Vertically Distributed Computing Technology

Kazuya Matsuo, Masaru Takagi, Ryota Nakada, and Koya Mori

Abstract

We developed a technology that quickly shares information found by a connected vehicle with other vehicles. For example, when a connected vehicle finds an obstacle on the road, the technology can transmit information about it quickly to other vehicles. This quick notification is achieved by offloading part of the processing of the collected obstacle information to network-edge nodes and transmitting interim results to vehicles near the obstacle. However, in urban areas, the delay in notification may become large because a large number of vehicles connect to a small number of network-edge nodes; thus, the volume of data received from these vehicles may overwhelm the computing resources of these nodes. To solve this problem, we also developed a technology that dynamically selects which computer will execute any particular notification processing on the basis of the states of the connected vehicles. Using this technology, obstacle information can be transmitted quickly even when a large number of vehicles are connected to a small number of network-edge nodes.

Keywords: distributed processing, edge computing, load balancing

1. Roles and challenges with an ICT platform for connected vehicles

The role of an information and communication technology (ICT) platform for connected vehicles is to collect videos captured with onboard cameras and CAN (controller area network) data from connected vehicles, analyze them, and send the analysis results to the other vehicles to provide these vehicles with more information than they can obtain from their sensors alone. This enables drivers to obtain information about their blind spots, preventing traffic accidents from occurring.

When connected vehicles are widely used, a large amount of data will be collected from a large number of such vehicles and may overwhelm the ICT platform. Even in such situations, the ICT platform must be able to quickly notify vehicles of urgent information, such as that relating to obstacles.

In the collaboration between the NTT Group and Toyota Motor Corporation, we conducted field trials for several use cases in which the ICT platform notifies relevant vehicles about the processing results for the data it has collected.

The vertically distributed computing introduced in this article consists of the *vertically distributed application architecture* and *technology for dynamically selecting processing nodes* we developed. The following sections describe the challenges each of these technologies addresses, solutions each provides, and verification results of the field trials.

2. Vertically distributed computing

2.1 Vertically distributed application architecture

(1) Obstacle detection and notification in the previous ICT platform

The previous ICT platform used in our collaboration with Toyota was built on the lambda architecture [1, 2], in which stream (real-time) data processing and batch data processing are executed in parallel, allowing both quick results from stream processing and detailed results from batch processing to be made available (**Fig. 1**). To be able to execute obstacle

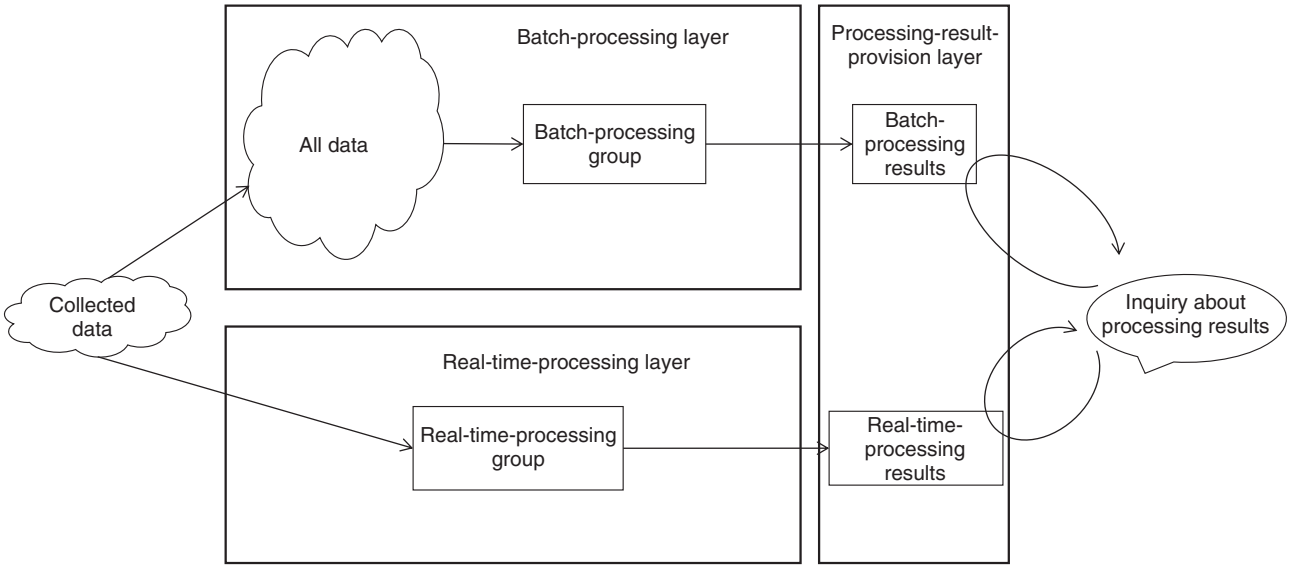


Fig. 1. Lambda architecture.

detection and notification within 7 s, the platform estimated the type and location of any obstacle from each frame (image) by stream processing. However, the lambda architecture does not take into account the requirements for processing time. Therefore, a system built on this architecture does not necessarily satisfy the requirement to execute obstacle detection and notification within 7 s. The actual processing time has been about 15 s.

(2) Vertically distributed application architecture

To solve this problem, we proposed an architecture that allows a processing-time threshold to be set in the lambda architecture. This new architecture is called a vertically distributed application architecture (**Fig. 2**). In this architecture, groups of processes are extracted from a series of processes in the order of their processing in such a way that the total processing time does not exceed the threshold. The results of these extracted processes can be obtained by the client. We revised the ICT platform to run on this architecture. In the revised ICT platform, object recognition is extracted from image-data processing, which consists of object recognition and object-location estimation. By doing so, connected vehicles can obtain information about the type of obstacle before the ICT platform completes the estimation of the obstacle location. To further increase the processing speed, the revised ICT platform offloads[*1] object recognition to network-edge nodes[*2], which are located closer to the relevant connected vehicles than the datacenter servers are. These nodes send obstacle information directly to the connected vehicles. In this architecture, obstacle detection and notification are executed in the following two stages (**Fig. 3**):

(i) First notification: Information about the type of obstacle and the rough obstacle location (actually, the location of the connected vehicle that took the image of the obstacle) is sent to connected vehicles near the obstacle.

(ii) Notification of details: The type of obstacle and more precise information about its location, as obtained from the obstacle location estimation, are sent to connected vehicles near the obstacle.

(3) Field trial

In a field trial, the vehicle that took the image of the obstacle and the vehicle that received the obstacle information were the same. We measured the time between when the onboard camera captured the obstacle and when the vehicle received the obstacle information from the ICT platform. The results are shown in **Table 1**. The first notification was sent within 7 s. As mentioned above, the first notification only contained information about the approximate location of the obstacle because the precise location had not yet been calculated at this point. Since this

*1 Offloading: A mechanism by which the processing load of a computer is reduced by passing it to another computer.

*2 Network-edge node: A set of computers that are located at NTT's base stations and central offices.

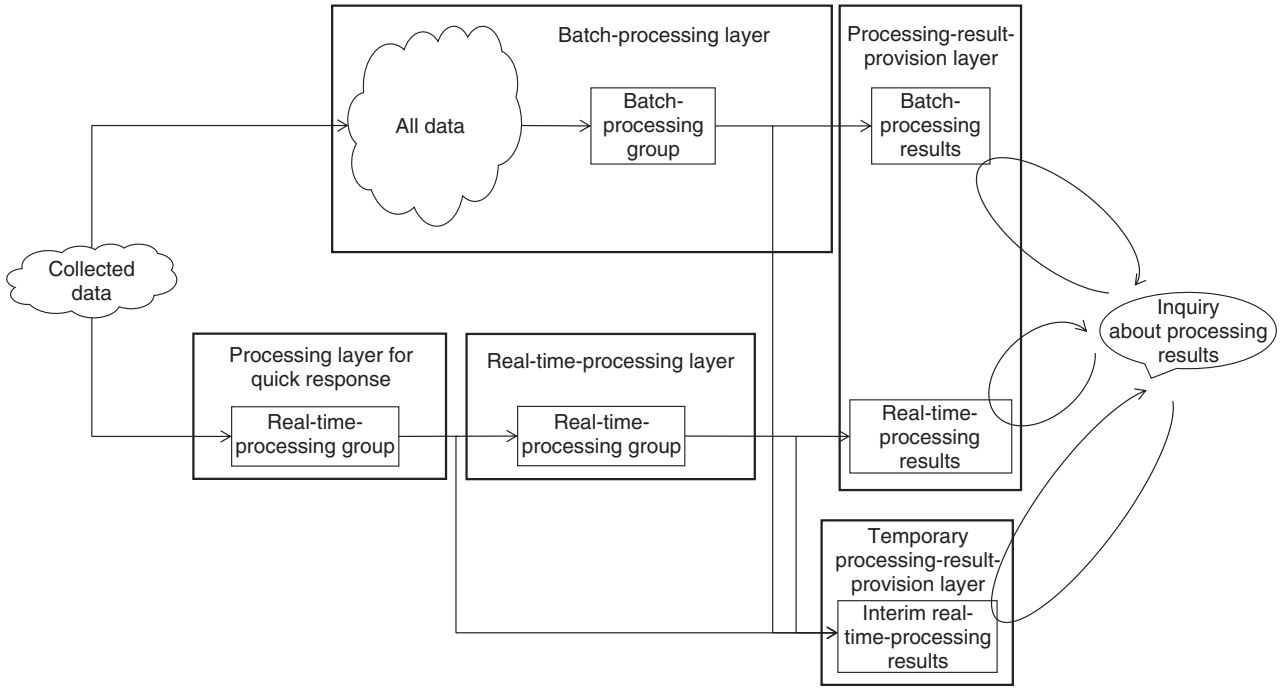


Fig. 2. Vertically distributed application architecture.

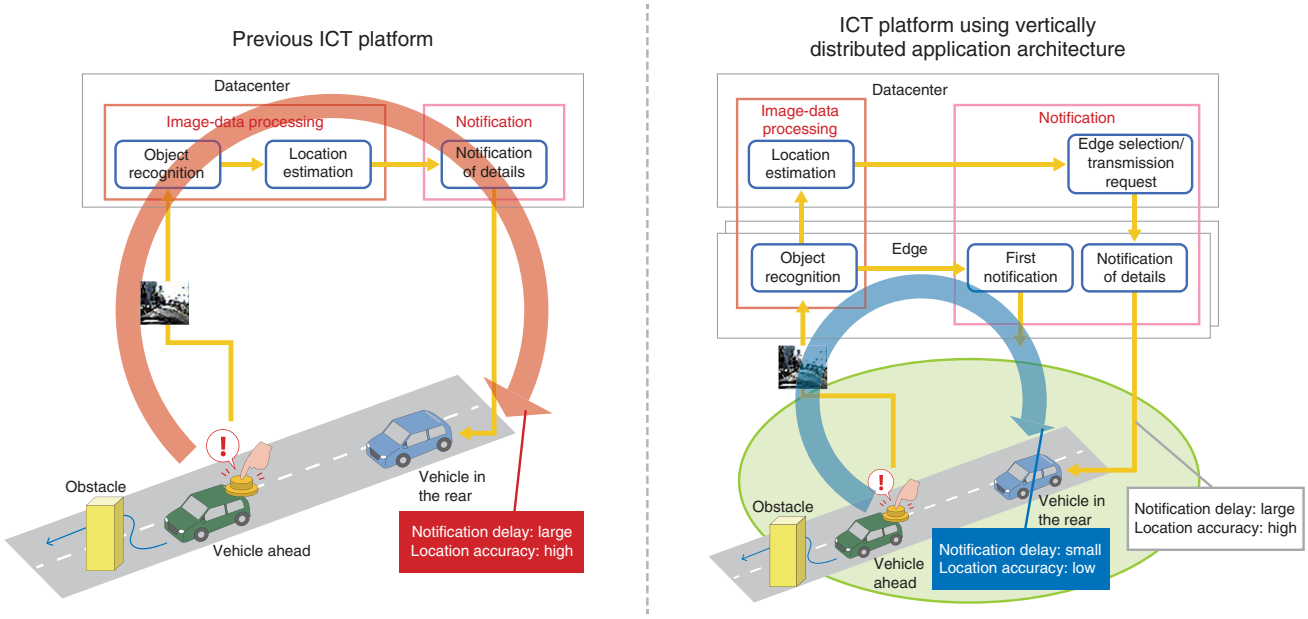


Fig. 3. A before-and-after comparison for the introduction of vertically distributed application architecture.

approximate location was the location of the vehicle that took the image of the obstacle, it was several to ten meters away from the real obstacle location. However, since vehicles can move more than 10 m/s, we believe that providing information about the approximate obstacle location more than 10 s earlier

Table 1. Results of the field trial of vertically distributed application architecture.

Processing time			
From obstacle detection to first notification		From obstacle detection to notification of details	
Average (s)	Fastest (s)	Average (s)	Fastest (s)
4.574	4.092	9.772	9.212

than had been possible, even at the cost of location accuracy, would be extremely valuable for assisting safe driving.

2.2 Dynamic selection of processing nodes

(1) Issues confronting the ICT platform after the adoption of the vertically distributed application architecture

The field trial showed that the vertically distributed application architecture made quick notification possible. However, the trial was conducted under conditions such that the computing resources of the network-edge nodes were not exhausted. When connected vehicles are widespread, a large number of them may be connected to a small number of network-edge nodes. The concentration of the processing load for object recognition on these edge nodes may overwhelm their computing resources, making it impossible to send notifications quickly.

One way to distribute a load that would otherwise concentrate on a small number of computers is to offload the processing load to the most appropriate computers on the basis of the acceptable processing delay and remaining computer resources. Many such technologies have been proposed (e.g., [3]). However, adopting only these conventional technologies cannot solve the above problem because vehicles can move more than 10 m/s, causing the surrounding conditions to change quickly from moment to moment, and the acceptable processing delay varies depending on these surrounding conditions. For example, if an obstacle is located on a blind curve, the risk of an accident is very high, and it is urgent to alert the vehicles near the obstacle. In contrast, if an obstacle is on a road with good visibility, the risk of an accident is low, and it is less urgent to send obstacle notifications to the vehicles near the obstacle. Conventional technologies do not take into account these client situations (i.e., the road situations mentioned above). Therefore, if the ICT platform's load balancing is based on conventional technologies, obstacle notification may not be sent quickly even in situations where quick notification is absolutely needed, or it may be sent quickly even when there is no urgent need for it to be.

(2) Overview of technology for dynamically selecting processing nodes

To address the issue mentioned above, we developed a technology for dynamically selecting processing nodes. It adds to the conventional load balancing technology the ability to take into account the surrounding conditions of the vehicles to be notified. This technology consists of the following three processes:

(i) Determine data-processing priority on the basis of the surrounding conditions of the vehicles to be notified (the part added in the proposed technology)
(ii) Select the computers to which the target processing is offloaded (conventional technology)
(iii) Execute the above offloading (conventional technology)

Various algorithms for Process (i) can be conceived for each use case. In the field trial, we adopted an algorithm designed for the obstacle detection and notification use case and verified its effectiveness.

(3) Application of the technology for dynamically selecting processing nodes to the ICT platform

In the field trial, we adopted an algorithm that takes into account the requirement that connected vehicles moving at high speed without being able to detect the obstacle should be urgently notified. Specifically, the platform searches for connected vehicles in the vicinity of the vehicle that transmitted the relevant onboard camera image. These are the targets for obstacle notification. The platform then determines whether it is urgent to notify these connected vehicles on the basis of the following two conditions (**Fig. 4**):

(i) Whether the average speed of the target connected vehicle exceeds the threshold.
(ii) Whether the obstacle is visible from the driver's seat (to be more precise, whether the driver can see the location of the connected vehicle that sent the relevant obstacle-containing image when the object is first detected, or whether the driver can see the estimated location of the obstacle when the obstacle has already been precisely detected and is being monitored).

To implement Condition (ii), the platform uses an algorithm similar to the selective vehicle-data-collection algorithm to determine whether the view of the obstacle is blocked. If quick notification is required,

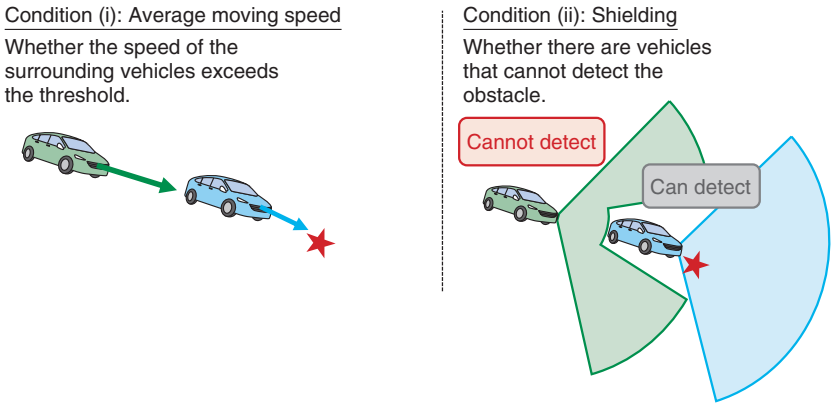


Fig. 4. Determining whether there are connected vehicles requiring quick notification.

Table 2. Setting of processing priority for images from onboard cameras.

(i) Average moving speed	(ii) Shielding	Priority
High	Yes	High
	No	Medium
Low	Yes	Medium
	No	Low

Table 3. Selection of the processing computer based on priority.

Priority	Processing computer
High	Nearby network-edge node with remaining resources
Medium	Nearby computer with remaining resources (network-edge node or datacenter server)
Low	Datacenter server

the processing priority is set to high. If quick notification is not required, the processing priority is set to low. For in-between cases, the processing priority is set to medium. This priority setting is summarized in **Table 2**.

The next step is to determine, based on the priority setting just described, which computer should execute the object recognition. As with the conventional technology, our technology for dynamically selecting processing nodes monitors the remaining resources of each computer in the system and selects the appropriate computer on the basis of the remaining resources of each computer and the processing priority, as shown in **Table 3**. Finally, the platform offloads the object-recognition process to the selected computer and the latter executes the process.

Figure 5 illustrates an example computer configuration in the ICT platform and how the computers to which the target process is offloaded are selected. We assume that the network-edge node receives the onboard camera videos in the sequence of videos 1 to 6 in the figure and that the object-recognition priority for each video is set as shown in the figure. Object recognition 4, which processes video 4, has low priority thus offloaded to a datacenter server. Object recognition 5 has medium priority. It is executed by the network-edge node because the above offloading has freed up its resources. This exhausts the computing resources at the network-edge node. Therefore, object recognition 6, which has medium priority, is offloaded to a datacenter server.

(4) Field trial

In the field trial, we examined how the number of onboard camera videos sent to the network-edge node affects the image-processing time. The results are shown in **Fig. 6**. The horizontal axis shows the number of videos received per second. The number of high-priority videos was varied between 1 and 3, and the number of medium-priority videos was varied between 1 and 9. The vertical axis shows the increase in processing time from a condition under which computing resources are sufficiently available. As can be seen from this figure, when no dynamic selection of processing nodes was used, the processing time at the network-edge node increased when more than three videos were received per second. In contrast, when dynamic selection of processing nodes

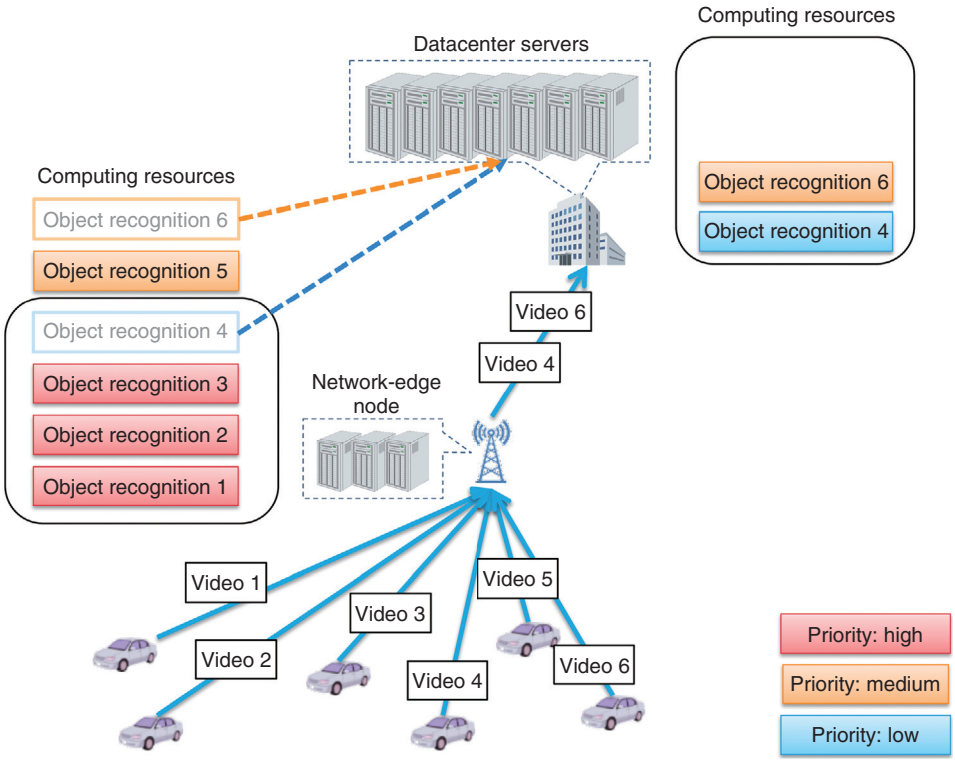


Fig. 5. Example of the dynamic selection of processing nodes.

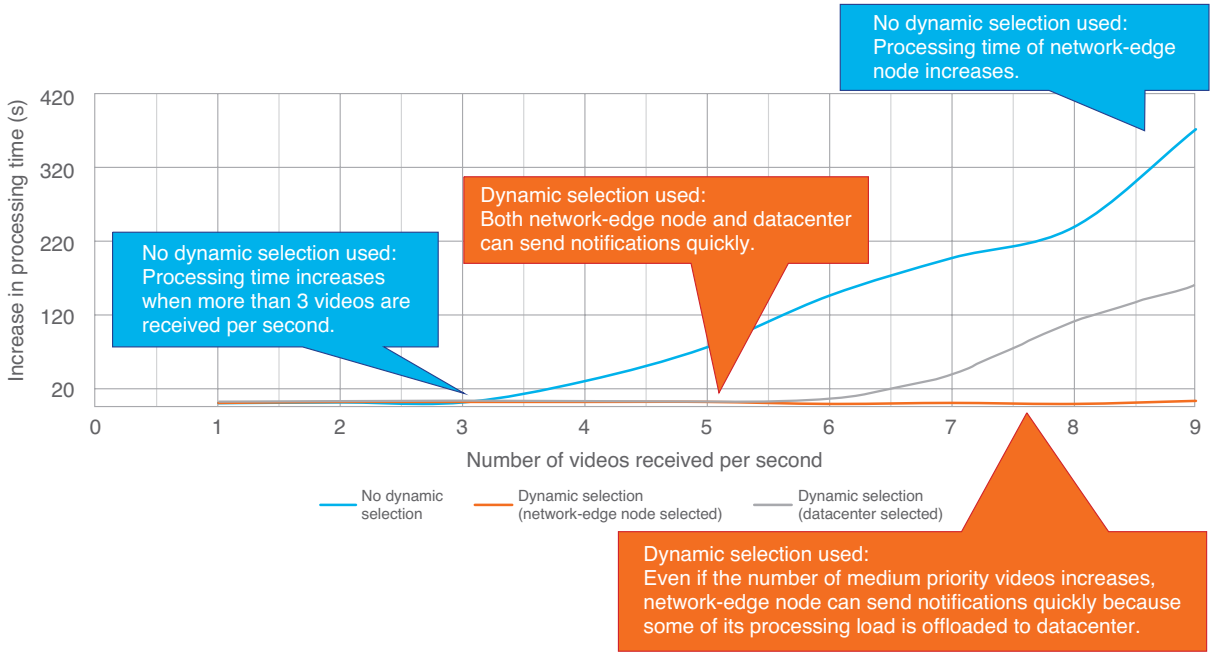


Fig. 6. Results of the field trial of dynamically selecting processing nodes.

was used, the processing time at the network-edge node did not increase until nine[*3] videos were received per second. This was because the network-edge node processed only high-priority processes and other processes were offloaded to the datacenter. Because of this offloading, the processing time at the datacenter began to increase when six videos were received per second.

3. Future outlook

Since computer technology is advancing daily, the computers in vehicles have become powerful enough to process the video from their onboard cameras. However, it is not assumed with the current vertically distributed computing technology that connected vehicles can process onboard camera videos. To make more effective use of the computing resources at both network-edge nodes and datacenters, we will study the extended use of the computing resources in connected vehicles.

In the obstacle detection and notification use case, an additional type of processing is required: when the platform detects an obstacle in the image from a connected vehicle, it needs to determine whether the obstacle is being identified for the first time or has already been found in the image from another vehicle. This is because, if the obstacle has already been found, the connected vehicles near the obstacle will have already been notified of the obstacle, and so there is less urgency. We are also studying technology for determining whether the objects detected by multiple sensors are identical [4].

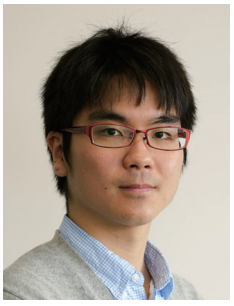
References

[1] N. Marz, "Big Data Lambda Architecture," 2012. http://www.databasetube.com/database/big-data-lambda-architecture/
[2] K. Dutta and M. Jayapal, "Big Data Analytics for Real Time Systems," Big Data Analytics Seminar, pp. 1–13, 2015.
[3] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks," IEEE Internet of Things Journal, Vol. 6, No. 3, pp. 4377–4387, June 2019.
[4] K. Matsuo, M. Takaki, R. Nakata, and K. Mori, "A Study on a Dynamic Object Detection and Identification System That Uses Point Cloud Alignment for Creating a High-precision Dynamic Map," SIG Technical Reports on Human-Computer Interaction, Vol. 2021, No. 6, pp. 1–8, 2021 (in Japanese).

*3 This load is very small for a world in which connected vehicles are widely used. This small value was derived because the field trials were intended to confirm the effectiveness of each technology studied in this collaboration, thus were conducted using the minimum computer configuration. When the ICT platform is built with a large-scale computer configuration to confirm practical feasibility in the future, a larger load will be applied when verifying the effectiveness of each technology.

Kazuya Matsuo
Researcher, NTT Digital Twin Computing Research Center.

He received a Ph.D. in information science from Osaka University in 2017 and joined NTT the same year. While at university (2012–2017), he studied the collection of data in mobile sensor networks. He studied modeling of vertically distributed applications in NTT until 2019. Since 2018, he has been involved in collaborative work between Toyota and NTT. Since 2020, he has also been working on a Digital Twin Computing project. He received a Young Researcher's Award from the Institute of Electronics, Information and Communication Engineers (IEICE) in 2018. He is a member of the Information Processing Society of Japan (IPSJ).

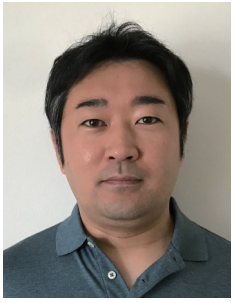


Masaru Takagi
Researcher, NTT Digital Twin Computing Research Center.

He received a B.E. in engineering and M.E. and Ph.D. in information science and technology from the University of Tokyo in 2013, 2015, and 2018. He joined NTT in 2018 and was engaged in a collaboration project with Toyota Motor Corporation. He is currently studying multi-agent mobility simulation as part of Digital Twin Computing. He is a member of IEICE and IPSJ.

Ryota Nakada
Senior Research Engineer, NTT Digital Twin Computing Research Center.

He received a B.E. and M.E. in mathematics from Tokyo University of Science in 2008 and 2010. He joined NTT Network Innovation Laboratories in 2010 and researched a high-resolution image-transport system. He moved on to NTT WEST in 2013, where he studied an indoor-location information system. He rejoined NTT Network Innovation Laboratories in 2016 and has been researching edge computing. Since 2020, he has been in charge of research for Digital Twin Computing at NTT Digital Twin Computing Research Center.



Koya Mori
Senior Research Engineer, NTT Digital Twin Computing Research Center.

Since joining NTT in 2004, he has been involved in the research and development of an Internet-of-Things application based on software technologies, such as OSGi, OpenStack, and Edge Computing. Since 2020, he has been in charge of research for Digital Twin Computing at NTT Digital Twin Computing Research Center.