TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
1–19
© National Academy of Sciences: Transportation Research Board 2022
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/03611981221091762
journals.sagepub.com/home/trr
SAGE

Real-Time Distributed Cooperative Adaptive Cruise Control Model Considering Time Delays and Actuator Lag

Yingtong Tan[1] and Kuilin Zhang[1]

Abstract
Real-time control of a fleet of Connected and Automated Vehicles (CAV) for Cooperative Adaptive Cruise Control (CACC) is a challenging problem concerning time delays (from sensing, communication, and computation) and actuator lag. This paper proposes a real-time predictive distributed CACC control framework that addresses time delays and actuator lag issues in the real-time networked control systems. We first formulate a Kalman Filter-based real-time current driving state prediction model to provide more accurate initial conditions for the distributed CACC controller by compensating time delays using sensing data from multi-rate onboard sensors (e.g., Radar, GPS, wheel speed, and accelerometer), and status-sharing and intent-sharing data in BSM via V2V communication. We solve the prediction model using a sequential Kalman Filter update process for multi-rate sensing data to improve computational efficiency. We propose a real-time distributed MPC-based CACC controller with actuator lag and intent-sharing information for each CAV with the delay-compensated predicted current driving states as initial conditions. We implement the real-time predictive distributed CACC control algorithms and conduct numerical analyses to demonstrate the benefits of intent-sharing-based distributed computing, delay compensation, and actuator lag consideration on string stability under various traffic dynamics.

Keywords
operations, traffic flow

Real-time control of a fleet of Connected and Automated Vehicles (CAV) for Cooperative Adaptive Cruise Control (CACC) is a challenging problem concerning time delays (from sensing, communication, and computation) and actuator lag. These time delays and actuator lag deteriorate the control performance, especially for the time-critical control problem in the real-time networked control systems (NCS). Delayed sensing data and control commands may produce unstable traffic to incur more traffic oscillations that lead to collisions. This paper aims to model and compensate for the time delays and actuator lag in a real-time predictive distributed CACC controlling framework for Cooperative Driving Automation (CDA) of a fleet of CAVs.

In the real-time control system of CACC, the sensing delay is caused by the time needed for sensor measuring and processing. Ignoring the sensing delay results in an inaccurate estimation of the initial condition for the controller (*1*), which leads to a model mismatch issue that can degrade the control performance to unstable traffic (*2*). Wang compensates for the sensing delay by predicting the state evolution in a previous time instant (*3*). Xu et al. use an augmented model to compensate for the sensing delay (*2*).

Communication delay in Vehicle-to-Vehicle (V2V) communication, which is the network delay in the real-time NCS of CACC, is also a problem that affects the control performance. In the real-time networked CACC controller, where the control relies on sharing information among vehicles frequently, communication delay

[1]Department of Civil, Environmental, and Geospatial Engineering, Michigan Technological University, Houghton, MI

Corresponding Author:
Kuilin Zhang, klzhang@mtu.edu

significantly compromises the string stability. This issue commonly exists in NCS. Pin and Parisini present a predictive NCS scheme to compensate for the communication delay (*4*). However, this scheme cannot compensate for the delays smaller than a control time interval, so it is not suitable for the real-time networked CACC, where vehicles must respond to the frequently changing traffic condition. Other methods, including Long Short-Term Memory (LSTM) neural network (*5*) and the Smith predictor (*6*), are also used to deal with communication delay.

Computation delay, also known as control delay in a real-time control system, is caused by the time needed to solve the control problem. Most existing studies assume there is no computation delay, that is, the computation can finish instantaneously. However, the computation time for non-closed form controllers such as Model Predictive Control (MPC) can be significant. If the computation delay is ignored, the actual execution time instant of the control command is not consistent with the designed execution time instant. This may cause significant performance deterioration. To address this problem, Wang et al. propose to reserve sufficient computation time for each control interval (*7*). Zavala and Biegler reduce computation time using nonlinear programming sensitivity (*8*).

Actuator lag is caused by the limitation of the vehicle response to a control command. This is a first-order lag for vehicles to track the desired acceleration (*9*). Ignoring it leads to a mismatch between the controller and the actual vehicle states and over-optimistic evaluations of the controller performance (*3*). Existing studies formulate the actuator lag explicitly in the vehicle dynamics model (*3*, *10*, *11*) to address the lag issue in the real-time control systems.

Most existing MPC-based CACC controllers are centralized, requiring two-way communication between the central computer and all other CAVs in the real-time NCS (*12–14*). The computation time of a centralized MPC-based CACC controller will increase monotonically as the platoon size becomes larger. Distributed MPC (DMPC), which is not limited by the platoon size, has attracted increasing research interest (*15–17*). Zhou et al. propose a serial DMPC where the controller of the ego vehicle is time-dependent to the solution of the preceding vehicle for the same control interval (*15*). However, this serial scheme is not practical in real-time control because the communication delays accumulate along the string. Dunbar and Caveney design a DMPC scheme where every vehicle in the platoon shares its optimal predicted state trajectory to its follower, which is intent-sharing (*18*). It does not have the communication delay propagation issue but requires a precise state prediction model. This paper also adopts the distributed MPC-based CACC using intent-sharing data, which is reserved as an optional field in the Basic Safety Message (BSM) data for the intent-sharing level in CDA.

This paper proposes a real-time predictive distributed CACC control framework that addresses time delays (resulting from sensing, communication, and computation) and actuator lag issues in the real-time NCS. We first formulate a Kalman Filter-based real-time current driving state prediction model to provide more accurate initial conditions for the distributed CACC controller by compensating time delays using sensing data from multi-rate onboard sensors (e.g., Radar, GPS, wheel speed, and accelerometer), and status-sharing and intent-sharing data in BSM via V2V communication. We solve the prediction model using a sequential Kalman Filter update process for multi-rate sensing data to improve computational efficiency. We propose a real-time distributed MPC-based CACC controller with actuator lag and intent-sharing information for each CAV with the delay-compensated predicted current driving states as initial conditions. We implement the real-time predictive distributed CACC control algorithms and conduct numerical analyses to demonstrate the benefits of intent-sharing-based distributed computing, delay compensation, and actuator lag consideration on string stability under various traffic dynamics.

This paper makes two contributions to the literature. First, we formulate a real-time predictive distributed CACC model that explicitly considers time delay (resulting from sensing, communication, and computation) and actuator lag. To the best of our knowledge, this is the first model to address all the four critical issues in one model. Second, we integrate the intent-sharing data in the distributed MPC-based CACC model for better control performance. Experimental results show that intent-sharing can improve string stability under challenging traffic conditions. String stability is observed under traffic that is slow but moving smoothly (i.e., without shocks). The maximum tolerable delays and actuator lag are also provided under traffic oscillations (i.e., traffic shocks).

This paper is structured as follows. First, it describes the real-time CACC problem. Assumptions and notations are defined. Second, it introduces the real-time CACC control framework. Third, it presents a real-time solution algorithm for the proposed framework. Next, it evaluates the proposed real-time distributed MPC-based CACC control framework using real-world vehicle trajectories. Stability analysis is conducted. Finally, the conclusion of this paper is given.

Problem Statement and Assumptions

The Real-Time Cooperative Adaptive Cruise Control Problem

The problem of interest addressed in this paper is to propose a real-time distributed CACC controller with time delays and actuator lag. We use an intent-sharing-based communication topology for the real-time distributed CACC model to automatically control a platoon of CAVs (*19*). This distributed CACC communication topology in Figure 1 shows that each CAV only uses BSM with status-sharing (i.e., its observed current driving states) and intent-sharing information (i.e., its predicted driving states) via V2V communication from its immediately preceding vehicles. Each CAV i is equipped with multi-rate onboard sensors, including a Radar (e.g., 20–50 Hz), a GPS (e.g., 10–20 Hz), a wheel-speed sensor (e.g., 100–1k Hz), and an accelerometer (e.g., 100–1k Hz), and an Onboard Unit (OBU) for V2V communication to receive BSM messages (e.g., 10 Hz) from its immediately preceding vehicle $i-1$. The vehicle systems for the leading CAV (i.e., vehicle 1) and each following CAV (i.e., vehicle $i, \forall i>1$) are shown in Figures 2 and 3. Based on the onboard sensing data and BSM messages from its immediately preceding vehicle, each CAV solves a local distributed predictive control model with a constant time headway CACC policy using its onboard sensing data and BSM messages (with both status-sharing and intent-sharing). Each vehicle system includes a real-time current driving state prediction model to compensate for time delays from sensing, communication, and computation, a real-time MPC-based control model to address the actuator lag, and an actuator to execute the optimal control command at the current time. For the leading CAV, there is no intent-sharing data from its immediately preceding connected vehicle (IPCV), that is the vehicle 0. Thus, the distributed CACC control framework of the leading CAV is different from that of the following CAVs. Figures 4 and 5 illustrate the real-time predictive control framework for the leading and following CAVs, respectively.

Assumptions and Notation

We make the following assumptions for the real-time distributed CACC controller with time delays and actuator lag:

Assumption 1: The IPCV of the CACC platoon is assumed to be a Connected Vehicle (CV). The IPCV can broadcast its observed current driving states in its BSM data via V2V communication, which is the status-sharing level in a CDA system.

Assumption 2: Every vehicle in the platoon is assumed to be a CAV. It can broadcast its observed current driving states and predicted driving states during a prediction horizon in its BSM data via V2V communication, which is the intent-sharing level in a CDA system.

Assumption 3: Every onboard sensing data in each CAV carries a timestamp representing its measuring time.

Assumption 4: Onboard sensors have zero-mean Gaussian noises, and the noises are independent of each other.

We list the notation as follows:

Indices:

i, vehicle index, $i = 0, 1, 2, \ldots$ where $i = 0$ represents the IPCV, $i = 1$ stands for the leading CAV, and $i = 2, \ldots$ stand for following CAVs.

j, state update time step index, $j = 1, 2, \ldots$

k, control time interval index, $k = 1, 2, \ldots$

h, current control time interval index, $h = 1, 2, \ldots$

s, sensor index, $s = 1, 2, \ldots$

Parameters:

Δt, state update time interval, which is determined by the highest frequency sensor

Δt_c, control time interval

$N_c = \frac{\Delta t_c}{\Delta t}$, number of state update time intervals in a control time interval

τ_{l_i}, actuator lag of vehicle i

τ_c^{max}, maximum computation time delay (number of state update time intervals Δt)

L_i, the length of vehicle i

N_s, the total number of sensors

T_p, MPC prediction horizon (number of control time intervals Δt_c)

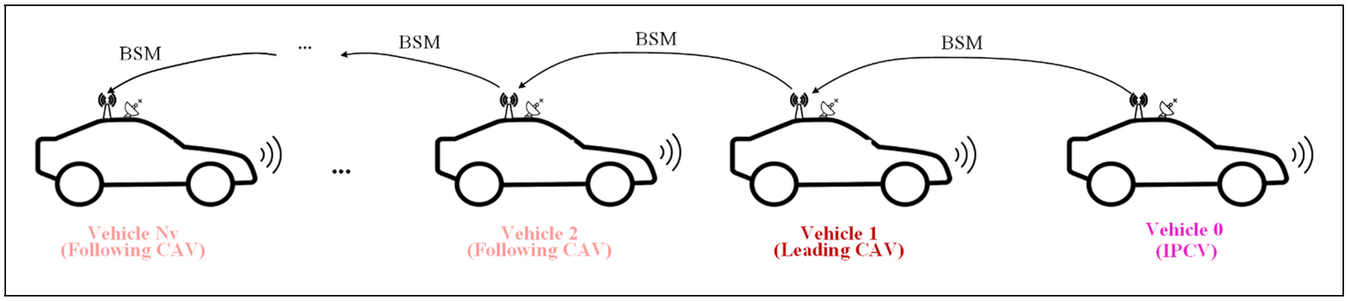

Figure 1. The communication topology for the distributed cooperative adaptive cruise control model.
Note: BSM = basic safety message; CAV = connected and automated vehicle; IPCV = immediately preceding connected vehicle.

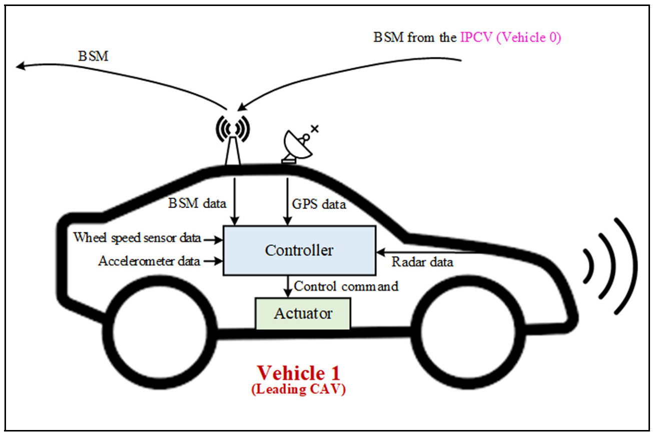

Figure 2. The leading CAV system—Vehicle 1.
Note: BSM = basic safety message; CAV = connected and automated vehicle; IPCV = immediately preceding connected vehicle; GPS = global positioning system.

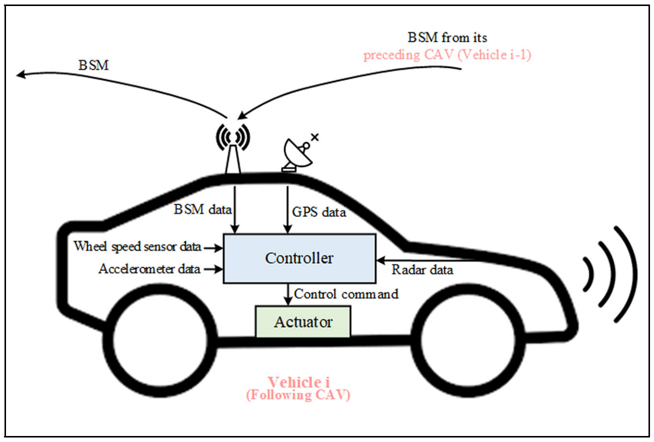

Figure 3. The following CAV system—Vehicle ***i*** (***i***>1).
Note: BSM = basic safety message; CAV = connected and automated vehicle; GPS = global positioning system.

t_{hw}, constant time headway
t_{hw}^{min}, minimum safe time headway
s_i^{min}, minimum safe spacing for stopped vehicles considering vehicle length for vehicle i
v^{limit}, speed limit
u_i^{min}, minimum acceleration for vehicle i
u_i^{max}, maximum acceleration for vehicle i
A_i, state transition matrix for vehicle i
B_i, control input matrix for vehicle i
C_i^s, observation matrix of sensor s for vehicle i
D_i^s, the constant vector in the observation equation of sensor s for vehicle i
R_i^s, observation noise covariance matrix of sensor s for vehicle i
Q_i, system noise covariance matrix for vehicle i
T_s, the time window size of sensing data collecting (number of state update time intervals Δt)

Variables for real-time current driving state prediction model:
$\hat{l}_i(j)$, the predicted location of vehicle i at state update time step j
$\hat{v}_i(j)$, the predicted velocity of vehicle i at state update time step j
$\hat{a}_i(j)$, the predicted acceleration of vehicle i at state update time step j
$\hat{l}_{i-1}(j)$, the predicted location of the immediately preceding vehicle of vehicle i at state update time step j
$\hat{v}_{i-1}(j)$, the predicted velocity of the immediately preceding vehicle of vehicle i at state update time step j
$\hat{a}_{i-1}(j)$, the predicted acceleration of the immediately preceding vehicle of vehicle i at state update time step j
$X_i(j)$, state vector of vehicle i at state update time step j
$U_i(j)$, control input vector of vehicle i at state update time step j
$Y_i^s(j)$, output vector of sensor s for vehicle i at state update time step j
$V_i^s(j)$, observation noise vector of sensor s for vehicle i at state update time step j
$W_i(j)$, system noise vector for vehicle i at state update time step j
$\hat{X}_i(j|j)$, the posteriori estimated state vector of vehicle i at state update time step j
$\hat{X}_i(j|j-1)$, the priori estimated state vector of vehicle i at state update time step j
$K_i(j)$, Kalman gain of vehicle i for state update time step j
$P_i(j|j)$, the posteriori estimate covariance matrix of vehicle i for state update time step j
$P_i(j|j-1)$, the priori estimate covariance matrix of vehicle i for state update time step j
$S_i(g,j)$, the set of sensing data measured at state update time step j that are collected before current control interval h, $g = N_c * h$
$H_i(g,j)$, the set of sensor indexes corresponding to $S_i(g,j)$
$\hat{X}_i^m(j|j)$, the posteriori state vector of vehicle i using the m^{th} piece of sensing data measured at state update time step j, $\forall m \in [1, |S_i(g,j)|]$
$P_i^m(j|j)$, the posteriori estimate covariance matrix of vehicle i using the m^{th} piece of sensing data measured at state update time step j, $\forall m \in [1, |S_i(g,j)|]$
$K_i^m(j)$, the Kalman gain matrix of vehicle i using the m^{th} piece of sensing data measured at state update time step j, $\forall m \in [1, |S_i(g,j)|]$

State variables for the real-time MPC-based control model:
$l_i(k)$, location of vehicle i at control time interval k, $\forall k \in [h,\ h + T_p - 1]$
$v_i(k)$, velocity of vehicle i at control time interval k, $\forall k \in [h,\ h + T_p - 1]$
$a_i(k)$, acceleration of vehicle i at control time interval k, $\forall k \in [h,\ h + T_p - 1]$

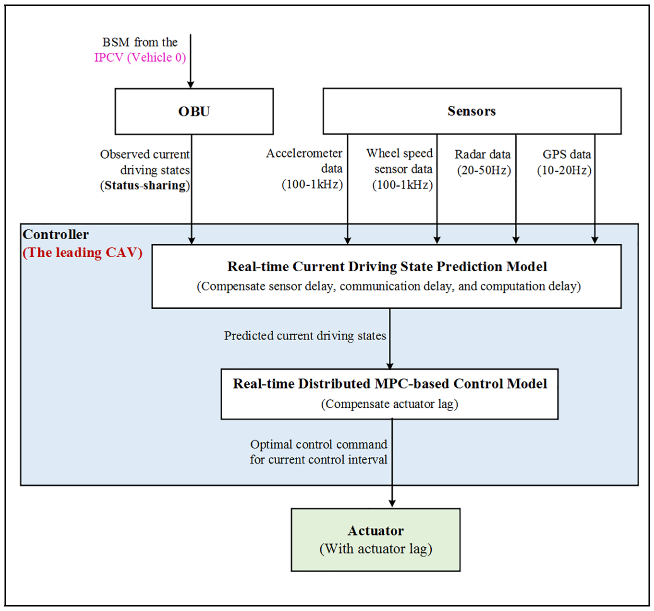
BSM from the IPCV (Vehicle 0)
OBU
Sensors
Observed current driving states (Status-sharing)
Accelerometer data (100-1kHz)
Wheel speed sensor data (100-1kHz)
Radar data (20-50Hz)
GPS data (10-20Hz)
Controller (The leading CAV)
Real-time Current Driving State Prediction Model (Compensate sensor delay, communication delay, and computation delay)
Predicted current driving states
Real-time Distributed MPC-based Control Model (Compensate actuator lag)
Optimal control command for current control interval
Actuator (With actuator lag)

Figure 4. The real-time distributed predictive control framework for the leading CAV.

Note: BSM = basic safety message; CAV = connected and automated vehicle; IPCV = immediately preceding connected vehicle; OBU = onboard unit; MPC = model predictive control; GPS = global positioning system.

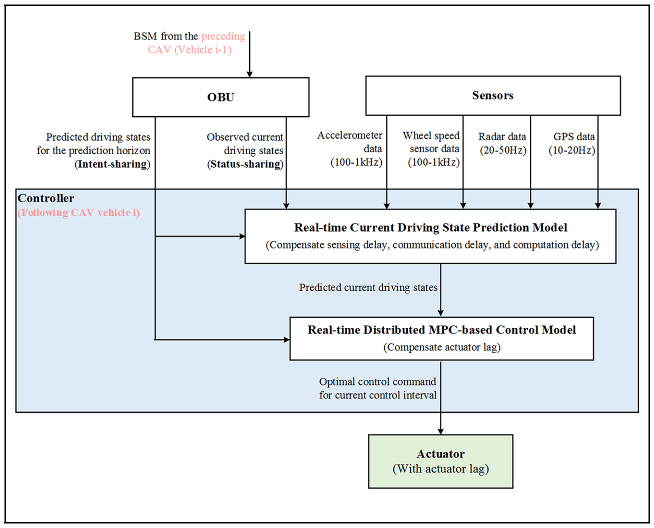
BSM from the preceding CAV (Vehicle i-1)
OBU
Sensors
Predicted driving states for the prediction horizon (Intent-sharing)
Observed current driving states (Status-sharing)
Accelerometer data (100-1kHz)
Wheel speed sensor data (100-1kHz)
Radar data (20-50Hz)
GPS data (10-20Hz)
Controller (Following CAV vehicle i)
Real-time Current Driving State Prediction Model (Compensate sensing delay, communication delay, and computation delay)
Predicted current driving states
Real-time Distributed MPC-based Control Model (Compensate actuator lag)
Optimal control command for current control interval
Actuator (With actuator lag)

Figure 5. The real-time distributed predictive control framework for each following CAV.

Note: BSM = basic safety message; CAV = connected and automated vehicle; GPS = global positioning system; MPC = model predictive control.

Control variable for the real-time MPC-based control model:

$u_i(k)$, control command of vehicle i for control time interval k, $\forall k \in [h,\ h + T_p - 1]$

Other variables for the real-time MPC-based control model:

$\hat{l}_{i-1}(k)$, location of the preceding vehicle of vehicle i at control time interval k, $\forall k \in [h,\ h + T_p - 1]$

$\hat{v}_{i-1}(k)$, velocity of the preceding vehicle of vehicle i at control time interval k, $\forall k \in [h,\ h + T_p - 1]$

Intent-sharing variables:

$u_i^*(k)$, predicted control command of vehicle i at control interval k, $\forall k \in [h,\ h + T_p - 1]$

$l_i^*(k)$, predicted location of vehicle i at control interval $\forall k \in [h,\ h + T_p - 1]$

$v_i^*(k)$, predicted velocity of vehicle i at control interval $\forall k \in [h,\ h + T_p - 1]$

Model Formulations

This section introduces the real-time distributed CAV control framework. We first present a Kalman Filter-based real-time current driving state prediction model to compensate time delays. Then, we formulate a real-time MPC-based control model with actuator lag for each CAV.

The Real-Time Current Driving State Prediction Model

The real-time current driving state prediction aims to provide more accurate current driving states at the initial conditions in the real-time MPC-based controller to compensate for time delays from sensing, communication, and computation. The prediction period covers the time delays for multi-rate sensing and communication to the start of computing, as well as the time delay from the start of computing to the start of actuating. This delay compensation predicts the initial conditions at the time to start actuating the control command, which avoids using delayed initial conditions in the MPC controller.

Figure 6 shows the time delays compensated in the state prediction model. The state prediction model uses Δt (i.e., state update interval) as the time interval, whereas the MPC model uses Δt_c (i.e., control interval) as the time interval. We set Δt to be the sensing period of the highest frequency sensors—the wheel-speed sensor and the accelerometer (e.g., 100–1k Hz). We do not consider delays smaller than Δt, as the wheel-speed sensor and the accelerometer generally have high sensing rates and very small sensing delay. Instead of using only the latest sensing data, this model would synthesize all multi-rate sensing data from all onboard sensors. This model collects streaming sensing data and BSM data τ_c^{max} steps before each control interval. τ_c^{max} is the maximum computation time reserved for all computations needed by the state prediction model and the MPC model. We use the timestamp of the sensor data and the BSM data to identify the actual measuring time of the data. After that, a Kalman Filter-based prediction model is applied to predict the current vehicle driving states. In Xu et al. an

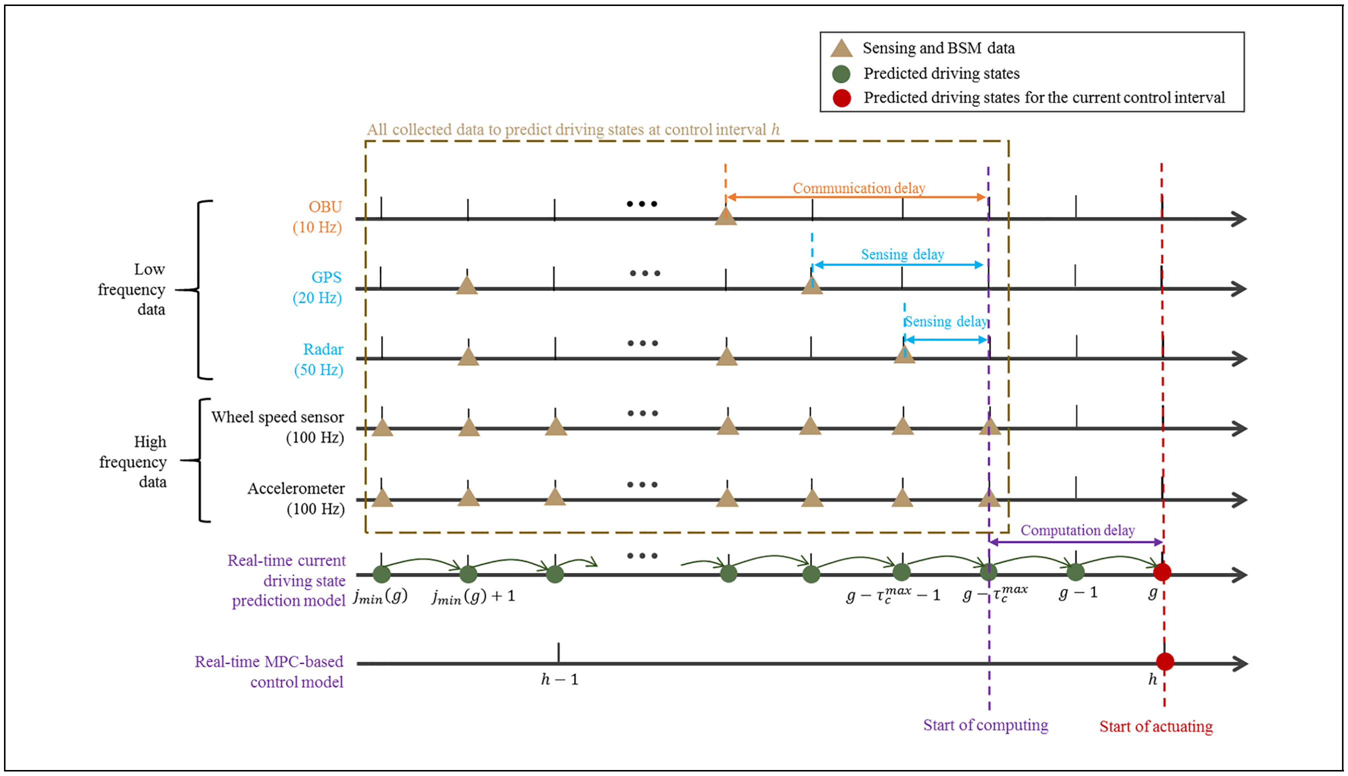

Figure 6. The real-time current driving state prediction to compensate for time delays.
Note: BSM = basic safety message; OBU = onboard unit; MPC = model predictive control; GPS = global positioning system.

augmented strategy is used, which is computationally costly in calculating the inverse of the augmented matrix with large dimensions (*2*). Unlike that, we sequentially use the data in a timely order based on the timestamps to update the estimation of the vehicle states based on the idea of Sequential Measurement Fusion to reduce the computation cost (*13*).

This paper uses multi-rate sensing data from Radar, GPS, wheel-speed sensor, accelerometer, and measurements in the BSM data. For each vehicle, six states, including the location, velocity, and acceleration of the ego vehicle, and the location, velocity, and acceleration of its immediately preceding vehicle, are predicted in this model. The Kalman Filter-based real-time current driving state prediction problem for each vehicle is formulated in **Problem 1**.

Problem 1: The Kalman Filter-based real-time current driving state prediction model for vehicle i

The system update equation of vehicle i is

$$\hat{X}_i(j) = A_i\hat{X}_i(j-1) + B_iU_i(j-1) + W_i(j) \tag{1}$$

The measurement update equation for sensor $s \in [1, N_s]$ of vehicle i is

$$Y_i^s(j) = C_i^sX_i(j) + D_i^s + V_i^s(j) \tag{2}$$

System Dynamics of the Prediction Model. In the system update equation in Equation 1, because every vehicle needs to estimate the state of itself and its immediately preceding vehicle, $\hat{X}_i(j)$ includes the state of itself and its immediately preceding vehicle. The first element $u_i(j)$ of $U_i(j)$ comes from the controller of vehicle i. The second element $u_{i-1}^*(j)$ of $U_i(j)$ comes from the intent-sharing data. The IPCV, denoted as vehicle 0, is not controlled by the MPC framework. We do not assume the IPCV performs intent-sharing, so the value of the A_i and B_i for the leading vehicle is different from those for the following vehicles.

$$\hat{X}_i(j) = \begin{bmatrix} \hat{l}_i(j) & \hat{v}_i(j) & \hat{a}_i(j) & \hat{l}_{i-1}(j) & \hat{v}_{i-1}(j) & \hat{a}_{i-1}(j) \end{bmatrix}^T \tag{3}$$

$$U_i(j-1) = \begin{bmatrix} u_i(j-1) & u_{i-1}^*(j-1) \end{bmatrix}^T \tag{4}$$

$$A_i = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 1-\frac{\Delta t}{\tau_{l_i}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \forall i = 1 \tag{5}$$

$$A_i = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 1 - \frac{\Delta t}{\tau_{l_i}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 - \frac{\Delta t}{\tau_{l_{i-1}}/N_c} \end{bmatrix}, \forall i \in [2, N_v] \tag{6}$$

$$B_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{\Delta t}{\tau_{l_i}/N_c} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \forall i = 1 \tag{7}$$

$$B_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{\Delta t}{\tau_{l_i}/N_c} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{\Delta t}{\tau_{l_{i-1}}/N_c} \end{bmatrix}, \forall i \in [2, N_v] \tag{8}$$

Measurement Updates Using Multiple Sensors. This model uses multiple onboard sensors, including Radar, GPS, wheel-speed sensor, and accelerometer, as well as BSM status-sharing and intent-sharing data via V2V communication, in the sensor fusion process.

For Radar, observations of the distance gap and relative velocity measured at step j can be obtained by Equations 9 and 10.

$$Y_i^1(j) = C_i^1 X_i(j) + D_i^1 + V_i^1(j) \tag{9}$$

$$C_i^1 = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \end{bmatrix},$$

$$D_i^1 = \begin{bmatrix} -L_{i-1} \\ 0 \end{bmatrix}, \ V_i^1(j) \sim (0, R_i^1) \tag{10}$$

For GPS, observations of location, velocity, and acceleration measured at step j can be obtained by Equations 11 and 12.

$$Y_i^2(j) = C_i^2 X_i(j) + V_i^2(j) \tag{11}$$

$$C_i^2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \ V_i^2(j) \sim \left(0, R_i^2\right) \tag{12}$$

For wheel-speed sensor, observations of velocity measured at step j can be obtained by Equations 13 and 14.

$$Y_i^3(j) = C_i^3 X_i(j) + V_i^3(j) \tag{13}$$

$$C_i^3 = [\, 0 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \,], V_i^3(j) \sim (0, \ R_i^3) \tag{14}$$

For the accelerometer, observations of acceleration measured at step j can be obtained by Equations 15 and 16.

$$Y_i^4(j) = C_i^4 X_i(j) + V_i^4(j) \tag{15}$$

$$C_i^4 = [\, 0 \ \ 0 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \,], \ V_i^4(j) \sim (0, R_i^4) \tag{16}$$

The observations of location, velocity, and acceleration of the immediately preceding vehicle measured at step j in BSM status-sharing and intent-sharing data can be obtained by Equations 17 and 18.

$$Y_i^5(j) = C_i^5 X_i(j) + V_i^5(j) \tag{17}$$

$$C_i^5 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, V_i^5(j) \sim \left(0, R_i^5\right) \tag{18}$$

A Solution Procedure for the Real-Time Current Driving State Prediction Model. To predict the current driving state for control interval h of vehicle $i = 1, 2, \ldots N_v$, the prediction model first collects all sensing data received from state update step $g - \tau_c^{max} - T_s + 1$ to $g - \tau_c^{max}$. T_s is the size of the time window for data collecting. $g = h \times N_c$ is the state update time step corresponding to the current control time interval h. Among the collected data, the oldest piece of sensor data has the largest delay, and we denote its corresponding measuring time step as $j_{min}(g)$. Then sensing data measured at time step j are organized into sets $S_i(g,j), \forall j \in [j_{min}(g), g]$. By the time this procedure starts, no measurement for $j \in [g - \tau_c^{max} + 1, g]$ is collected, and thus $S_i(g,j)$ and $H_i(g,j)$ are empty for $j \in [g - \tau_c^{max} + 1, g]$.

After organizing $S_i(g,j)$ and $H_i(g,j)$ for control interval h, this model restores its states to the historical states of step $j_{min}(g)$, and then conducts sequential Kalman Filter-based prediction for each state update step. Figure 7 shows the procedure of the state update. For each $j \in [j_{min}(g), g]$, the model applies Equations 19 and 20 to make a priori state estimation based on the system dynamic. If $S_i(g,j)$ is not empty, then the model applies Equations 21 to 23 to make a posteriori state estimation based on the sensing data. The index m in variables $K_i^m(j)$, $\hat{X}_i^m(j|j)$, and $P_i^m(j|j)$ means they are correspondent to the m^{th} piece of sensing data measured at j. This is the idea of sequential processing, which is more computationally efficient than simultaneous processing (*20*).

$$\hat{X}_i(j|j-1) = A_i \hat{X}_i(j-1|j-1) + B_i U_i(j-1) \tag{19}$$

$$P_i(j|j-1) = A_i P_i(j-1|j-1) A_i^T + Q_i \tag{20}$$

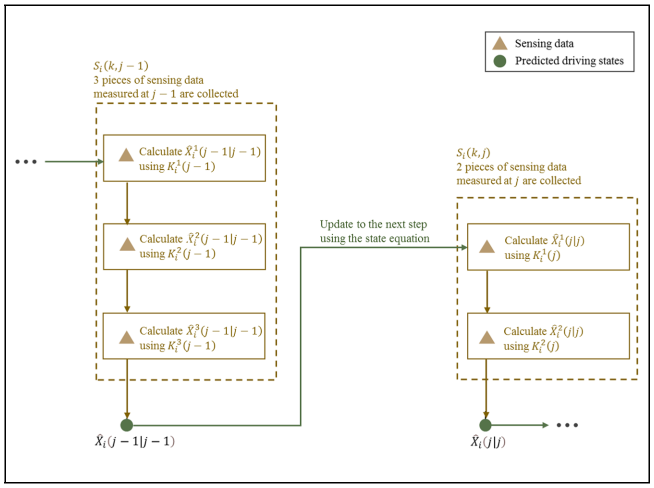
▲ Sensing data
● Predicted driving states
$S_i(k, j-1)$ 3 pieces of sensing data measured at $j-1$ are collected
Calculate $\hat{X}_i^1(j-1|j-1)$ using $K_i^1(j-1)$
Calculate $\hat{X}_i^2(j-1|j-1)$ using $K_i^2(j-1)$
Calculate $\hat{X}_i^3(j-1|j-1)$ using $K_i^3(j-1)$
$\hat{X}_i(j-1|j-1)$
Update to the next step using the state equation
$S_i(k, j)$ 2 pieces of sensing data measured at j are collected
Calculate $\hat{X}_i^1(j|j)$ using $K_i^1(j)$
Calculate $\hat{X}_i^2(j|j)$ using $K_i^2(j)$
$\hat{X}_i(j|j)$

Figure 7. The sequential Kalman Filter-based state prediction process with multi-sensor multi-rate sensing data.

$$K_i^m(j) = P_i^{m-1}(j|j)\left(C_i^s\right)^T\left[C_i^s P_i^{m-1}(j|j)\left(C_i^s\right)^T + R_i^s\right]^{-1}, \quad \forall m \in [1, |S_i(g,j)|] \tag{21}$$

$$\hat{X}_i^m(j|j) = \hat{X}_i^{m-1}(j|j) + K_i^m(j)[Y_i^s(j) - C_i^s\hat{X}_i^{m-1}(j|j) - D_i^s], \quad \forall m \in [1, |S_i(g,j)|] \tag{22}$$

$$P_i^m(j|j) = \left[I - K_i^m(j)C_i^s\right]P_i^{m-1}(j|j), \forall m \in [1, |S_i(g,j)|] \tag{23}$$

The Real-Time Distributed MPC-Based Control Model

This section presents a real-time distributed MPC-based control model for CACC. The real-time controller explicitly considers the actuator lag in its constraints, predicted current driving states at its initial conditions, and predicted driving states of its immediately preceding vehicle via intent-sharing in its safety constraints and objective function. The real-time distributed MPC-based CACC controller problem for vehicle $i = 1, 2, \ldots N_v$ at control interval h is formulated in **Problem 2**.

Problem 2: The Real-time Distributed MPC Model for vehicle i

$$\min_{u_i(k)} \sum_{k=h}^{h+T_p-1} \left(\hat{l}_{i-1}(k) - l_i(k) - v_i(k) \times t_{hw} - s_i^{min}\right)^2 + \left(v_i(k) - \hat{v}_{i-1}(k)\right)^2 + \left(u_i(k)\right)^2 \tag{24}$$

s.t.

$$l_i(k) = \hat{l}_i(g), \forall k = h,\ g = h \times N_c \tag{25}$$

$$v_i(k) = \hat{v}_i(g), \forall k = h,\ g = h \times N_c \tag{26}$$

$$a_i(k) = \hat{a}_i(g), \forall k = h,\ g = h \times N_c \tag{27}$$

$$\hat{l}_{i-1}(k) = \hat{l}_{i-1}(g), \forall\ k = h,\ g = h \times N_c \tag{28}$$

$$\hat{v}_{i-1}(k) = \hat{v}_{i-1}(g), \forall k = h,\ g = h \times N_c \tag{29}$$

$$\hat{l}_{i-1}(k) = \begin{cases} \hat{l}_{i-1}(k-1) + \hat{v}_{i-1}(k-1) \times \Delta t_c, \text{ when } i = 1 \\ \gamma_i(k|h) \times l_{i-1} \times\ (k) + (1 - \gamma_i(k|h)) \times \left(\hat{l}_{i-1}(k-1) + \hat{v}_{i-1}(k-1) \times \Delta t_c\right), \\ \text{when } i > 1, \end{cases}$$

$$\forall\ k \in [h+1, h+T_p-1] \tag{30}$$

$$\hat{v}_{i-1}(k) = \begin{cases} \hat{v}_{i-1}(k-1), \text{ when } i = 1 \\ \gamma_i(k|h) \times v_{i-1} \times (k) + (1 - \gamma_i(k|h)) \times \hat{v}_{i-1}(k-1), \quad \text{when } i > 1, \end{cases}$$

$$\forall\ k \in [h+1, h+T_p-1] \tag{31}$$

$$l_i(k) = l_i(k-1) + v_i(k-1) \times \Delta t_c + \frac{1}{2} \times a_i(k-1) \times \Delta t_c^2, \forall k \in [h+1,\ h+T_p-1] \tag{32}$$

$$v_i(k) = v_i(k-1) + a_i(k-1) \times \Delta t_c, \forall k \in [h+1,\ h+T_p-1] \tag{33}$$

$$a_i(k) = a_i(k-1) + \frac{u_i(k-1) - a_i(k-1)}{\tau_{l_i}} \times \Delta t_c, \quad \forall k \in [h+1,\ h+T_p-1] \tag{34}$$

$$\hat{l}_{i-1}(k) - l_i(k) \geqslant v_i(k) \times t_{hw}^{min} + s_i^{min},\ \forall k \in [h,\ h+T_p-1] \tag{35}$$

$$0 \leqslant v_i(k) \leqslant v^{limit},\ \forall k \in [h,\ h+T_p-1] \tag{36}$$

$$u_i^{min} \leqslant u_i(k) \leqslant u_i^{max}, \forall k \in [h,\ h+T_p-1]. \tag{37}$$

$u_i(k)$ is the control variable. $l_i(k)$, $v_i(k)$, and $a_i(k)$ the state variables. Equations 25 to 29 represent the initial conditions, where $\hat{l}_i(g)$, $\hat{v}_i(g)$, $\hat{a}_i(g)$, $\hat{l}_{i-1}(g)$, and $\hat{v}_{i-1}(g)$ are the output of the Kalman Filter-based real-time current driving state prediction model compensating for sensing, communication, and computation delays for control interval h. $g = h \times N_c$ represents the state update time step corresponding to the current control time interval h. In Equations 30 and 31, $\hat{l}_{i-1}(k)$, and $\hat{v}_{i-1}(k)$ are the driving states of the immediately preceding vehicle, which are constrained by the intent-sharing data $l_{i-1} \times\ (k)$ and $v_{i-1} \times (k)$ from the BSM data. $\gamma_i(k|h)$ is an indicator of the availability of the intent-sharing data. If the intent-sharing data are not available, $\gamma_i(k|h)$ is 0, in which case, $\hat{l}_{i-1}(k)$ and $\hat{v}_{i-1}(k)$ are determined by assuming the immediately preceding vehicle to drive at the same velocity as its previous time step. Equations 32 to 34 are the vehicle dynamic constraints. Equation 35 is the safety constraint. Equation 36 is the speed limit constraint. Equation 37 reflects the vehicle acceleration and deceleration capability.

Solution Algorithm 1: Solving the real-time predictive distributed CACC model (for time *h*)

```
Step 1: Start computing
1:  Set state update step $j = g - \tau_c^{max} = N_c * h - \tau_c^{max}$.
2:  Trigger the computing at time step $j = g - \tau_c^{max}$.

Step 2: Compute the Kalman Filter-based real-time current driving state prediction model
3:  Collect all sensing data received during $[g - \tau_c^{max} - T_s + 1, g - \tau_c^{max}]$. Sensing data comes from onboard
      sensors and received BSM.
4:  Find out the sensing data with the largest delay. Denote its measuring time step as $j_{min}(g)$.
5:  Based on the measuring time step derived from the timestamp of the sensing data, organize the collected
      sensing data into sets $S_i(g, j), \forall j \in [j_{min}(g), g]$.
6:  Generate the sets of sensor indexes $H_i(g, j), \forall j \in [j_{min}(g), g]$ according to the elements in $S_i(g, j)$.
7:  Assign a value to $u_i(j), \forall j \in [j_{min}(g), g]$ according to the control command that this vehicle had applied.
8:  If vehicle $i$ is the leading CAV
9:      Assign 0 to $u_{i-1}^*(j), \forall j \in [j_{min}(g), g]$.
10: Else if vehicle $i$ is a following CAV
11:     Assign the predicted control command from intent-sharing to $u_{i-1}^*(j), \forall j \in [j_{min}(g), g]$.
12: Restore the system states to the historical states of step $j_{min}(g)$.
13: For each $j \in [j_{min}(g), g]$
14:       Apply Equation 19 and Equation 20 to make a priori state estimation based on the system dynamic.
15:       If $S_i(g, j)$ is not empty
16:             For every $m \in [1, |S_i(g, j)|]$
17:                   Apply Equation 21, Equation 22, and Equation 23 to make a posterior estimation using sensing data $m$.
18: Output $\hat{X}_i(g) = \hat{X}_i(g|g)$ to the real-time MPC model.

Step 3: Compute real-time MPC-based control model
19: Use the predicted driving states in $\hat{X}_i(g)$ as the state variable initial condition of the MPC model for time $h$.
20: If vehicle $i$ is the leading CAV
21:       Determine the preceding vehicle states used in the MPC constraints by assuming the IPCV drives
            in the same velocity as its previous time over the prediction horizon.
22: Else if vehicle $i$ is a following CAV
23:       Determine the preceding vehicle states used in the MPC constraints by using the predicted driving states
            in intent-sharing data.
24: Solve the MPC problem described in Equation 24 to Equation 37 and get a feasible solution $u_i^*(h)$ before the
      start of the control interval $h$.
25: Output the control command $u_i^*(h)$ to the actuator.

Step 4: Execute the current optimal control command in the actuator
26: The actuator applies $u_i^*(h)$ at the start of control interval $h$.

Step 5: Broadcast intent-sharing and status-sharing messages via the OBU
27: Intent-sharing: update the predicted driving states over the prediction horizon, including $u_i^*(k)$, $l_i^*(k)$,
      and $v_i^*(k), \forall k = [h, h + T_p - 1]$ derived from the feasible MPC solution in BSM.
28: Status-sharing: update the current onboard sensor data in BSM.
29: Broadcast the BSM data via the OBU.
```

After solving the MPC for each control interval, vehicle i will execute the current optimal control command in the actuator and share the predicted variables in Equations 38 to 40 in BSM data for intent-sharing to its immediately following vehicle $i + 1$, in the communication topology in Figure 1.

$$u_i \times (k), \forall k \in [h, \ h + T_p - 1] \tag{38}$$

$$l_i \times (k) = l_i(k), \forall k \in [h, \ h + T_p - 1] \tag{39}$$

$$v_i \times (k) = v_i(k), \forall k \in [h, \ h + T_p - 1] \tag{40}$$

Solution Algorithm

This section presents the procedure of the proposed real-time predictive distributed CACC controlling framework. It consists of 5 steps. **Step 1** starts at the state update step $j = g - \tau_c^{max}$ because the proposed framework reserves τ_c^{max} state update steps for the computation. This framework makes sure **Step 1** to **Step 3** are finished before the start of the control interval h, that is, in a true real-time manner. **Step 2** is to compute the real-time current driving prediction. It should be noted that **Line 3** uses data from BSM status-sharing, and **Line 10** to **Line 11** use data from BSM intent-sharing. **Step 3** solves the real-time MPC-based control problem. Notably, **Line 22** to **Line 23** use the BSM intent-sharing data. **Step 4** applies the current optimal control command to the actuator. **Step 5** is to perform intent-sharing and status-sharing.

Numerical Analyses

This section evaluates the proposed real-time predictive distributed CACC control framework using real-world vehicle trajectories from the NGSIM dataset.

Experimental Setup

The proposed method is implemented in C++ on a computer with Intel Xeon E5-2680 v2 2.8 GHz CPU (22 cores). We simulate a platoon of seven vehicles. Two vehicle trajectories from the NGSIM I-80 dataset are used as the trajectory of the IPCV to generate traffic conditions in our simulation. The values of the average velocity of both trajectories are much lower than the actual speed limit, which is 65 mph (29.06 m/s), meaning that the vehicles are under congestion. Trajectory 2 consists of an obvious traffic shock. We denote the traffic scenario using trajectory 1 and trajectory 2 as the "no-shock" traffic (i.e., traffic is slow but smoothly moving) and "one-shock" traffic (i.e., traffic is slow and uncertain), respectively.

In our simulation, v^{limit}, u_i^{max}, and u_i^{min} are set to 15 m/s (33.55 mph), 3 m/s^2 and -5 m/s^2. Besides, t_{hw}, t_{hw}^{min}, s_i^{min}, and T_p are set to 1.6 s, 0.2 s, 4.7 m, and 10 steps. Δt_c and Δt are 0.1 s and 0.01 s. The sensing rate of Radar, GPS, wheel-speed sensor, and accelerometer are 50 Hz, 20 Hz, 100 Hz, and 100 Hz, respectively. The distance gap and relative velocity measurement errors (i.e., standard deviation) of Radar are 0.3 m and 0.25 m/s. The location, velocity, and acceleration measurement errors of GPS are 0.2 m, 0.1 m/s, and 0.08 m/s^2. The velocity measurement error of the wheel-speed sensor is 0.3 m/s. The acceleration measurement error of the accelerometer is 0.1 m/s^2. The location, velocity, and acceleration measurement in the BSM data has the same error as the GPS sensing data.

The string stability is analyzed in both the time domain and frequency domain. In relation to frequency domain analysis, this section uses the magnitude of spacing error string stability transfer function (Equation 42) as an indicator of the stability performance (*21*). $E_i(j\omega)$ is the Fourier transform of the spacing error $e_i(k)$ calculated by Equation 41.

$$e_i(k) = l_{i-1}(k) - l_i(k) - v_i(k) \times t_{hw} - s_i^{min} \tag{41}$$

$$|\Gamma_i(j\omega)| = \left|\frac{E_i(j\omega)}{E_{i-1}(j\omega)}\right| \tag{42}$$

Benefits of Intent-Sharing

This experiment compares the performance without and with intent-sharing on different traffic conditions. In no intent-sharing case, we simply assume the preceding vehicle to drive at constant velocity. Figure 8 shows the time-domain profile and frequency-domain plots of the

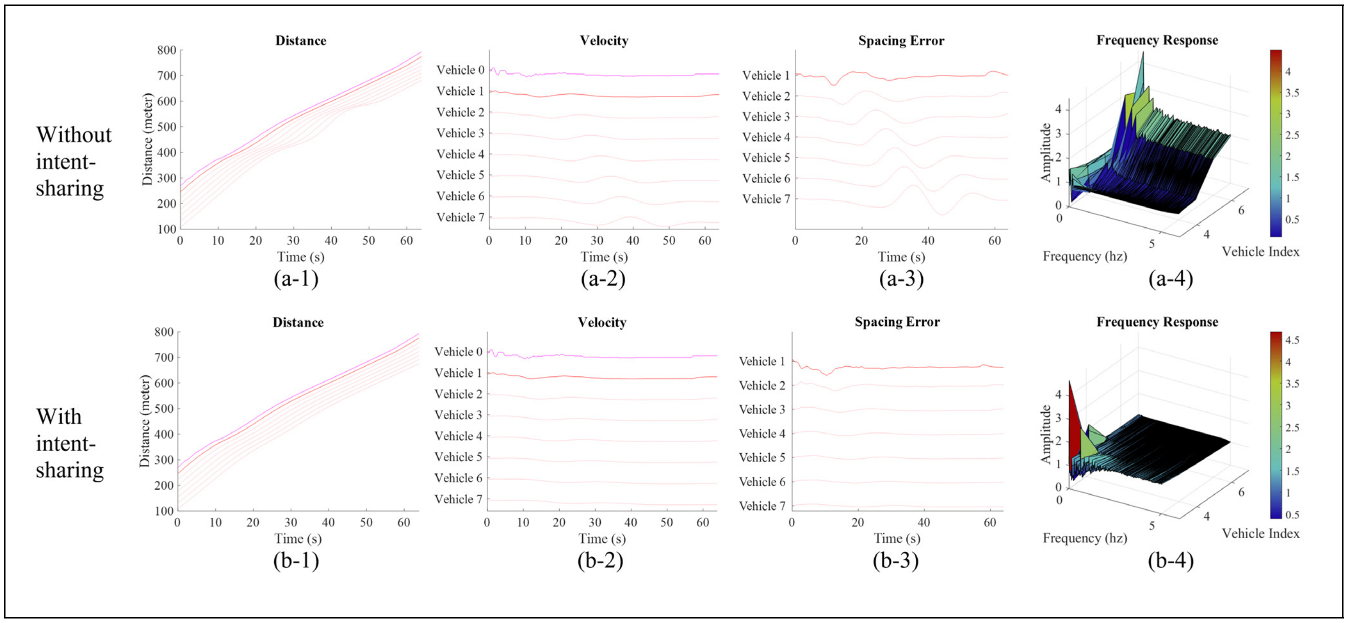

Figure 8. Performance without and with intent-sharing under no-shock traffic condition.

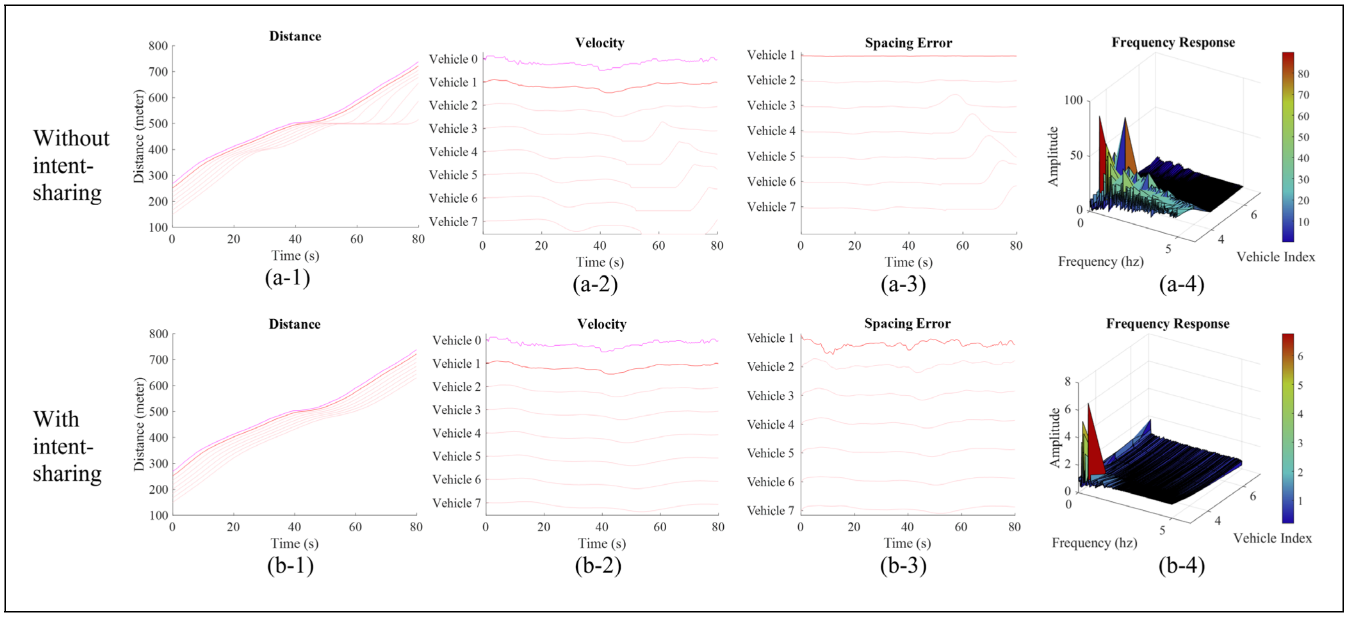

Figure 9. Performance without and with intent-sharing under one-shock traffic condition.

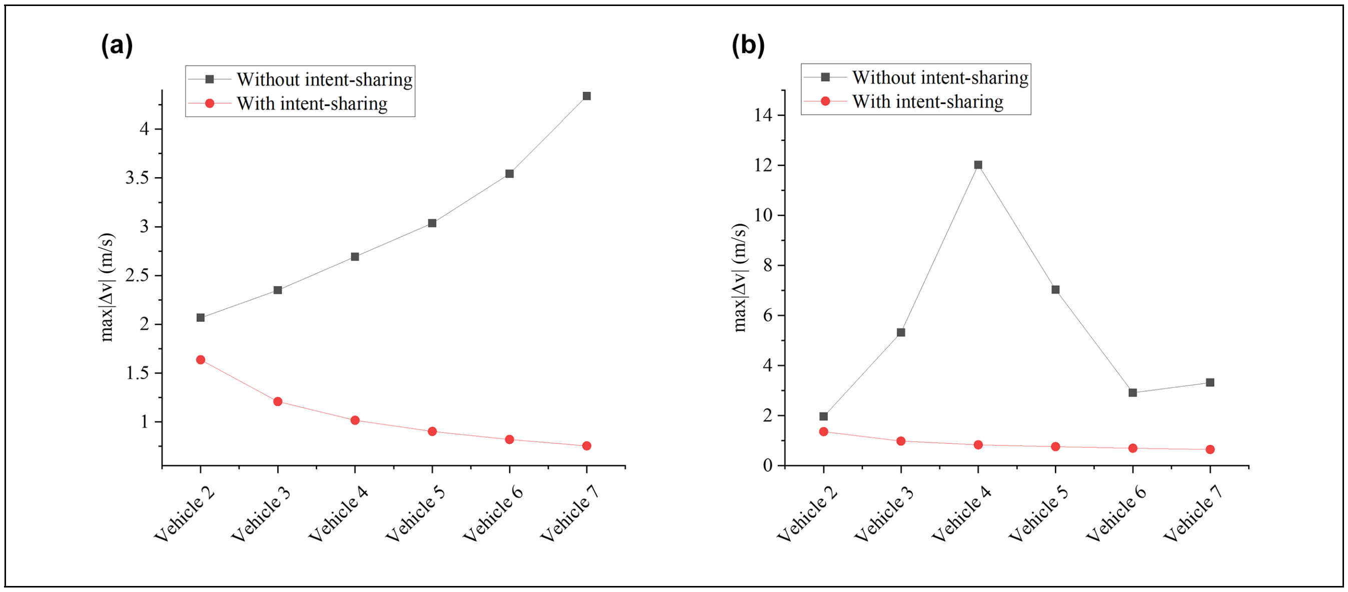

Figure 10. Maximum absolute relative velocity without and with intent-sharing under various traffic conditions: (*a*) no-shock traffic condition and (*b*) one-shock traffic condition.

vehicles under no-shock traffic. The purple line represents IPCV, the red line represents the leading CAV, and the pink lines represent the following CAVs. From Figure 8, the platoon is unstable without intent-sharing. The perturbation of the downstream traffic is amplified along the string. The amplitude of the transfer function increases along the string significantly, meaning that the spacing error is amplified. On the other hand, when we adopt intent-sharing, the platoon is stable. The perturbation from the downstream traffic is attenuated along the string. From Figure 9, when there is no intent-sharing, the platoon is unstable. The transfer function amplitude of vehicle 3 is very large, meaning that the spacing error is greatly amplified. In contrast, when there is intent-sharing, the platoon remains stable. The perturbations from the downstream traffic are attenuated. Figure 10 shows the maximum absolute relative velocity of every following CAV. In both traffic condition

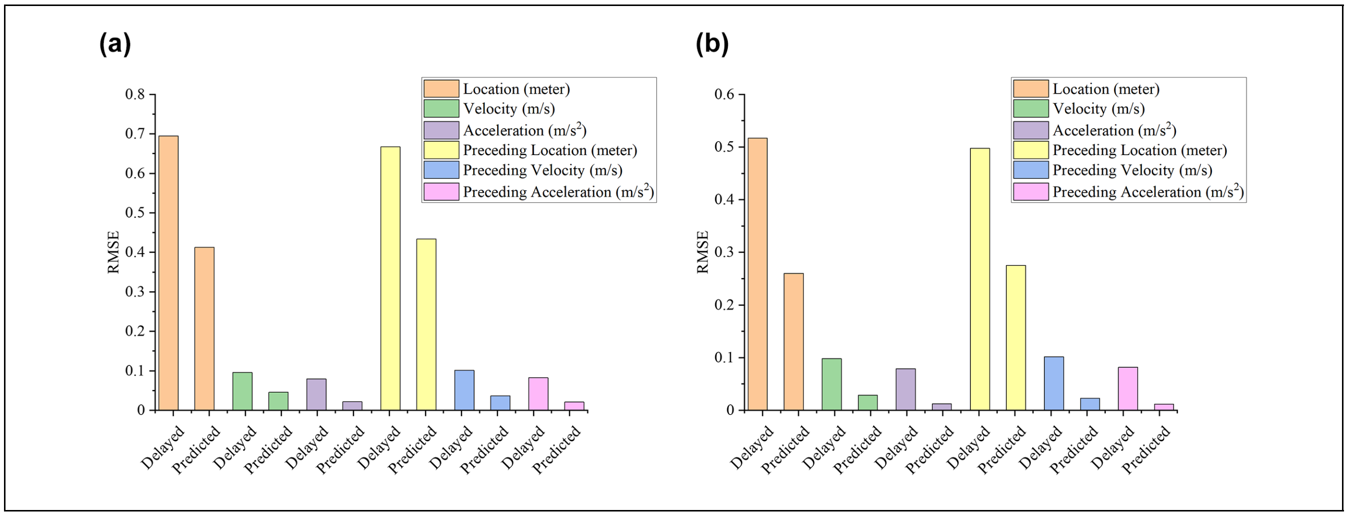

Figure 11. Root mean square error (RMSE) of the vehicle states without and with prediction under various traffic conditions: (*a*) no-shock and (*b*) one-shock.

scenarios, the maximum absolute relative velocities increase dramatically along the string when there is no intent-sharing. On the contrary, the maximum absolute relative velocity of every following CAV is smaller than that of its preceding CAV when intent-sharing is applied.

Benefits of the Real-Time Current Driving State Prediction for Delay Compensation

This experiment compares the performance without and with driving state prediction for delay compensation of sensing, communication, and computation. In the no-prediction case, we directly use the latest observed sensing data as the initial conditions in the MPC algorithm. We set GPS delay as 0.06 s, Radar delay as 0.01 s, and communication delay as 0.06 s. We calculate Root Mean Square Error (RMSE) for the six states in Equation 43. $\hat{x}_i$ can be any of the six states from the MPC model. x_i is the ground truth value of any of the six states.

$$RMSE(\hat{x}_i) = \sqrt{\frac{\sum_{j=1}^{N_T} (\hat{x}_i(j) - x_i(j))^2}{N_T}} \tag{43}$$

RMSEs of the vehicle states without and with prediction are shown in Figure 11. The RMSEs of all predicted states are smaller than those of the no-prediction states. This validates the capability of the real-time driving state prediction model in providing more accurate vehicle states.

Figures 12 and 13 show the stability performance without and with prediction under no-shock traffic and one-shock traffic. We can see a lot of ripples in the acceleration plot and spacing error plot of vehicles without prediction. In contrast, there are no ripples in the acceleration and spacing plot of the vehicles when prediction is used. This shows that the predicted current driving state leads to a smoother control. From the perspective of the frequency domain, sensing error would increase the magnitude at some frequencies (*22*). The amplitude without prediction is much larger than 1. This indicates that vehicles without the prediction model would significantly amplify the spacing error in some frequency bands. In contrast, the proposed prediction model is effective in attenuating the spacing error from the preceding vehicle. Therefore, the prediction model can not only smooth the control but also improve the string stability.

String Stability by Varying Sensing Delays

We evaluate the string stability of the proposed model given different sensing delays. The values of sensing delay in different scenarios are given in Figure 14. Communication delay, actuator lag, and computation delay are 0.06 s, 0.5 s, and 0.01 s, respectively. From Figure 14, the proposed framework can ensure good stability performance for GPS delay no larger than 0.22 s and Radar delay no larger than 0.03 s under no-shock traffic. When there is one shock, the platoon is still stable in small and medium delay. However, in the large delay scenario, the amplitude of vehicle 5 is much larger than its preceding vehicle in some frequencies. This indicates that the stability performance declines when the sensing delay is large. Based on our findings, the proposed framework can ensure good stability performance for GPS delay no larger than 0.16 s and Radar delay no larger than 0.02 s under one-shock traffic. According to

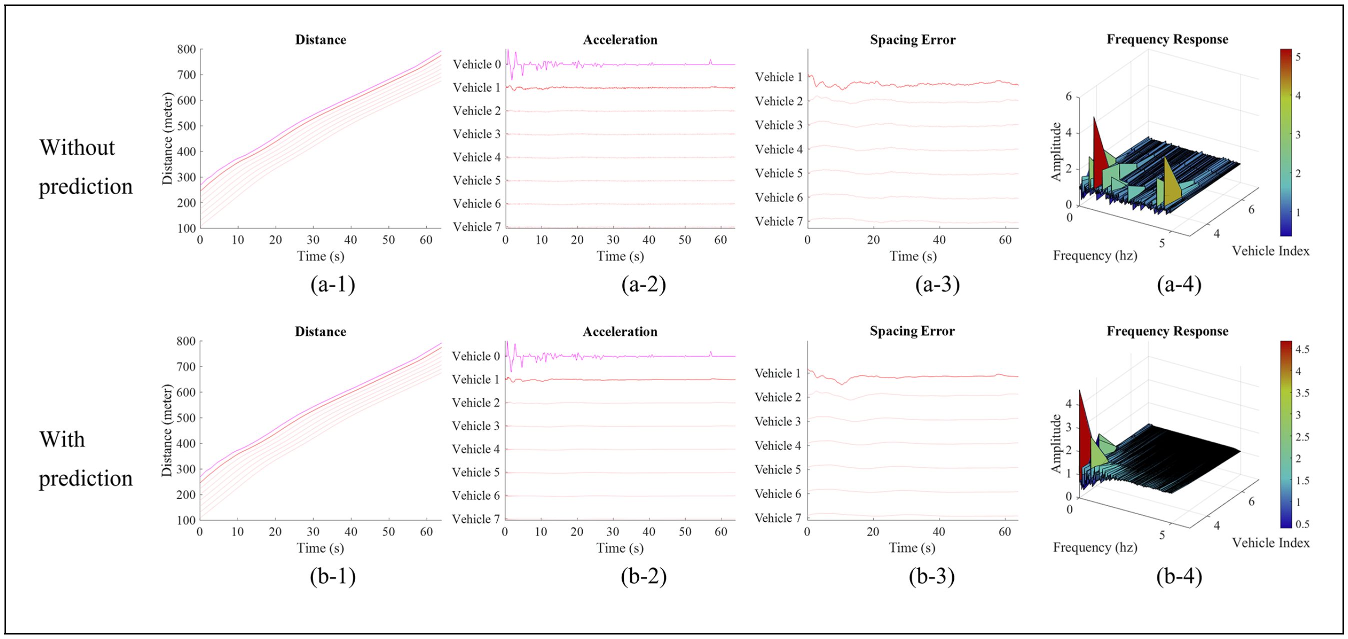

Figure 12. Stability analysis without and with prediction under no-shock traffic condition.

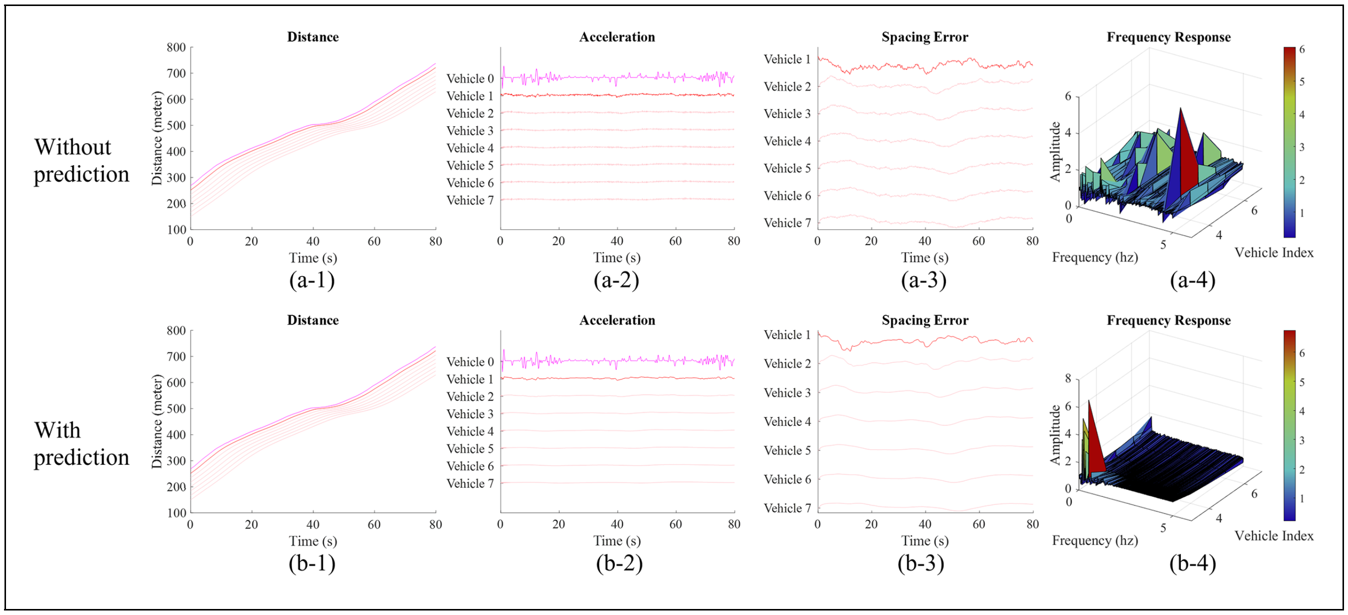

Figure 13. Stability analysis without and with prediction under one-shock traffic condition.

Figure 15, for all scenarios, the maximum absolute relative velocities decrease along the string. The results of all scenarios seem string stable in the time domain, whereas it is not the case in the frequency domain.

String Stability by Varying the Communication Delay

This section evaluates the stability of the proposed framework given different communication delays, which are 0.06 s, 0.14 s, and 0.22 s. According to Figure 16, the proposed framework can ensure stability for communication delay no larger than 0.22 s under no-shock traffic. In the one-shock traffic case, when the communication goes up to 0.22 s, the amplitude of the last following CAV becomes very large. Thus, the proposed framework can ensure stability for communication no larger than 1.4 s under the one-shock traffic condition. The results in the time domain shown in Figure 17 are very similar.

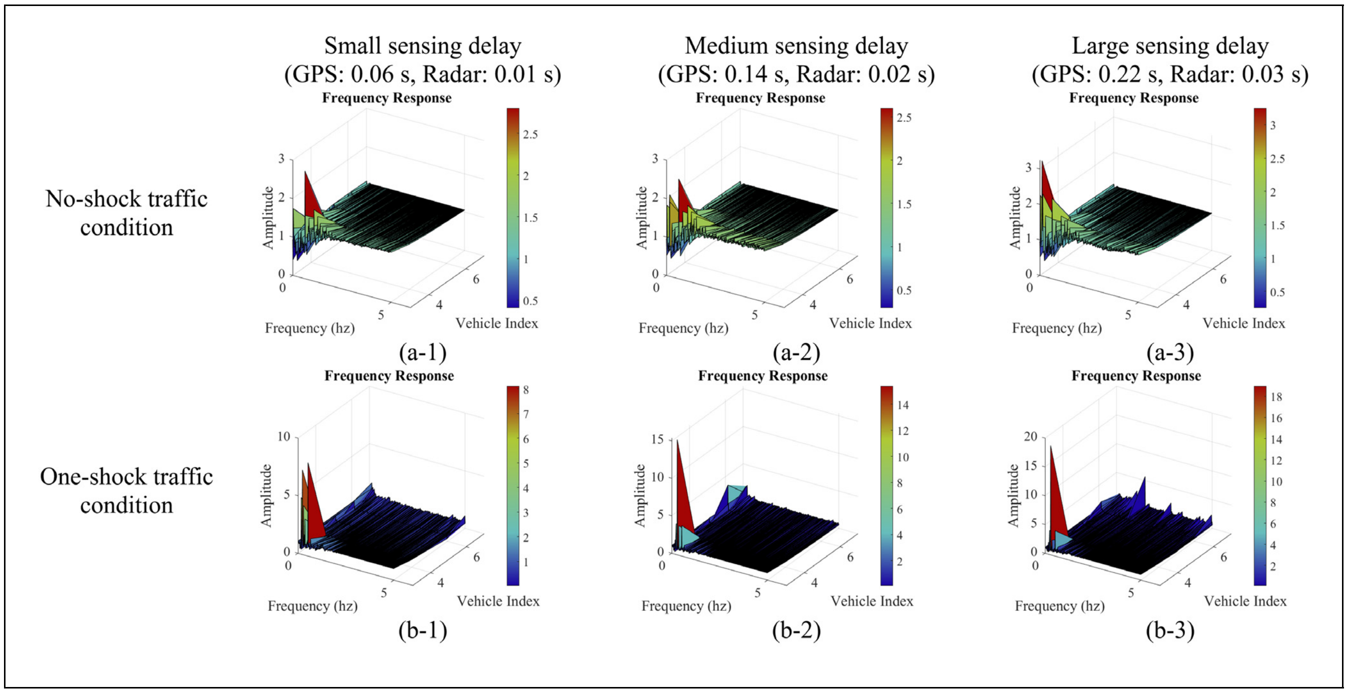

Figure 14. Stability analysis with different sensing delays under various traffic conditions.

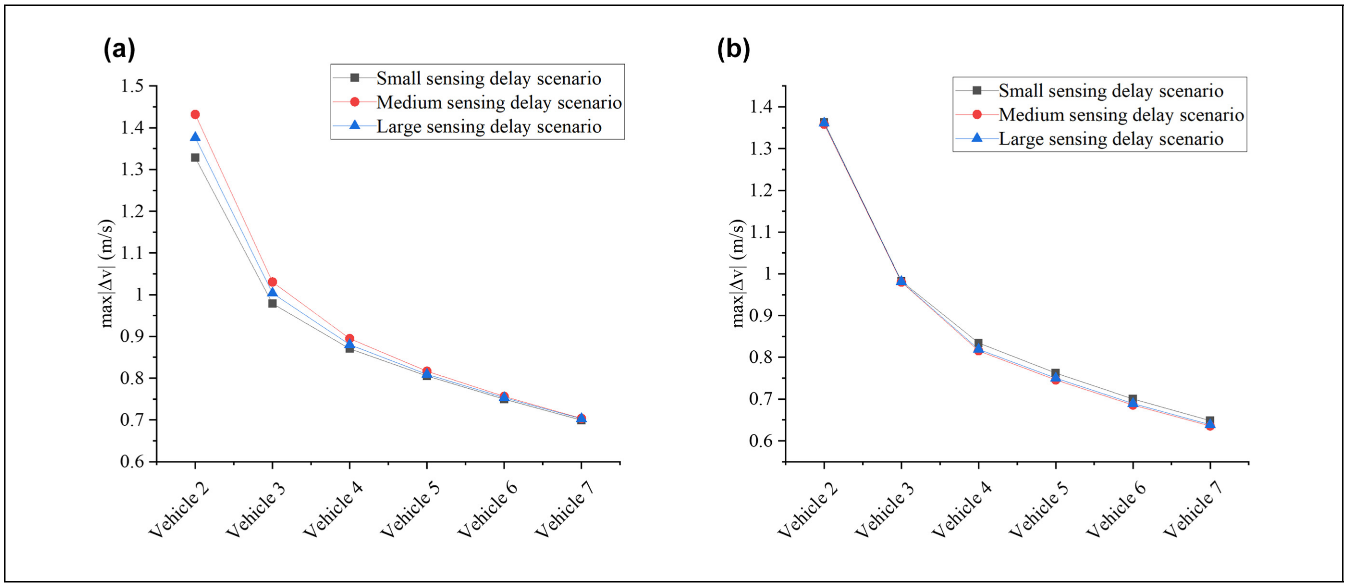

Figure 15. Maximum absolute relative velocity with different sensing delays under various traffic conditions: (*a*) no-shock traffic condition and (*b*) one-shock traffic condition.
Note: GPS = global positioning system.

However, in the frequency domain, we can see noticeable differences in different scenarios.

String Stability by Varying the Actuator Lag

This section evaluates the string stability of the proposed framework given different vehicle actuator lags, which are 0.5 s, 0.6 s, and 0.7 s. By Figure 18, in the no-shock traffic case, as the actuator lag increase, the amplitude of vehicle 3 becomes extensive. However, the amplitudes of the rest of the following CAVs are still close to 1. This means the perturbation from vehicle 3 does not propagate along the string. Thus, the proposed framework can remain stable for actuator lag no larger than 0.7 s in

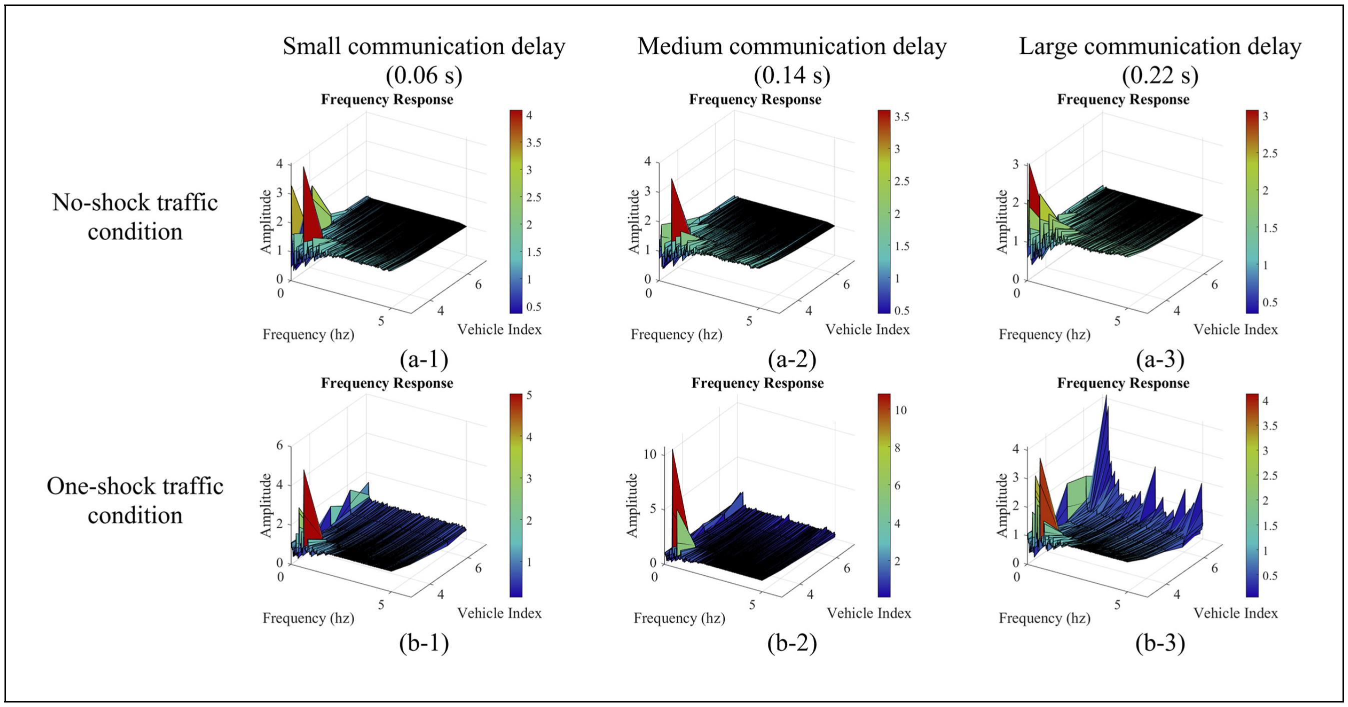

Figure 16. Stability analysis with different communication delays under various traffic conditions.

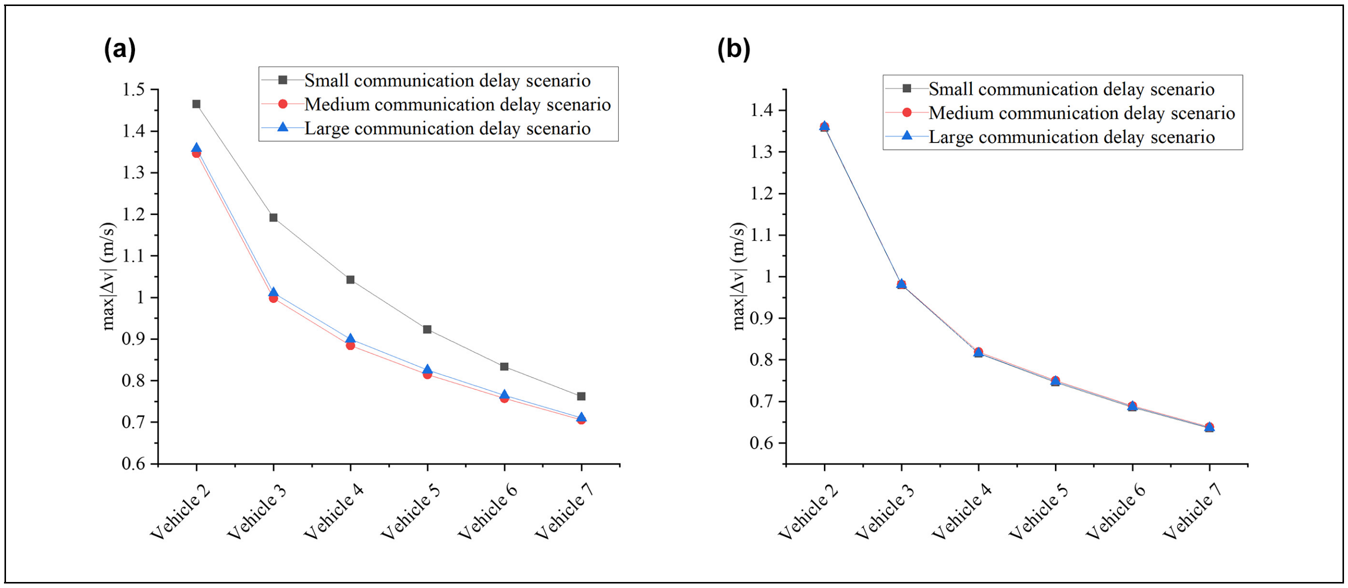

Figure 17. Maximum absolute relative velocity with different communication delays under various traffic conditions: (*a*) no-shock traffic condition and (*b*) one-shock traffic condition.

no-shock traffic. Likewise, it can ensure stability for actuator lag no larger than 0.6 s in the one-shock traffic condition. The results in the time domain shown in Figure 19 are again very similar, but we can see a noticeable difference in the frequency domain figures in Figure 18.

String Stability by Varying the Computation Delay

We measure the string stability under different computation delays, which are 0.02 s, 0.06 s, and 0.14 s. From Figure 20, in the no-shock traffic case, the amplitudes of vehicle 3 reduce to around 1 for frequency larger than

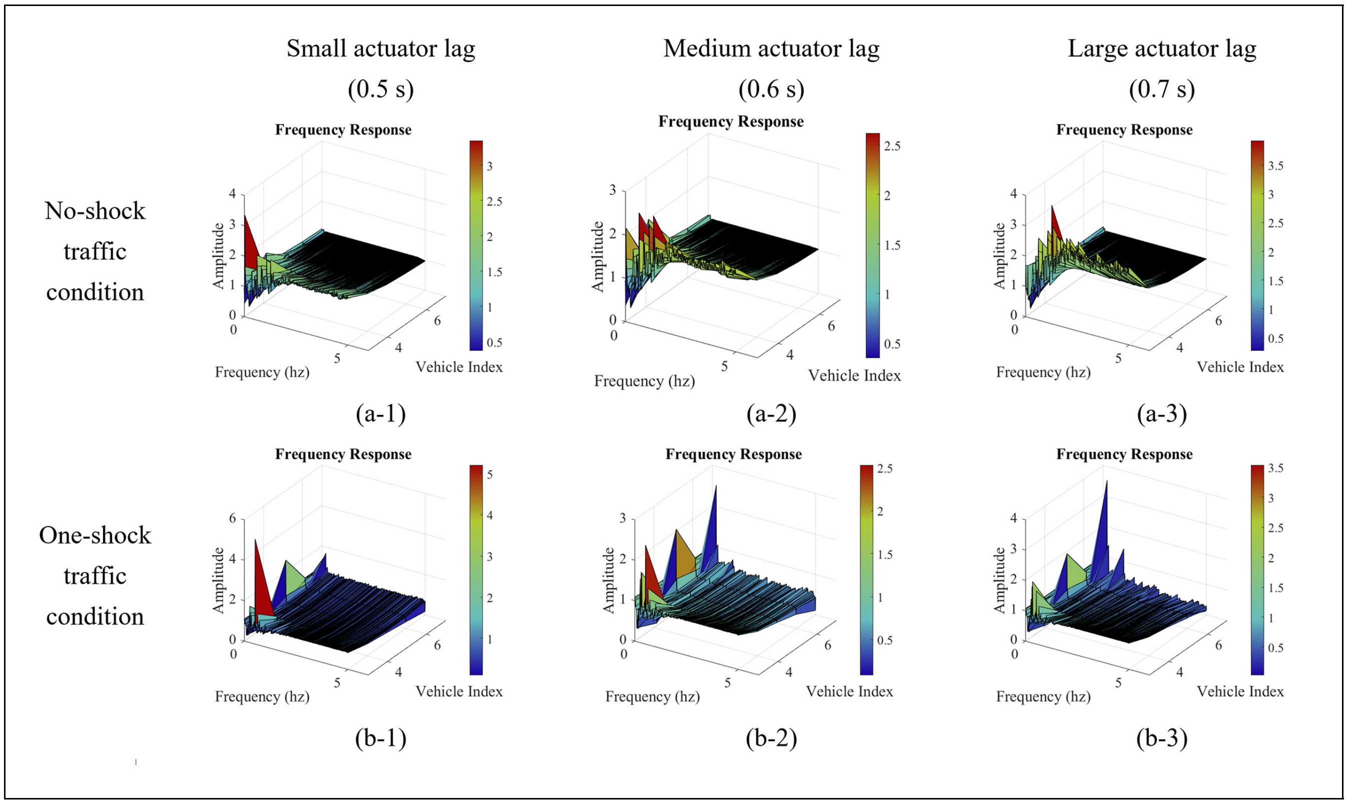

Figure 18. Stability analysis with different actuator lags under various traffic conditions.

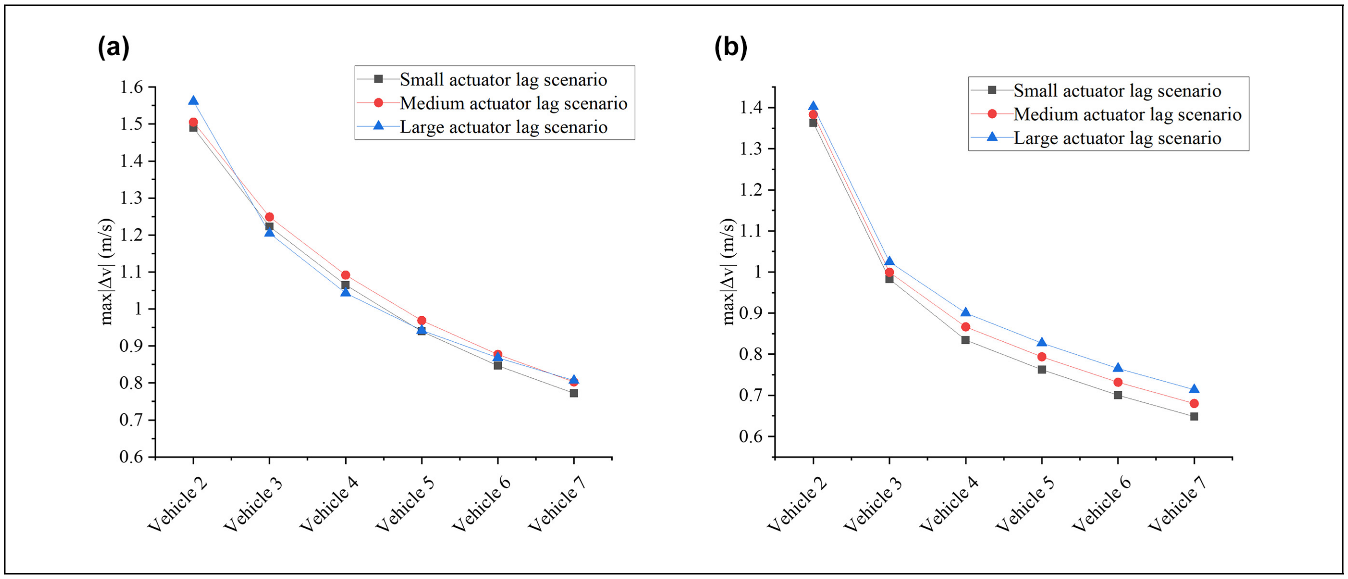

Figure 19. Maximum absolute relative velocity with different actuator lags under various traffic conditions: (*a*) no-shock traffic condition and (*b*) one-shock traffic condition.

2 Hz, which means it can ensure stability for computation delay no larger than 0.14 s under no-shock traffic. In the one-shock case, the amplitudes of all vehicles are close to 1 for frequency larger than 1 Hz, which means the platoons work well in attenuating high-frequency perturbations. When computation delay is 0.14 s, the

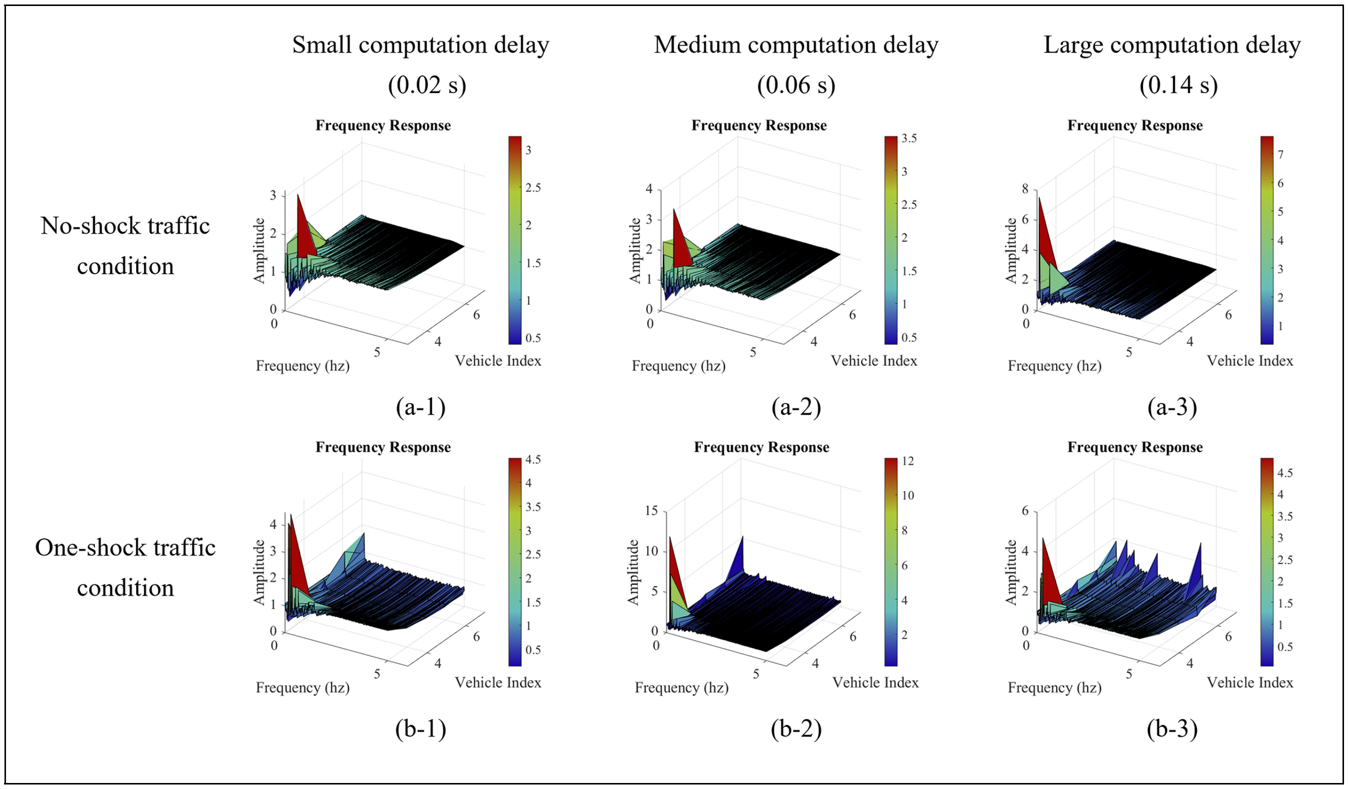

Figure 20. Stability analysis with different computation delays under various traffic conditions.

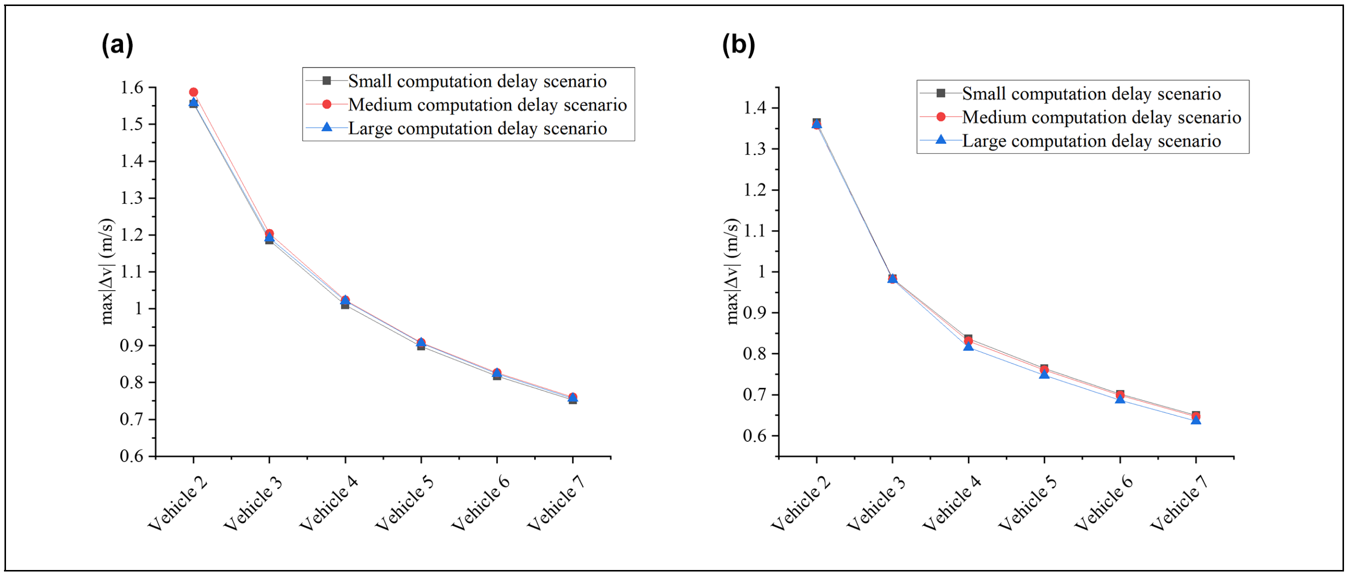

Figure 21. Maximum absolute relative velocity with different computation delays under various traffic conditions: (*a*) no-shock traffic condition and (*b*) one-shock traffic condition.

spacing error from vehicle 6 propagates to vehicle 7. So, the proposed framework can ensure stability for computation delay no larger than 0.06 s under the one-shock traffic condition. By Figure 21, for all scenarios, the maximum absolute relative velocities decrease along the string. The results are very similar in the time domain whereas they are different in the frequency domain.

Conclusions

In this paper, we propose a real-time predictive distributed CACC control framework for CDA of a fleet of CAVs to address the time delays and actuator lag issues. The framework uses a Kalman Filter-based real-time current driving state prediction model to compensate for the delays of sensing, communication, and computation. It provides a more accurate initial condition to the real-time MPC model with actuator lag. Intent-sharing via V2V communication is used in this framework with the distributed CACC communication topology. Numerical experiments shows that intent-sharing can improve string stability. Experiments on delay compensation by the real-time current driving state prediction model validate the capability of the proposed real-time driving state prediction model in providing more accurate driving states as the initial conditions in the MPC-based CACC controller. String stability is observed under the slow but smoothly moving traffic (i.e., without shocks). The maximum tolerable delays and actuator lag are also provided under traffic oscillations (i.e., traffic shocks).

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: Y. Tan and K. Zhang; data collection: Y. Tan and K. Zhang; analysis and interpretation of results: Y. Tan and K. Zhang; draft manuscript preparation: Y. Tan and K. Zhang. All authors reviewed the results and approved the final version of the manuscript.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was supported by a grant from the National Science Foundation (NSF) under an NSF CAREER award—CMMI-1846795.

ORCID iDs

Yingtong Tan https://orcid.org/0000-0003-3569-3487
Kuilin Zhang https://orcid.org/0000-0002-8180-5016

References

1. Wang, M., S. P. Hoogendoorn, W. Daamen, B. van Arem, B. Shyrokau, and R. Happee. Delay-Compensating Strategy to Enhance String Stability of Adaptive Cruise Controlled Vehicles. *Transportmetrica B: Transport Dynamics*, Vol. 6, No. 3, 2018, pp. 211-229.
2. Xu, S., H. Peng, and Y. Tang. Preview Path Tracking Control With Delay Compensation for Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 5, 2020, pp. 2979-2989.
3. Wang, M. Infrastructure Assisted Adaptive Driving to Stabilise Heterogeneous Vehicle Strings. *Transportation Research Part C: Emerging Technologies*, Vol. 91, 2018, pp. 276-295.
4. Pin, G., and T. Parisini. Networked Predictive Control of Uncertain Constrained Nonlinear Systems: Recursive Feasibility and Input-to-State Stability Analysis. *IEEE Transactions on Automatic Control*, Vol. 56, No. 1, 2010, pp. 72-87.
5. Tian, B., G. Wang, Z. Xu, Y. Zhang, and X. Zhao. Communication Delay Compensation for String Stability of CACC System Using LSTM Prediction. *Vehicular Communications*, Vol. 29, 2021, p. 100333.
6. Xing, H., J. Ploeg, and H. Nijmeijer. Smith Predictor Compensating for Vehicle Actuator Delays in Cooperative ACC Systems. *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 2, 2018, pp. 1106-1115.
7. Wang, J., S. Gong, S. Peeta, and L. Lu. A Real-Time Deployable Model Predictive Control-Based Cooperative Platooning Approach for Connected and Autonomous Vehicles. *Transportation Research Part B: Methodological*, Vol. 128, 2019, pp. 271-301.
8. Zavala, V. M., and L. T. Biegler. The Advanced-Step NMPC Controller: Optimality, Stability and Robustness. *Automatica*, Vol. 45, No. 1, 2009, pp. 86-93.
9. Rajamani, R. *Vehicle Dynamics and Control*. Springer Science & Business Media, Berlin, Heidelberg, 2011.
10. Tak, S., S. Kim, and H. Yeo. A Study on the Traffic Predictive Cruise Control Strategy With Downstream Traffic Information. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, No. 7, 2016, pp. 1932-1943.
11. Zhou, Y., and S. Ahn. Robust Local and String Stability for a Decentralized Car Following Control Strategy for Connected Automated Vehicles. *Transportation Research Part B: Methodological*, Vol. 125, 2019, pp. 175-196.
12. Zhao, S., and K. Zhang. A Distributionally Robust Stochastic Optimization-Based Model Predictive Control With Distributionally Robust Chance Constraints for Cooperative Adaptive Cruise Control Under Uncertain Traffic Conditions. *Transportation Research Part B: Methodological*, Vol. 138, 2020, pp. 144-178.
13. Zhang, W. A., B. Chen, and M. Z. Chen. Hierarchical Fusion Estimation for Clustered Asynchronous Sensor Networks. *IEEE Transactions on Automatic Control*, Vol. 61, No. 10, 2015, pp. 3064-3069.
14. Negenborn, R. R., and J. M. Maestre. Distributed Model Predictive Control: An Overview and Roadmap of Future Research Opportunities. *IEEE Control Systems Magazine*, Vol. 34, No. 4, 2014, pp. 87-97.
15. Zhou, Y., M. Wang, and S. Ahn. Distributed Model Predictive Control Approach for Cooperative Car-Following With Guaranteed Local and String Stability. *Transportation Research Part B: Methodological*, Vol. 128, 2019, pp. 69-86.
16. Maxim, A., O. Pauca, C. F. Caruntu, and C. Lazar. Distributed Model Predictive Control Algorithm With

Time-Varying Communication Delays for a CACC Vehicle Platoon. *Proc., 24th International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, Romania, IEEE, New York, October 8, 2020, pp. 775-780.

17. Zheng, Y., S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick. Distributed Model Predictive Control for Heterogeneous Vehicle Platoons Under Unidirectional Topologies. *IEEE Transactions on Control Systems Technology*, Vol. 25, No. 3, 2016, pp. 899-910.
18. Dunbar, W. B., and D. S. Caveney. Distributed Receding Horizon Control of Vehicle Platoons: Stability and String Stability. *IEEE Transactions on Automatic Control*, Vol. 57, No. 3, 2011, pp. 620-633.
19. Ploeg, J., D. P. Shukla, N. van de Wouw, and H. Nijmeijer. Controller Synthesis for String Stability of Vehicle Platoons. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 15, No. 2, 2013, pp. 854-865.
20. Anderson, B. D., and J. B. Moore. *Optimal Filtering*. Courier Corporation, Chelmsford, MA, 2012.
21. Naus, G. J., R. P. Vugts, J. Ploeg, M. J. van De Molengraft, and M. Steinbuch. String-Stable CACC Design and Experimental Validation: A Frequency-Domain Approach. *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 9, 2010, pp. 4268-4279.
22. Öncü, S., J. Ploeg, N. Van de Wouw, and H. Nijmeijer. Cooperative Adaptive Cruise Control: Network-Aware Analysis of String Stability. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 15, No. 4, 2014, pp. 1527-1537.

The authors are solely responsible for the content of the paper.