InPrime: Indonesian Journal of Pure and Applied Mathematics
Vol. 5, No. 1 (2023), pp. 47 – 59, doi: 10.15408/inprime.v5i1.28313
p-ISSN 2686-5335, e-ISSN: 2716-2478

Moving Horizon State Estimation for Linear System with Application to Autonomous Vehicle

Heri Purnawan[1]*, Ulul Ilmi[1], Rifky Aisyatul Faroh[1], Ahmad Bustanul Ali Ar Rizqi[1] and Fitroh Resmi[2]
[1]Department of Electrical Engineering, Universitas Islam Lamongan, Lamongan, Indonesia
[2]Department of Mathematics, Universitas Billfath, Lamongan, Indonesia
Email: *heripurnawan@unisla.ac.id

Abstract

This paper proposes moving horizon estimation (MHE) to estimate the state variables of autonomous vehicle linear systems under measurement noises. To solve the MHE optimization problem, quadratic programming is employed. The steering angle, yaw angle, and global position constraints of an autonomous vehicle are considered in the estimation design. According to the simulation results, it can be observed that although the longer MHE step can give better results compared to the shorter MHE step, the difference in the MHE step only slightly affects the estimated results. However, the longer MHE step can increase the computational time. Additionally, the proposed MHE scheme is compared to the Kalman filter (KF) estimator. Based on the obtained results, the KF gives a better estimation than the MHE, but this notion must be verified for other case studies.
Keywords: autonomous vehicle; Kalman filter; linear system; MHE; quadratic programming.

Abstrak

Paper ini mengusulkan moving horizon estimation *(MHE) untuk mengestimasi variabel keadaan sistem linier kendaraan otonom karena pengaruh noise pengukuran. Untuk menyelesaikan masalah optimasi MHE, digunakan pemrograman kuadratik. Kendala sudut kemudi, sudut yaw dan posisi global dari kendaraan otonom dipertimbangkan dalam desain estimasi. Dari hasil simulasi dapat diketahui bahwa meskipun langkah MHE yang lebih panjang dapat memberikan hasil yang lebih baik dibandingkan dengan langkah MHE yang lebih pendek, perbedaan langkah MHE hanya sedikit mempengaruhi hasil estimasi. Namun, langkah MHE yang semakin panjang dapat meningkatkan waktu komputasi. Selain itu, skema MHE yang diusulkan dibandingkan dengan estimator* Kalman filter *(KF). Berdasarkan hasil yang diperoleh, KF memberikan estimasi yang lebih baik daripada MHE, tetapi gagasan ini harus diverifikasi untuk studi kasus lainnya.*
Kata Kunci: *kendaraan otonom;* Kalman filter*; sistem linier; MHE; pemrograman kuadratik.*

2020MSC: 62P35, 90C20

1. INTRODUCTION

With the sophistication of technology in the information and communication field, the development of autonomous and intelligent vehicle systems is increasingly being carried out. The purpose of developing these vehicles is to make it easier for humans to drive and to increase traffic efficiency as well [1]. Autonomous vehicles are intelligent vehicles that can steer themselves from one location to another without a driver. To develop an autonomous vehicle, several technical aspects in engineering are needed to support the concept, including Navigation, Guidance, and Control (NGC).

* Corresponding author
Submitted September 29th, 2022, Revised May 25th, 2023, Accepted for publication May 29th, 2023.

To design a feedback control system, complete knowledge of the assumptions of a given system is essential. Measuring all system states and parameters may be impossible due to economic reasons [2]. In addition, the measurement results from a sensor can also be affected by error or noise, so the resulting output can be inaccurate. To handle this issue, it is possible to estimate the state variables or parameters. Various estimation methods can be used, including Kalman Filter (KF) and its modification. Generally, KF is designed for linear systems, while its modifications, such as Extended Kalman Filter (EKF), Ensemble Kalman Filter (EnKF), and Unscented Kalman Filter (UKF), are for nonlinear systems [3]–[6]. If the estimation problem does not consider the state and input constraints and the error distribution of the estimated state variables is Gaussian, KF and its modifications can give optimal estimation [7]. However, the limitations of the system should be considered when we design an estimation method to estimate the state variables or parameters. The estimation method that can be utilized to overcome this problem is Moving Horizon Estimation (MHE).

MHE is an optimization-based estimation method under system constraints. MHE uses all past measurements within the estimation horizon to minimize the deviation of the model subject to the state and input constraints via online optimization [8]. Noted that to solve the MHE optimization problem by converting it into nonlinear programming at each time, the computational time burden is highlighted in the literature. Due to this drawback, MHE does not have the feasibility to be implemented in real problems [9]. Through algorithm development, the solution to this problem can be addressed rapidly. One of the software packages that can be used to able to apply this estimation method is the ACADO toolkit. ACADO toolkit is an open-source efficient package that can be implemented as standalone C++ code and a user-friendly MATLAB interface [10]. Recently, the study about MHE for an autonomous vehicle can be found in [11]–[13]. Furthermore, MHE has been applied to several unmanned vehicles and combined with the Model Predictive Control (MPC) control method [14]–[16]. The main contribution of this paper is how to implement MHE in a linear system by utilizing the quadratic programming method to obtain the solution to the optimization problem. Additionally, the estimation results of MHE are compared to the Kalman filter to ensure the obtained solutions are according to the transformation process carried out.

The outline of this article is formulated as follows. In Section 2, the method section describes the general formulation of the proposed estimation method, namely MHE, and its transformation into quadratic programming formula. Additionally, the procedure of the MHE optimization problem is also presented in Section 2. Then, the results and discussions are presented in Section 3. The last section of this paper presents the conclusion and suggestions.

2. METHODS

This section contains the optimization-based estimation method, namely moving horizon estimation and its transformation into quadratic programming.

2.1 Moving Horizon Estimation

MHE is an estimation method utilizing optimization [17] with a series of measurements observed over time containing noises and other inaccuracies to produce the estimation of unknown state variables and parameters. Different from the deterministic approaches to finding solutions, MHE is based on an iterative approach that relies on linear, quadratic, or nonlinear programming [18]. MHE is a multivariable estimation method that uses (1) a dynamical model of a process, (2) previous measurement values, and (3) objective function along the estimation horizon to calculate optimal values of state variables and parameters. MHE can overcome physical limitations that cannot be ignored in a system, namely by defining those constraints in optimization problems [19].

A linear system and measurement equation in discrete time used in this study are defined as follows:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_d \boldsymbol{x}_k + \boldsymbol{B}_d \boldsymbol{u}_k + \boldsymbol{w}_k, \tag{1}$$

$$\boldsymbol{z}_k = \widetilde{\boldsymbol{H}} \boldsymbol{x}_k + \boldsymbol{v}_k, \tag{2}$$

where $\boldsymbol{x}_k, \boldsymbol{x}_{k+1} \in \mathbb{R}^{n_x}$ are the state variables at time-k and $k+1$, the state and input matrices in discrete time are denoted as $\boldsymbol{A}_d \in \mathbb{R}^{n_x \times n_x}$ and $\boldsymbol{B}_d \in \mathbb{R}^{n_x \times n_u}$, respectively, $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ is the manipulated or input variables at time-k, $\boldsymbol{w}_k$, and $\boldsymbol{v}_k$ are the additive Gaussian noise with zero mean and standard deviation $\boldsymbol{\sigma}_w$ and $\boldsymbol{\sigma}_v$, respectively, $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ is the measurement variables at time-k, $\widetilde{\boldsymbol{H}} \in \mathbb{R}^{n_z \times n_x}$ is the measurement matrix, and n_x, n_u, and n_z denote the number of the state, input, and measurement variables, respectively. The purpose of MHE is to minimize the error between the actual value of the measurement and the measurement values of the system by considering the constraints on the state and input variables. The actual values in MHE can be obtained from the sensor device installed on an object.

The formulation of the MHE optimization problem at each time-k, where $k \leq N$, can be written as follows:

$$J_{MHE} = \min_{\substack{\boldsymbol{x}_{k-N},\cdots,\boldsymbol{x}_k \\ \boldsymbol{u}_{k-N},\cdots,\boldsymbol{u}_{k-1}}} \sum_{j=k-N}^{k} \left\| \tilde{\boldsymbol{z}}_j - \widetilde{\boldsymbol{H}} \boldsymbol{x}_j \right\|_{\boldsymbol{V}}^{2} + \sum_{j=k-N}^{k-1} \left\| \widetilde{\boldsymbol{u}}_j - \boldsymbol{u}_j \right\|_{\boldsymbol{W}}^{2} \tag{3}$$

subject to

$$\boldsymbol{x}_{j+1} = \boldsymbol{A}_d \boldsymbol{x}_j + \boldsymbol{B}_d \boldsymbol{u}_j, \qquad j \in [k-N, k-1] \tag{4}$$

$$\boldsymbol{x}^{min} \leq \boldsymbol{x}_j \leq \boldsymbol{x}^{max}, \qquad j \in [k-N, k] \tag{5}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}_j \leq \boldsymbol{u}^{max}, \qquad j \in [k-N, k-1] \tag{6}$$

where $\tilde{\boldsymbol{z}}_j$ represents the actual values of the measurement at time-j, $\widetilde{\boldsymbol{u}}_j$ is the optimal input obtained from a controller at time-j, N is the estimation horizon or MHE step, $\boldsymbol{x}^{min}$ and $\boldsymbol{x}^{max}$ are the lower and upper bounds of the state variables, respectively, $\boldsymbol{V} \in \mathbb{R}^{n_z \times n_z}$ and $\boldsymbol{W} \in \mathbb{R}^{n_u \times n_u}$ are the weighting matrices for measurement and input variables, respectively, $\boldsymbol{u}^{min}$ and $\boldsymbol{u}^{max}$ are the lower and upper bounds of the input variables, respectively, and

$$\left\| \tilde{\boldsymbol{z}}_j - \widetilde{\boldsymbol{H}} \boldsymbol{x}_j \right\|_{\boldsymbol{V}}^{2} = \left(\tilde{\boldsymbol{z}}_j - \widetilde{\boldsymbol{H}} \boldsymbol{x}_j \right)^T \boldsymbol{V} \left(\tilde{\boldsymbol{z}}_j - \widetilde{\boldsymbol{H}} \boldsymbol{x}_j \right)$$

$$\left\| \widetilde{\boldsymbol{u}}_j - \boldsymbol{u}_j \right\|_{\boldsymbol{W}}^{2} = \left(\widetilde{\boldsymbol{u}}_j - \boldsymbol{u}_j \right)^T \boldsymbol{W} \left(\widetilde{\boldsymbol{u}}_j - \boldsymbol{u}_j \right)$$

The MHE optimization problem in Equations (3)-(6) is then transformed into quadratic programming form so that the solution process is easier to find.

2.2 Quadratic Programming of MHE Optimization Problem

The process to solve mathematical optimization problems in terms of a quadratic objective function is known as quadratic programming. The optimization process is to minimize or maximize an objective function by considering the linear constraints of the variables. Quadratic programming can be categorized into nonlinear programming types [20]. In general, the form of quadratic programming can be written as follows [21]:

$$J = \frac{1}{2}\boldsymbol{\chi}^T \boldsymbol{H}\boldsymbol{\chi} + \boldsymbol{f}^T\boldsymbol{\chi} \tag{7}$$

subject to

$$\boldsymbol{A}\cdot\boldsymbol{\chi} \leq \boldsymbol{b}, \tag{8}$$

$$\boldsymbol{A}_{eq}\cdot\boldsymbol{\chi} = \boldsymbol{b}_{eq}, \tag{9}$$

$$\boldsymbol{lb} \leq \boldsymbol{\chi} \leq \boldsymbol{ub}, \tag{10}$$

where $\boldsymbol{H}, \boldsymbol{A},$ dan $\boldsymbol{A}_{eq}$ are the matrices, and $\boldsymbol{f}, \boldsymbol{b}, \boldsymbol{b}_{eq}, \boldsymbol{lb}, \boldsymbol{ub}$, and $\boldsymbol{\chi}$ are the vectors. One of the alternative tools to solve the optimization problem in Equation (7)-(10) is by utilizing a toolbox in MATLAB, namely "quadprog" syntax.

The steps to transform the MHE optimization problem (3)-(6) into quadratic programming form are as follows:

1. Objective function

$$\begin{aligned} J_{MHE} &= \left(\tilde{\boldsymbol{z}}_{k-N} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N}\right)^T \boldsymbol{V}\left(\tilde{\boldsymbol{z}}_{k-N} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N}\right) + \cdots + \left(\tilde{\boldsymbol{z}}_{k} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k}\right)^T \boldsymbol{V}\left(\tilde{\boldsymbol{z}}_{k} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k}\right) + \\ &\quad (\widetilde{\boldsymbol{u}}_{k-N} - \boldsymbol{u}_{k-N})^T \boldsymbol{W}(\widetilde{\boldsymbol{u}}_{k-N} - \boldsymbol{u}_{k-N}) + \cdots + (\widetilde{\boldsymbol{u}}_{k-1} - \boldsymbol{u}_{k-1})^T \boldsymbol{W}(\widetilde{\boldsymbol{u}}_{k-1} - \boldsymbol{u}_{k-1}) \\ &= \left(\tilde{\boldsymbol{z}}_{k-N}^T - \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\right) \boldsymbol{V}\left(\tilde{\boldsymbol{z}}_{k-N} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N}\right) + \cdots + \left(\tilde{\boldsymbol{z}}_{k}^T - \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\right) \boldsymbol{V}\left(\tilde{\boldsymbol{z}}_{k} - \widetilde{\boldsymbol{H}}\boldsymbol{x}_{k}\right) + \\ &\quad (\widetilde{\boldsymbol{u}}_{k-N}^T - \boldsymbol{u}_{k-N}^T)\, \boldsymbol{W}\,(\widetilde{\boldsymbol{u}}_{k-N} - \boldsymbol{u}_{k-N}) + \cdots + (\widetilde{\boldsymbol{u}}_{k-1}^T - \boldsymbol{u}_{k-1}^T)\, \boldsymbol{W}\,(\widetilde{\boldsymbol{u}}_{k-1} - \boldsymbol{u}_{k-1}) \\ &= \tilde{\boldsymbol{z}}_{k-N}^T \boldsymbol{V}\,\tilde{\boldsymbol{z}}_{k-N} - \tilde{\boldsymbol{z}}_{k-N}^T \boldsymbol{V}\,\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N} - \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\,\tilde{\boldsymbol{z}}_{k-N} + \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\,\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N} + \cdots + \\ &\quad \tilde{\boldsymbol{z}}_{k}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k} - \tilde{\boldsymbol{z}}_{k}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k} - \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k} + \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k} + \widetilde{\boldsymbol{u}}_{k-N}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-N} - \widetilde{\boldsymbol{u}}_{k-N}^T\boldsymbol{W}\boldsymbol{u}_{k-N} \\ &\quad -\boldsymbol{u}_{k-N}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-N} + \boldsymbol{u}_{k-N}^T\boldsymbol{W}\boldsymbol{u}_{k-N} + \cdots + \widetilde{\boldsymbol{u}}_{k-1}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-1} - \widetilde{\boldsymbol{u}}_{k-1}^T\boldsymbol{W}\boldsymbol{u}_{k-1} - \\ &\quad \boldsymbol{u}_{k-1}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-1} + \boldsymbol{u}_{k-1}^T\boldsymbol{W}\boldsymbol{u}_{k-1} \end{aligned}$$

The forms that do not contain $\boldsymbol{x}$ and $\boldsymbol{u}$ can be removed from the optimization, so we obtain,

$$\begin{aligned} J_{MHE} &= -\tilde{\boldsymbol{z}}_{k-N}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N} - \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k-N} + \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N} - \cdots - \tilde{\boldsymbol{z}}_{k}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k} - \\ &\quad \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k} + \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k} - \widetilde{\boldsymbol{u}}_{k-N}^T\boldsymbol{W}\boldsymbol{u}_{k-N} - \boldsymbol{u}_{k-N}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-N} + \boldsymbol{u}_{k-N}^T\boldsymbol{W}\boldsymbol{u}_{k-N} \\ &\quad - \cdots - \widetilde{\boldsymbol{u}}_{k-1}^T\boldsymbol{W}\boldsymbol{u}_{k-1} - \boldsymbol{u}_{k-1}^T\boldsymbol{W}\,\widetilde{\boldsymbol{u}}_{k-1} + \boldsymbol{u}_{k-1}^T\boldsymbol{W}\boldsymbol{u}_{k-1} \\ &= -2\boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k-N} + \boldsymbol{x}_{k-N}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k-N} - \cdots - 2\boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\tilde{\boldsymbol{z}}_{k} + \boldsymbol{x}_{k}^T\widetilde{\boldsymbol{H}}^T\boldsymbol{V}\widetilde{\boldsymbol{H}}\boldsymbol{x}_{k} \\ &\quad -2\boldsymbol{u}_{k-N}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-N} + \boldsymbol{u}_{k-N}^T\boldsymbol{W}\boldsymbol{u}_{k-N} - \cdots - 2\boldsymbol{u}_{k-1}^T\boldsymbol{W}\widetilde{\boldsymbol{u}}_{k-1} + \boldsymbol{u}_{k-1}^T\boldsymbol{W}\boldsymbol{u}_{k-1} \end{aligned} \tag{13}$$

Then, Equation (11) can be written into Equation (7) by defining,

$$\boldsymbol{\chi} = \begin{bmatrix} \boldsymbol{x}_{k-N} \\ \boldsymbol{u}_{k-N} \\ \boldsymbol{x}_{k-N+1} \\ \vdots \\ \boldsymbol{u}_{k-1} \\ \boldsymbol{x}_k \end{bmatrix}_{((n_u+n_x)N+n_x)\times 1}, \boldsymbol{f} = \begin{bmatrix} -2\widetilde{\boldsymbol{H}}^T \boldsymbol{V}\, \tilde{\boldsymbol{z}}_{k-N} \\ -2\boldsymbol{W}\, \tilde{\boldsymbol{u}}_{k-N} \\ -2\widetilde{\boldsymbol{H}}^T \boldsymbol{V}\, \tilde{\boldsymbol{z}}_{k-N+1} \\ \vdots \\ -2\boldsymbol{W}\, \tilde{\boldsymbol{u}}_{k-1} \\ -2\widetilde{\boldsymbol{H}}^T \boldsymbol{V}\, \tilde{\boldsymbol{z}}_k \end{bmatrix}_{((n_u+n_x)N+n_x)\times 1}, \text{ and}$$

$$\boldsymbol{H} = \begin{bmatrix} 2\widetilde{\boldsymbol{H}}\,\boldsymbol{V}\,\widetilde{\boldsymbol{H}} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & 2\boldsymbol{W} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & 2\widetilde{\boldsymbol{H}}\,\boldsymbol{V}\,\widetilde{\boldsymbol{H}} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & 2\widetilde{\boldsymbol{H}}\,\boldsymbol{V}\,\widetilde{\boldsymbol{H}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & 2\boldsymbol{W} \end{bmatrix}_{((n_u+n_x)N+n_x)\times((n_u+n_x)N+n_x)}$$

2. System constraint

   Considering the multiple shooting approach, i.e., $\boldsymbol{x}_{j+1} - \boldsymbol{A}_d\boldsymbol{x}_j - \boldsymbol{B}_d\boldsymbol{u}_j = \boldsymbol{0}, j \in [k-N, k-1]$ For every j, we can obtain Equation (9) by defining,

$$\boldsymbol{A}_{eq} = \begin{bmatrix} -\boldsymbol{A}_d & -\boldsymbol{B}_d & \boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{A}_d & -\boldsymbol{B}_d & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & -\boldsymbol{B}_d & \boldsymbol{I} \end{bmatrix}_{(n_xN)\times((n_u+n_x)N+n_x)}, \boldsymbol{b}_{eq} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix}_{(n_xN)\times 1}$$

3. The lower and upper bounds in Equation (10) for the decision variables are,

$$\boldsymbol{lb} = \begin{bmatrix} \boldsymbol{x}_{k-N}^{min} \\ \boldsymbol{u}_{k-N}^{min} \\ \boldsymbol{x}_{k-N+1}^{min} \\ \vdots \\ \boldsymbol{u}_{k-1}^{min} \\ \boldsymbol{x}_k^{min} \end{bmatrix}_{((n_u+n_x)N+n_x)\times 1}, \boldsymbol{ub} = \begin{bmatrix} \boldsymbol{x}_{k-N}^{max} \\ \boldsymbol{u}_{k-N}^{max} \\ \boldsymbol{x}_{k-N+1}^{max} \\ \vdots \\ \boldsymbol{u}_{k-1}^{max} \\ \boldsymbol{x}_k^{max} \end{bmatrix}_{((n_u+n_x)N+n_x)\times 1}$$

For this case, there is no inequality constraint as in Equation (8).

The procedure to perform the MHE method can be described in Figure 1. The estimation is done offline, i.e., the measurements are synthesized first and then the state estimator (MHE) can be implemented. The data are obtained from MPC simulation and the number of measurement data is denoted by n.

3. RESULTS AND DISCUSSION

This section discusses how to implement the proposed method that was developed in the previous section. The mathematical model used in this study is the lateral vehicle dynamics which can be represented by the following state space model [22]:

$$\begin{bmatrix} \dot{V}_y \\ \dot{\psi} \\ \dot{r} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -4.4021 & 0 & -12.4603 & 0 \\ 0 & 0 & 1 & 0 \\ 1.3913 & 0 & -5.1868 & 0 \\ 1 & 15 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_y \\ \psi \\ r \\ y \end{bmatrix} + \begin{bmatrix} 24.1270 \\ 0 \\ 15.8609 \\ 0 \end{bmatrix} \delta$$

where V_y is the lateral velocity of a car, ψ represents the heading or yaw angle, r is the yaw rate, y denotes the global position of a car on the y-axis, and δ is the steering angle. The state variables can be defined as $\boldsymbol{\chi} = \begin{bmatrix} V_y & \psi & r & y \end{bmatrix}^T$, while the input variable of the model is $\boldsymbol{u} = \delta$.

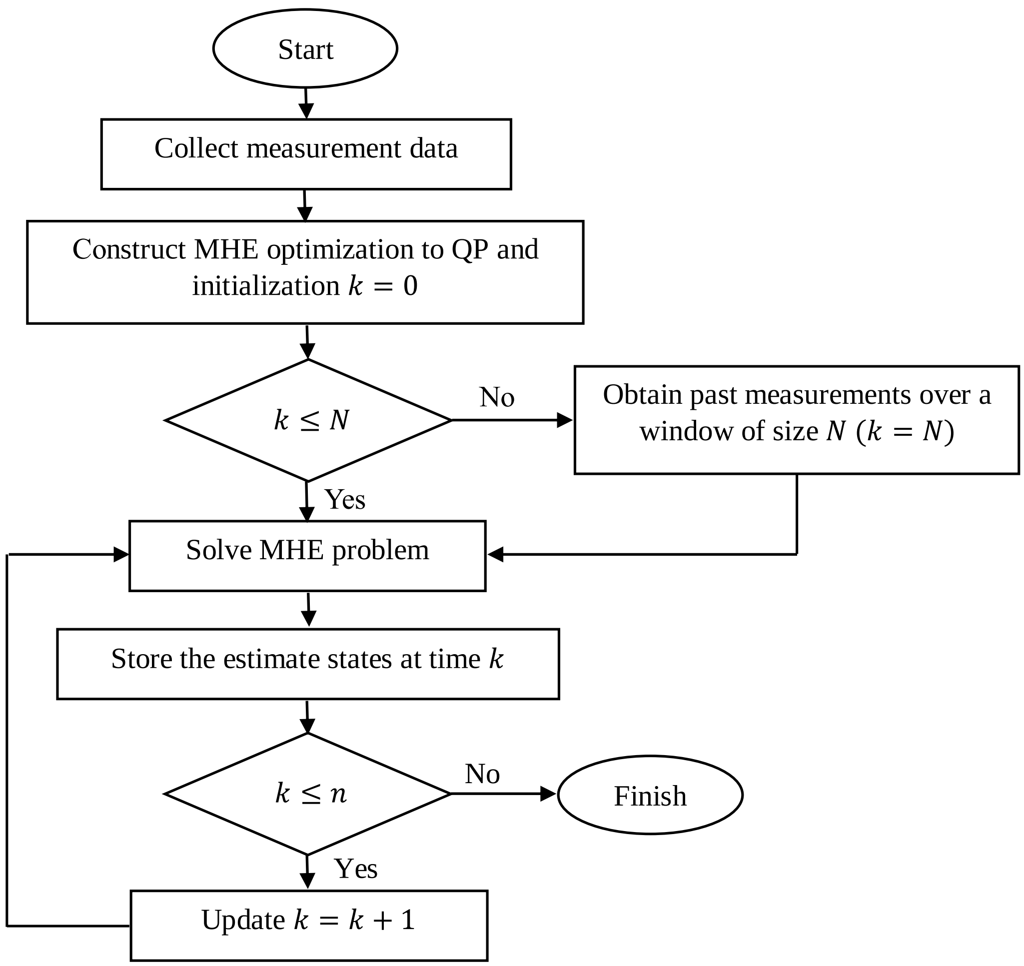

Figure 1. MHE procedure

The simulation in this study is performed through two scenarios. In the first scenario, we estimate the state variables in one MHE step, and the second one, we vary the MHE steps. In this case study, it is assumed that only the yaw angle and the global position can be measured, so the measurement matrix is $\widetilde{\boldsymbol{H}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$. To obtain the simulation results, the measurement covariance is set to

$\text{diag}([\text{deg2rad}(0.1) \quad 0.1])^2$. Then, the weighting matrices are set to $\boldsymbol{V} = \begin{bmatrix} \text{deg2rad}(0.1) & 0 \\ 0 & 0.1 \end{bmatrix}^{-1}$ and $\boldsymbol{W} = [\text{deg2rad}(0.1)]^{-1}$. The sampling time to transform the continuous system into a discrete system is $T_s = 0.1\ s$. The constraints of yaw angle and global position are $-0.2 \leq \psi \leq 0.2$ (in radian) and $-2 \leq y \leq 6$ (in meter). In addition, the input constraints are $-\frac{\pi}{6} \leq \delta \leq \frac{\pi}{6}$ (in radian).

3.1 Scenario 1: One MHE Step

In this scenario, the MHE step value is set to $N = n - 1$. Through this scheme, the solution obtained from the optimization is directly depicted in a figure to observe the state estimate results of MHE. Even though this strategy is not common in the field of estimation, the results of this scenario must be analyzed. The results of this scenario are shown in Figure 2.

Figure 2 shows the estimated results of one MHE step. Based on the figure, it can be said that the state estimates are close to the actual state values. The trajectories of true values are taken from the closed-loop system without the influence of noise. According to the figures of yaw angle and global position, due to the noises, the measurements do not approximate the true values. The weighting matrices $\boldsymbol{V}$ and $\boldsymbol{W}$ in the objective function of the optimization formula affect the estimated results as well. To observe the errors of measurements and MHE estimator results, it is essential to plot in a figure.

The errors of measurements and MHE are obtained based on the difference between the actual values and the measurement values as well as the MHE values. According to Figure 3, the errors of MHE for each time are smaller than the errors of measurements, except at the end of the simulation for the yaw angle error. This is caused by the MHE estimator that rejecting the noises in a system. In the next subsection, we observe the influence of MHE step variation to estimate the state variables.

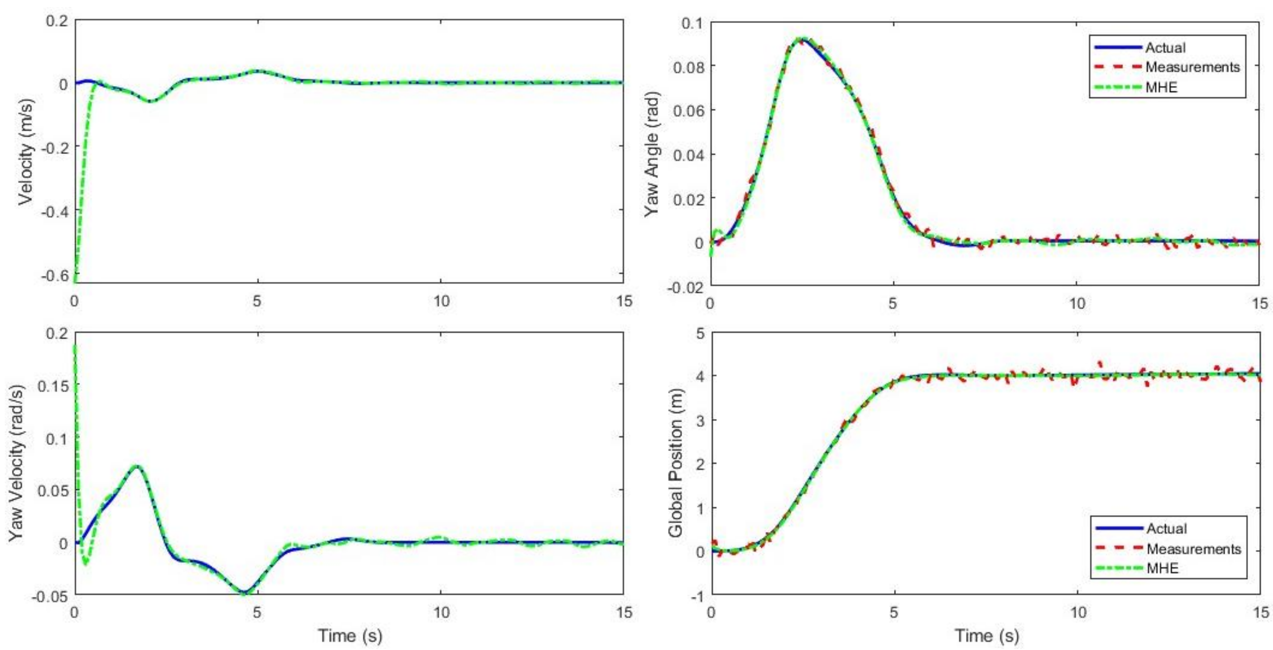

Figure 2. The state estimates one MHE step.

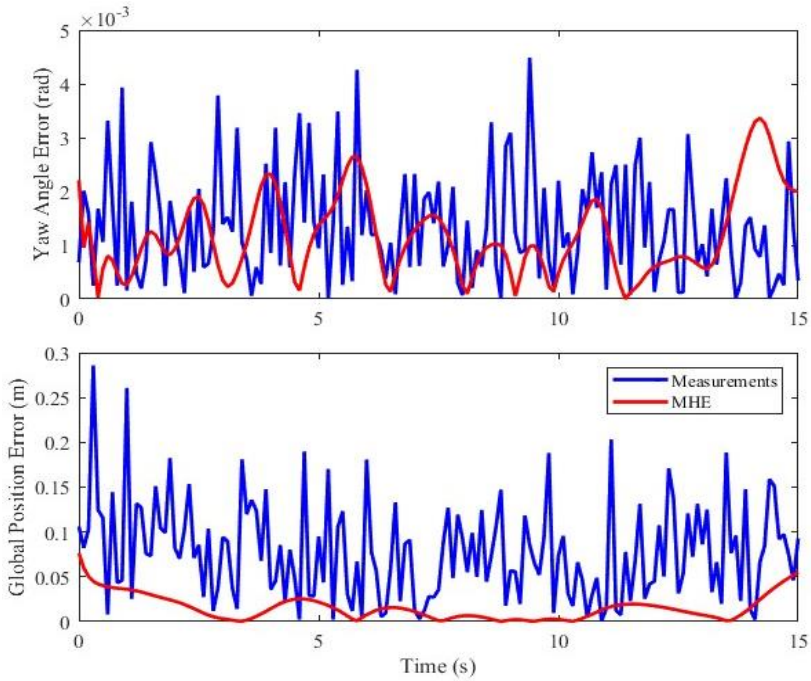

Figure 3. Error in measurements and MHE.

3.2 Scenario 2: The Variations of MHE Step

The procedure in this scenario is the same as in the first scenario, except that we vary the MHE steps. This concept is considered to understand the effect of the MHE steps when producing the state estimates. This idea is more relevant in the real case since the MHE step is set by a user. The MHE steps used in this scenario are $N = 10$ and $N = 15$. Using these MHE step values, the estimated results can be analyzed to find the proper MHE step.

Figure 4 shows the estimated results of MHE with different N. In Figure 4, it can be seen the MHE can estimate the state variables both with and without known measurement values affected by noises. Based on the simulation results, the state estimates of MHE are close to the actual values, indicating that the MHE has good performance for an autonomous vehicle. For $N = 10$ and $N = 15$, both yield almost similar results. The errors from the two results are depicted in Figure 5, including the errors of all state variables.

The errors are obtained from the difference between the actual and estimated values of MHE. According to Figure 5, the error amplitude from the two different MHE steps is very varied. It is too difficult to observe the differences between them from the plot. This difficulty can be caused by at a certain time, $N = 15$ has low amplitude values, but at certain other times, $N = 10$ has lower amplitude values. To be able to find the best performance, the RMSE value is used to compare beween the two cases.

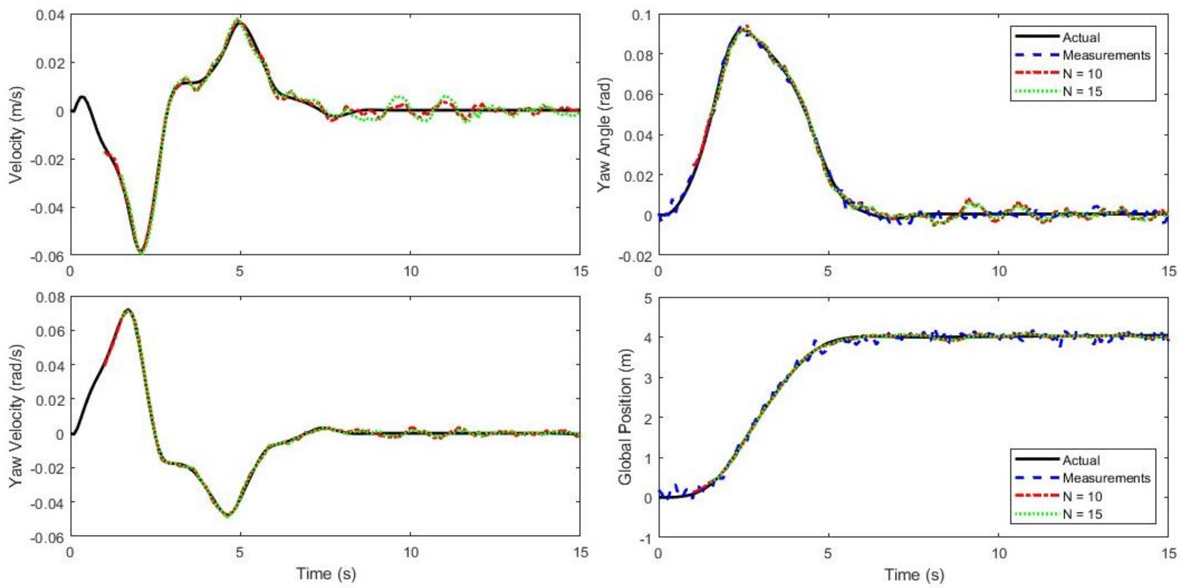

Figure 4. Simulation results of MHE with different N.

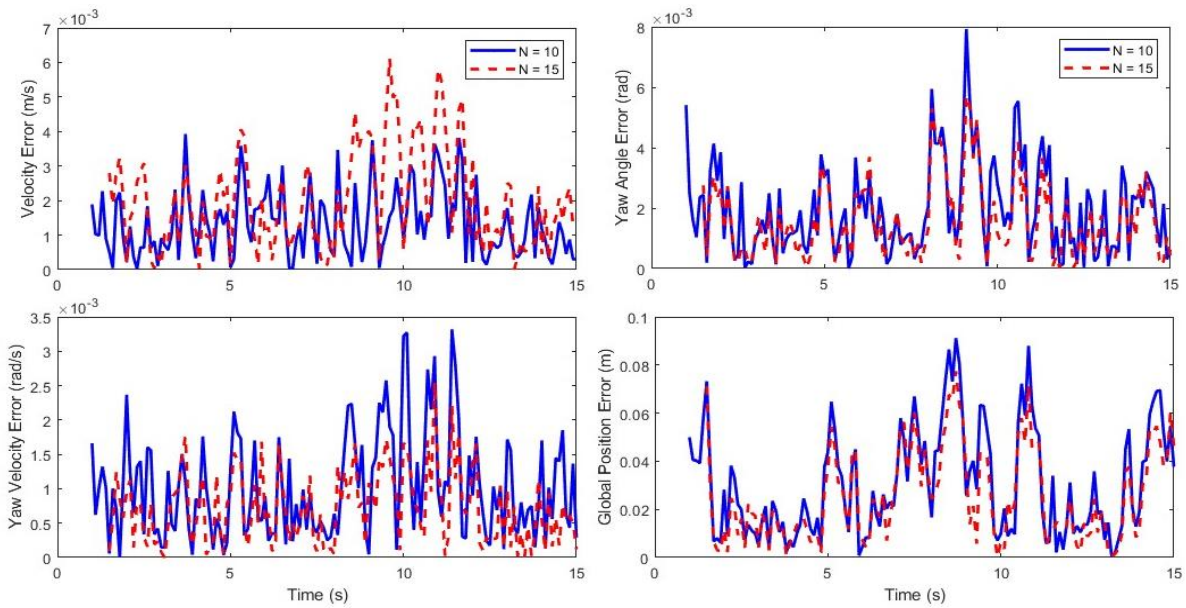

Figure 5. Error comparison for different N.

Table 1. RMSE values for different N

State variables	$N = 10$	$N = 15$
velocity	0.0016	0.0024
yaw angle	0.0024	0.0020
yaw velocity	0.0013	8.9535×10^{-4}
global position	0.0375	0.0318

Based on Table 1, there are no significant differences between the RMSE values for $N = 10$ and $N = 15$. Therefore, the different N values considered in this study only slightly affects the estimated results. However, the longer the MHE steps used, the longer the computational time. In other words, the computational time becomes heavy as MHE steps increase. In the next subsection, $N = 10$ is used to compare the performance of MHE with the Kalman filter.

3.3 Comparison of MHE and Kalman Filter

In this simulation, the MHE is compared to the Kalman filter (KF) to estimate the yaw angle and global position of an autonomous vehicle. In this case, the initial state error covariance is $\boldsymbol{P}_0 = 1e - 3 \cdot I_4$, where I_4 is the identity matrix with size 4×4, and the measurement noise covariance is set the same as the MHE estimator. The comparison of simulation results between MHE and KF can be seen in Figure 6.

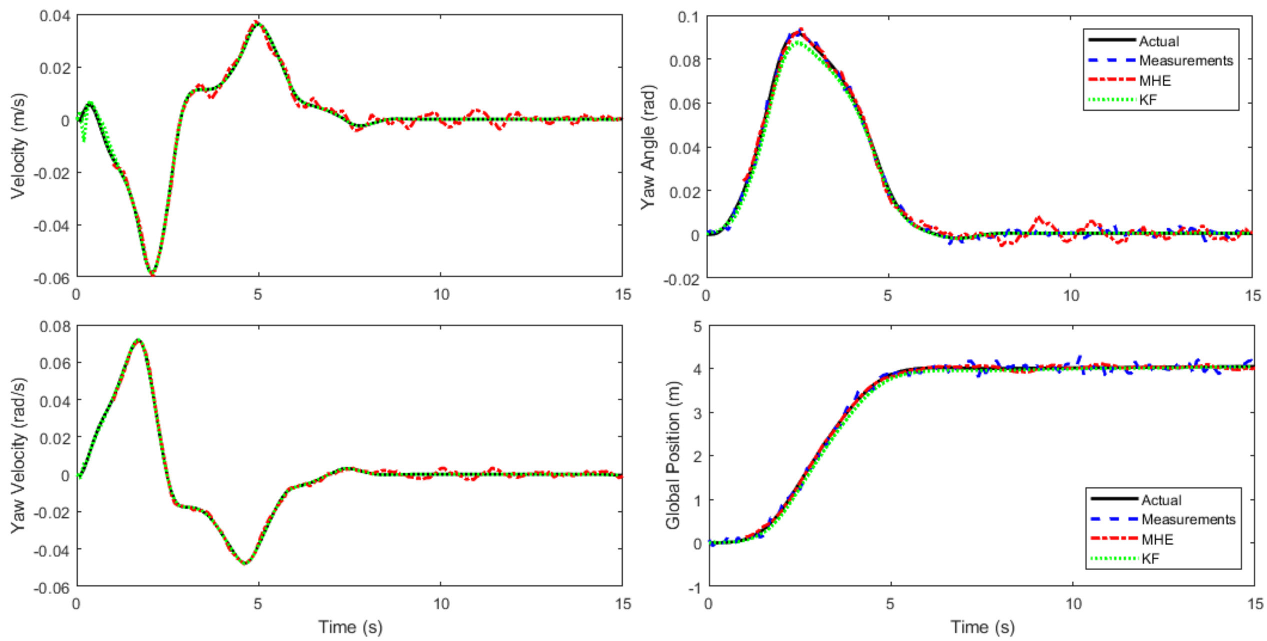

Figure 6. Comparison of MHE and KF

In Figure 6, it can be observed that both KF and MHE have good results, but the KF is better than MHE to estimate the state variables compared to MHE with $N = 10$. This statement is based on the simulation results, especially at the end of simulation time for each state variable. To support this finding, we report the errors of MHE and KF calculated from the difference between the actual and the estimated values.

Figure 7 shows the errors of estimation results between MHE and KF. The plot shows that KF has a lower error amplitude, and its errors are almost close to zero compared to MHE, particularly for velocity, yaw angle, and yaw velocity. Noted that the error of global position for KF is greater than MHE at the beginning of simulation time, but in the end, it converges to zero. Table 2 shows the comparison of RMSE between MHE and KF.

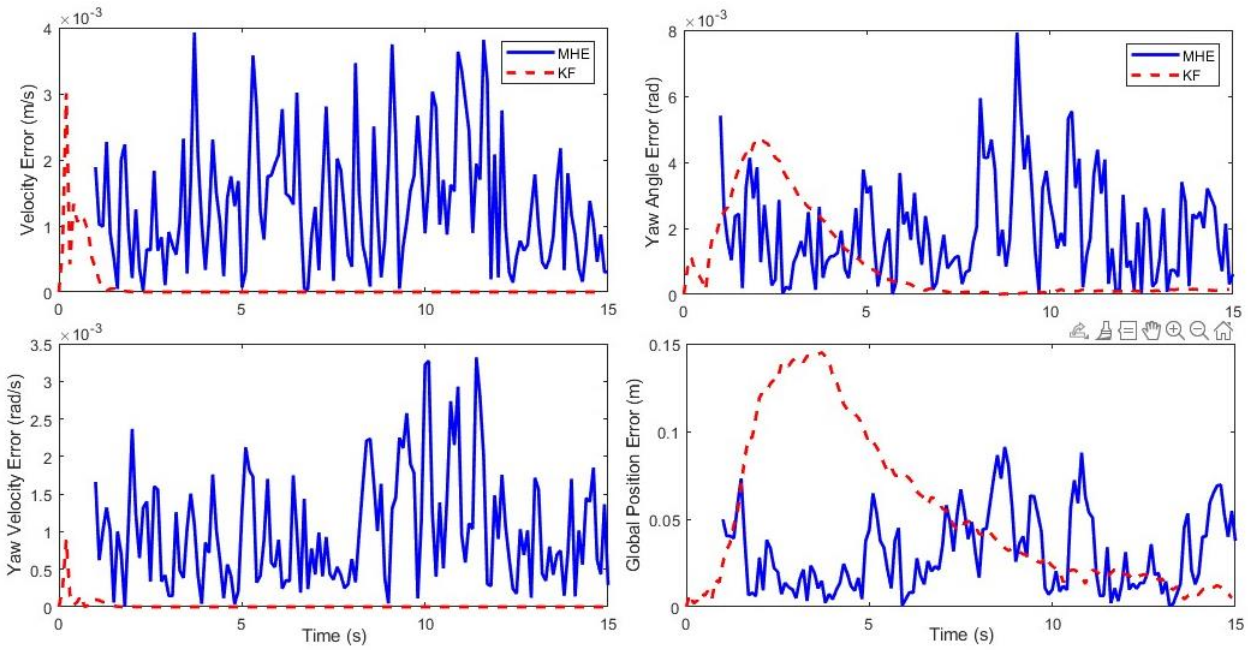

Figure 7. The comparison of error between MHE and KF

Table 2. RMSE values of MHE and KF

State variables	MHE	KF
velocity	0.0016	3.4720×10^{-5}
yaw angle	0.0024	0.0017
yaw velocity	0.0013	1.3150×10^{-5}
global position	0.0375	0.0721

Table 2 gives information about the RMSE values of MHE and KF for each state variable. The yaw angle and global position have very different RMSE values for both MHE and KF, while for velocity and yaw velocity, the KF has a smaller RMSE than MHE. By referring to Figure 6, the KF produces a better estimation since the errors go to zero. These findings can be caused by several factors. Firstly, we perform the estimation by offline as the data is collected first based on MPC simulation. Secondly, in the KF method, there are two stages, namely prediction and correction. These stages can produce better estimation results. Thirdly, in the MHE optimization, the estimation horizon and weighting matrices in the objective function must be tuned, so the obtained results can estimate the state variables perfectly.

4. CONCLUSIONS

This study applies the MHE estimator to estimate the state variables of an autonomous vehicle. The formulation of the MHE optimization problem transformed into quadratic programming is presented in this paper. The MHE gives better performance to estimate the state variables under measurement noises. It can also estimate unmeasured state variables, i.e., velocity and yaw velocity.

According to the simulation results, the difference in the MHE steps only slightly affects the estimation results. In this research, the comparison of MHE and KF is also reported. Based on the used case study, the KF produces better estimation results than MHE. However, this judgment must be further analyzed for other certain cases and approaches, such as the estimation process carried out online. For future research, we can perform comparison studies with KF modifications, such as the Extended Kalman filter (EKF) and Unscented Kalman filter (UKF) for nonlinear systems.

ACKNOWLEDGEMENTS

This research was supported by the Institute for Research, Development and Community Service (LITBANG PEMAS), Universitas Islam Lamongan (UNISLA) via the internal HIBAH scheme (070/SP2H/PDP/LITBANGPEMAS/2022).

REFERENCES

[1] M. Park, S. Lee, and W. Han, "Development of Steering Control System for Autonomous Vehicle Using Geometry-Based Path Tracking Algorithm," *ETRI J.*, vol. 37, no. 3, pp. 617–625, 2015, doi: 10.4218/etrij.15.0114.0123.

[2] M. W. Mehrez, G. K. Mann, and R. G. Gosine, "Nonlinear Moving Horizon State Estimation for Multi-Robot Relative Localization," *IEEE 27th Can. Conf. Electr. Comput. Eng.*, 2014, doi: 10.1109/CCECE.2014.6901134.

[3] S. A. Talla Ouambo, A. T. Boum, and A. Moukengue Imano, "Parameters and States Estimation by Moving Horizon Estimation, High Gain Observer and Unscented Kalman Filter of a Doubly-Fed Induction Generator Driven by Wind Turbine: A Comparative Study.," *J. Eng. Sci. \& Technol. Rev.*, vol. 11, no. 2, 2018.

[4] R. Alexander, G. Campani, S. Dinh, and F. V Lima, "Challenges and opportunities on nonlinear state estimation of chemical and biochemical processes," *Processes*, vol. 8, no. 11, p. 1462, 2020.

[5] X. Li, A. J. Cheng, and H. X. Lin, "Sample Regenerating Particle Filter Combined With Unequal Weight Ensemble Kalman Filter for Nonlinear Systems," *IEEE Access*, vol. 9, pp. 109612–109623, 2021.

[6] H. Purnawan, U. Ilmi, R. A. Faroh, and A. B. A. A. Rizqi, "Positioning Estimation of Autonomous Car using Extended Kalman Filter," *1st ICEHST 2022*, vol. 1, no. 02, pp. 1–8, 2022.

[7] H. Joachim Ferreau, T. Kraus, M. Vukov, W. Saeys, and M. Diehl, "High-Speed Moving Horizon Estimation based on Automatic Code Generation," in *51st IEEE Conference on Decision and Control*, pp. 687–692, 2012.

[8] C. V. Rao and J. B. Rawlings, "Nonlinear Moving Horizon State Estimation," in *Nonlinear Model Predictive Control*, Springer, pp. 45–69, 2000.

[9] R. Kandepu, L. Imsland, and B. A. Foss, "Constrained State Estimation Using the Unscented Kalman Filter," in *16th Mediterranean Conference on Control and Automation*, pp. 1453–1458, 2008.

[10] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO Toolkit - An Open Source Framework for Automatic Control and Dynamic Optimization," *Optim. Control Appl. Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[11] W. Zhang, Z. Wang, C. Zou, L. Drugge, and M. Nybacka, "Advanced vehicle state monitoring: Evaluating moving horizon estimators and unscented Kalman filter," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5430–5442, 2019.

[12] H. Liu, P. Wang, J. Lin, H. Ding, H. Chen, and F. Xu, “Real-time longitudinal and lateral state estimation of preceding vehicle based on moving horizon estimation,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 8755–8768, 2021.

[13] D. Mori and Y. Hattori, “Simultaneous estimation of vehicle position and data delays using gaussian process based moving horizon estimation,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2303–2308, 2020.

[14] M. Gulan, M. Salaj, M. Abdollahpouri, and B. Rohal-Ilkiv, “Real-Time MHE-based Nonlinear MPC of A Pendubot System,” *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 422–427, 2015.

[15] S. A. P. Quintero, D. A. Copp, and P. Hespanha, “Robust UAV Coordination for Target Tracking using Output-Feedback Model Predictive Control with Moving Horizon Estimation,” pp. 3758–3764, 2015.

[16] T. Kraus *et al.*, “Moving Horizon Estimation and Nonlinear Model Predictive Control for Autonomous Agricultural Vehicles,” *Comput. Electron. Agric.*, vol. 98, pp. 25–33, 2013, doi: 10.1016/j.compag.2013.06.009.

[17] D. A. Allan and B. R. James, “Moving Horizon Estimation,” in *Handbook of Model Predictive Control*, Birkhäuser, Cham, pp. 99–124, 2019.

[18] J. D. Hedengren, R. A. Shishavan, K. M. Powell, and T. F. Edgar, “Nonlinear Modeling, Estimation and Predictive Control in APMonitor,” *Computers & Chemical Engineering*, 2014.

[19] N. Hashemian and A. Armaou, “Fast Moving Horizon Estimation of Nonlinear Processes via Carleman Linearization,” pp. 3379–3385, 2015.

[20] M. S. Bazaraa, D. S. Hanif, and M. S. Chitharanjan, *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 2013.

[21] I. The Mathworks, “MATLAB version 9.10.0.1739362 (R2021a),” 2021.

[22] U. Melda, “Designing an MPC Controller with Simulink,” MATLAB Central File Exchange. Retrieved July 27, 2022.