sensors

Article

Energy-Optimal Adaptive Control Based on Model Predictive Control

Yuxi Li [1] and Gang Hao [1,2,*]

[1] School of Electronic Engineering, Heilongjiang University, Harbin 150080, China; 2211763@s.hlju.edu.cn
[2] Key Laboratory of Information Fusion Estimation and Detection, Harbin 150080, China
* Correspondence: haogang@hlju.edu.cn; Tel.: +86-187-4568-1046

Abstract: Energy-optimal adaptive cruise control (EACC) is becoming increasingly popular due to its ability to save energy. Considering the negative impacts of system noise on the EACC, an improved modified model predictive control (MPC) is proposed, which combines the Sage-Husaadaptive Kalman filter (SHAKF), the cubature Kalman filter (CKF), and the back-propagation neural network (BPNN). The proposed MPC improves safety and tracking performance while further reducing energy consumption. The final simulation results show that the proposed algorithm has a stronger energy-saving capability compared to previous studies and always maintains an appropriate relative distance and relative speed to the vehicle in front, verifying the effectiveness of the proposed algorithm.

Keywords: model predictive control (MPC); cubature Kalman filter; energy-optimal cruise control; artificial neural network

check for updates

Citation: Li, Y.; Hao, G. Energy-Optimal Adaptive Control Based on Model Predictive Control. *Sensors* **2023**, *23*, 4568. https://doi.org/10.3390/s23094568

Academic Editors: Zhiheng Li, Hailong Zhu, Yilong Ren and Jiyuan Tan

Received: 19 April 2023
Revised: 4 May 2023
Accepted: 5 May 2023
Published: 8 May 2023

1. Introduction

Electric and hybrid vehicles are gaining popularity as environmentally friendly alternatives to conventional fuel vehicles due to their reduced emissions and lower operating costs [1–3]. However, electric vehicles face issues such as limited range, extended charging times, and high battery costs [4]. As a result, it is crucial to develop eco-driving assistance systems that optimize energy consumption in electric vehicles while maintaining safety and comfort [5].

Adaptive cruise control (ACC) is a widely used driving assistance system that automatically adjusts a vehicle's speed and maintains a safe distance from the vehicle ahead. Nevertheless, conventional ACC systems have limitations in terms of energy efficiency, driver comfort, and tracking stability. To address these challenges, researchers have proposed various ACC extensions, including cooperative adaptive cruise control (CACC) [6–8], personalized adaptive cruise control (PACC) [9–11], and energy-optimal adaptive cruise control (EACC) [1,12–15]. CACC enables multi-vehicle cooperation through vehicle communication systems, while PACC tailor sand simulates individual driving habits. EACC focuses on energy consumption and aims to conserve energy while tracking the leading vehicle. Several studies have applied model predictive control (MPC) to ACC problems, yielding promising results [12,13].

EACC is a crucial aspect of ACC, but it faces challenges due to the uncertainty of the leading vehicle's state. One approach to this issue involves combining MPC and dynamic programming (DP) algorithms. Weißmann A et al. use DP to plan the speed and route from the cloud based on the starting position, while the host vehicle uses the cloud information to estimate the leading vehicle's speed in real time and applies MPC for calculation and control [14]. Alternatively, Pan C et al. combine an economical linearized energy consumption equation with MPC, proposing the use of a nonlinear autoregressive model with exogenous inputs (NARX) to predict the leading vehicle's speed [15]. Similar methods employ the conditional linear Gauss (CLG) model to predict the leading vehicle's speed and control the host vehicle using chance constraints, stochastic MPC, and randomized MPC [7].

Some existing ACC studies only consider vehicle measurement noise [16], while most others disregard noise altogether [1,7,12–14]. A few papers assume the use of filters for preprocessing [17]. ACC algorithms for cars with ultrasonic sensors have been researched and considered [18], but these sensors have limited range and accuracy and can only measure vehicle distance. Noise is pervasive and can significantly impact ACC performance. Inaccurate system process state estimation can lead to control system instability, resulting in frequent acceleration and deceleration and wasting energy. Hence, it is essential to develop EACC algorithms capable of handling noisy conditions.

Some recent studies have considered control systems with noise. Aubeck F et al. addressed the plug-in hybrid vehicle energy management problem using a generalized stochastic particle filtering algorithm for filtering, followed by a two-level MPC for coordinated vehicle fuel use and charging management [19]. Another example is [20], where Yan D et al. combined a Kalman-consistent filter and a fixed-time disturbance observer with a multi-constraint MPC strategy to control the formation flight of unmanned aerial vehicles. To solve the problem above, an EACC system algorithm is proposed for solving vehicle systems with noise. The main contributions of this paper are as follows:

(1) In order to address the effects of process noise and measurement noise in vehicle nonlinear systems, a Sage-Husa adaptive cubature Kalman filter (SHACKF) is proposed. By filtering, the leading vehicle's speed prediction model is improved to tackle the issue of diminished multi-step prediction accuracy.

(2) Secondly, a back-propagation neural network (BPNN) for trend prediction is incorporated, which can be combined with various leading vehicle speed prediction models. Additionally, the energy consumption equation and kinetic energy recovery system (KERS) are considered in order to address the problem of frequent deceleration due to noise. The control is limited to a range that excludes mechanical braking, and interpolation is used to fit the motor's efficiency. This is combined with MPC to calculate the most energy-efficient operating point.

Comparison experiments were conducted under two types of cycles, and the results show that the proposed algorithm significantly improves energy savings, following distance, and following relative speed compared to previous studies.

The rest of this paper is organized as follows: the system model is presented in Section 2; the filter and controller are designed in Section 3; the simulation results are shown in Section 4; and the conclusions are given in Section 5.

2. Modeling

In this paper, we focus on a vehicle-following scenario on a single lane, aiming to control the host vehicle to maintain an optimal distance from the front vehicle while minimizing energy consumption. To implement the algorithm, the first step is to model the vehicle cruise process equation, measurement equation, and energy consumption equation.

2.1. Vehicle Longitudinal Dynamics Model

Figure 1 depicts a schematic illustration of two consecutive vehicles in the traffic flow, where k represents discrete time, $V_{f,k}$ represents the velocity of the front vehicle, d_k represents the distance difference between the two vehicles, $V_{h,k}$ denotes the velocity of the host vehicle, and $a_{h,k}$ denotes the acceleration of the host vehicle. The considered system process vector $\boldsymbol{x}_k$ can be modeled as follows:

$$\boldsymbol{x}_k = \begin{bmatrix} V_{f,k} & d_k & V_{h,k} & a_{h,k} \end{bmatrix}^T \tag{1}$$

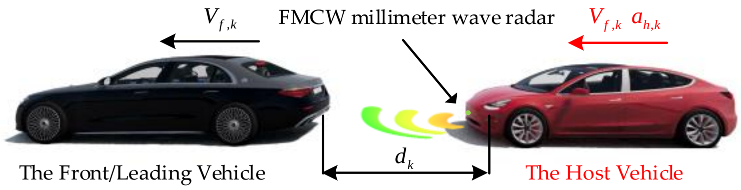


Figure 1. Schematic of two consecutive vehicles.

Firstly, the state of the host vehicle is considered. Numerous papers have proposed models for the vehicle-following scenario [1,21]. Specifically, the relationship between the velocity and control input of the host vehicle is as follows:

$$ma_{h,k} = \frac{T_m(u_k)i\eta}{r} - F_{rolling} - F_{air}(V_{h,k}) - F_{grade} \tag{2}$$

$$V_{h,k+1} = V_{h,k} + T_s a_{h,k} \tag{3}$$

where m denotes the vehicle mass, T_s is the sampling and control period, $T_m(u_k)$ is the motor output torque affected by control input u_k, which will be described in detail in Section 2.3, i is the transmission ratio product of the gearbox and the main reducer, η is the transmission system efficiency, and r is the wheel radius. The resistances $F_{rolling}$, $F_{air}\left(V_{h,k}\right)$, and F_{grade} represent the rolling resistance, the air resistance function, and the grade resistance, respectively [1,21]. They are calculated as follows:

$$F_{rolling} = mgf_r \cos\alpha \tag{4}$$

$$F_{air}\left(V_{h,k}\right) = \frac{C_w \rho A_{window} V_{h,k}^2}{2} \tag{5}$$

$$F_{grade} = mg \sin\alpha \tag{6}$$

where f_r is the rolling resistance coefficient, g is the gravitational acceleration, α is the road slope angle, ρ is the air density, C_w is the drag coefficient, and A_{window} is the vehicle's windward area.

Various solutions have been offered to predict the front vehicle's velocity for model predictive control (MPC), such as the Coordinated Leading Guidance (CLG) method [7] and the Nonlinear Auto Regressive with Exogenous Inputs (NARX) method [15]. These algorithms are constructive, but they have some limitations. Using these algorithms for systems with noise may lead to noise accumulation in multi-step predictions, resulting in inaccurate predictions.

The BPNN is combined with the forward vehicle speed prediction model proposed in the current study. As an example, we improve the NARX model suggested in [15]. The NARX and the BPNN are used for single-step and multi-step predictions, respectively. The BPNN aims to fit the possible increments for each step. The BPNN's input is the front vehicle's velocity from time $k - p_p$ to time k, defined as follows:

$$\boldsymbol{h}_{pre} \triangleq \left[\hat{V}_{f,k-p_h|k-p_h}, \cdots, \hat{V}_{f,k-1|k-1}, \hat{V}_{f,k|k}\right] \tag{7}$$

where p_h represents the setting historical horizon.

The structure of the model for front vehicle velocity prediction is shown in Figure 2, where "Act" represents the activation layer.

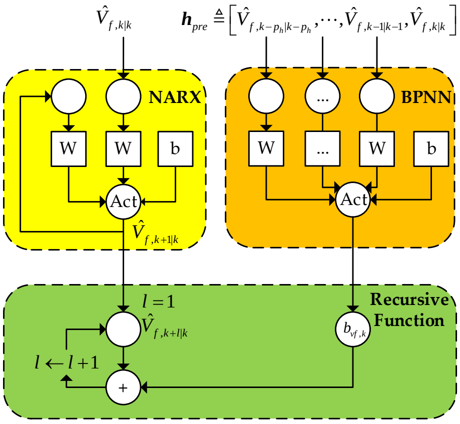


Figure 2. Structure diagram of the velocity of the front vehicle prediction model.

Using this approach, the l step at time k can be predicted as follows:

$$\hat{V}_{f,k+l|k} = \text{NARX}\left(\hat{V}_{f,k|k}\right) + (l-1)\text{BPNN}\left(\boldsymbol{h}_{pre}\right) \tag{8}$$

where $\text{NARX}\left(\hat{V}_{f,k|k}\right)$ and $\text{BPNN}\left(\boldsymbol{h}_{pre}\right)$ denote the predictions of the NARX and the BPNN, respectively.$\text{NARX}\left(\hat{V}_{f,k|k}\right)$ and $\text{BPNN}\left(\boldsymbol{h}_{pre}\right)$ are considered special functions. Equation (8) can be converted into a recursion formula as follows:

$$\hat{V}_{f,k+l|k} = \begin{cases} \text{NARX}\left(\hat{V}_{f,k|k}\right) & l=1 \\ \hat{V}_{f,k+l-1|k} + \text{BPNN}\left(\boldsymbol{h}_{pre}\right) & l>1 \end{cases} \tag{9}$$

Furthermore, the relative distance between the front vehicle and the host vehicle can be calculated as follows:

$$d_{k+l+1} = T_s\left(\hat{V}_{f,k+l|k} - \hat{V}_{h,k+l}\right) \tag{10}$$

The approach used in the [7] algorithm of the CLG can also be applied here. These sequences of traffic light signals and the historical speed sequence of the previous vehicle's velocity are combined and input into the BPNN. At the same time, the CLG is simplified to perform Bayesian network operations, executing only one step of prediction.

The above Equations (1)–(9) are ideal, but the actual situation is much more complicated. The primary sources of errors are:

Vehicle variation, which includes changes in the vehicle mass due to changes in the weight of the passengers and changes in the rolling resistance coefficient due to changing tires; environment variation, which includes changes in gravitational acceleration, wind resistance coefficient, and air density; and modeling errors, which include the effect of lateral direction control on velocity [8], the effect of state of charge (SOC) changes on battery voltage [22], and the effect of tire pressure on grip during acceleration and braking. The velocity of the front vehicle cannot be directly modeled, and the prediction of speed must have errors. Although considering more variables results in less process noise, accounting for too many factors will cause the matrix dimension to become too large.

Therefore, a balance between accuracy and complexity should be struck. The first two points affect the parameters in Equations (2) and (4)–(6). There are specific patterns of change over a short period of time. The last point causes random noise. These errors are described using w_k, thus w_k is an additive noise with a non-zero mean that can be considered to obey a Gaussian process. Section 3.1 will demonstrate that the filter is statistically consistent between the Gaussian process assumption and the actual situation through a series of tests [23,24].

According to the series of derivations above, the EACC system process equation can be written as follows:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} V_{f,k+1} \\ d_{k+1} \\ V_{h,k+1} \\ a_{h,k+1} \end{bmatrix} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k = \begin{bmatrix} \mathrm{NARX}\left(V_{f,k}\right) \\ T_s V_{f,k} + d_k - T_s V_{h,k} \\ V_{h,k} + T_s a_{h,k} \\ \frac{\frac{T_k(u_k) i \eta}{r} - F_{rolling} - F_{air}\left(V_{h,k}\right) - F_{grade}}{m} \end{bmatrix} + \boldsymbol{w}_k \tag{11}$$

where $\mathrm{BPNN}\left(\boldsymbol{h}_{pre}\right)$ in Equation (9) is not used in (11) but is used to make predictions in the controller, which will be described in detail in Section 3.2.

2.2. Vehicle Sensor Measurement Model

The goal of this section is to deduce the measurement equation by modeling sensors. Typically, most vehicles are equipped with a comprehensive set of sensors. The FMCW radar measures the relative velocity and distance between two vehicles, the wheel velocity sensor provides the host vehicle's velocity, and the inertial measurement unit supplies the host vehicle's acceleration.

Assuming that the measurement noise covariances $\boldsymbol{v}_k$ for these sensors are Gaussian white noise with known values, the values of the measurement noise covariances can be obtained from sensor manuals or prior studies. The initial measurement noise covariance diagonal matrix $\boldsymbol{Q}_{v0}$ is given by [16,25,26]:

$$\boldsymbol{Q}_{v0} = \mathrm{diag}(0.055, 0.28, 1, 0.005) \tag{12}$$

However, this assumption may not hold in actual situations, where the measurement noise covariances may vary over time and have a non-zero mean. For example, severe tire slippage on wet roads can increase the wheel velocity sensor's error [25]. Likewise, rain and fog may cause increased noise, reducing the radar's penetration effect [18,27,28]. This situation is described in Section 3.1.

The EACC system measurement equation is written as follows:

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{30}{\pi r} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \boldsymbol{x}_k + \boldsymbol{v}_k \tag{13}$$

where $30/\pi r$ is the conversion of velocity (unit: m/s) to wheel velocity (unit: rpm).

2.3. Energy Model

In this section, the equations for energy consumption and recovery are derived. The efficiency of the motor at the working points is given by:

$$P_m = \begin{cases} \frac{T_m(u_k) n_k}{9549 \eta_m (T_m(u_k), n_k)} & T_m(u_k) \geq 0 \\ \frac{T_m(u_k) n_{m,k}}{9549} \eta_m (T_m(u_k), n_k) & T_m(u_k) \leq 0 \end{cases} \tag{14}$$

where n_k is the motor's rotation speed, $\eta_m(T_m(u_k), n_k)$ is the electrical energy conversion efficiency, a quantity related to the motor speed and torque, and 9549 is the coefficient of unit conversion. Equation (14) is adapted from [15].

The KERS can recover some of the vehicle's kinetic energy during braking and convert it into electrical energy. The braking process is divided into two stages based on braking strength [1,29]:pure electric braking and electro-hydraulic hybrid braking. The hybrid braking stage uses a combination of motor braking and hydraulic braking. The first stage saves more energy than the second stage because hydraulic braking converts part of the kinetic energy into thermal energy, resulting in energy waste.

In this paper, when u_k is positive, the motor performs energy output; when it is negative, it signifies that the KERS performs energy recovery. Furthermore, $u_k = 1$ and $u_k = -1$ represent the maximum throttle opening and the critical point between the two stages of braking, respectively. In this paper, the output torque and the input control quantity are considered linearly related, so the torque output function $T_k(u_k)$ and the control input u_k exhibit the following relationship:

$$T_m(u_k) = \begin{cases} u_k T_m^{(+)} & u_k > 0 \\ u_k T_m^{(-)} & u_k < 0 \end{cases} \tag{15}$$

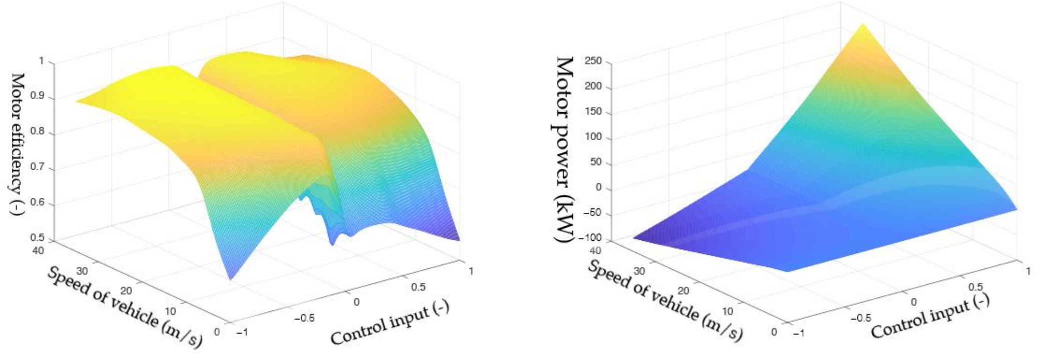
where $T_m^{(+)}$ and $T_m^{(-)}$ are the maximum torque for energy output and energy recovery input, respectively. Since only one set of gears exists for electric vehicles, the relationship between motor speed n_k and vehicle velocity $V_{f,k}$ is as follows:

$$V_{h,k} = \frac{n_k \pi r}{30i} \tag{16}$$

Substituting (15) and (16) into (14) yields:

$$P_m(u_k, V_{h,k}) = \begin{cases} \frac{10u_k T_m^{(+)} iV_{h,k}}{3183\pi r\eta_m(u_k, V_{h,k})} & u_k \geq 0 \\ \frac{10u_k T_m^{(-)} iV_{h,k}}{3183\pi r}\eta_m(u_k, V_{h,k}) & u_k \leq 0 \end{cases} \tag{17}$$

Since the efficiency function $\eta_m(u_k, V_{h,k})$ is nonlinear, the equation is fitted using Thin-plate spline interpolation, and the final result is shown in Figure 3a. The $P_m(u_k, V_{h,k})$ obtained by Equation (17) is shown in Figure 3b.



(**a**) (**b**)

Figure 3. (**a**) Interpolation fitting surface for the relationship between motor efficiency and control input and motor speed; (**b**) The surface of the relationship between motor power and control input and motor speed.

Similar fitting methods can be applied to hybrid vehicles, with special consideration required for the product of the gearbox and main gearbox ratios.

3. Filter and Controller Algorithm

3.1. Filter Algorithm

Equations (11) and (13) present challenges for conventional nonlinear filtering methods, such as the extended Kalman filter (EKF) and the cubature Kalman filter (CKF) [30], as they assume that the noise is zero-mean Gaussian white noise with fixed covariance. Additionally, linear adaptive filtering methods, including the Sage-Husa adaptive Kalman filter (SHAKF) [31–33], cannot handle nonlinear functions. To address these limitations, we propose the SHACKF, a combination of the CKF and the SHAKF. The CKF approximates nonlinear functions using cubature points. The SHAKF algorithm parameters are updated as follows:

$$\hat{\boldsymbol{q}}_{w,k} = (1-\delta_{k-1})\hat{\boldsymbol{q}}_{w,k-1} + \delta_{k-1}\left(\hat{\boldsymbol{x}}_{k|k} - \boldsymbol{A}_k\hat{\boldsymbol{x}}_{k-1|k-1}\right) \tag{18}$$

$$\hat{\boldsymbol{Q}}_{w,k} = (1-\delta_{k-1})\hat{\boldsymbol{Q}}_{w,k-1} + \delta_{k-1}\left(\boldsymbol{K}_k\epsilon_k\epsilon_k^T\boldsymbol{K}_k^T + \boldsymbol{P}_{k|k} - \boldsymbol{A}_k\boldsymbol{P}_{k|k}\boldsymbol{A}_k\right) \tag{19}$$

$$\hat{\boldsymbol{q}}_{v,k} = (1-\delta_{k-1})\hat{\boldsymbol{q}}_{v,k-1} + \delta_{k-1}\left(\hat{\boldsymbol{z}}_{k|k} - \boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k-1}\right) \tag{20}$$

$$\hat{\boldsymbol{Q}}_{v,k} = (1-\delta_{k-1})\hat{\boldsymbol{Q}}_{v,k-1} + \delta_{k-1}\left(\epsilon_k\epsilon_k^T - \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T\right) \tag{21}$$

where $\hat{\boldsymbol{q}}_{w,k}$ and $\hat{\boldsymbol{Q}}_{w,k}$ denote the estimated process noise mean and covariances, respectively; $\hat{\boldsymbol{q}}_{v,k}$ and $\hat{\boldsymbol{Q}}_{v,k}$ denote the estimated measurement noise mean and covariances, respectively; $\epsilon_k = \boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1} - q_{v,k}$ is the residual vector; $\delta_{k-1} = (1-b)/\left(1-b^{k+1}\right)$ and b represent the amnestic factor and the forgetting factor, respectively; $\boldsymbol{A}_k$ and $\boldsymbol{H}_k$ are the state transition matrix and measurement matrix, respectively; $\boldsymbol{K}_k$ and $\boldsymbol{P}_{k|k-1}$ are the Kalman gain and predicted error covariance matrix, respectively; and $\hat{\boldsymbol{z}}_{k|k}$, $\hat{\boldsymbol{x}}_{k|k-1}$, and $\hat{\boldsymbol{x}}_{k|k}$ are the predicted measurement, predicted state, and posterior state, respectively.

To address this challenge, the SHACKF integrates the CKF and the SHAKF into a unified framework. The update process for the SHACKF is given in the following steps:

$$\hat{\boldsymbol{q}}_{w,k} = (1-\delta_{k-1})\hat{\boldsymbol{q}}_{w,k-1} + \delta_{k-1}\left(\hat{\boldsymbol{x}}_{k|k} - \frac{1}{2n}\sum_{\mu=1}^{2n}\boldsymbol{Z}_{\mu,\,k|k-1}\right) \tag{22}$$

$$\hat{\boldsymbol{Q}}_{w,k} = (1-\delta_{k-1})\hat{\boldsymbol{Q}}_{w,k-1} + \delta_{k-1}\left(\boldsymbol{K}_k\epsilon_k\epsilon_k^T\boldsymbol{K}_k^T + \boldsymbol{P}_{k|k} - \left(\frac{1}{2n}\sum_{\mu=1}^{2n}\boldsymbol{X}_{\mu,\,k|k-1}\left(\boldsymbol{X}_{\mu,\,k|k-1}\right)^T - \hat{\boldsymbol{x}}_{k|k-1}\left(\hat{\boldsymbol{x}}_{k|k-1}\right)^T\right)\right) \tag{23}$$

$$\hat{\boldsymbol{q}}_{v,k} = (1-\delta_{k-1})\boldsymbol{q}_{v,k-1} + \delta_{k-1}\left(\hat{\boldsymbol{z}}_{k|k} - \frac{1}{2n}\sum_{\mu=1}^{2n}\boldsymbol{Z}_{\mu,\,k|k-1}\right) \tag{24}$$

$$\hat{\boldsymbol{Q}}_{v,k} = (1-\delta_{k-1})\boldsymbol{Q}_{v,k-1} + \delta_{k-1}\left(\epsilon_k\epsilon_k^T - \left(\frac{1}{2n}\sum_{\mu=1}^{2n}\boldsymbol{Z}_{\mu,\,k|k-1}\left(\boldsymbol{Z}_{\mu,\,k|k-1}\right)^T - \hat{\boldsymbol{z}}_{k|k-1}\left(\hat{\boldsymbol{z}}_{k|k-1}\right)^T\right)\right) \tag{25}$$

where $\boldsymbol{Z}_{\mu,\,k|k-1}$ and $\boldsymbol{X}_{\mu,\,k|k-1}$ indicate propagated cubature point sand cubature points, respectively [30].

To enhance the stability of the SHACKF, the innovation sequence ϵ_k and its theoretical statistical features are used to make judgments. If the filtering process is abnormal, the noise parameters are reinitialized [31]. The judgment is shown below:

$$\epsilon_k\epsilon_k^T > \gamma tr\left(E\left[\epsilon_k\epsilon_k^T\right]\right) = \gamma tr\left(\boldsymbol{P}_{zz,k|k-1}\right) \tag{26}$$

where γ is a coefficient, which is taken as 1 in this paper.

As mentioned in Section 2.1, the error of the EACC system is assumed to obey a Gaussian process. The results are shown in Figure 4. Figure 4a–d show the autocorrelation of the normalized innovation sequence in the simulation [23]. The yellow dashed line represents the 95% confidence interval. All four components fall within the confidence interval of approximately 95%, which indicates that they are uncorrelated. Figure 4e–h represent the distribution histograms of the innovation sequence after the normalization of the four state components of the EACC system. The green lines represent the standard normal distribution. It can be seen that the four components are almost identical to the normal distribution. Therefore, there is sufficient reason to believe that the process noise obeys a Gaussian process.

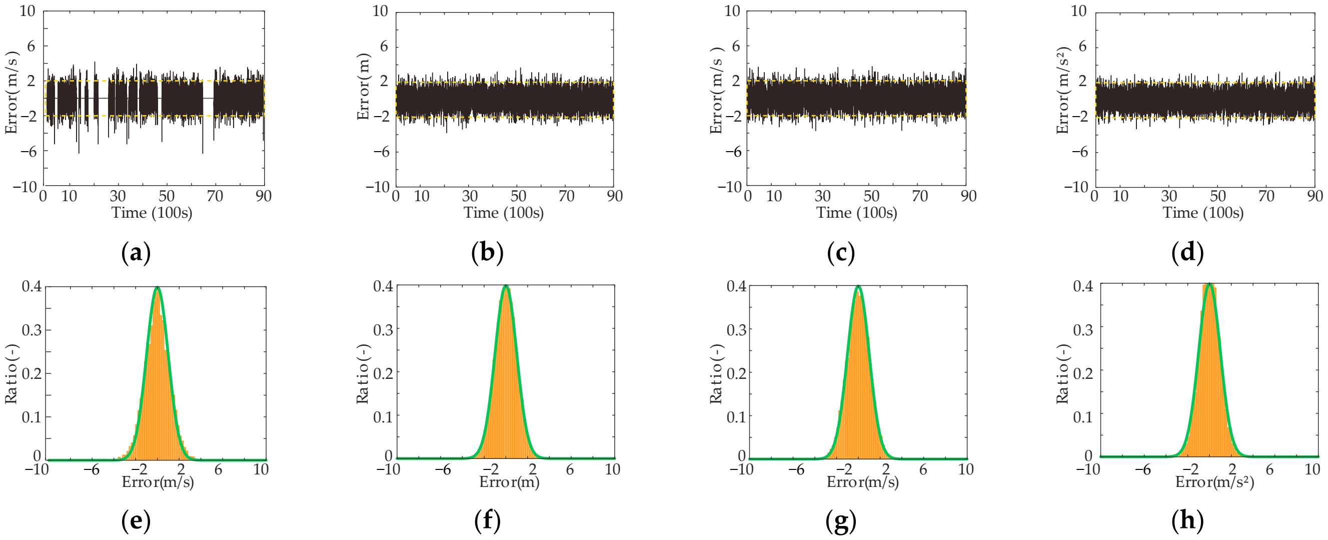


Figure 4. (**a**–**d**) The innovation sequences after normalization of the state of the system; (**e**–**h**) The distribution histograms.

However, there is an exception to the experiment. When the front vehicle stops, the vehicle velocity prediction at the next moment is also zero, which is almost correct and meaningless. This situation is reflected in the noiseless region in Figure 4a. Therefore, this situation is not considered in this paper.

3.2. MPC Controller Algorithm

In this section, the design of the control system is presented based on the previous discussion. The system framework is shown in Figure 5,and it consists of three main components: the front vehicle velocity prediction model, the filter, and the controller.

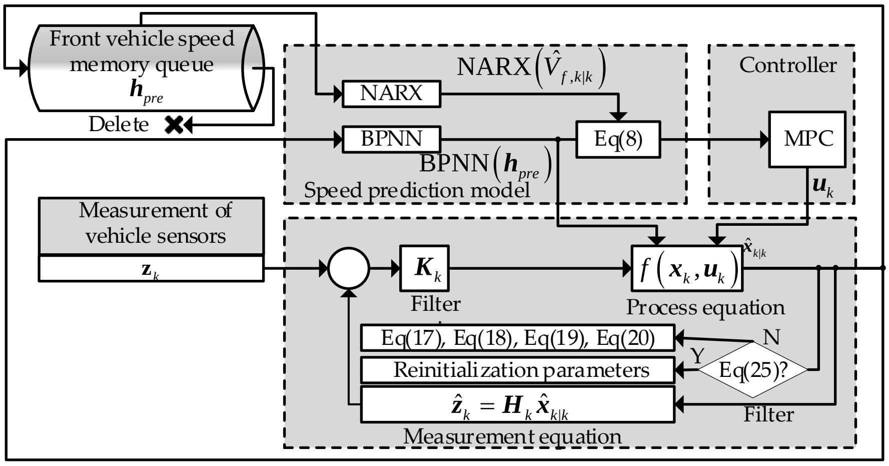

Front vehicle speed memory queue
h_{pre}
Delete
NARX$\left(\hat{V}_{f,k|k}\right)$
NARX
BPNN
BPNN$\left(h_{pre}\right)$
Speed prediction model
Eq(8)
Controller
MPC
u_k
Measurement of vehicle sensors
z_k
K_k
Filter
$f\left(x_k, u_k\right)$
$\hat{x}_{k|k}$
Process equation
Eq(17), Eq(18), Eq(19), Eq(20)
N
Y
Eq(25)?
Reinitialization parameters
$\hat{z}_k = H_k \hat{x}_{k|k}$
Filter
Measurement equation

Figure 5. Schematic diagram of the algorithm proposed in this article.

The front vehicle velocity prediction model, described in Section 2.1, uses a queue to store historical data and follows the FIFO (first-in, first-out) principle. When a new value arrives, it deletes the earliest value in the queue. This queue is initialized to zero.

The filter is described in Section 3.1. To begin, the system state is estimated by the CKF algorithm. Then, it determines whether the noise parameters need to be reinitialized according to Equation (26). If not, it updates them according to Equations (22)–(25). The values obtained after filtering are provided to the controller and the prediction model.

The controller uses an MPC algorithm and is designed based on the model in Section 2. It solves the optimal control input according to the current state estimation and the desired state. Then, the control input is applied to the system to achieve EACC. The slack variables are introduced into the MPC to reduce non-solutions, and the NMPC toolbox provided by MATLAB has been used to solve the nonlinear model predictive control problem. The loss function and constraints of the MPC problem are given by:

$$\min_{|u_{k+l}|} \varpi_1 \sum_{l=1}^{p_p} T_s P_m\left(u_{k+l}, \hat{V}_{h,k+l|k}\right) + \varpi_2 \sum_{l=1}^{p_p} \left|\hat{V}_{h,k+l|k} - \hat{V}_{f,k+l|k}\right| + \varpi_3 \sum_{l=1}^{p_p} \left|\hat{a}_{k+l|k}\right| + \varpi_4 \sum_{l=1}^{p_p} (|\boldsymbol{\varepsilon}_d| + |\boldsymbol{\varepsilon}_V|) \tag{27}$$

$$\hat{\boldsymbol{x}}_{k+l+1|k} = f\left(\hat{\boldsymbol{x}}_{k+l|k}, \boldsymbol{u}_{k+l}\right) \tag{27a}$$

$$\underline{u} \leq u_{k+l} \leq \overline{u} \tag{27b}$$

$$\underline{d} + \boldsymbol{\varepsilon}_d < \hat{d}_{k+l|k} \leq \overline{d} - \boldsymbol{\varepsilon}_d \tag{27c}$$

$$0 \leq \hat{V}_{h,k+l|k} \leq \overline{V}_h + \boldsymbol{\varepsilon}_V \tag{27d}$$

$$0 \leq \boldsymbol{\varepsilon}_d \leq \overline{\boldsymbol{\varepsilon}}_d \tag{27e}$$

$$0 \leq \boldsymbol{\varepsilon}_V \leq \overline{\boldsymbol{\varepsilon}}_V \tag{27f}$$

where p_p is the MPC prediction horizon; $\hat{V}_{h,k+l|k}$ denotes the prediction of the speed of the preceding vehicle for $k+l$ time under the premise of k time, and the same for the other similar subscripts; $\boldsymbol{\varepsilon}_d$ and $\boldsymbol{\varepsilon}_V$ represent the slack variables of relative distance and the host vehicle's velocity, respectively; $\overline{d}$ and $\underline{d}$ represent the upper and lower bounds of the relative distance, respectively; $\overline{u}$ and $\underline{u}$ represent the upper and lower bounds of the control input, respectively; and $\overline{V}_h$ represents the upper bounds of the host vehicle's velocity.

The four terms in Equation (27) consider energy consumption, the distance difference between the two vehicles, the velocity difference between the two vehicles, the host vehicle's acceleration magnitude, and the slack variable magnitude. Equation (27a) is the process equation. The control input range is restricted by Equation (27b). The relative distance is restricted and maintained between cars by Equation (27c). The maximum vehicle velocity is limited by Equation (27d). The slack variable range is limited by Equations (27e) and (27f).

Although slack variables are introduced, there may be situations where no solution can be found. In such cases, the algorithm first tries to find a suboptimal solution. If that fails, the algorithm attempts to relax the energy, distance, or velocity constraint conditions. If that still fails, the algorithm reinitializes the MPC. This is because there may be an inconsistency between the control input and Equation (11) when the front vehicle completely stops. The reinitialization clears the previous input data and solves the problem again.

In summary, the proposed MPC controller algorithm addresses the challenge of EACC under noise uncertainty, and it effectively integrates the front vehicle velocity prediction model and the filter to achieve tracking control. Additionally, the algorithm can also be combined with studies [7,15].

4. Performance Analysis

The performance of the algorithm is analyzed in detail in this section using CarSim and Simulink. To solve the MPC problem, the NLMPC toolbox is utilized, and all simulations are conducted on a personal computer with an i5-8600 processor.

The host vehicle simulation model is based on the Tesla Model 3 Rear-Wheel Drive series with a single rear-mounted IPM-SynRM electric motor. Table 1 displays the simulation parameters for the host vehicle and environment. The "Model Value" column represents values used in the calculation, whereas the "Simulation Value" column denotes the values configured in the software during the simulation. This approach is reasonable since the electronic control unit cannot accurately measure relevant environmental parameters in real time. Table 2 shows the calculation parameters for the MPC.

Table 1. Vehicle and environmental parameters.

Parameter	Symbol	Model Value	Simulation Value	Unit
Maximum torque for energy output	T_m^+	340		Nm
Maximum torque for energy recovery input	T_m^-	135		Nm
Tire specification	-	235/55 R18	225/55 R18	-
Air resistance coefficient	C_w	0.23		-
Transmission ratio	i	7.2288		-
Vehicle and passenger mass	m	1440	1580	kg
Windward area	A_{window}	0.7035		m^2
Rolling resistance coefficient	-	0.9	0.75	-

Table 2. The parameters of the MPC algorithm.

Parameter	Symbol	Value		Unit
Historical horizon	p_h	10		-
Prediction horizon	p_p	10		-
Relative distance, upper bound	$\overline{d}$	$25 + 0.7V_{f,k}$		m
Relative distance, lower bound	$\underline{d}$	10		m
Sampling time	T_s	0.5		s
Host vehicle's velocity, upper bound	$\overline{V}_h$	$V_{f,k} + 5$		m/s
Upper bound on the relaxation variable of the distance	ε_d	0.5		m
Upper bound on the relaxation variable of the velocity	ε_V	0.3		m/s
Weights	ϖ_1 0.1	ϖ_2 1	ϖ_3 0.6	ϖ_4 0.01

To predict the front vehicle's velocity, two types of neural networks, the BPNN and the NARX, are trained. The training data set is collected by Beamng.tech [34], a free simulation software for the academic community. The data set consisted of seventeen scenarios with various vehicles, weather, congestion, and road conditions, each with a driving time ranging from 75 s to 5 min. Beamng.tech has been used in simulations in previous studies [34,35].

In order to conduct a comprehensive evaluation of the algorithm's performance, comparative experiments are performed between the algorithms proposed in [7,15] and the proposed algorithm under the same conditions. The algorithm is comprehensively tested for energy consumption, tracking, velocity following, and filtering using two types of cycles: the NEDC and the CLTC.

The NEDC is a driving cycle with four urban segments and one suburban segment, lasting 1180 s and covering a distance of 11.022 km. The average velocity is 33.6 km/h. The CLTC [36], short for China Light Vehicle Test Cycle, is composed of urban, suburban, and expressway segments with a total duration of 1800 s and a total distance of 14.48 km, with more random velocity variation than the NEDC.

4.1. Analysis of Energy Consumption

In this section, the energy consumption of the host vehicle is analyzed. The stochastic model predictive control (SMPC) method proposed by Moser, D. et al. [7] and the MPC+ method proposed by Zhou, H. et al. [15] are compared with the proposed algorithm under the same conditions. The host vehicle has a total battery capacity of 60 kWh, with 36 kWh remaining at the initial moment. Figure 6a,b show the energy consumption and recovery under the NEDC and CLTC driving cycles, respectively.

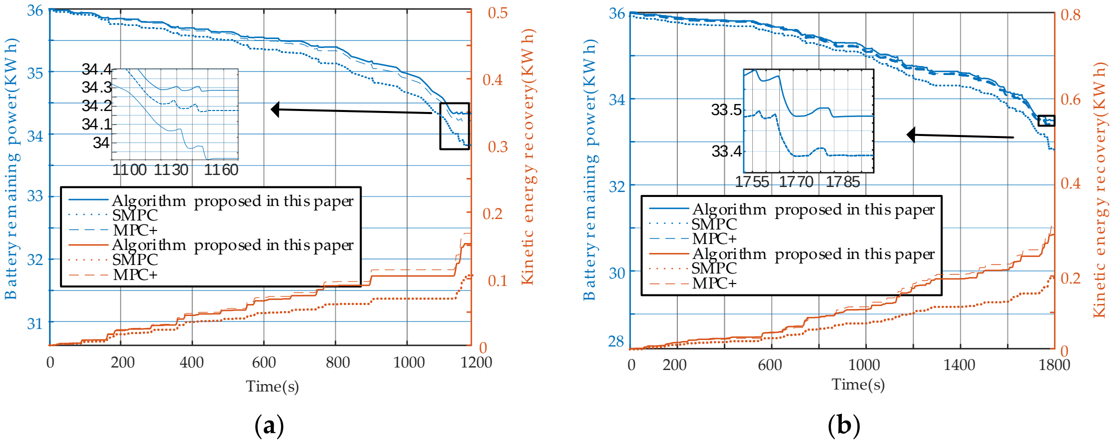


Figure 6. (**a**) Electricity consumption comparison under NEDC conditions; (**b**) Electricity consumption comparison under CLTC conditions.

The energy consumption gap between the proposed algorithm and the algorithms in [7,15] is mainly due to the inaccurate estimation. It is worth mentioning that the KERS in this algorithm recovers less energy than that of the algorithm proposed in [15], as the inaccurate prediction of the front vehicle's velocity leads to frequent deceleration. Since the efficiency of the vehicle is less than 100%, accelerating and then decelerating to the same speed leads to a waste of energy. The study in [7] can be applied in the context of CACC or when communication is absent. However, this paper does not take into account the kinetic energy recovery system, which results in significantly lower energy recovery compared to the algorithms proposed in this paper and [15]. Additionally, higher energy consumption is observed due to the inaccuracies in the estimation.

The PID algorithm is used to simulate the vehicle's driving under NEDC and CLTC conditions. The simulation results are considered the standard energy consumption, and the results are presented in Table 3.

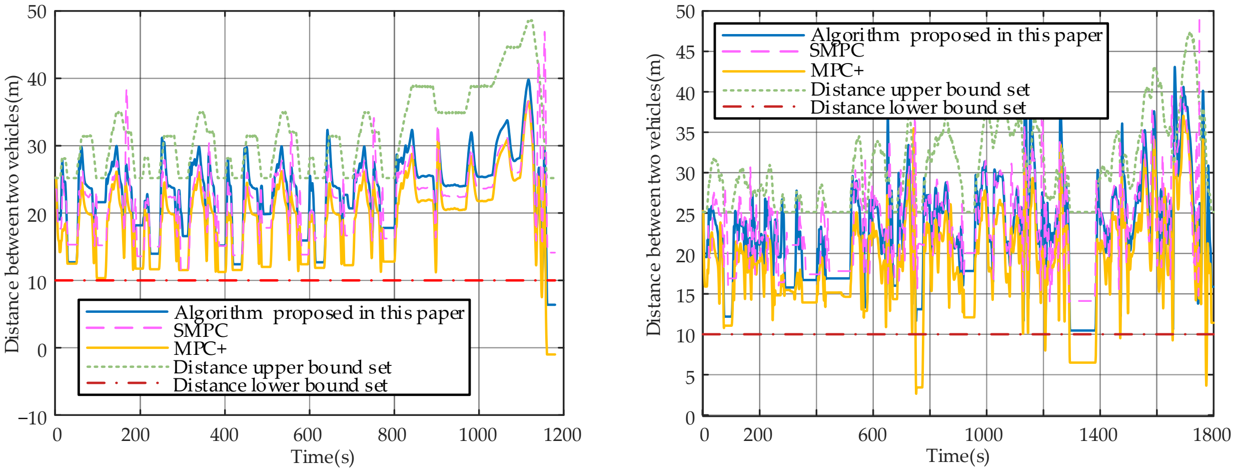
Table 3. Comparison of energy consumption and recovery.

Circulation	Energy Consumption in This Paper	Energy Consumption in [15]	Energy Consumption in [7]	Energy Recovery in This Paper	Energy Recovery in [15]	Energy Recovery in [7]	Standard
Unit	kWh	kWh	kWh	kWh	kWh	kWh	kWh
NEDC	1.6728	1.786	2.0923	0.1526	0.1682	0.1035	1.926
CLTC	2.505	2.6000	3.0070	0.2862	0.3087	0.1921	2.154

The CLTC driving cycle has more complex environmental conditions than the NEDC, necessitating a more accurate front vehicle velocity prediction model. Inaccurate predictions can degrade overall performance. Noise filtering is crucial for the EACC system, as an accurate prediction of the dynamic system is a prerequisite for stable and efficient MPC operation [37]. All EACC algorithms consume more energy than the standard under the CLTC driving cycle due to the more complex and error-prone velocity prediction in the tracking case. However, by filtering and improving the EACC system, more energy can be saved by the algorithm proposed in this paper. Compared to the algorithm proposed in [15], the algorithm proposed in this paper can achieve an energy saving of 0.095 kWh per single CLTC. With a 60 kWh battery, the algorithm in [15] can run 347 km, the algorithm in [7] can run 289 km, and the algorithm in this paper can run 334 km. Energy can therefore be saved by 0.006 kWh/km and 0.03 kWh/km relative to the algorithms proposed in [7,15], respectively.

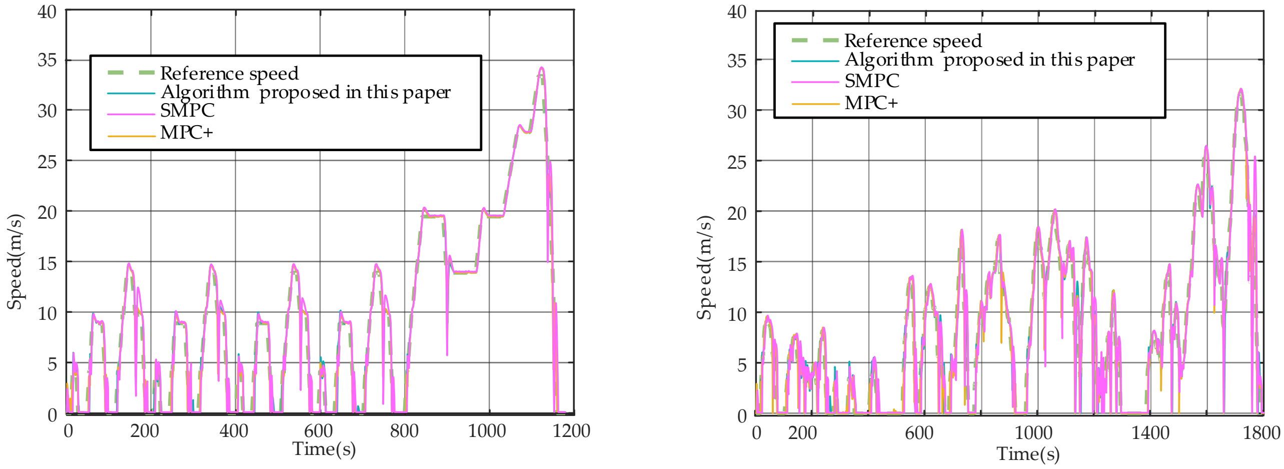
4.2. Analysis of Tracking Performance

The tracking performance of the vehicle is analyzed in this section. Figure 7a,b illustrate the inter-vehicle distance tracking under the NEDC and CLTC driving cycles, respectively. The green line represents the upper distance boundary, while the red line represents the lower distance boundary. Similarly, Figure 8a,b show the velocity tracking curves.



(**a**) (**b**)

Figure 7. (**a**) Inter-vehicle distance tracking curve under NEDC conditions; (**b**) Inter-vehicle distance tracking curve under CLTC conditions.



(**a**) (**b**)

Figure 8. (**a**) Speed-tracking curve under NEDC conditions; (**b**) Speed-tracking curve under CLTC conditions.

Constraint violations are observed in Figure 7a,b, particularly during the leading vehicle's acceleration and deceleration phases, resulting from the inaccurate prediction of the front vehicle's velocity. A method is proposed for a condition that failed to solve the MPC equation in this paper, allowing the vehicle to track or stop within acceptable limits even in violation of constraints. In contrast, the algorithms proposed in [7,15] do not consider this problem and noise, which led to over-near distance and even rear-end collisions at the last moment (1160 s) during the NEDC simulation.

Figure 6a,b show that all three algorithms predicted that the velocity continued to increase after almost all end-of-ascent time moments (e.g., 143 s, 338 s, and 533 s under

NEDC conditions; 723 s, 998 s, and 1057sunderCLTC conditions). After a change occurred, they rapidly decelerated to follow the front vehicle's velocity. Compared with [15], a variable distance limit is introduced, while a fixed one is used in [15]. The relative distances are similar at relatively low velocities, but the developed algorithm has a farther distance than [15] at faster velocities. Most of the relative distance differences between the two algorithms are within 5 m, which is acceptable. However, fixed limits may have led to prediction errors with noise, resulting in dangerous distances (e.g., 751 s, 1293 s, and 1774 s in the CLTC; 1160sin the NEDC, as mentioned earlier). The dangerous distance situations did not occur in [7], which can be attributed to the fact that the algorithm in [7] has no limit on the KERS and applies significant braking when approaching dangerous distances, leading to energy consumption.

By analyzing the tracking capabilities of all algorithms, it is evident that the proposed method minimizes the occurrence of accidents and enhances system reliability. However, this may introduce a computational burden. Each calculation time is recorded, with an average calculation time of 0.1141 s. This is still sufficient for current mainstream vehicle chips.

4.3. Analysis of Modeling and Filtering

In this section, the model developed in Section 2 and the SHACKF proposed in Section 3.1 will be analyzed.

Figure 7a shows the estimation of the velocity of the host vehicle by filtering and modeling algorithms. As can be seen from the Figure 9, several filtering algorithms are effective, as almost all are the same as the true value. The modeling algorithm does not use the observations to adjust the estimates, and this curve is always around the true value, which also proves that the model proposed in Section 2 is valid.

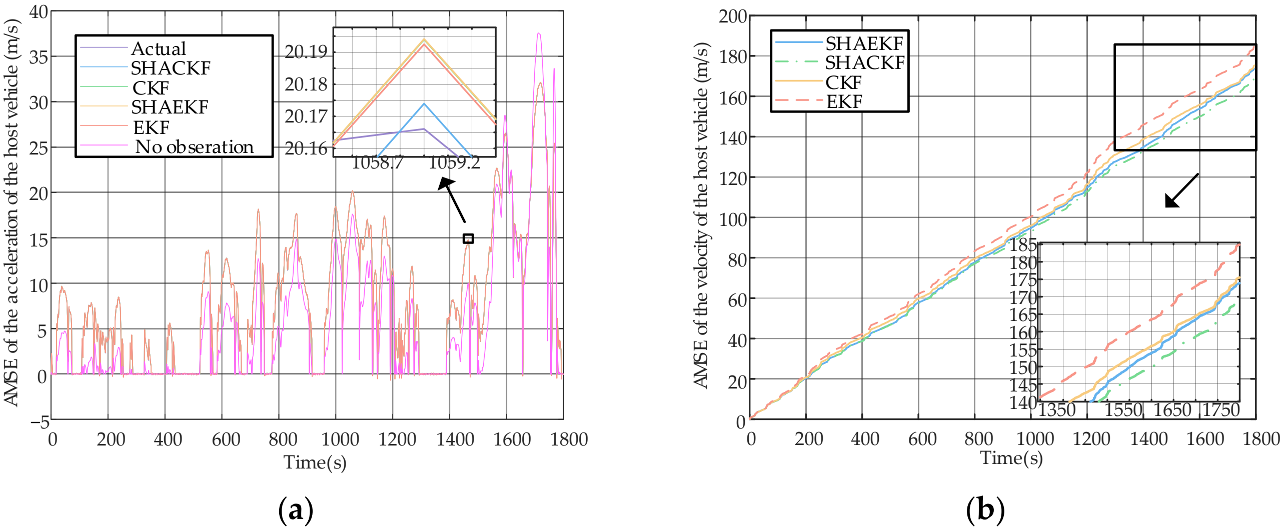


Figure 9. (**a**) Filtering and modeling algorithms for the host vehicle speed estimation curve; (**b**)AMSE of the host vehicle speed estimation curve.

Figure 7b shows the accumulative mean square error (AMSE) of the SHAKF, SHACKF, CKF, and EKF, where the AMSE is as follows:

$$\mathrm{AMSE}(k) = \sum_{t=0}^{k} \left\| \hat{x}_{t|t} - x_t \right\|_2^2 \tag{28}$$

In the figure, we can see that the SHACK algorithm has the best filtering performance, the CKF and the SHAKF are almost the same, and the EKF algorithm is the worst. It is understandable that the CKF uses the nonlinear property of the model, the SHAKF compensates for the error, and the SHACKF combines the advantages of these two algorithms.

5. Conclusions

In this paper, the design of anEACC system considering vehicle process and measurement noise is studied, focusing on how to minimize the impact of noise on the EACC system and save more energy. This study addresses non-zero mean noise in nonlinear vehicle systems by employing the SHACKF algorithm, which integrates the CKF and the SHAKF algorithms. To boost multi-step prediction accuracy, the vehicle speed prediction model is augmented with a BPNN. The issue of frequent acceleration and deceleration is tackled by considering vehicle kinetic energy recovery systems and interpolating motor efficiency. Furthermore, slack variables are employed to manage the infeasibility of MPC.

The simulation results provided valuable insights into the performance of our proposed algorithm. In terms of energy savings, the algorithm demonstrated significant improvements compared to previous methods, with numerical results showing an energy saving of 0.006–0.03 kWh/km. The proposed algorithm also outperformed previous algorithms in terms of stability in distance tracking, maintaining a consistent and appropriate distance from the leading vehicle. In velocity tracking, minor fluctuations were observed only during the final stages of acceleration and deceleration, while the tracking remained stable throughout other periods. It is worth mentioning that this algorithm is versatile and can be integrated with various current MPC-based EACC studies.

Future research may explore incorporating this algorithm into CACC studies by using information fusion algorithms to enhance traffic efficiency. Additionally, although the SHACKF algorithm is effective, it has the drawback of estimating covariance and mean inaccurately. To address this issue, more precise filters, such as robust adaptive filters and H∞ filters, could be employed to eliminate noise.

Author Contributions: Conceptualization and methodology: Y.L. and G.H.; software and validation: Y.L.; formal analysis and writing—original draft preparation: G.H.; writing—review and editing supervision: Y.L. and G.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by the National Natural Science Foundation (NNSF) of China under Grant No. 61503127, by the University Basic Research Foundation of Heilongjiang Province (2021-KYYWF-0027), by the Outstanding Youth Foundation of Heilongjiang University, and by the Key Laboratory of Information Fusion Estimation and Detection, Heilongjiang Province.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the Key Laboratory of Information Fusion Estimation and Detection, Heilongjiang Province, for providing the conditions used in this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. He, Q.; Yang, Y.; Luo, C.; Zhai, J.; Luo, R.; Fu, C. Energy recovery strategy optimization of dual-motor drive electric vehicle based on braking safety and efficient recovery. *Energy* **2022**, *248*, 123543. [CrossRef]
2. Yu, Y.; Zhou, D.; Zha, D.; Wang, Q.; Zhu, Q. Optimal production and pricing strategies in auto supply chain when dual credit policy is substituted for subsidy policy. *Energy* **2021**, *226*, 120369. [CrossRef]
3. Xie, Y.; Li, Y.; Zhao, Z.; Dong, H.; Wang, S.; Liu, J.; Guan, J.; Duan, X. Microsimulation of electric vehicle energy consumption and driving range. *Appl. Energy* **2020**, *267*, 115081. [CrossRef]
4. Yang, Y.; Zhang, Y.; Tian, J.; Li, T. Adaptive real-time optimal energy management strategy for extender range electric vehicle. *Energy* **2020**, *197*, 117237. [CrossRef]
5. Zhang, H.; Chen, D.; Zhang, H.; Liu, Y. Research on the influence factors of brake regenerative energy of pure electric vehicles based on the CLTC. *Energy Rep.* **2022**, *8*, 85–93. [CrossRef]

6. Hidalgo, C.; Lattarulo, R.; Flores, C.; Perez Rastelli, J. Platoon Merging Approach Based on Hybrid Trajectory Planning and CACC Strategies. *Sensors* **2021**, *21*, 2626. [CrossRef] [PubMed]
7. Moser, D.; Schmied, R.; Waschl, H.; del Re, L. Flexible Spacing Adaptive Cruise Control Using Stochastic Model Predictive Control. *IEEE Trans. Control Syst. Technol.* **2018**, *26*, 114–127. [CrossRef]
8. Ren, P.; Jiang, H.; Xu, X. Research on a Cooperative Adaptive Cruise Control (CACC) Algorithm Based on Frenet Frame with Lateral and Longitudinal Directions. *Sensors* **2023**, *23*, 1888. [CrossRef]
9. Kim, H.; Min, K.; Sunwoo, M. Driver Characteristics Oriented Autonomous Longitudinal Driving System in Car-Following Situation. *Sensors* **2020**, *20*, 6376. [CrossRef]
10. Zhu, B.; Jiang, Y.; Zhao, J.; He, R.; Bian, N.; Deng, W. Typical-driving-style-oriented Personalized Adaptive Cruise Control design based on human driving data. *Transp. Res. Part C Emerg. Technol.* **2019**, *100*, 274–288. [CrossRef]
11. Jiang, J.; Ding, F.; Zhou, Y.; Wu, J.; Tan, H. A Personalized Human Drivers' Risk Sensitive Characteristics Depicting Stochastic Optimal Control Algorithm for Adaptive Cruise Control. *IEEE Access* **2020**, *8*, 145056–145066. [CrossRef]
12. Liu, J.; Dong, S.; Liu, Q.; Gao, B.; Kawabe, T.; Chen, H. Fast warm-start of F-MPC strategy for automotive cruise control with mode switching. *Control Eng. Pract.* **2022**, *128*, 105344. [CrossRef]
13. Zhou, X.; Sun, F.; Zhang, C.; Sun, C. Stochastically predictive co-optimization of the speed planning and powertrain controls for electric vehicles driving in random traffic environment safely and efficiently. *J. Power Sources* **2022**, *528*, 231200. [CrossRef]
14. Weißmann, A.; Görges, D.; Lin, X. Energy-optimal adaptive cruise control combining model predictive control and dynamic programming. *Control Eng. Pract.* **2018**, *72*, 125–137. [CrossRef]
15. Pan, C.; Huang, A.; Wang, J.; Chen, L.; Liang, J.; Zhou, W.; Wang, L.; Yang, J. Energy-optimal adaptive cruise control strategy for electric vehicles based on model predictive control. *Energy* **2022**, *241*, 122793. [CrossRef]
16. Li, Y.; Yin, G.; Zhuang, W.; Zhang, N.; Wang, J.; Geng, K. Compensating Delays and Noises in Motion Control of Autonomous Electric Vehicles by Using Deep Learning and Unscented Kalman Predictor. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *50*, 4326–4338. [CrossRef]
17. Zhou, H.; Zhou, A.; Li, T.; Chen, D.; Peeta, S.; Laval, J. Congestion-mitigating MPC design for adaptive cruise control based on Newell's car following model: History outperforms prediction. *Transp. Res. Part C Emerg. Technol.* **2022**, *142*, 103801. [CrossRef]
18. Alonso, L.; Milanes, V.; Torre-Ferrero, C.; Godoy, J.; Oria, J.P.; de Pedro, T. Ultrasonic sensors in urban traffic driving-aid systems. *Sensors* **2011**, *11*, 661–673. [CrossRef]
19. Aubeck, F.; Lenz, M.; Mertes, S.; Zylka, K.; Pischinger, S. Generic stochastic particle filter algorithm for predictive energy optimization of a Plug-in Hybrid Electric Vehicle extended by a battery temperature control and implemented on a Hardware-in-the-Loop system. *Control Eng. Pract.* **2022**, *120*, 105002. [CrossRef]
20. Yan, D.; Zhang, W.; Chen, H.; Shi, J. Robust control strategy for multi-UAVs system using MPC combined with Kalman-consensus filter and disturbance observer. *ISA Trans.* **2023**, *135*, 35–51. [CrossRef]
21. Shakouri, P.; Ordys, A. Nonlinear Model Predictive Control approach in design of Adaptive Cruise Control with automated switching to cruise control. *Control Eng. Pract.* **2014**, *26*, 160–177. [CrossRef]
22. Zülke, A.; Li, Y.; Keil, P.; Burrell, R.; Belaisch, S.; Nagarathinam, M.; Mercer, M.P.; Hoster, H.E. High-Energy Nickel-Cobalt-Aluminium Oxide (NCA) Cells on Idle: Anode- versus Cathode-Driven Side Reactions. *Batter. Supercaps* **2021**, *4*, 934–947. [CrossRef]
23. Lio, W.H.; Li, A.; Meng, F. Real-time rotor effective wind speed estimation using Gaussian process regression and Kalman filtering. *Renew. Energy* **2021**, *169*, 670–686. [CrossRef]
24. Bar-Shalom, Y.; Kirubarajan, T.; Li, X.R. *Estimation with Applications to Tracking and Navigation*; Estimation with Applications to Tracking and Navigation: New York, NY, USA, 2001.
25. Wan, R.; Song, Y.; Mu, T.; Wang, Z. Moving Target Detection Using the 2D-FFT Algorithm for Automotive FMCW Ra-dars. In Proceedings of the 2019 International Conference on Communications, Information System and Computer Engineering (CISCE), Haikou, China, 5–7 July 2019.
26. Jang, J. Pavement slipperiness detection using wheel speed and acceleration sensor data. *Transp. Res. Interdiscip. Perspect.* **2021**, *11*, 100431. [CrossRef]
27. Ivanov, S.I.; Kuptsov, V.D.; Fedotov, A.A. The signal processing algorithm of automotive FMCW radars with an extended range of speed estimation. *J. Phys. Conf. Ser.* **2019**, *1236*, 012081. [CrossRef]
28. Drummond, O.E.; Gale, N.; Hong, L.; Roy, A. Design and analysis of an FMCW radar system for vehicle tracking. In *Signal and Data Processing of Small Targets 2011*; SPIE: Bellingham, WA, USA, 2011.
29. Ji, Y.; Zhang, J.; He, C.; Ma, R.; Hou, X.; Hu, H. Constraint performance pressure tracking control with asymmetric continuous friction compensation for booster based brake-by-wire system. *Mech. Syst. Signal Process.* **2022**, *174*, 109083. [CrossRef]
30. Arasaratnam, I.; Haykin, S. Cubature Kalman Filters. *IEEE Trans. Autom. Control* **2009**, *54*, 1254–1269. [CrossRef]
31. Fan, Y.; Qiao, S.; Wang, G.; Chen, S.; Zhang, H. A modified adaptive Kalman filtering method for maneuvering target tracking of unmanned surface vehicles. *Ocean. Eng.* **2022**, *266*, 112890. [CrossRef]
32. Ding, W.; Wang, J.; Rizos, C.; Kinlyside, D. Improving Adaptive Kalman Estimation in GPS/INS Integration. *J. Navig.* **2007**, *60*, 517–529. [CrossRef]
33. Wang, D.; Yang, Y.; Gu, T. A hierarchical adaptive extended Kalman filter algorithm for lithium-ion battery state of charge estimation. *J. Energy Storage* **2023**, *62*, 106831. [CrossRef]

34. Humeniuk, D.; Khomh, F.; Antoniol, G. A search-based framework for automatic generation of testing environments for cyber–physical systems. *Inf. Softw. Technol.* **2022**, *149*, 106936. [CrossRef]
35. Riccio, V.; Tonella, P. Model-based exploration of the frontier of behaviours for deep learning system testing. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, 8–13 November 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 876–888.
36. State Administration for Market Regulation; Standardization Administration of China. *China Automotive Test Cycle Part 1: Light-Duty Vehicles: GB/T 38146.1-2019*; Standards Press of China: Beijing, China, 2019.
37. Sun, X.; Zhou, P.; Ding, J.; Qiao, J. Enhanced MPC based on unknown state estimation and control compensation. *J. Process Control* **2023**, *121*, 60–72. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.