, and game engine-based simulators CARLA (Dosovitskiy *et al.*, 2017) and Unity, are used in this architecture to emulate the physical devices of the MDT framework. Note these simulators in the device layer only play the role as physical entities in the physical space, i.e., only for sampling and actuation processes. However, simulators can also be deployed in the cloud layer or the edge layer to play the role as digital entities, i.e., for modeling, learning, simulation and prediction.

2.3 Application: Personalized Adaptive Cruise Control with Human/Vehicle/Traffic Digital Twins

This section introduces a case study of the Personalized Adaptive Cruise Control (P-ACC) system is conducted. Real-world experimental results showcase the effectiveness of P-ACC while functioning in the MDT framework, compared to traditional ACC systems that purely rely on on-board sensing and computing.

The P-ACC system integrate micro-services from *Human*, *Vehicle*, and *Traffic Digital Twins*. Traditional ACC systems allow the ego vehicle to travel at a set speed and/or maintain a desirable gap with its preceding vehicle with the help of ranging sensors like radars, however, they require drivers to manually adjust speed and gap settings based on dynamic environments (Toyota, 2021; Ford, 2021; Volkswagen, 2021). They do not consider each driver's personalized driving style, and cannot account for the changes of environmental factors, such as traffic conditions, road types, and weather types (Wang *et al.*, 2021b).

As shown in Figure 2.2, the P-ACC system is proposed with an edge layer on the vehicle and a cloud layer with AWS. While the computer on the edge layer (i.e., Nvidia Jetson TX2) conducts some basic edge computing tasks to filter the raw data generated by the vehicle CAN BUS, most computing tasks are offloaded from the edge layer to the cloud layer (i.e., AWS), where the *Human/Vehicle/Traffic* Digital Twins get deployed.

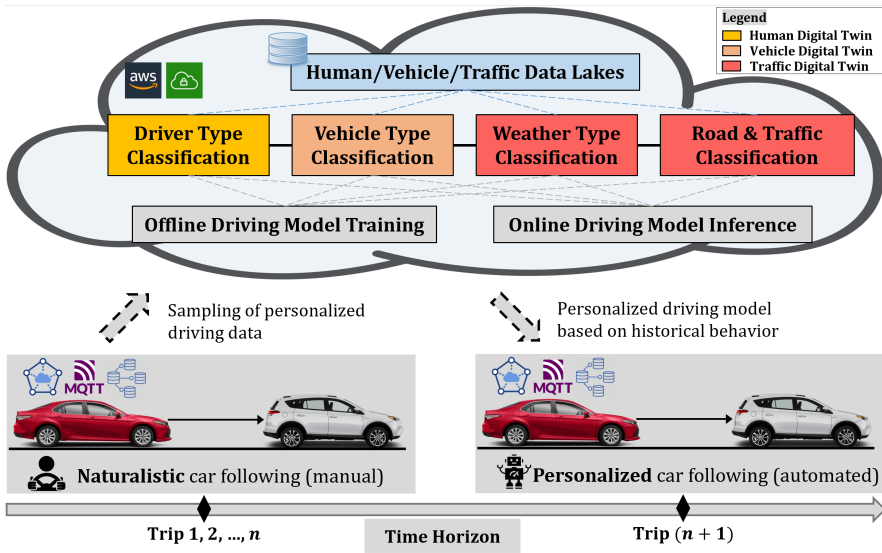


Figure 2.2: The Personalized Adaptive Cruise Control system integrates micro-services from the Driver Digital Twin, the Vehicle Digital Twin, and the Traffic Digital Twin.

Once the personalized driving data sampled from a driver's naturalistic car-following behavior gets filtered on the edge layer and uploaded to the cloud layer, it is further classified by the following Digital Twins:

- The *Human Digital Twin*, with the micro-service of driver type classification (aggressive, neutral, conservative, etc.).
- The *Vehicle Digital Twin*, with the micro-service of vehicle type classification (ego/preceding vehicle being a sedan, SUV, truck, etc.).
- The *Traffic Digital Twin*, with the micro-services of weather type classification (sunny, cloudy, rainy, snowy, foggy, etc.), road type classification (rural, urban, highway, etc.) and traffic type classification (congested, moderate, light, etc.).

Aforementioned classifications can be conducted either by simple rule-based algorithms or complex machine learning algorithms (Wang

et al., 2020e). Once the driving data is classified into various clusters, the data in each cluster can be used by machine learning algorithms to train a personalized driving model. In this case study, the maximum entropy Inverse Reinforcement Learning (Max-Ent IRL) algorithm (Zhao *et al.*, 2022) is adopted, where the probability of a trajectory is proportional to the sum of the exponential rewards accumulated along the trajectory.

Based on the recovered reward function from the Max-Ent IRL algorithm, the particular driver's desired car-following gap (e.g., 70 m) at each speed value (e.g., 35 m/s) can be calculated by the following equation and then put into a look-up table:

$$d_{\text{des}}(v) = \arg \max_g r(v) \quad (2.1)$$

This $d_{\text{des}}-v$ table is the trained driving model for the certain cluster of driving data, which can be inferred in real time with the match of driver, vehicle, weather, road, and traffic types.

A real-world end-to-end test is conducted, which utilizes the CAN BUS data generated from a Lexus LS test vehicle, including 15 data fields such as position, speed, acceleration, and so on. Time-series data generated by naturalistic driving gets filtered on the edge and sent to AWS through edge communication module every 10 minutes. When the driver turns on the ACC feature, the cloud layer infers the particular driving model trained with the data in the particular driver/vehicle/weather/road/traffic type, and pushes that model to the vehicle through the edge communication module. ACC at this time will satisfy this driver's personalized car-following preference, and hence acts as a P-ACC.

The benefit of the proposed P-ACC with the MDT framework is quantitatively measured using the takeover percentage. Takeover denotes the status where the driver steps onto the acceleration pedal or brake pedal when he/she feels uncomfortable. The takeover percentage is the takeover time divided by the overall P-ACC (or ACC in the baseline) activation time, and the results are presented in Table 2.1. Based on the test results from three different drivers in different environments, the takeover event happens 86.4% less frequently when drivers use the proposed IRL-based P-ACC, compared to a traditional PID-based ACC that purely relies on real-time on-board computing.

Table 2.1: Takeover Results of P-ACC Compared to Traditional ACC (Percentage of Takeover Duration during a Trip)

Driver	P-ACC (Proposed)	ACC (Baseline)	Improvement
A	3.5%	18.5%	81.1%
B	1.5%	17.4%	91.4%
C	1.6%	12.0%	86.7%
Average	2.2%	16%	86.4%

Table 2.2: Latency Results of Processing 10-min CAN BUS Data and Querying the Data Lakes

Data Frequency	Uplink Data Total Size	Uplink Latency	Cloud Computing+ Downlink Latency	Overall Latency
0.5 Hz	12 MB	1.4 s	16.1 s	17.5 s
1 Hz	24 MB	2.1 s	16.1 s	18.2 s
5 Hz	120 MB	10.6 s	16.1 s	26.7 s
10 Hz	240 MB	21.1 s	16.1 s	33.2 s

The end-to-end latency results are shown in Table 2.2, indicating that under different sampling frequency of the CAN BUS data, the uplink latency is relatively proportional to the uplink batch size, with a minimum of 1.4 s for 12 MB and a maximum of 21.1 s for 240 MB. However, the majority of the end-to-end latency is made up by the aforementioned cloud computing processes, which roughly takes 16 s under different data frequency. This means once the P-ACC feature is switched on by a driver, it needs a 16 s warm-up period to get the corresponding IRL-based personalized model. During this warm-up period, the baseline PID model can be run to guarantee car-following safety.

In a nutshell, the end-to-end testing showcases the capability of the proposed MDT framework to execute complex cloud computing processes based on bulk uplink data within a rational range of latency, and the computing results improve the performance of commercially available vehicle applications.

2.4 Summary

In this section, a Mobility Digital Twin (MDT) framework has been developed for connected vehicles with the help of cloud computing and edge computing. In this section, the proposed MDT framework has been demonstrated with details regarding all of its components: *Human*, *Vehicle*, *Traffic*, together with their associated *Human Digital Twin*, *Vehicle Digital Twin*, and *Traffic Digital Twin*.

The public cloud service Amazon Web Services (AWS) has been adopted in this study to design the cloud-edge architecture, which deploys the MDT framework and makes it a reality rather than a concept. To showcase the effectiveness of the MDT framework, a case study of Personalized Adaptive Cruise Control (P-ACC) has been conducted leveraging the key micro-services of the MDT framework: user management and driver type classification of the *Human Digital Twin*, cloud-based Advanced Driver-Assistance Systems (ADAS) of the *Vehicle Digital Twin*, and traffic flow monitoring and variable speed limit of the *Traffic Digital Twin*.

In conclusion, scalability, reliability, security, cost, latency, and fidelity are the key factors to consider when designing any Digital Twin frameworks. Along with the rapid development of the Digital Twin, it can be envisioned that more related studies will be conducted in the mobility domain to tackle the real-world challenges in the near future.

3

Practical Deployments of Edge and Cloud Computing in CAVs

In the past decade, we have developed and commercially deployed various types of connected and automated vehicle (CAV) systems. In the following three sections, we delve into the practical deployment and development experiences of edge and cloud computing systems for CAVs. We will introduce different use cases of CAVs, review the design methodology, as well as introduce specific design examples. It is our hope that this tutorial will shed light on advancing systems design for CAVs.

3.1 Introduction

In 2017, at the beginning of the autonomous driving boom, we wrote an article to predict the development of CAVs (Liu *et al.*, 2017a). We predicted that from 2017 to 2020, the whole industry would explore different ways to develop commercial CAVs, focusing on algorithm design, edge computing system design, and the development of cloud platforms. Then we predicted that starting 2020, we would enter the mixed-mode era, in which manned vehicles and CAVs will co-exist. Given that the lifetime of a vehicle is around ten to fifteen years, we foresaw that this era would last about two decades. Then we predicted

that by 2040, all vehicles should be completely autonomous and it would be illegal for human to drive on public roads. By then, we may have created completely new traffic ecosystems where all vehicles are centrally controlled in an infrastructure-vehicle cooperative autonomous driving (IVCAD) system. We also predicted that by 2040 autonomous transportation would become a basic utility just like electricity and water. Consequently, traffic fatality would drop to almost zero from more than a million a year today. We would need far fewer vehicles as they can be shared efficiently. Finally, we would have much less air pollution because all vehicles would be powered by new energy sources instead of fossil fuels.

Revisiting these predictions today, we still feel that these predictions are valid as much more developments are required for the edge computing systems, cloud computing systems, and IVCAD systems for ubiquitous commercial deployment of CAVs. Also, in the past few years, as a startup in the autonomous driving space, we have evolved our technology stack significantly to cope with the rapid changes of autonomous driving use cases (Yu *et al.*, 2021; Liu *et al.*, 2020b), throughout the process we have learned a lot of lessons from the perspective of engineering management for autonomous driving technology development (Liu, 2020a).

The most important insight we have gained in the past decade is that, as an increasing number and kinds of autonomous machines enter our daily life, the age of autonomous machines is upon us and a whole new era of information technology begins (Liu and Gaudiot, 2022). Yet, autonomous machines are extremely complex systems that integrate many pieces of technologies and for autonomous machines to become an integral part of our daily life, we are still facing many technical and non-technical challenges. Particularly, CAVs are the crown jewel of the autonomous machine age. For instance, in 2021, the global automotive industry enjoyed a market size of \$2.86 trillion from car sales alone, but less than 5 percent of the cars sold are equipped with some degree of autonomous driving capabilities. That leaves tremendous room for future growth. Also, the underlying edge computing system for CAVs can also be applied to different form factors of autonomous machines, so as the cloud computing systems, and the IVCAD systems.

We believe the development and maturation of edge computing systems, cloud computing systems, and IVCADs are essential to drive the growth of the autonomy economy, or the use of autonomous machines, such as CAVs, delivery robots, industrial robots, drones, home service robots, etc. to provide goods and services. We firmly believe that the autonomy economy will succeed the digital economy, or the use of information technology to create, market, distribute, and consume goods and services, to fuel the world's economic growth in the coming decades.

The following three sections attempt to delve into the practical deployment and development experiences of edge and cloud computing systems for CAVs. We will first introduce various emerging use cases of CAVs in this section. Then, we review the design and implementation of edge computing systems for CAVs in Section 4, including a brief discussion of IVCAD. We examine the challenges and use cases of cloud computing for CAVs in Section 5. All these sections come with specific examples and technical details. It is our hope that they will shed light on advancing systems design for CAVs.

3.2 Emerging Use Cases of CAVs

In addition to full-fledged Level-4 autonomous passenger vehicles, such as the ones developed by Tesla, Waymo, Cruise, and many other autonomous driving companies, there are many emerging use cases for CAVs. In this section, we review a few interesting use cases of CAVs. Note that although these CAVs serve different purposes, the design of the underlying edge computing systems as well as the cloud computing systems supporting these CAVs are similar. We will review the edge computing and cloud computing systems design in later sections.

3.2.1 Autonomous Vehicles Lite

Autonomous micromobility services, or autonomous vehicles lite, are a way to provide autonomous transportation for trips less than 5 miles (Liu and Gaudiot, 2020). In this subsection, we review the design and implementation of autonomous vehicles lite.

Many young urbanites don't want to own a car, and unlike earlier generations, they don't have to rely on mass transit. Instead they treat mobility as a service. When they need to travel significant distances, say, more than 5 miles, they use their phones to summon an Uber (or a car from a similar ride-sharing company). If they have less than a mile or so to go, they either walk or use various "micromobility" services, such as the increasingly ubiquitous Lime and Bird scooters or, in some cities, bike sharing (Liu and Gaudiot, 2020). The problem is that today's mobility-as-a-service ecosystem often doesn't do a good job covering intermediate distances, say a few miles. Hiring an Uber or Lyft for such short trips proves awkwardly expensive, and riding a scooter or bike more than a mile or so can be taxing to many people. So getting yourself to a destination that is from 1 to 5 miles away can be a challenge. Yet such trips account for about half of the total passenger miles traveled.

Many of these intermediate-distance trips take place in environments with limited traffic, such as university campuses and industrial parks, where it is now both economically reasonable and technologically possible to deploy small, low-speed CAVs powered by electricity.

Because these diminutive autonomous vehicles never exceed 20 miles per hour and don't mix with high-speed traffic, autonomous vehicles lite don't engender the same kind of safety concerns that arise with autonomous cars that travel on regular roads and highways. Given how expensive autonomous cars still are in the quantities that they are currently being produced—an experimental model can cost you in the neighborhood of US \$800,000—you might not think it possible to sell a CAV of any kind for much less. Our experience over the past few years shows that, in fact, it is possible today to produce a self-driving passenger vehicle much more economically.

Let's start by explaining why CAVs are normally so expensive. In a nutshell, it's because the sensors and computers they carry are very pricey. The suite of sensors required for autonomous driving normally includes a high-end satellite-navigation receiver, LiDAR (light detection and ranging), one or more video cameras, radar, and sonar. The vehicle also requires at least one very powerful computer. The satellite-navigation receivers used in this context aren't the same as the one found in your phone. The kind built into CAVs have what is called

real-time kinematic capabilities for high-precision position fixes—down to 10 centimeters. These devices cost well over \$10,000. Even so, such satellite-navigation receivers can't be entirely relied on to tell the vehicle where it is. The fixes it gets could be off in situations where the satellite signals bounce off of nearby buildings, introducing noise and delays.

In any case, satellite navigation requires an unobstructed view of the sky. In closed environments, such as tunnels, that just doesn't work. Fortunately, CAVs have other ways to figure out where they are, in particular they can use LiDAR. LiDAR determines distances to things by bouncing a laser beam off them and measuring the time it takes for the light to reflect back. A typical LiDAR unit for CAVs covers a range of 150 meters and samples more than 1 million spatial points per second. Such LiDAR scans can be used to identify different shapes in the local environment. The vehicle's computer then compares the observed shapes with the shapes recorded in a high-definition digital map of the area, allowing it to track the exact position of the vehicle at all times. LiDAR can also be used to identify and avoid transient obstacles, such as pedestrians and other cars.

LiDAR is wonderful technology, but it suffers from two problems. First, these units are extremely expensive: A high-end LiDAR for autonomous driving can easily cost more than \$80,000. Also, LiDAR, being an optical device, can fail to provide reasonable measurements in bad weather, such as heavy rain or fog. The same is true for the cameras found on these vehicles, which are mostly used to recognize and track different objects, such as the boundaries of driving lanes, traffic lights, and pedestrians. Usually, multiple cameras are mounted around the vehicle. These cameras typically run at 60 frames per second, and the multiple cameras used can generate over 1 gigabyte of raw data each second. Processing this vast amount of information, of course, creates very large computational demands on the vehicle's computer. On the plus side, cameras aren't very expensive.

The radar and sonar systems found in CAVs are used for obstacle avoidance. The data generated by radar and sonar shows the distance from the nearest object in the vehicle's path. A major advantage of radar and sonar is that they work in all weather conditions. Sonar usually covers a range of 0 to 10 meters whereas radar covers a range of

3 to 150 meters. Like cameras, these sensors are relatively inexpensive, costing less than \$1,000.

The many measurements such sensors supply are fed into the vehicle's computers, which have to integrate all this information to produce an understanding the environment. Artificial neural networks and deep learning, an approach that has seen rapid growth in recent years, play a large role here. With those techniques, the computer can keep track of other vehicles moving nearby, as well as of pedestrians crossing the road, ensuring the autonomous vehicle doesn't collide with anything or anyone.

Of course, the computers that direct CAVs have to do a lot more than just avoid hitting something. They have to make a vast number of decisions about where to steer and how fast to go. For that, the vehicle's computers generate predictions about the upcoming movement of nearby vehicles before deciding on an action plan based on those predictions and on where the occupant needs to go. Lastly, an autonomous vehicle needs a good map. Traditional digital maps are usually generated from satellite imagery and have meter-level accuracy. Although that's more than sufficient for human drivers, CAVs demand higher accuracy for lane-level information. Therefore, special high-definition maps are needed.

Just as with traditional digital maps, these HD maps contain many layers of information. The bottom layer is a map with grid cells that are about 5 centimeters by 5 centimeters square; it's generated from raw LiDAR data collected using special cars. This grid basically records elevation and reflection information of the objects in the environment. On top of that basal grid, there are several layers of additional information. For instance, lane information is added to the grid map to allow CAVs to determine whether they are in the correct lane. On top of the lane information, traffic sign labels are added to notify the CAVs of the local speed limit, whether they are approaching traffic lights, and so forth. This helps in cases where cameras on the vehicle fail to be able to read the signs. Traditional digital maps are updated every six to twelve months. To make sure the maps that CAVs use contain up-to-date information, the refresh cycle for their HD maps should be no longer than one week. As a result, generating and maintaining HD maps can cost millions of dollars per year for a mid-size city.

Given that the sensors and computers alone can easily cost more than \$100,000, it's not hard to understand why CAVs are so expensive, at least today. Sure, the price will come down as the total number manufactured increases. But it's still unclear how the costs of creating and maintaining HD maps will be passed along. In any case, it will take time yet for better technology to address all the obvious safety concerns that come with autonomous driving on normal roads and highways.

As shown in Figure 3.1, we have been trying to address these challenges by focusing on small, slow-speed vehicles that operate in limited areas and don't have to mix with high-speed traffic—university campuses and industrial parks, for example. The main tactic we've used to reduce costs is to do away with LiDAR entirely and instead use more affordable sensors: cameras, inertial measurement units, satellite positioning receivers, wheel encoders, radars, and sonars. The data that each of these sensors provides can then be combined through a process called sensor fusion. Not only do these sensors each have their own drawbacks and advantages but they also tend to complement one another so that when one fails or otherwise malfunctions, others can immediately take over to ensure that the system remains reliable. With this sensor-fusion approach, sensor costs could drop eventually to something like \$2,000.

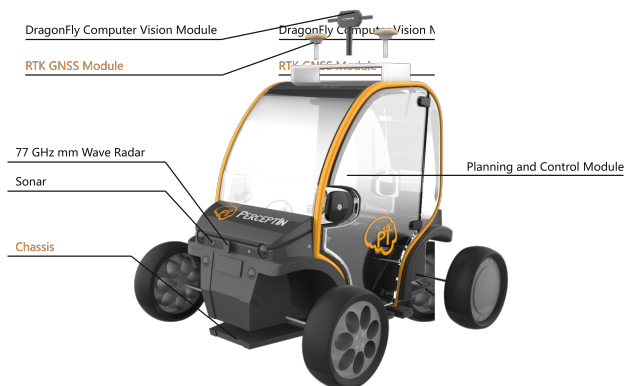


Figure 3.1: Autonomous Vehicles Lite

Our vehicles use satellite positioning to provide an initial localization. While not as accurate as the systems found on highway-capable

autonomous cars, the satellite navigation receivers used still provide sub-meter accuracy. Using a combination of camera images and data from inertial measurement units (in a technique called visual inertial odometry), the vehicle's computer further improves the accuracy, providing position fixes down to the decimeter level.

For imaging, we have integrated four cameras into one hardware module. One pair of cameras faces the front of the vehicle, and another pair of cameras faces the rear. Each pair provide binocular vision, allowing it to capture the kind of spatial information that LiDAR would normally provide. What's more, the four cameras together can capture a 360-degree view of the environment, with enough overlapping spatial regions between frames to ensure that visual odometry works in any direction.

Even if visual odometry were to fail and satellite-positioning signals were to drop out, all would not be lost. The vehicle could still work out position updates using rotary encoders attached to its wheels—a process sailors used to refer to as dead reckoning. Data from all these sensors is combined to give the vehicle an overall understanding its environment. Based on this understanding, the vehicle's computer can make the decisions it needs to make to ensure a smooth and safe trip.

The vehicle also has an anti-collision system that operates independently of its main computer, providing a last line of defense. This uses a combination of millimeter-wave radars and sonars to sense objects within 5 meters, in which case the vehicle is immediately brought to a stop to avoid colliding with anything. Relying on less expensive sensors is just one strategy that PerceptIn has pursued to reduce costs. Another has been to push computing to the sensors so as to reduce the demands on the vehicle's main computer

Our camera module, for example, can generate 400 megabytes of image information per second. If all this data were transferred to the main computer for processing, that computer would have to be extremely complex, which would have significant consequences in terms of reliability, power, and cost. We instead have each sensor module perform as much computing as possible. This reduces the burden on the main computer and a simplifies its design.

More specifically, a GPU is embedded into the camera module to extract features from the raw images. Then, only the extracted features are sent to the main computer, reducing the data transfer rate a thousand-fold. Applying the same design principles to the satellite positioning receiver and the radar system reduces the cost of main computer to less than \$2,000.

Another area for limiting costs involves the creation and maintenance of the HD maps. Rather than using vehicles outfitted with LiDAR units to provide data for such maps, Our approach is to enhance existing digital maps with visual information to achieve decimeter-level accuracy. The resultant high-precision visual maps, like the LiDAR-based HD maps they replace, consist of multiple layers. The bottom layer can be any of the existing digital maps, such as those from the Open Street Map project. This bottom layer has a resolution of about 1 meter. The second layer records the visual features of the road surfaces to improve mapping resolution to the decimeter level. The third layer, also saved at decimeter-resolution, records the visual features reflecting other parts of the environment—signs, buildings, trees, fences, light poles, and the like. The fourth layer is the semantic layer, which contains lane markings, traffic sign labels, and so forth.

As a result of these optimizations, we managed to reduce the manufacturing cost of each autonomous vehicle lite down to under \$10,000, allowing large scale commercial deployment. For more detailed information of building autonomous vehicles lite, please refer to Liu (2020b).

3.2.2 Autonomous Last-Mile Delivery Vehicles

Logistics services are part of the daily lives of many Asian countries. People place orders online and expect their parcels sitting at their doorsteps when they get back from work. Hence, autonomous logistics services empowered by autonomous driving technologies have enormous commercial potentials. In this subsection, we review the design and implementation of autonomous last-mile delivery vehicles (Figure 3.2) (Li *et al.*, 2020).

Logistics service contributes to a satisfactory e-commerce experience because consumers commonly expect the shopped goods to be delivered

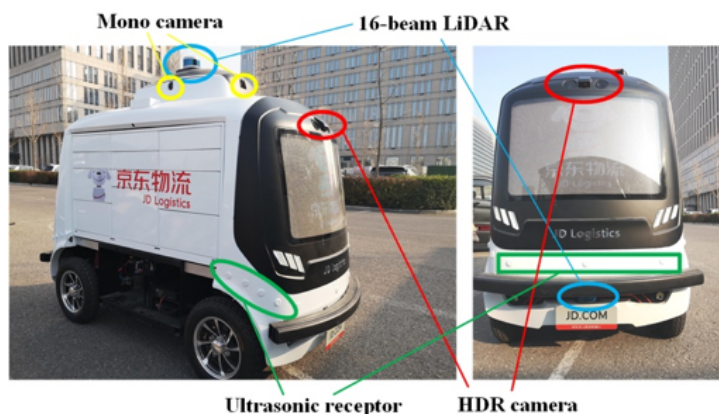


Figure 3.2: Autonomous Last-Mile Delivery Vehicles (Li *et al.*, 2020)

safely, quickly and privately. Here last-mile delivery refers to the delivery of goods ordered online from a local distribution center to the consumers. As the final stage of an e-commerce delivery process, autonomous last-mile delivery is extremely challenging as it needs to handle complex traffic environments.

The main motivations for autonomous last-mile delivery services are the intrinsic disadvantages suffered by traditional last-mile deliveries: first, the growing labor cost has gradually become unbearable for service providers. A contracted delivery clerk with an annual salary of \$17,000 USD can deliver 110 e-commerce parcels per day, which renders that each delivery order costs nearly \$0.5 USD. This cost is expected to continue increasing as the demographic dividend has reached its end.

Secondly, a delivery clerk has to spend time repeatedly on contacting the consumers, waiting for their pick-ups, and traveling on the road, where time is wasted and pre-vents human from other creative work. Fortunately, autonomous driving technologies target this exact problem, the benefits of utilizing an CAV to replace a delivery clerk lie in the following aspects: 1) The delivery is no longer interrupted by weather condition or time. Ideally the unmanned vehicle can respond to the consumers' 24/7 e-commerce orders in a timely manner. Particularly, delivering in the night or during the legal holidays would not have extra costs. 2) Following the first aspect, since the unmanned vehicle

is designated to work 24/7, it has more temporal flexibility than the conventional 8-hour working time to accomplish the delivery tasks. 3) The cost in labor recruitment, training, and management are completely removed. 4) The vehicle system improves the safety and efficiency of both delivery clerks and other people sharing the public transportation system. Therefore, deploying CAVs for the last-mile delivery is a promising solution to over-come the aforementioned disadvantages.

Autonomous driving technologies have been extensively studied in the past few years. However, the traffic conditions in unruly environments are more challenging, primarily because of the large numbers of heterogeneous traffic participants, including pedestrians, bicycles, and automobiles, sharing the roads, and these traffic participants do not necessarily follow traffic rules. These unruly environments are common in various metropolis of Asia, which is distinct from most of the cases in more developed countries, e.g., North America or Europe. Specifically,

- Since the urban population is large in Asia, urban residents typically live in apartments rather than individual houses. This naturally means that the population density is high around apartment complexes, which is quite distinct from the situations in more developed countries, which have most of their population in the suburbs. As a result, when traveling in the unstructured environment around apartment environments, an autonomous vehicle would commonly encounter large numbers of complicated interactive objects such as automatic barrier gates in parking lots, pedestrians, bicycle riders, etc.
- There are typically multiple types of traffic participants including bicycles, electric bicycles, motorcycles on urban roads, etc. Each traffic participant has its own kinematic feature. To make the situation worse, some residents may use their vehicles in unusual, unsafe or even illegal ways.
- Traffic jams often happen in large cities. This is not surprising because motor vehicle ownership keeps growing rapidly.

We target specifically the traffic conditions in unruly environments. The environment is far more complex than those in suburban areas, and

the delivery vehicle should be able to drive on bicycle lanes, university campus, city lanes, residential areas, etc. Existing autonomous driving solutions may not be directly applicable in a chaotic environment. The technologies should be enhanced and adapted to the application scenarios and budget constraints.

Safety should obviously be a primary concern in developing and maintaining CAVs. Compared with CAVs that carry passengers, delivery vehicles have unique safety requirements. They should at the same time obey traffic laws and social conventions to make the traffic on the road proceed in a normal, unblocked, and safe manner. If this is not possible, the delivery CAVs should not cause an impediment or a danger to the other vehicles, especially other vehicles carrying human passengers. Instead, a delivery vehicle might have to sacrifice itself as an alternative choice, which is a special safety requirement for autonomous delivery vehicles.

In a last-mile delivery scheme, a delivery address is assigned to the autonomous vehicle. Thereafter, the current location of the vehicle, as well as the route from the current position to the destination are specified through the navigation service platform. The vehicle then begins to move. During its trip towards the destination, the autonomous vehicle implements localization, local perception, local prediction, local decision, local planning, control, etc., in the algorithmic platform to guarantee that the trip is safe, smooth, efficient and predictable. The hardware platform is a supporting platform for the functioning of the aforementioned elements. The hardware platform consists of the concrete devices, as well as the connections and managements among them. As a general robotic platform, the system is designed for both Ackerman steering and differential steering vehicles. The offline modules, on the other hand, prepare for the aforementioned online modules. For example, the map production platform is responsible for the creation of the High-Definition (HD) map; the simulation platform and visual debugging tools help the developers debug.

Localization is responsible for deriving the current location of the vehicle. In our autonomous driving solution, there are broadly five sources of information which contribute to localization. The first one is the GPS signal, which is used as a startup signal to activate the

request of an HD map for later usage. The remaining four localization sub-modules include localization algorithms based on multi-line LiDAR, cameras, chassis-based odometry, and Inertial Measurement Unit (IMU) Information. In using a multi-line LiDAR algorithm, the generated point clouds are matched to the pre-stored ones in the requested HD map. The matching is implemented via the generalized Iterative Closest Point (ICP) method online, where the LiDAR point clouds are pre-separated into ground and non-ground point clouds. With multi-line LiDAR, camera, and chassis data at hand, the derivative of the data indicates the movement, which is odometry. Through synthesizing the independent sources of odometry information, a reliable odometry result is derived, otherwise known as odometry fusion. The fused result, when synthesized with IMU results, is utilized to calibrate the ICP method in a Kalman filter framework. In turn, the ICP method could be further used to calibrate the fusion result so that they could be both closer to the ground truth.

The perception module is responsible for recognizing and tracking the dynamics of the obstacles in the environment. The perception module plays a key role in making the whole autonomous system capable of running through the complex traffic environments. The hardware setup is a common multi-source solution. Our vehicle is equipped with one 1-beam LiDAR, one 16-beam LiDAR, four mono cameras to detect the surrounding objects. A high-resolution HDR camera is used for traffic light detection. Ultrasonic receptors are used to prevent immediate collisions with other objects or road elements.

The prediction, decision and planning module is critical to enabling our CAVs to safely navigate through complex traffic conditions. Prediction refers to approximating the future trajectories of the tracked moving obstacles. Prediction is done in two cooperative layers. Concretely in the first layer, the behaviors of the well tracked vehicles are predicted with the utilization of the routing information and lane information in the HD map. Herein, we build assumptions on how well the tracked objects will abide by the traffic rules based on the historic road data on this road. The first layer is commonly suitable to handle the regulated vehicles. The second layer targets the prediction of abnormal object behavior, which is achieved through machine learning based methods

and deduction methods. The two layers work together to provide the trajectories of the moving objects of interest in the sighting scale of the vehicle. The predicted trajectory is also attached with a variance to measure the prediction confidence.

Decision and planning play a critical role in directly guiding the local driving maneuvers of the vehicle. In dealing with the complex scenarios in a chaotic environment, the decision module should find a rough but reliable homotopic trajectory, while the planning module should run fast. In the decision module, a sample-and-search based method is adopted. In more details, while searching in a graph consisting of sampled nodes, a dynamic programming algorithm is used, and the cost function considers the uncertainties in the perception and prediction modules. Through this, a robust and safe trajectory decision is made.

In deploying this autonomous vehicle system, a few lessons have been learned, the first being that the algorithms should be explainable, which make their performance easy to estimate, predict, and thus acceptable for other users sharing the road. Having said this, we find that deep-learning based end-to-end solutions are not practical at this stage, but Machine-Learning based methods are extensively used in each sub-module with clearly defined boundaries. Secondly, the routes of last-mile delivery vehicles are mostly fixed, so that we heavily rely on HD maps to record fine details along the route. The third lesson is that, during the real-world on-road tests, the pursuit for higher taken-over mile index is misleading because it may lead the developers to hide the risks or problems rather than to find and conquer them. In our viewpoint, accurately recognizing a risk and then request a manual taken-over is highly meaningful, which is a critical part of the entire safety guarantee system. The fourth lesson is that it makes sense to clearly separate the jobs that are suitable for humans and for an automated machine. After a long time of operations in trial, we have learned that it is feasible to allocate the vehicles to handle the complicated scenarios that repeatedly appear and the human monitors can take over the vehicles when required. In addition, autonomous driving does not mean that human beings become useless, instead they can do innovative work highly related to maintaining an autonomous delivery system with CAVs.

3.2.3 Autonomous Mobile Clinics

Autonomous driving technologies enable many emerging usage scenarios, such as autonomous healthcare delivery, or autonomous mobile clinics. In this subsection, we review the design and implementation of autonomous mobile clinics (Liu *et al.*, 2022b; Liu *et al.*, 2022d; Liu *et al.*, 2022c).

As shown in Figure 3.3, Autonomous Mobile Clinics (AMCs) are CAVs equipped with medical diagnostic equipment, AI doctor software, and telemedicine capability. AMCs have the potential to revolutionize healthcare delivery by bringing healthcare services to the patient by the order of the patient’s fingertips. Specifically, AMCs consist of two parts: the “autonomous” part allows AMCs to autonomously navigate to places where health care services are most needed, thus solving the mobility problem. The “mobile clinic” part allows health care access on-site, thus solving the access problem. In addition, the AI doctor within AMCs can serve unlimited patients, thus potentially eliminating the cost barrier (Yang *et al.*, 2022).

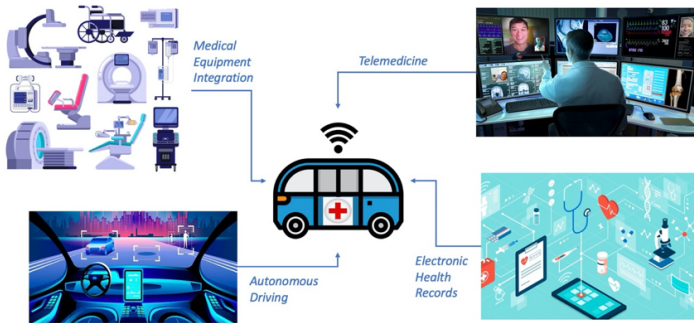


Figure 3.3: Autonomous Mobile Clinics (Liu *et al.*, 2022b)

We first introduce the technologies integrated in each AMC, including telemedicine, medical equipment integration, electronic health records, AI doctors software, and autonomous driving:

- **Telemedicine:** traditional telemedicine⁴ is only a part of the AMC technology stack. There are two categories of data communication for AMCs: Healthcare sensing data involves transmitting raw sensing data collected from medical equipment, whereas video

streams allow interactive communications between patients and remote doctors.

- **Medical Equipment Integration:** a mobile clinic requires the integration of various types of medical equipment to perform point-of-care testing. However, each medical equipment manufacturer enforces its own data standard, preventing the seamless healthcare delivery experience. A standard method of integrating and making sense of the enormous amount of sensing data is required.
- **Electronic Health Records:** due to the availability of on-vehicle point-of-care devices, integrated electronic health records, instead of fragmented health data from each department, can be obtained to enable integrated care through AI doctors or telemedicine. In addition, the valuable diagnostics from remote doctors, along with the integrated electronic health records are essential for training AI doctors.
- **AI Doctors:** today the training of effective AI doctors is often constrained by the availability of integrated health records. AMCs provide not only integrated health records but also diagnostics by remote doctors as training labels to facilitate the training of effective AI doctors. Note that unlike human doctors, AI doctors software can serve unlimited patients, thus effectively eliminating the cost barrier. For instance, many tropical diseases, such as Dengue disease, can already be diagnosed by AI doctors.
- **Autonomous Driving:** in some healthcare delivery situations, such as during outbreaks of infectious diseases, autonomous driving can provide the necessary isolation when handling or transporting patients.

The main technical challenge for AMCs is the integration of autonomous driving technologies and health care technologies into the same computing system. In automotive applications, a domain controller is a computer that controls a set of vehicular functions related to a specific area, or domain. Functional domains requiring domain controllers are typically compute-intensive and connect to a large number

of input/output (I/O) devices. Examples of relevant domains include autonomous driving, smart cockpit, and chassis. For instance, an autonomous driving domain controller receives inputs from sensors around the vehicle, such as radars and cameras, and uses that input to create a model of the surrounding environment. Autonomous driving software in the domain controller then makes planning decisions to generate actions the vehicle should take.

Intelligent vehicles are the best space to deliver healthcare services for three reasons: first, in many countries, people spend a considerable amount of time in cars. The average daily time is about one hour in the U.S. This allows continuous monitoring of one's health conditions. Second, intelligent vehicles are one of the best-equipped environments in our daily life for health monitoring. Third, vehicles are private spaces suitable for healthcare service delivery. Yet, intelligent vehicles have not been utilized for healthcare service scenarios due to the lack of an automotive health domain controller (AHDC) to integrate all necessary technologies to enable a seamless healthcare service experience for the drivers and riders.

Particularly, AHDCs will enable built-in healthcare features by integrating healthcare sensors in the vehicle manufacturing process. AHDCs will support brought-in healthcare features by integrating with healthcare devices that the drivers and passengers carry with them into vehicles, such as smartphones and smart watches. AHDCs will facilitate beamed-in, or cloud-enabled healthcare services through reliable connectivity.

To fulfill the aforementioned requirements, we have defined the architecture of AHDC as illustrated in Figure 3.4. The AHDC consists of an healthcare I/O interface, along with a data specification, to integrate both the built-in and the brought-in healthcare sensors. With the influx of heterogeneous healthcare sensing data, a health AI engine performs automatic abnormality detection, prognosis, and sensor data analysis. The health AI engine is key to enable future in-vehicle AI doctors to improve healthcare efficiency. Then a storage engine along with encryption capability is required to store the private and sensitive healthcare data, both raw sensing data and AI-prognosis data. The storage engine is to guarantee the patient owns his healthcare data in

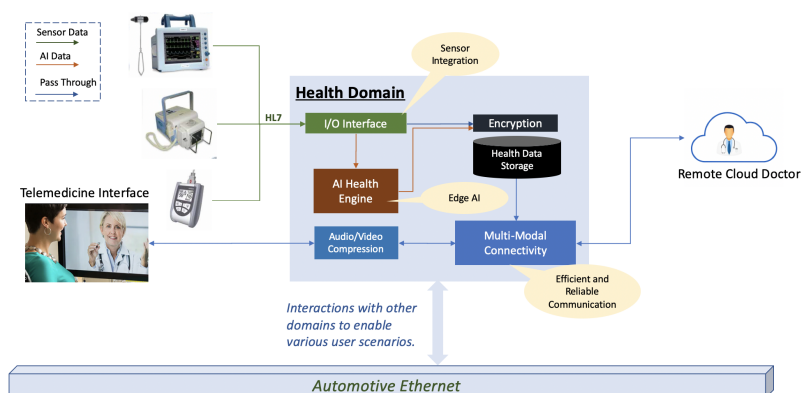


Figure 3.4: Architecture of Automotive Health Domain Controller (Liu *et al.*, 2022b)

case he chooses not to upload the data to the cloud. Next, a connectivity engine is required to enable beamed-in or cloud-based healthcare services, such as telemedicine and the transmission of health sensing data.

The key challenge for AHDC is to integrate all these technical pillars into a domain controller with limited compute power, energy consumption, cost, but with the strictest reliability requirements. In the rest of the tutorial, we will delve into each of these technical areas.

One challenge is to integrate various healthcare sensors into intelligent vehicles to deliver a seamless healthcare delivery experience, especially that the sensor and equipment deployment needs to be highly customized to the specific healthcare service scenarios. Taking Acute Myocardial Infarction and Stroke as an example, various sensors are required for vital sign monitoring, facial expression tracking, as well as infarction warning. This deployment involves built-in sensors, including millimeter wave radars, IR cameras, thermal cameras, photoelectric sensors, conductive textiles, as well as brought-in sensors, such as mobile phones and wearable devices. In the scenario of Acute Myocardial Infarction, vital sign monitoring tracks a patient's day-to-day vital information and generates basic health report and identifies any abnormality. Facial expression tracking specifically targets symptoms such as facial palsy, a sign of stroke. When a sign of facial palsy is detected, the

vehicle will issue a warning and may connect to a remote doctor for symptom confirmation. Infarction warning is one step further to confirm the patient's medical situation, when a patient gets to the infarction warning stage, he or she may already be unconscious, the intelligent vehicle with autonomous driving function may directly send the patient to a nearby emergency department while sending the healthcare data along the way, or the intelligent vehicle summons an ambulance to save the patient's life as soon as possible.

The second challenge is to deliver edge AI capabilities. There are multiple benefits of performing edge AI for healthcare services in vehicle. First, many users do not want to have their medical data uploaded to the cloud due to privacy concerns. By integrating edge AI in intelligent vehicles, the raw medical data of the patient can be consumed *in situ*, thus effectively addressing the privacy concerns. Second, edge AI is connectivity-prone. In many locations with limited network connectivity, edge AI is still able to provide healthcare services, thus guaranteeing quality of service under different situations. Third, and the most important reason, edge AI is able to greatly improve healthcare service efficiency, as edge AI can perform prognosis for the doctors, and thus significantly increase doctors' service bandwidth.

The third challenge is connectivity, as connectivity is key to guarantee the Quality-of-Service (QoS) of in-vehicle applications, especially for healthcare services. There are two categories of data communication for in-vehicle healthcare services, healthcare sensing data and telemedicine data stream. Healthcare sensing data involves the transmission of raw sensing data collected from healthcare sensors, such as imaging devices. This type of communication is bandwidth constrained but less latency sensitive, it demands a communication bandwidth of 15 MB/s or more and a latency of 1 s or less. Telemedicine data stream is mostly High-Definition video conferencing workloads to allow interactive communication between the patient and the remote doctor. This type of communication is highly latency sensitive, it demands a communication bandwidth of 3 MB/s or more and a latency of 100 ms or less.

Solving all these technical challenges for in-vehicle healthcare service delivery, and integrating these functionalities along with autonomous driving capabilities into one central edge computing system presents unique challenges to the design of edge computing systems for CAVs.