sensors

MDPI

Article

A Rapid and Adaptive Alignment under Mooring Condition Using Adaptive EKF and CNN-Based Learning

Jong Nam Lim and Chan Gook Park *

Department of Aerospace Engineering, Seoul National University, Seoul 08826, Korea; ljn78@snu.ac.kr

* Correspondence: chanpark@snu.ac.kr; Tel.: +82-2-880-1675

Received: 25 May 2020; Accepted: 15 July 2020; Published: 22 July 2020

Abstract: Alignment of the inertial navigation system (INS) in the mooring environment should take into account the movements of the waves or wind. The alignment of the INS is performed through an extended Kalman filter (EKF) using zero velocity as a measurement. However, in the mooring condition, this is not perfect stationary, thus the measurement error covariance matrix should be adjusted. In addition, if the measurement error covariance matrix is fixed to one value, the alignment time may take longer or the performance may be reduced depending on the change in mooring conditions. To solve this problem, we propose an alignment method using adaptive Kalman filter and convolution neural network (CNN)-based learning. The proposed method was verified for the superiority of alignment time and accuracy through Monte Carlo simulation in a mooring environment.

Keywords: convolution neural network (CNN); adaptive extended Kalman filter (EKF); alignment; mooring environment; inertial navigation system (INS)

1. Introduction

The inertial navigation system (INS) is the most fundamental navigation system that provides position, velocity, and attitude information using the gyro to measure angular velocity and the accelerometer to measure the linear acceleration of the vehicle. The strapdown-based INS calculates navigation solution in the navigation frame from the output of the inertial sensors attached to the body frame through recursive integration and coordinate transformation based on initial information. Therefore, it is necessary to find the initial attitude precisely, and this process is called alignment [1,2].

Alignment in the stationary state is performed using the Earth's rate and gravitational acceleration. To perform the alignment, a gyrocompassing method using a control law or an estimator based on a Kalman filter technique capable of estimating errors of sensors as well as an attitude during alignment has been applied.

The stationary state has a complete zero velocity, thus the alignment process is performed using the zero-velocity information as a measurement in Kalman filter. However, if the alignment is performed in a sea environment, the velocity and attitude are continuously changed by the wind and wave. Therefore, it is essential for navigation systems operated on marine platforms, such as a ship and a submarine, to apply the technique of performing alignment in sea environment [1–3].

We conducted a study on Extended Kalman Filter (EKF)-based alignment using INS in a mooring environment. Here, EKF, which is a linearized Kalman filter, is applied with consideration to the integration with the nonlinear system INS. The most crucial point is how to handle this linear velocity and rotation that repeatedly occur in the mooring environment with wind or wave. These movements appear as errors or disturbances that interfere with a stationary condition, thus EKF-based alignment can be performed by setting the measurement error covariance large.

However, in the continuously changing mooring environment, this method may degrade performance depending on changing situations and takes a long time to finish the alignment. Therefore, we propose adaptive methods using adaptive EKF and CNN-based learning for alignment in the mooring environment. The adaptive EKF uses innovation of the measurement to estimate the measurement error covariance matrix continuously, and outputs an adaptive result according to the condition of the measurement. The CNN-based learning method learns the optimal measurement error covariance matrix to minimize the error in changing the mooring environment based on a large number of data.

2. Related Work

Various methods of using digital filters such as FIR or IIR have been developed to deal with this problem. Lian et al. proposed a method to reduce the effect of acceleration disturbance by using a finite impulse response (FIR) filter [4]. Feng et al. applied the infinite impulse response (IIR) filter and power spectrum analysis to compensate for the disadvantages of the FIR filter that must have a huge order [5]. However, this digital filter method has a delay problem in real-time applications and has limitations due to a fixed filter coefficient in a sea environment in which the magnitude or period of motion continuously changes.

Gaiffe et al. proposed an inertial frame-based alignment (IFBA) technique to project gravity on an inertial frame to estimate the coordinate transformation matrix between the inertial frame and the body frame [6]. Since then, this IFBA technique has been used by many researchers for alignment in mooring [5,7,8]. Gao et al supposed a fast alignment method using IFBA, and bidirectional Kalman filter [7], and Sun et al applied the IFBA and hidden Markov model to prevent filter delay for alignment in mooring [8].

If the conventional EKF is used for alignment in the mooring environment, the alignment is disturbed by rate and moving. Therefore, it is necessary to set an appropriate error covariance, and the time required for alignment is increased. The adaptive EKF has been applied to various cases in which the environment or condition changes. It is classified into a case where the measurement changes and a case where the system model changes [9,10]. The alignment in mooring uses zero velocity as a measurement, and since the disturbance caused by the waves or wind can be regarded as an error in the measurement, we use a measurement dependent adaptive EKF.

The deep learning technique has been widely used in fields that solve classification, recognition, and estimation problems in various nonlinear environments using many data. Among various deep learning techniques, CNN was first introduced by Lecun and applied to handwriting zip code recognition. Then, it was generalized and expanded in concept, making it widely used in image processing, recognition, and classification [11–13].

Recently, research to apply deep learning techniques to the Kalman filter has been conducted. In the process of applying the Kalman filter to the pose estimation problem, Coskun et al. proposed a long short-term memory–Kalman filter (LSTM-KF) that learn motion model, measurement model and noise covariance of Kalman filter through three independent LSTM modules [14]. Martin proposed a method to reduce navigation errors when driving a car by learning [15].

Lee and Bang calculated the filter coefficient of RBPF in terrain-aided navigation (TAN) through the LSTM-based learning [16]. By estimating the measurement error covariance, system error covariance, and measurement model using four LSTM modules, it showed high performance in the TAN environment with strong nonlinearity.

3. Conventional Extended Kalman Filter Based Alignment in Mooring Environment

Since alignment is performed based on INS, the model of INS is used as the system model, and the zero-velocity model is used as the measurement model for the Kalman filter. Furthermore, we apply an indirect method based Conventional Extended Kalman filter (CEKF) that linearizes the Kalman filter for the nonlinear INS model.

As mentioned above, the easiest Kalman filter-based method to perform alignment in the mooring environment is to set the filter coefficients with consideration of movement conservatively. When the zero velocity is used as a measurement in the mooring environment, repetitive changes in velocity or attitude due to wind or waves occur in the form of a sin wave. Since this can be regarded as the large error of the measurement, the Kalman filter should be performed by setting the measurement error covariance matrix R to be larger than the conventional case.

The conventional EKF recursively performs the measurement update and time propagation. The nonlinear system model and measurement model are as follows.

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{w}(t) \tag{1}$$

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t)) + \mathbf{v}(t) \tag{2}$$

$$\mathbf{w}(t) \sim N(0,\ \mathbf{Q}), \mathbf{v} \sim N(0,\ \mathbf{R}) \tag{3}$$

where **f** is a nonlinear function for the system equation; **h** is a nonlinear function for the measurement equation; **x** is a state variable; **y** is a measurement; **w** is a noise of the system model; **v** is noise of the measurement model, which follows a Gaussian Probability Density Function (PDF); **Q** is system error covariance matrix; and **R** is measurement error covariance matrix.

The results of linearizing and discretizing Equations (1) and (2) and expressing them in error state are as follows [1,3].

$$\delta\hat{\mathbf{x}}_{k+1} = (\mathbf{I} + \mathbf{F}_k\Delta t)\delta\hat{\mathbf{x}}_k + \mathbf{w}_k \tag{4}$$

$$\delta\mathbf{y}_k = \mathbf{H}_k\delta\hat{\mathbf{x}}_k + \mathbf{v}_k \tag{5}$$

where

$$\mathbf{F}_k = \left.\frac{\partial\mathbf{f}_k}{\partial\mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_k},\ \mathbf{H}_k = \left.\frac{\partial\mathbf{h}_k}{\partial\mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_k} \tag{6}$$

The time update equation of EKF is as follows.

$$\delta\hat{\mathbf{x}}_{k+1|k} = \mathbf{F}_k\delta\hat{\mathbf{x}}_k \tag{7}$$

$$\mathbf{P}_{k+1|k} = \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k^T + \mathbf{Q}_k \tag{8}$$

The measurement update equation of EKF is as follows.

$$\delta\hat{\mathbf{x}}_{k+1} = \delta\hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_k(\delta\mathbf{z}_k - \mathbf{H}_k\delta\hat{\mathbf{x}}_{k+1|k}) \tag{9}$$

$$\mathbf{S}_{k+1} = \mathbf{H}_{k+1}\mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T + \mathbf{R}_{k+1} \tag{10}$$

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T\mathbf{S}_{k+1}^{-1} \tag{11}$$

$$\mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} - \mathbf{K}_{k+1}\mathbf{S}_{k+1}\mathbf{K}_{k+1}^T \tag{12}$$

The nonlinear error model of the INS system is as follows [1,2], and the system matrix $\mathbf{F}_k$ refers to Appendix A.

$$\begin{aligned}
\delta\dot{\mathbf{V}}^n &= \left[\mathbf{C}_b^n\mathbf{f}^b\right]\times\delta\mathbf{\Phi}^n - \left[2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n\right]\times\delta\mathbf{V}^n + \mathbf{C}_b^n\delta\mathbf{f}^b + \mathbf{V}^n\times[\delta\boldsymbol{\omega}_{en}^n] \\
\delta\dot{\mathbf{\Phi}}^n &= \delta\boldsymbol{\omega}_{in}^n - \boldsymbol{\omega}_{in}^n\times\delta\mathbf{\Phi}^n - \mathbf{C}_b^n\delta\boldsymbol{\omega}_{ib}^b \\
\delta\dot{\nabla} &= 0 \\
\delta\dot{\varepsilon} &= 0
\end{aligned} \tag{13}$$

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{V}^n & \delta\Phi^n & \delta\nabla & \delta\varepsilon \end{bmatrix}^T \tag{14}$$

where $\delta\mathbf{V}^n$ is a velocity error, $\delta\mathbf{\Phi}^n$ is an attitude error, $\delta\nabla$ is an accelerometer error, $\delta\varepsilon$ is a gyro error, $\mathbf{C}_b^n$ is a coordinate transformation matrix that converts from body frame to navigation frame, $\boldsymbol{\omega}_{ie}^n$ is an angular rate due to the Earth rate in navigation frame, $\boldsymbol{\omega}_{en}^n$ is an angular rate due to the transpose rate in navigation frame, $\mathbf{g}^n$ is a gravity acceleration in navigation frame, and $\mathbf{f}^b$ is a specific force in body frame.

$$\begin{aligned}\delta\mathbf{y}_k &= \begin{bmatrix} \delta\hat{V}_k^n & \delta\hat{V}_k^e & \delta\hat{V}_k^d \end{bmatrix} \\ \delta\mathbf{z}_k &= \mathbf{H}_k\delta\mathbf{x}_k + \mathbf{v}_k = \begin{bmatrix} \delta\overline{V}_k^n & \delta\overline{V}_k^e & \delta\overline{V}_k^d \end{bmatrix}^T = \begin{bmatrix} \overline{V}_k^n - 0 & \overline{V}_k^e - 0 & \overline{V}_k^d - 0 \end{bmatrix}^T\end{aligned} \tag{15}$$

where $\delta\hat{V}_k^n$ is an error of propagated north-direction velocity, $\delta\hat{V}_k^e$ is an error of propagated east-direction velocity, $\delta\hat{V}_k^d$ is an error of propagated down-direction velocity. $\overline{V}_k^n$ is a measured north-direction velocity, $\overline{V}_k^e$ is a measured east-direction velocity, $\overline{V}_k^d$ is a measured down-direction velocity, $\delta\overline{V}_k^n$ is an error of measured north-direction velocity, $\delta\overline{V}_k^e$ is an error of measured east-direction velocity, and $\delta\overline{V}_k^d$ is an error of measured down-direction velocity.

However, in the case of using the fixed measurement error covariance matrix, performance and convergence speed of the Kalman filter change according to sea conditions, which change with time or position. In general, assuming that system error covariance matrix Q is appropriately fixed, if the measurement error covariance matrix R becomes small, the filter believes the measurement more. Since it is judged that the measurement is accurate, the performance is improved, and convergence speed is fast. However, if the error of the actual measurement is larger than the expected measurement error, performance of the filter degrades, and even the filter may diverge.

Conversely, if the measurement error covariance matrix R is large, the filter judges that the error of the measurement is significant. Thus, the convergence speed becomes slow, but the divergence of the filter can be avoided. In other words, if the error of the actual measurement in the mooring environment is similar to the error of the expected measurement, the best result is estimated. Since the error of actual measurement in the mooring environment is changing, conventional EKF using a fixed measurement error covariance matrix may degrade performance.

Therefore, we propose two methods for alignment under the mooring condition, as described in the next sections.

4. Adaptive EKF Based Alignment in Mooring Environment

The adaptive EKF is to adaptively change measurement error covariance matrix R using an innovation sequence, which is defined as the difference between the measured value and the estimated value. The innovation sequence is as follows [9].

$$\mathbf{i}_k = \delta\mathbf{z}_k - \mathbf{H}\delta\hat{\mathbf{x}}_k = \delta\mathbf{z}_k - \mathbf{HF}\delta\hat{\mathbf{x}}_{k-1} \tag{16}$$

where $\mathbf{i}_k$ is an innovation sequence, $\mathbf{z}_k$ is a measurement, $\mathbf{H}$ is a measurement model matrix, and $\mathbf{F}$ is a system model matrix.

Substituting the measurement in Equation (15) into Equation (16),

$$\begin{aligned}\mathbf{i}_k &= \delta\mathbf{z}_k - \mathbf{HF}\delta\hat{\mathbf{x}}_{k-1} = \mathbf{H}\delta\mathbf{x}_k + v_k - \mathbf{HF}\delta\hat{\mathbf{x}}_{k-1} \\ &= \mathbf{H}(\delta\mathbf{x}_k - \mathbf{F}\delta\hat{\mathbf{x}}_{k-1}) + v_k = \mathbf{H}\tilde{\mathbf{x}}_{k|k-1} + v_k\end{aligned} \tag{17}$$

Innovation sequence is a kind of indicator of real estimation errors that can be used for the adaptive algorithm. The covariance of innovation is calculated using the above equation and is as follows [9,10].

$$\mathbf{I}_k = E\left[\mathbf{i}_k\mathbf{i}_k^T\right] = \mathbf{HP}_{k|k-1}\mathbf{H}^T + \mathbf{R}_k \tag{18}$$

Therefore, an adaptive measurement error covariance matrix is as follows.

$$\mathbf{R}_k = \mathbf{I}_k - \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T \tag{19}$$

where $\mathbf{I}_k$ is a covariance of innovation, $\mathbf{i}_k$ is an innovation, $\mathbf{H}$ is an observation matrix, $\mathbf{P}_{k|k-1}$ is a state error covariance, and $\mathbf{R}_k$ is a measurement error covariance matrix.

In the stationary system and noise environment, $\mathbf{I}_k$ can be calculated by using the average as following equation [17].

$$\hat{\mathbf{I}}_k = \frac{1}{k}\sum_{m=1}^{k} \mathbf{i}_m \mathbf{i}_m^T \tag{20}$$

Moreover, converting the above equation to recurrent form is as follows.

$$\hat{\mathbf{I}}_k = \frac{k-1}{k}\hat{\mathbf{I}}_{k-1} + \frac{1}{k}\mathbf{i}_k \mathbf{i}_k^T \tag{21}$$

Substituting Equation (21) into Equation (19), we can obtain the estimated adaptive measurement error covariance matrix as follows.

$$\hat{\mathbf{R}}_k = \hat{\mathbf{I}}_k - \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T \tag{22}$$

At this time, to prevent R from becoming negative in the case $\hat{\mathbf{I}}_k$ is too small, the following equation is added.

$$if\ diag\left(\hat{\mathbf{R}}_k\right) < 0,\ then\ diag\left(\hat{\mathbf{R}}_k\right) = 0 \tag{23}$$

An innovation is obtained from the measurement, and the covariance of innovation is calculated through data. Since the calculated covariance of innovation represents the condition of the measurement, the measurement error covariance matrix is automatically adjusted according to the current measurement. In other words, if the wave is large, the measurement error is large, so the measurement error covariance matrix R is increased. Conversely, if the wave is small, the measurement error is small, and then the measurement error covariance matrix R is decreased.

5. Learning-Based Alignment in Mooring Environment

5.1. Introduction of CNN

In this section, the measurement error covariance matrix R is adjusted according to the wave condition using the trained network. In this paper, convolutional neural network (CNN) is applied to learn time-series sensor data. Recursive neural network (RNN) is generally used to process this kind of time series data. However, this time series data can also be processed as a convolution feature, thus this study uses a one-dimensional convolution network. Compared to RNN, CNN has the advantage of less complexity and reduced computation, and better extraction of data characteristics according to conditions. In fact, in the field of audio generation and machine translation, CNN has shown good results in the past few years.

CNN extracts important characteristic among related data. In other words, CNN is not just learning a list of data as in the multi-layer perceptron (MLP) but learning spatial correlation by assigning local connections between neurons in the adjacent layer. CNN compresses the characteristics of the original signal through convolution. At this time, according to the filter used, a feature map is constructed to output only the characteristics that can best represent the data. The feature map starts from low-level feature information and synthesizes low-level features as the layer deepens to generate high-level features.

Learning using this CNN has the advantage of being able to process many data at a time through a convolution operation, and significantly reducing the number of parameters to be learned through the sharing of weights and the receptive field. In addition, CNN can extract and learn spatial and

temporal characteristics of data through spatial/temporal correlation, thus it has a robustness against noise or external disturbance [18].

The input through sensors is continuous time-series data, and data of the previous time and current time are dependent and have a relationship by system model.

5.2. Architecture of CNN

5.2.1. Architecture

In this paper, CNN is composed of an input layer that receives sensor input, a convolution layer that performs synthesis, a ReLU activation function that adds nonlinearity, a dropout layer, and an output layer. The output layer consists of a fully connected layer and outputs the measurement error covariance matrix R, as shown in Figure 1.

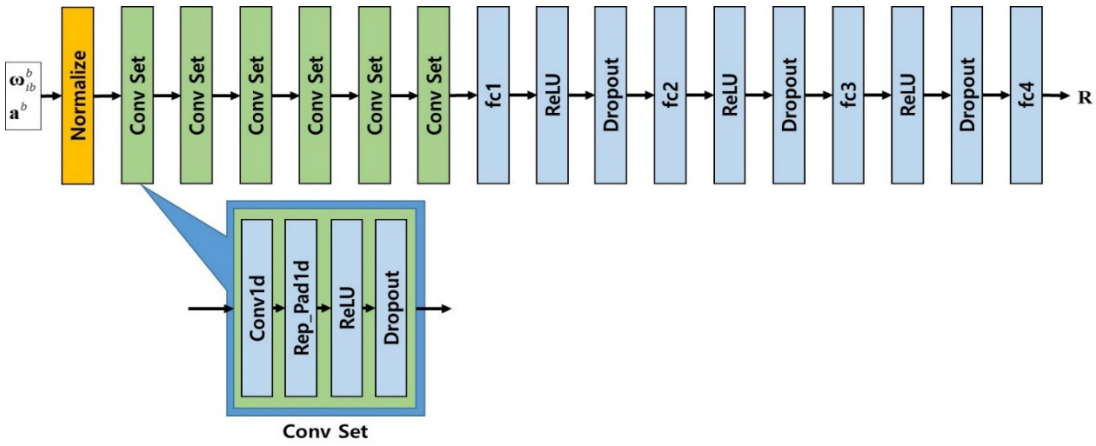

Figure 1. Architecture of CNN layers.

$\boldsymbol{\omega}_{ib}^{b}$ is an angular rate that is the output of gyro, $\mathbf{a}^{b}$ is a linear acceleration that is an output of accelerometer, and Conv Set is composed of Conv1d representing one-dimensional convolution layer, Rep_Pad1d, which creates one-dimensional padding by copying the boundary values, ReLU activation function, and Dropout layer.

The input of learning is the output of the inertial sensors, which is linear acceleration and angular rate. Since the values have different characteristics, normalization preprocessing is performed using the average and variance of the sensor output. The output is measurement error covariance matrix R, which is the parameter that controls the EKF.

5.2.2. Input Data and Normalization

The network inputs consist of three-axis gyros and accelerometers. When performing alignment during mooring, the ship moves repeatedly depending on the size of the sea waves, so the parameter of the filter must be adjusted according to this movement. Gyros measure rotation and accelerometers measure linear acceleration, thus the networks input is the physical quantity measured by the gyros and accelerometers.

The scale of the physical quantity measured by the gyros and accelerometers is not the same. In the static condition, the acceleration on the horizontal x and y axes is not large. However, the z axis accelerometer measures the Earth's gravitational acceleration, thus it outputs a relatively large value. In addition, the gyros measure the Earth rate, but its size is very small.

Therefore, normalization is performed to make the range of all physical quantities equal. Normalization is performed using the mean and standard deviation as shown in Equations (24)–(27).

$$\mathbf{u}_n = (\mathbf{u} - \mathbf{u}_{mean}) / \mathbf{u}_{std} \tag{24}$$

where

$$\mathbf{u} = \begin{bmatrix} \omega_{ib}^{bx} & \omega_{ib}^{by} & \omega_{ib}^{bz} & | & \mathbf{a}^{bx} & \mathbf{a}^{by} & \mathbf{a}^{bz} \end{bmatrix}^T \tag{25}$$

$$\mathbf{u}_{mean} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{u}^{(i)} \tag{26}$$

$$\mathbf{u}_{std} = \sqrt{\frac{1}{N}\left[\sum_{i=1}^{N}\left(\mathbf{u}^{(i)} - \mathbf{u}_{mean}\right)^2\right]} \tag{27}$$

Since the calculation frequency of the inertial navigation and EKF filter is 10 Hz, the sampling time of the input data for training is 0.1 s. For continuous time series data to output the measurement error covariance matrix R, continuous sensor data of 20 s were used.

5.2.3. Convolution Layer

One-dimensional convolution layer is applied to perform the convolution operation of time series sensor data. The output of the convolution layer is input to the ReLU activation function to add nonlinearity, and the dropout layer is connected to the activation function to reduce the size and give generality. Total convolution layers are composed of six consecutive Conv Sets.

One Conv Set is composed of one-dimensional convolution layer, one-dimensional padding layer, one ReLU, and one dropout layer, as shown in Figure 2.

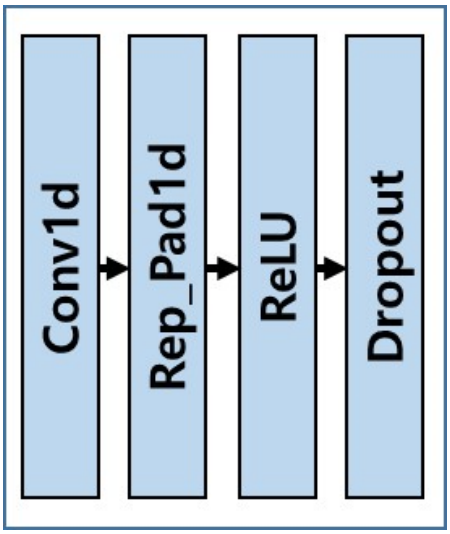

Figure 2. Block diagram of Conv Set.

In this paper, the conv1d function provided by PyTorch is applied to construct one-dimensional convolution layer; the length of output sequence is determined by Equation (28).

$$L_{out} = \left[\frac{L_{in} + 2 \times P - D \times (K-1) - 1}{S} + 1\right] \tag{28}$$

where L_{out} is the length of output sequence, L_{in} is the length of input sequence, P is the length of padding, D is the length of dilation, K is the length of kernel (filter), S: length of stride.

The values of each parameter for CNN layer used here are shown in Table 1 and the length of the output sequence was designed to remain the same as the length of the input sequence.

Table 1. Convolution layer parameters.

Parameter	Value
L_{out}: length of output sequence	200
L_{in}: length of input sequence	200
P: length of padding	4
D: length of dilation	2
K: length of kernel (filter)	5
S: length of stride	1

5.2.4. Fully Connected Layer and Network Output

The output of the convolutional layer is connected to the fully connected layers. The final output, the measurement error covariance matrix R, is calculated through the output of a fully connected layer. All fully connected layers are composed of four consecutive layer sets.

The output of the fully connected layer is connected to the ReLU activation function and goes through the dropout layer to add generality. The last fully connected layer omits the ReLU and dropout layer to output the measurement error covariance R.

The size of each fully connected layer is [200 × 2000], and this size remains the same in each fully connected layer. The output size of the final fully connected layer is [200 × 6]. The output value of network is finally converted to the measurement error covariance R using Equation (29). Here, dimension of R is [200 × 3].

$$\mathbf{R} = \mathbf{R}_0 \times 10^{\mathbf{z}} \tag{29}$$

where $\mathbf{R}$ is the measurement error covariance matrix, $\mathbf{R}_0$ is the initial measurement error covariance matrix, $\mathbf{z}$ is output of network.

5.2.5. Loss Function

Mean squared error applies to the loss function. Since the measurement of the EKF becomes the zero velocity, the error is the difference between the velocity estimate and the true velocity, as shown in Equation (30).

$$Loss(\gamma) = \frac{1}{n}\sum_{i=1}^{n} \|\hat{\mathbf{V}}_i^n(\gamma) - \mathbf{V}_{i,True}^n\|^2 \tag{30}$$

where $\hat{\mathbf{V}}_i^n(\gamma)$ is estimated velocity vector at time i, $\mathbf{V}_{i,True}^n$ is true velocity vector at time i.

5.2.6. Optimizer

The purpose of this learning is to optimize the measurement error covariance matrix R for EKF that minimizes the loss of Equation (30). This model updates the parameters using the gradient obtained through backpropagation. In this paper, PyTorch framework is applied as a tool for learning, and the optimizer used Adam optimizer that reflects the adaptive learning rate algorithm that combines the ideas of momentum optimization and RMSprop for fast and stable learning.

Adam optimizer use two parameters (learning rate and weight decay) for optimization. In this study, the learning rate of 0.00001 and the weight decay of 0.000001 were selected to satisfy the learning speed and accuracy.

5.2.7. Parameter Tuning for CNN Architecture

If the length of this sequence is too long, the characteristics of the rapidly changing sea wave cannot be learned well. It is also known that the size of an excessively long input window can over-smooth the result [19]. Conversely, if the length of this sequence is too short, the coefficient of the covariance changes too quickly, which can degrade the stability of the filter. Therefore, considering the above characteristics, the input length of the sequence data for training was set to 20 s.

In this paper, the architecture of the overall filter is superposition on the Conv Set structure to which the one-dimensional filter of 5 × 1 size is applied. This repetitive placement of the Conv Set utilizes non-linearity thorough the ReLU block and Dropout block in the middle of the Conv Set, thus it can make the desired feature stand out more and is advantages in terms of computation amount.

Conv Set was designed by configuring filter block, padding block, and dilation block so that the length of the input sequence and the output sequence are the same to keep the computation constant and balance the system. That is, even after passing through one Conv Set, 200 (20 s × 10 Hz) sequence data maintain their length.

Learning parameters for CNN architecture are shown in Table 2.

Table 2. Learning parameters.

Sub Parameter	Value
Maximum epoch	200
Input dimension	6
Output dimension	3
Initial bias of CNN layer	$u\left(-\sqrt{\frac{1}{5C_{in}}}, \sqrt{\frac{1}{5C_{in}}},\right)$
Initial weights of CNN layer	$u\left(-\sqrt{\frac{1}{5C_{in}}}, \sqrt{\frac{1}{5C_{in}}},\right)$
Initial bias of fully connected layer	$u\left(-\sqrt{\frac{1}{5F_{in}}}, \sqrt{\frac{1}{5F_{in}}},\right)$
Initial weights of fully connected layer	$u\left(-\sqrt{\frac{1}{5F_{in}}}, \sqrt{\frac{1}{5F_{in}}},\right)$
Learning rate	0.00001
Gradient decay factor	0.000001
Optimizer	Adam
Drop ratio of learning rate	0.9

C_{in}, size of input channel; F_{in}, size of each input sample; U, uniform distribution.

6. Simulation

6.1. Simulation Environment

To verify the performance of a proposed method, the mooring environment was simulated, and convergence speed and performance accuracy were compared by performing the alignment for the case of using a fixed measurement error covariance matrix, adaptive EKF, and learning as shown in Figures 3–5.

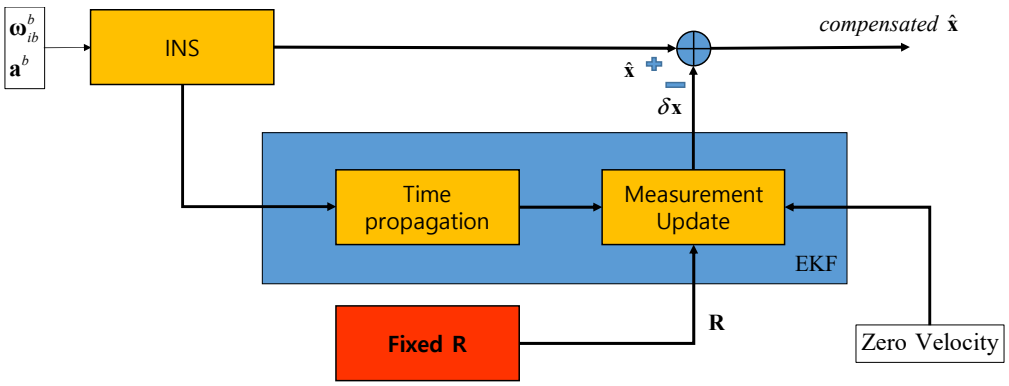

Figure 3. Test Case 1: CEKF with fixed R.

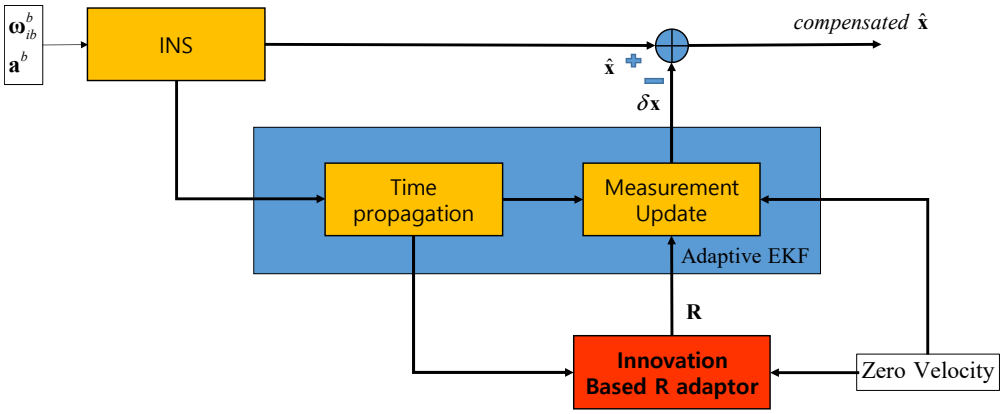

Figure 4. Test Case 2: Adaptive EKF.

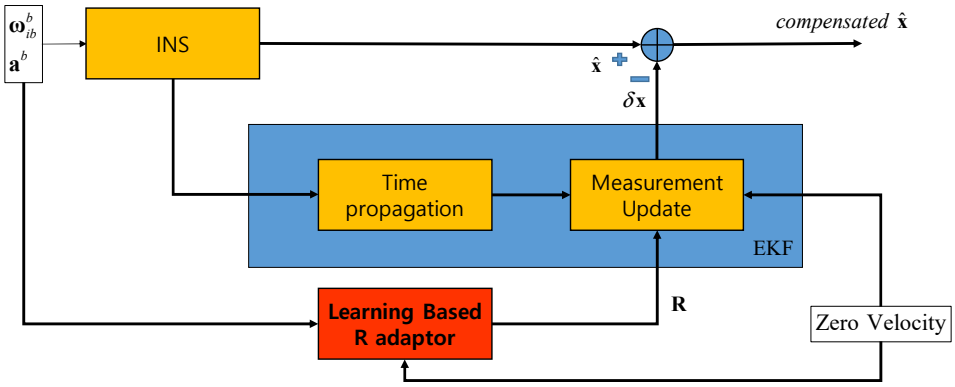

Figure 5. Test Case 3: Learning-based EKF.

Figure 3 shows the case of using a fixed error covariance R. INS uses the acceleration and angular velocity to calculate the navigation solutions. The EKF module located in the middle consists of the time propagation and measurement update process. The measurement update process is performed by applying the zero-velocity measurement and the fixed R value indicated in the red box. The compensated is obtained by subtracting the estimated error state from EKF to the navigation solution from INS.

Figure 4 shows the case where the adaptive EKF described in the Section 4 is used. The other parts are the same as in Figure 3. However, R is calculated adaptively in the red box labeled "Innovation Based R adaptor". The calculated R is used in the EKF measurement update in EKF.

Figure 5 shows the case where R calculated by learning is used. The error covariance matrix R is calculated thorough the network updated by learning in the red box marked "Learning-based R adaptor". The calculated R is used in the EKF measurement update in EKF.

To simulate the mooring, linear motion and angular rate are generated by the following equations and added to the gyro and accelerometer outputs.

$$\phi = A_{\phi}\cos\left[\frac{2\pi}{T_{\phi}}t\right],\ \theta = A_{\theta}\cos\left[\frac{2\pi}{T_{\theta}}t\right],\ \psi = A_{\psi}\cos\left[\frac{2\pi}{T_{\psi}}t\right] \tag{31}$$

$$P_x^b = A_{Pbx}\sin\left[\frac{2\pi}{T_{Pbx}}t\right],\ P_y^b = A_{Pby}\sin\left[\frac{2\pi}{T_{Pby}}t\right],\ P_z^b = A_{Pbz}\sin\left[\frac{2\pi}{T_{Pbz}}t\right] \tag{32}$$

Velocity outputs can be expressed as follows.

$$V_x^b = A_{Pbx}\frac{2\pi}{T_{Pbx}}\cos\left[\frac{2\pi}{T_{Pbx}}t\right],\ V_y^b = A_{Pby}\frac{2\pi}{T_{Pby}}\cos\left[\frac{2\pi}{T_{Pby}}t\right],\ V_z^b = A_{Pbz}\frac{2\pi}{T_{Pbz}}\cos\left[\frac{2\pi}{T_{Pbz}}t\right] \tag{33}$$

where A_ϕ, A_θ, A_ψ are the amplitude of attitude movement [roll, pitch, heading], A_{Pbx}, A_{Pby}, A_{Pbz} are the amplitude of position movement [x, y, z], T_ϕ, T_θ, T_ψ are the duration time of attitude movement [roll, pitch, heading], and T_{Pbx}, T_{Pby}, T_{Pbz} are the duration time of position movement [x, y, z].

The IMU sensors (gyros and accelerometers), initial navigation error, and parameters used in the simulation are shown in Table 3. To simulate the mooring environment, reference values for position, velocity, and attitude were generated using the parameter values defined in Table 3 and Equations (31)–(33). Figure 6 shows the position of a vehicle in the mooring condition. Figures 7 and 8 show the reference position and attitude generated under mooring conditions. Figures 9 and 10 show the true output of the gyro sensors and accelerometers sensors generated under mooring conditions.

Table 3. Simulation conditions.

	Sub Parameter	Value
Sensor Error	Gyro Bias	0.001°/hr (1σ, x/y/z axis)
	Gyro ARW	0.0005°/ $\sqrt{\text{hr}}$ (1σ, x/y/z axis)
	Accelerometer Bias	30 ug (1σ, x/y/z axis)
	Accelerometer Noise	20 ug (1σ, x/y/z axis)
Initial Error	Position	10 m (1σ, latitude/longitude axis)
	Velocity	0.1 m/s (1σ, north/east axis)
	Attitude (After Coarse Alignment)	Roll/Pitch 0.1° (1σ), Heading 5° (1σ)
Wave Condition	Attitude Amplitude	$A_\phi = 6°$, $A_\theta = 8°$, $A_\psi = 6°$
	Attitude duration	$T_\phi = 7$ s, $T_\theta = 5$ s, $T_\psi = 6$ s
	Velocity Amplitude	$A_{Pbx} = 0.3$ m, $A_{Pby} = 1.0$ m, $A_{Pbz} = 1.2$ m
	Velocity duration	$T_{Pbx} = 7$ s, $T_{Pby} = 6$ s, $T_{Pbz} = 8$ s
Mooring Condition	Small size of movement	10% of $[A_\phi, A_\theta, A_{Pbx}, A_{Pby}, A_{Pbz}]$
	Medium size of movement	50% of $[A_\phi, A_\theta, A_{Pbx}, A_{Pby}, A_{Pbz}]$
	large size of movement	100% of $[A_\phi, A_\theta, A_{Pbx}, A_{Pby}, A_{Pbz}]$
Case	Case 1	CEKF with fixed R (R = [0.1 m/s 0.1 m/s 0.1 m/s]^2)
	Case 2	Adaptive EKF
	Case 3	Learning-based EKF
Simulation Time		600 s

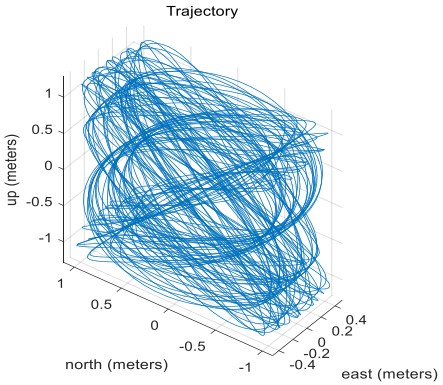

Figure 6. Trajectory in mooring environment.

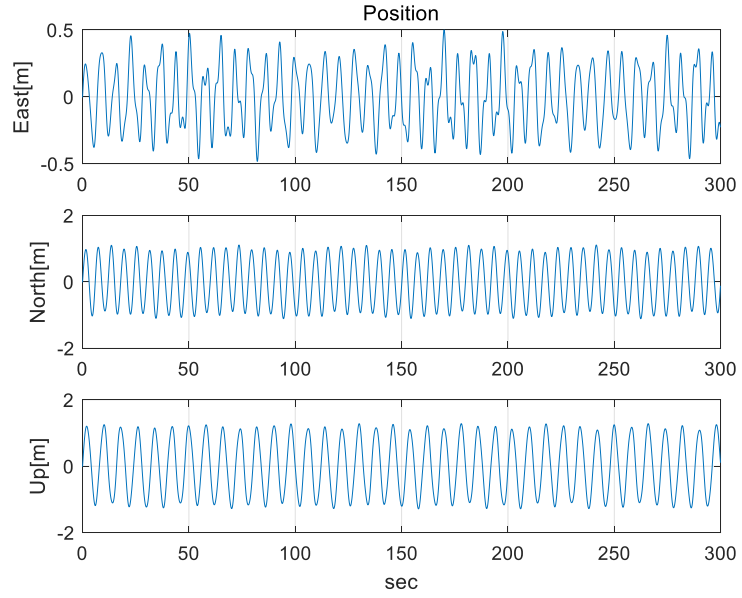

Figure 7. Reference Position in mooring environment. (Top: East-directional position, Middle: North-directional position, Bottom: Up-directional position).

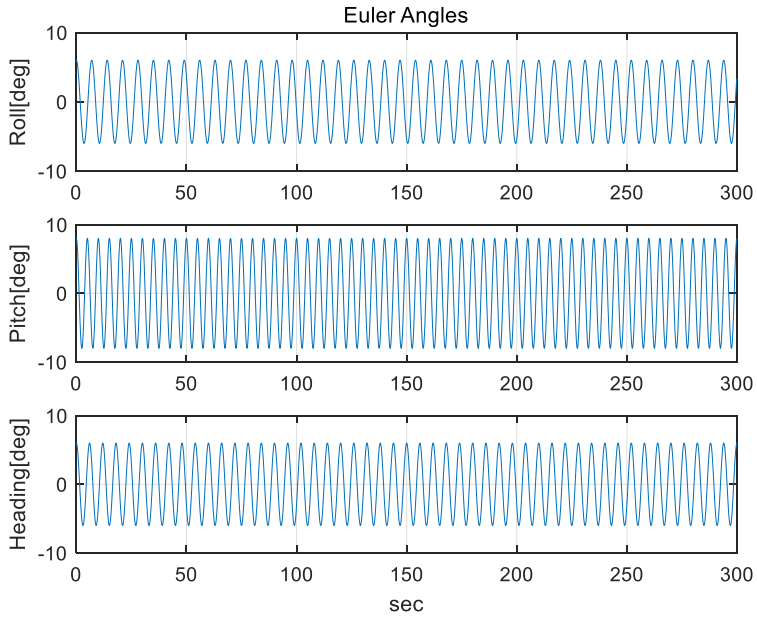

Figure 8. Reference Attitude in mooring environment. (Top: Roll angle, Middle: Pitch angle, Bottom: Heading angle).

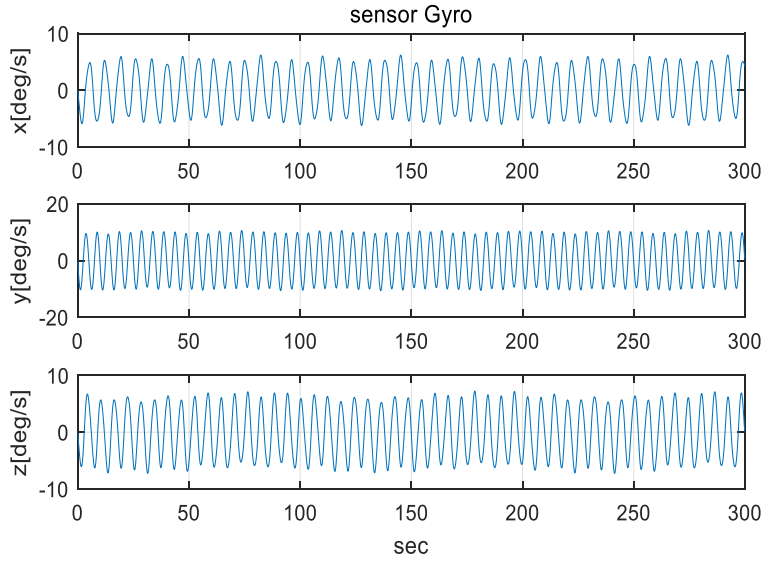

Figure 9. Output of Gyro Sensor in mooring environment. (Top: X-directional gyro output, Middle: Y-directional gyro output, Bottom: Z-directional gyro output).

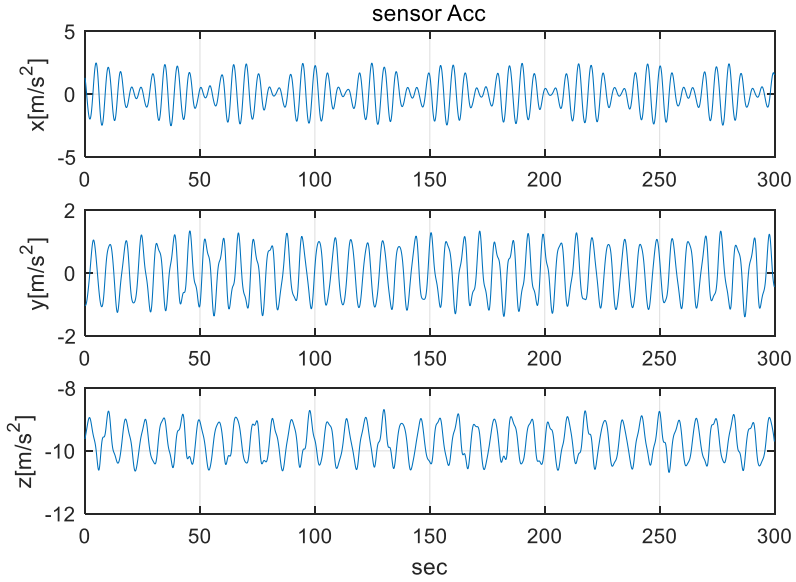

Figure 10. Output of Accelerometer in mooring environment. (Top: X-directional accelerometer output, Middle: Y-directional accelerometer output, Bottom: Z-directional accelerometer output).

Input data for learning were randomly generated using the values in Table 3. Based on the same reference trajectory, sensor errors, initial navigation errors, and mooring conditions were changed. The inertial sensor measurements were generated by randomly defining bias and noise with values that follow a normal distribution. In the same way, the initial navigation errors were generated by randomly defining the initial position errors, initial velocity errors, and initial attitude errors with values that follow a normal distribution. The mooring condition was generated by randomly defining the size and duration time of waves with values follow uniform distribution. The network was trained using the generated input data, and the training was repeatedly performed up to the maximum epoch, as defined in Table 2 for each data sequence.

The simulation was performed by dividing the mooring condition into three movements: (1) small size of movement; (2) medium size of movement; and (3) large size of movement. As shown in Table 3, the small size of the movement is defined as 10% of the amplitude of the large size of movement, and medium-size movement is defined as 50% of the amplitude of the large size of movement.

Alignment is performed in each case to compare the alignment accuracy and speed according to the method of applying the measurement error covariance matrix. As described above, it is divided into a method using CEKF with a fixed measurement error covariance matrix R, a method using adaptive EKF, and a method using CNN-based learning.

For coarse alignment, a feedback-based alignment technique can generally be applied in a mooring environment [9,10]. However, this paper focuses on fine alignment using Kalman filter and zero-velocity measurement, assuming that coarse alignment has been performed.

In consideration of the mooring environment, the initial attitude error after coarse alignment was set to 0.1° (roll/pitch, 1 σ) and 5° (heading, 1 σ). To perform the Monte Carlo simulation, sensor errors and initial errors were randomly generated. Repeated simulation was performed 100 times in each case, and the results were analyzed for the RMS error, alignment speed, and final error values.

6.2. Simulation Results

Simulation results are shown in Figures 11–19 and Tables 4–9. In the figures, the green lines show the results of 100 evolutions of Monte Carlo simulations with randomly set initial values. The red line is the RMS value of the result of 100 evolutions. Table 4, Table 6, and Table 8 show the RMS value of the error during the entire time for each case. Table 5, Table 7, and Table 9 show the RMS value of the last error for each case. In Table 4, Table 6, and Table 8, the times of convergence is defined as the time when the heading error (RMS) is within 20% of the initial heading error.

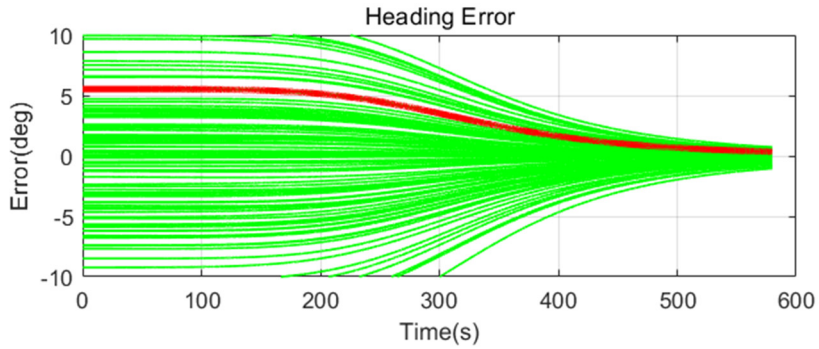

Figure 11. Heading error of alignment with CEKF (Case 1) in small size of movement.

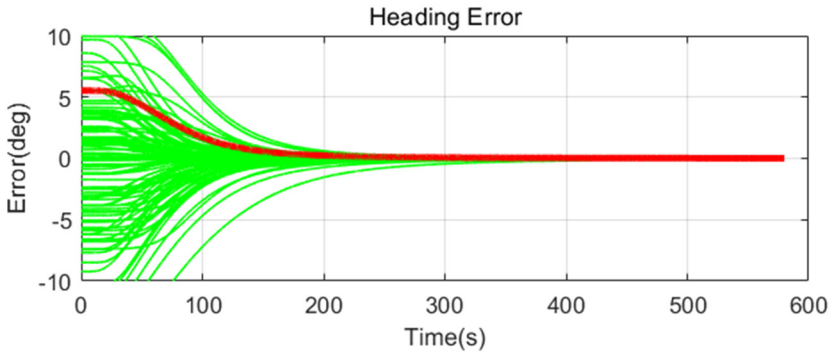

Figure 12. Heading error of alignment with Adaptive EKF (Case 2) in small size of movement.

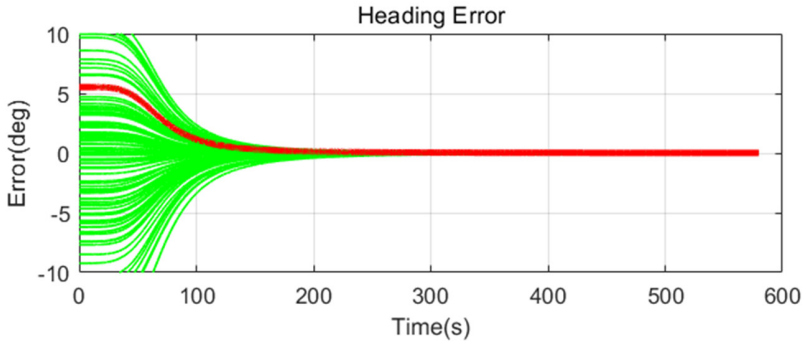

Figure 13. Heading error of alignment with learning-based EKF (Case 3) in small size of movement.

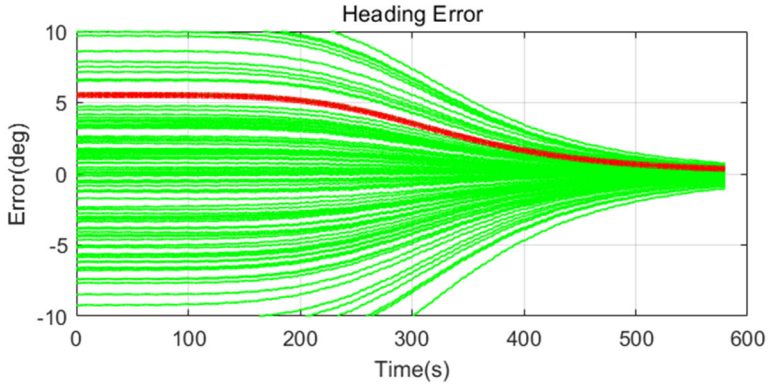

Figure 14. Heading error of alignment with CEKF (Case 1) in medium size of movement.

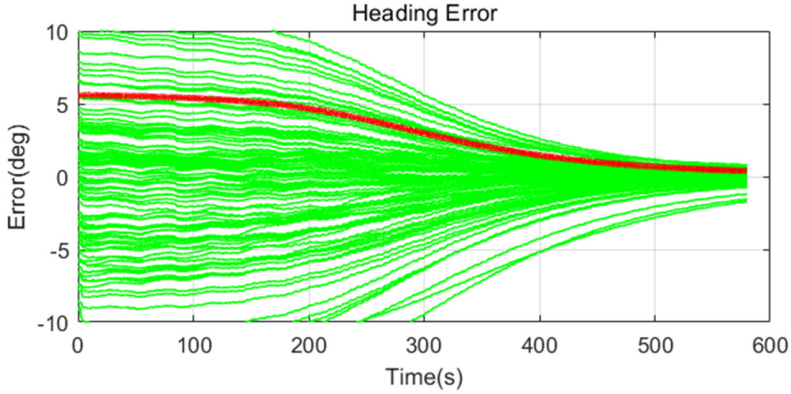

Figure 15. Heading error of alignment with Adaptive EKF (Case 2) in medium size of movement.

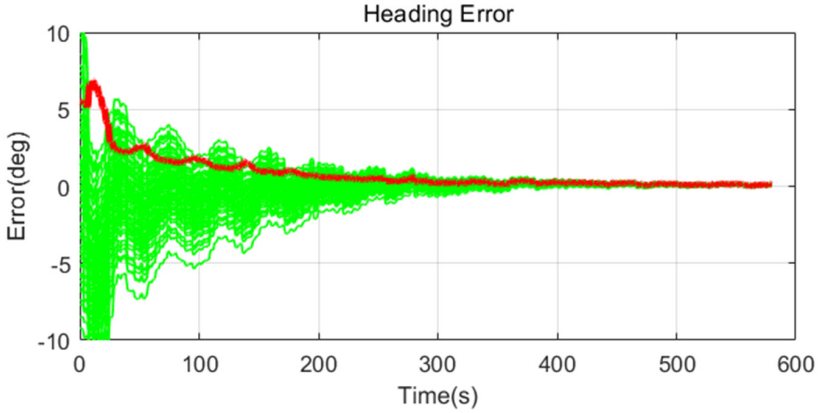

Figure 16. Heading error of alignment with learning-based EKF (Case 3) in medium size of movement.

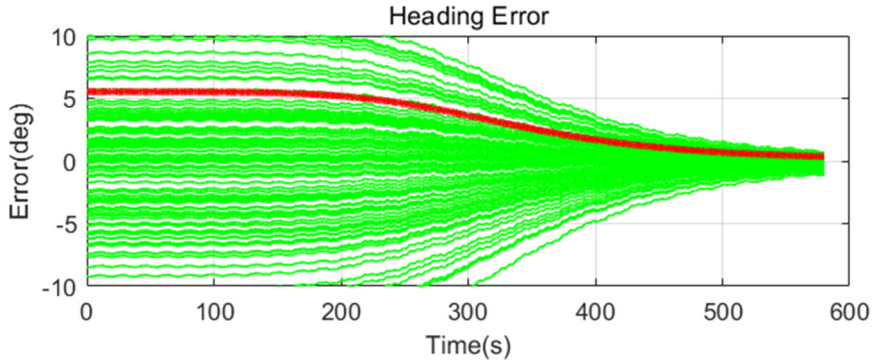

Figure 17. Heading error of alignment with CEKF (Case 1) in large size of movement.

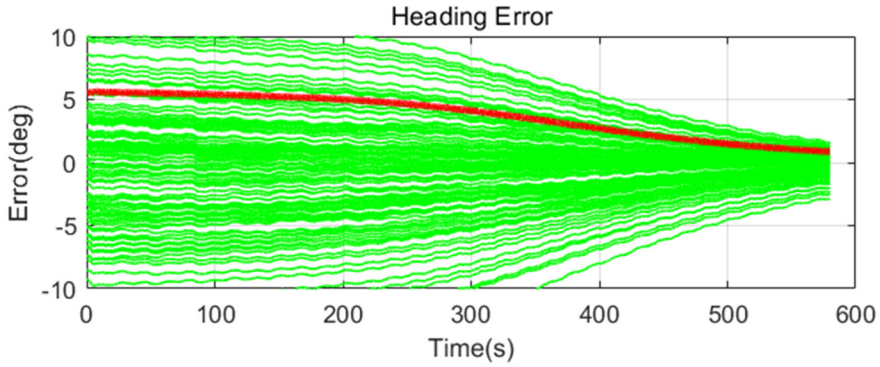

Figure 18. Heading error of alignment with Adaptive EKF (Case 2) in large size of movement.

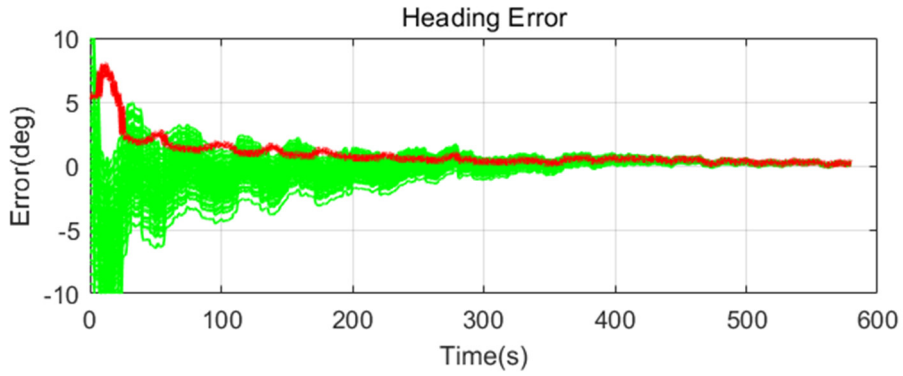

Figure 19. Heading error of alignment with learning-based EKF (Case 3) in large size of movement.

Table 4. Attitude RMS error of alignment in small size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)	Convergence Times (s)
CEKF	0.01914	0.02767	3.88458	443.5
Adaptive EKF	0.00773	0.00698	1.81203	119.5
Learning-based EKF	0.00524	0.00517	1.78866	101.6

Table 5. Last attitude error of alignment in small size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)
CEKF	0.00235	0.00619	0.34707
Adaptive EKF	0.00159	0.00162	0.01579
Learning-based EKF	0.00159	0.00163	0.01252

Table 6. Attitude RMS error of alignment in medium size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)	Convergence Times (s)
CEKF	0.03147	0.03316	3.89140	444.3
Adaptive EKF	0.02995	0.03286	3.66340	435.6
Learning-based EKF	0.03479	0.04540	1.48509	146.7

Table 7. Last attitude error of alignment in medium size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)
CEKF	0.00220	0.02694	0.34814
Adaptive EKF	0.00197	0.02722	0.39209
Learning-based EKF	0.00186	0.02362	0.10507

Table 8. Attitude RMS error of alignment in large size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)	Convergence Times (s)
CEKF	0.09804	0.07607	3.91701	447.6
Adaptive EKF	0.09842	0.07782	4.06675	545.3
Learning-based EKF	0.09978	0.08042	1.54649	112.1

Table 9. Last attitude error of alignment in large size of movement (100 evolutions).

Case	Roll (°)	Pitch (°)	Heading (°)
CEKF	0.00879	0.10111	0.36676
Adaptive EKF	0.01031	0.10460	0.87981
Learning-based EKF	0.00963	0.09532	0.16823

The figures show the heading angle error, which has a relatively large initial error and is the main goal of fine alignment. Moreover, the errors of the horizontal angle, such as roll and pitch, are summarized in the tables.

In Figures 11–13 and Tables 4 and 5, which are the cases where there is almost no disturbance, Case 1 (CEKF) converges very slowly. In contrast, Cases 2 (Adaptive EKF) and 3 (Learning-based EKF) converge quickly, and the last value shows a small error. This shows that the current mooring condition is almost static, thus the smaller is the measurement error covariance matrix, the better is the convergence and accuracy. In Cases 2 (Adaptive EKF) and 3 (Learning-based EKF), the measurement error covariance matrix value was adjusted to be small to adapt to the static environment, while Case 1 (CEKF) was maintained at a relatively large value in consideration of disturbance, thus reducing performance.

In Figures 14–16 and Tables 6 and 7, which are the cases where the size of the disturbance is medium, Cases 1 (CEKF) and 2 (Adaptive EKF) show that the alignment was performed despite the disturbance. However, in Case 1 (CEKF), it can be seen that the alignment was performed slowly by a relatively large measurement error covariance matrix designed for stability against the disturbance. In Case 2 (Adaptive EKF), the measurement error covariance matrix value was adaptively increased for the disturbance, but this matrix value is not considered to be optimized. In Case 3 (Learning-based EKF), since the optimization was performed to reduce the error by learning, the alignment was performed quickly and accurately despite the disturbance.

In Figures 17–19 and Tables 8 and 9, which are the cases where the size of the disturbance is relatively large, the overall result is similar to that of medium size of the movement, except that the performance is slightly worse as the size of the disturbance increase. In Case 2 (Adaptive EKF), the measurement error covariance matrix value was adaptively adjusted according to the size of the disturbance, but the performance was worse than in Case 1 due to the unoptimized value.

7. Conclusions

In this paper, we propose adaptive EKF and learning-based EKF to perform alignment using a Kalman filter in the mooring environment. Since the size of the wave in the mooring environment continuously changes with time and position, the conventional alignment method using a fixed measurement error covariance matrix has limitations. Therefore, as a method of adaptively adjusting the measurement error covariance matrix according to the disturbance, an alignment using an innovation-based adaptive EKF and a CNN-based learning method was applied.

The Monte Carlo simulation was performed by changing the initial errors and sensor errors and dividing the mooring condition into three types: small, medium, and large size of waves.

As a result, in Case 1 (CEKF), the alignment was performed without the filter diverging despite the existence of disturbance. However, in all mooring conditions, alignment was performed very slowly regardless of the disturbance. In Case 2 (Adaptive EKF), a convergence of filter was fast and accurate when there was little disturbance, and it showed adaptive results that alignment was performed even when there was a disturbance. However, alignment was performed relatively slowly depending on the size of the disturbance because optimization was not applied. In Case 3 (Learning-based EKF), when there was little disturbance, alignment was high-speed and accurate, as if the alignment was performed in a static condition. Moreover, even if a disturbance occurs, it shows the best result regardless of the size of disturbance.

In the case of CNN-based alignment, it is necessary to learn various data in changing mooring conditions, but it enables fast and accurate alignment when performing alignment in a mooring. Therefore, it can be usefully applied to systems requiring fast and accurate alignment according to missions, such as surface vessel, ship, and submarine.

Author Contributions: Conceptualization, J.N.L. and C.G.P.; methodology, software, validation, and writing—original draft preparation, J.N.L.; and writing—review and Supervision, C.G.P. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the Hanwha Corporation of South Korea.

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

Appendix A

The system matrix for the Kalman filter is as follows:

$$\mathbf{F}_k = \begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} & \mathbf{F}_{13} & 0_{3\times3} \\ \mathbf{F}_{21} & \mathbf{F}_{22} & 0_{3\times3} & \mathbf{F}_{24} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix}$$

$$\mathbf{F}_{11} = \begin{bmatrix} \frac{v_D}{R_m+h} & 2\rho_D + 2\Omega_D & -\rho_E \\ -2\Omega_D - \rho_D & \frac{v_N \tan L + v_D}{R_t+h} & 2\Omega_N + \rho_N \\ 2\rho_E & -2\Omega_N - 2\rho_N & 0 \end{bmatrix}$$

$$\mathbf{F}_{12} = \begin{bmatrix} 0 & -f_D & f_E \\ f_D & 0 & -f_N \\ -f_E & f_N & 0 \end{bmatrix}$$

$$\mathbf{F}_{13} = C_b^n$$

$$\mathbf{F}_{21} = \begin{bmatrix} 0 & \frac{1}{R_t+h} & 0 \\ -\frac{1}{R_m+h} & 0 & 0 \\ 0 & -\frac{\tan L}{R_t+h} & 0 \end{bmatrix}$$

$$\mathbf{F}_{22} = \begin{bmatrix} 0 & \Omega_D + \rho_D & -\rho_E \\ -\Omega_D - \rho_D & 0 & \Omega_N + \rho_N \\ \rho_E & -\Omega_N - \rho_N & 0 \end{bmatrix}$$

$$\mathbf{F}_{24} = -C_b^n$$

$$0_{3\times3} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{H} = \left[\begin{array}{ccc|c|c|c} 1 & 0 & 0 & & & \\ 0 & 1 & 0 & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0 & 0 & 1 & & & \end{array}\right]$$

where R_m is the meridian radius of the curvature, R_t is the transverse radius of the curvature in prime vertical, L is the latitude, h is the height, C_b^n is the direction cosine matrix that converts from the body frame to the navigation frame, $\begin{bmatrix} f_N & f_E & f_D \end{bmatrix}^T$ is the specific force in the NED navigation frame, $\begin{bmatrix} \Omega_N & \Omega_E & \Omega_D \end{bmatrix}^T$ is the Earth's rotation rate in the NED navigation frame, $\begin{bmatrix} \rho_N & \rho_E & \rho_D \end{bmatrix}^T$ is the transport rate in the NED navigation frame [1], and H is a measurement matrix.

References

1. Titterton, D.H.; Weston, J.L. *Strapdown Inertial Navigation Technology*; Institution of Electronic Engineers: Stevenage, UK, 2004.
2. Siouris, G.M. *Missile Guidance and Control Systems*; Springer: New York, NY, USA, 2004.
3. Chen, Z. From Kalman filters to particle filters, and beyond. *Statistics* **2003**, *182*, 1–69.
4. Lian, X.; Hu, D.; Wu, Y.; Hu, X. Research on SINS alignment algorithm based on FIR filters. *J. Beijing Inst. Technol.* **2007**, *16*, 437–442.
5. Sun, F.; Sun, W. Mooring alignment for marine SINS using the digital filter. *Measurement* **2010**, *43*, 1489–1494. [CrossRef]
6. Gaiffe, T.; Cottreau, Y.; Faussot, N.; Hardy, G.; Simonpietri, P.; Arditty, H. Highly compact fiber optic gyrocompass for applications at depths up to 3000 meters. In Proceedings of the 2000 International Symposium on Underwater Technology (Cat. No. 00EX418), Tokyo, Japan, 23–26 May 2000; pp. 155–160.
7. Gao, W.; Ben, Y.; Zhang, X.; Li, Q.; Fu, F. Rapid fine strapdown INS alignment method under marine mooring condition. *IEEE Trans. Aerosp. Electron. Syst.* **2011**, *47*, 2887–2896. [CrossRef]
8. Sun, F.; Lan, H.; Yu, C.; El-Sheimy, N.; Zhou, G.; Cao, T.; Liu, H. A robust self-alignment method for ship's strapdown INS under mooring conditions. *Sensors* **2013**, *13*, 8103–8139. [CrossRef] [PubMed]
9. Salychev, O.S. *Applied Inertial Navigation: Problems and Solutions*; BMSTU Press: Moscow, Russia, 2004.
10. Salychev, O.S. *MEMS-Based Inertial Navigation: Expectations and Reality*; BMSTU Press: Moscow, Russia, 2012.
11. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
12. Behnke, S. *Hierarchical Neural Networks for Image Interpretation*; Springer: Berlin/Heidelberg, Germany, 2003.
13. Simard, P.Y.; Steinkraus, D.; Platt, J.C. Best practices for convolutional neural networks applied to visual document analysis. In Proceedings of the Seventh International Conference on Document Analysis and Recognition, Edinburgh, UK, 6 August 2003.
14. Coskun, H.; Achilles, F.; DiPietro, R.; Navab, N.; Tombari, F. Long short-term memory kalman filters: Recurrent neural estimators for pose regularization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5524–5532.

15. Brossard, M.; Barrau, A.; Bonnabel, S. AI-IMU dead-reckoning. *arXiv* **2019**, arXiv:1904.06064. [CrossRef]
16. Lee, J.; Bang, H. A robust terrain aided navigation using the Rao-Blackwellized particle filter trained by long short-term memory networks. *Sensors* **2018**, *18*, 2886. [CrossRef] [PubMed]
17. Jazwinski, A.H. *Stochastic Processes and Filtering Theory*; Courier Corporation: Mineola, NY, USA, 2007.
18. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
19. Xie, B.; Zhang, H.; Xue, J. Deep convolutional neural network for mapping smallholder agriculture using high spatial resolution satellite image. *Sensors* **2019**, *19*, 2398. [CrossRef] [PubMed]