aerospace

MDPI

Article

Vision-Based Spacecraft Pose Estimation via a Deep Convolutional Neural Network for Noncooperative Docking Operations

Thaweerath Phisannupawong [1,2], Patcharin Kamsing [1,*], Peerapong Torteeka [3], Sittiporn Channumsin [4], Utane Sawangwit [3], Warunyu Hematulin [1], Tanatthep Jarawan [1], Thanaporn Somjit [1], Soemsak Yooyen [1], Daniel Delahaye [5] and Pisit Boonsrimuang [6]

1 Air-Space Control, Optimization, and Management Laboratory, Department of Aeronautical Engineering, International Academy of Aviation Industry, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand; thaweerath2009@gmail.com (T.P.); thezeroyou@gmail.com (W.H.); Tanatthep001@gmail.com (T.J.); newpai1.12@gmail.com (T.S.); soemsak.yo@kmitl.ac.th (S.Y.)
2 Internship Program, National Astronomical Research Institute of Thailand, ChiangMai 50180, Thailand
3 Research Group, National Astronomical Research Institute of Thailand, ChiangMai 50180, Thailand; peerapong@narit.or.th (P.T.); u.sawangwit@gmail.com (U.S.)
4 Astrodynamics Research Laboratory, Geo-Informatics and Space Technology Development Agency (GISTDA), Chonburi 20230, Thailand; sittiporn@gistda.or.th
5 Ecole Nationale de l'Aviation Civile, 31400 Toulouse, France; daniel.delahaye@enac.fr
6 Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand; kbpisit@kmitl.ac.th
* Correspondence: Patcharin.ka@kmitl.ac.th; Tel.: +66-89-9629998

Received: 8 July 2020; Accepted: 22 August 2020; Published: 30 August 2020

Abstract: The capture of a target spacecraft by a chaser is an on-orbit docking operation that requires an accurate, reliable, and robust object recognition algorithm. Vision-based guided spacecraft relative motion during close-proximity maneuvers has been consecutively applied using dynamic modeling as a spacecraft on-orbit service system. This research constructs a vision-based pose estimation model that performs image processing via a deep convolutional neural network. The pose estimation model was constructed by repurposing a modified pretrained GoogLeNet model with the available Unreal Engine 4 rendered dataset of the Soyuz spacecraft. In the implementation, the convolutional neural network learns from the data samples to create correlations between the images and the spacecraft's six degrees-of-freedom parameters. The experiment has compared an exponential-based loss function and a weighted Euclidean-based loss function. Using the weighted Euclidean-based loss function, the implemented pose estimation model achieved moderately high performance with a position accuracy of 92.53 percent and an error of 1.2 m. The in-attitude prediction accuracy can reach 87.93 percent, and the errors in the three Euler angles do not exceed 7.6 degrees. This research can contribute to spacecraft detection and tracking problems. Although the finished vision-based model is specific to the environment of synthetic dataset, the model could be trained further to address actual docking operations in the future.

Keywords: spacecraft docking operation; on-orbit services; pose estimation; deep convolutional neural network

1. Introduction

In one, docking is defined as "when one incoming spacecraft rendezvous with another spacecraft and flies a controlled collision trajectory in such a manner to align and mesh the interface mechanisms",

and [1] defined docking as an on-orbital service to connect two free-flying man-made space objects. The service should be supported by an accurate, reliable, and robust positioning and orientation (pose) estimation system. Therefore, pose estimation is an essential process in an on-orbit spacecraft docking operation. The position estimation can be obtained by the most well-known cooperative measurement, a Global Positioning System (GPS), while the spacecraft attitude can be measured by an installed Inertial Measurement Unit (IMU). However, these methods are not applicable to non-cooperative targets. Many studies and missions have been performed by focusing on mutually cooperative satellites. However, the demand for non-cooperative satellites may increase in the future. Therefore, determining the attitude of non-cooperative spacecrafts is a challenging technological research problem that can improve spacecraft docking operations [2]. One traditional method, which is based on spacecraft control principles, is to estimate the position and attitude of a spacecraft using the equations of motion, which are a function of time. However, the prediction using a spacecraft equation of motion needs support from the sensor fusion to achieve the highest accuracy of the state estimation algorithm. For non-cooperative spacecraft, a vision-based pose estimator is currently developing for space application with a faster and more powerful computational resource [3].

From this demand, the computer vision field is currently developing as an alternative way for estimating the pose of a spacecraft. A vision-based detection system is a non-cooperative method that takes images of a target object using a camera and then processes them using estimation software. The estimator extracts numerical data from the images based on the constructed relation. When a mathematical model is unavailable, a deep learning algorithm can construct an empirical mathematical model by learning from the data samples. The resulting mathematical model represents the relation between the input image data and the numerical output data. A vision-based estimator needs input and output data samples instead of the exact relationship among the training parameters. The primary precondition of deep learning algorithms is that they need massive amounts of data for training, and the cost of acquiring real spacecraft image data is exceptionally high. Undoubtedly, there identifying the position and attitude while real photos are being taken is problematic. However, pretrained convolutional neural network models, are available that require less data for fine tuning. Many researchers prefer to use public data instead of generating the data themselves because the public data have been well validated and are ready to use. Thus, using public data to construct the estimation algorithm is an excellent choice.

2. Related Works

Currently, deep learning algorithms are widely applied to aerospace information engineering problems. Moreover, applications involving deep Convolutional Neural Network (CNN) architectures have been demonstrated in many studies, for example, processing satellite images to detect forest-fire hazard areas [4], estimating and forecasting air travel demand [5], determining the crack length in aerospace-grade aluminum samples [6], aircraft maintenance and aircraft health management applications [7], and so on; however, their applications in pose estimation is limited compared with aerospace information applications. In this study, we apply deep learning to solve the problems involved in spacecraft pose estimation. Several pose estimation methods have been demonstrated in various fields in prior studies.

The pose estimation of spacecraft has been a problem of considerable interest in various applications. In satellite image registration tasks via push-broom sensors, the variations in registration shifts occur when the attitude of the satellite is changed. Bamber et al. [8] constructed the attitude determination model for a low-orbit satellite by modeling the changes in attitude and rates of the image registration shifting. Before the deep learning algorithm became well known, there was an attempt to apply the computer vision technique via artificial intelligence to estimate the spacecraft poses. Casonato and Palmerini [3] demonstrated the application of artificial intelligence in low-level processing to detect the edges of an Automatic Transfer Vehicle (ATV). After the edge detection, a Hough transformation was employed to identify the basic shape of the vehicle, and the relative position and attitude

parameters were determined using the mathematical formulation of the detected features. The relative position and attitude data were considered real-time navigation data and accomplished with the Clohessy–Whiltshire relative motion equations to estimate the rendezvous trajectory of the ATV to the international space station.

In addition to deep learning algorithms, several methods did not attempt to process the entire image; instead, they utilized the image only for feature detection. Liu et al. [9] applied the edge detection algorithm to extract meaningful features from the images of a cylinder-shaped spacecraft. The ellipses, which are obtained from arc detection, were employed to estimate spacecraft poses by the manipulation of shape, size, and position of features. For a similar method, Aumann [10] developed a pose estimation algorithm using Open source Computer Vision (OpenCV) to detect two longitudinal lines on the sides of a cylinder-shaped object. Then, the author manipulated the positions, directions, and parallelism of the two lines to acquire the pose of the cylindrical object. Sharma et al. [11] employed a Gaussian filter to detect edge lines and cutting point of the spacecraft in 2D images. Later, using principles of spacecraft kinematics, they manipulated the governed points and lines via the efficient perspective-n-point (EPnP) method to solve the 3D parameters from 2D images. Kelsey et al. [12] developed the Vision System for the Autonomous Rendezvous and Docking (VISARD) algorithm by implementing a model-based technique and edge detection for image preprocessing. For pose refinement, the researchers employed Iterative Reweighted Least Squares (IRLS) to estimate the motion of the model. The research also applied a tracking algorithm and used an Extended Kalman Filter (EKF) to predict the model pose. Nevertheless, all the prior studies have some implementation limitations. For example, the edge lines of an object with a complicated shape leads to complexities in the mathematical formulation. As a result that numerous points and lines are detected, in harsh lighting conditions, the feature detection performance may be reduced. Transfer learning is a technique to train the machine learning model using a learning agent, which contains the knowledge of a related task. This accumulated knowledge is theoretically able to accelerate learning with a similar task [13]. Therefore, to reduce the implementation complexities, transfer learning using a pretrained model as a learning agent is preferable for constructing the pose estimation algorithm.

Image regression through deep learning algorithms has been widely applied to pose estimation model construction, and the basic algorithms and mathematical models have been developed in several works. The regression method demonstrated in [14] derived equations for constructing convolutional neural network models. This study used various orientation estimation models for rotation in different dimensions. According to the methodology, the estimation algorithms for viewpoint estimation, surface-normal estimation, and 3D rotation have different rotation parameters and operations. Spacecraft usually behave as 3D rotating objects. Thus, the implementation of spacecraft pose is beyond the determination via Euler angles, as shown in surface-normal estimation. Instead, quaternions are required to represent the object's angle of rotation.

Many public datasets contain images with labeled data that are positioned and oriented in a representation of quaternions. Proença and Gao [15] generated a dataset by using Unreal Rendered Spacecraft On-Orbit (URSO). The tool proposed in that study is a simulator built in Unreal Engine 4 that creates realistic images of the spacecraft surroundings by mimicking the appearance of outer space. The generated images can visualize these outer space conditions for the spacecraft under harsh lighting conditions and can use realistic earth-surface images as the background. They also demonstrated a method that uses a ResNet architecture based on a pretrained CNN model as a backbone. This method achieved high accuracy but also has high complexity. Consequently, it consumes large amounts of computational resources.

Another previous work on spacecraft datasets can be found in [16], which introduced the spacecraft pose network (SPN), a custom CNN whose architecture includes three separate branches. They trained the CNN model using a public dataset, Spacecraft PosE Estimation Dataset (SPEED) and estimated the six degrees of freedom parameters separately. The position was estimated from a 2D bounding box on a target detected by one branch of the CNN model using the Gauss–Newton algorithm. The relative

attitude was determined directly from the other two branches using a hybrid discrete-continuous method. Although the custom convolutional network is beyond the research scope, it provides a significant contribution and performs estimation using separated parameters classification and regression method.

Kendall et al. [17] presented a deep neural model that employed a convolutional neural network to perform pose estimation for a camera. The dataset preparation process considers the pose as parameters relative to the scene and a practical algorithm for pose estimation is developed. The researchers implemented this process using a modified GoogLeNet architecture, which is a CNN model developed by Google. The pose estimation model was initially trained with interior data and subsequently required less outdoor data to train the model. Moreover, it was successful at performing relocalization and pose prediction from the camera images. Although spacecraft attitude estimation must be manipulated using data regarding the spacecraft's position and orientation relative to the camera rather than calculating the pose of the camera itself, the principle is still applicable. Artificial intelligence (AI) studies are concerned with constructing correlations between input and output data.

Mahendran et al. [18] developed a pose estimation algorithm for single objects using a pretrained VGG-M model. Using the Pascal 3D+ dataset, the training process adopted geodesic distance as the loss function. The next year [19], used a ResNet-50 model as the base architecture and demonstrated the use of various loss functions such as simple/naïve, log-Euclidean, geodesic, and probabilistic loss on the same dataset (Pascal 3D+). Another work involving single object detection in [20] applied state-of-the-art AI methods to medical science. They implemented CNNs to estimate six degrees of freedom, including the position and attitude of the human brain, from MRI scans. The pose estimation model was constructed using a ResNet18 model, which reduced the required size of the training dataset. In the training stage, the position loss was the mean-squared error, while the orientation loss was the geodesic distance. In addition, some works have addressed multiple-object pose estimation, such as [21–24]. The contributions from these works could be applied to multiple-object detection in space. For example, in situations where multiple objects need to interact, such a pose estimation system may need to estimate the poses of the various objects individually. Such situations might include space debris collection or vision-based docking operations involving multiple detected objects. Although this research concerns one spacecraft detection, the multiple object detection task could be applied to future works of advanced aerospace image sensing. Due to the lack of data samples for multiple space objects, pose estimation for a single object is more applicable. In the many prior works, different applications have been implemented by different techniques. However, the efficiencies of pretrained models have been evaluated by many research works. Based on that information, the base pretrained model was selected with respect to high efficiency and minimal computational resource consumption.

Another consideration of this research is the formulation of the loss function. Various works have used different formulations to address the terms of position loss and orientation loss. For position loss, most of the works implemented mean squared error [20,24] as the loss function. However, some research was successful using the Euclidean distance [15,17] and a multiplication of the scaling coefficient, as shown in Equation (1).

$$\mathrm{Loss}_{\mathrm{position}} = \beta_x \left\| x_i - x_{gt} \right\|, \tag{1}$$

where x_i is the trial position vector from the layers of the CNN model and x_{gt} is the ground-truth position vector available in the dataset. In many studies [14,18,20], the orientation loss was formulated as the geodesic loss in Equation (2).

$$\mathrm{Loss}_{\mathrm{orientation}} = \beta_q \arccos\left(\left| q_i^T q_{gt} \right| \right), \tag{2}$$

where q_i is the trial quaternion component extracted by the layers of the CNN model and q_{gt} is the ground-truth quaternion component, which is available in the dataset. The total loss defined in Equation (3) is a summation of Equations (1) and (2)

$$\text{Loss}_{\text{total}} = \text{Loss}_{\text{position}} + \text{Loss}_{\text{orientation}}. \quad (3)$$

To minimize the prediction error, the scaling factors β_x and β_q, must be optimized. Using the most straightforward method, β_x and β_q could be fine-tuned by trial and error.

3. Materials and Methods

This research aims to construct an attitude and position estimation model by repurposing a pretrained model. Figure 1 describes the brief information about the construction of the pose estimation model through the training and testing process with the dataset. The details of implementation are described in this section, including dataset preparation, pose estimation algorithm and preprocessing, and construction of the pose estimation algorithm with different loss functions, which consists of both (1) the exponential-based model and (2) the weighted Euclidean-based model.

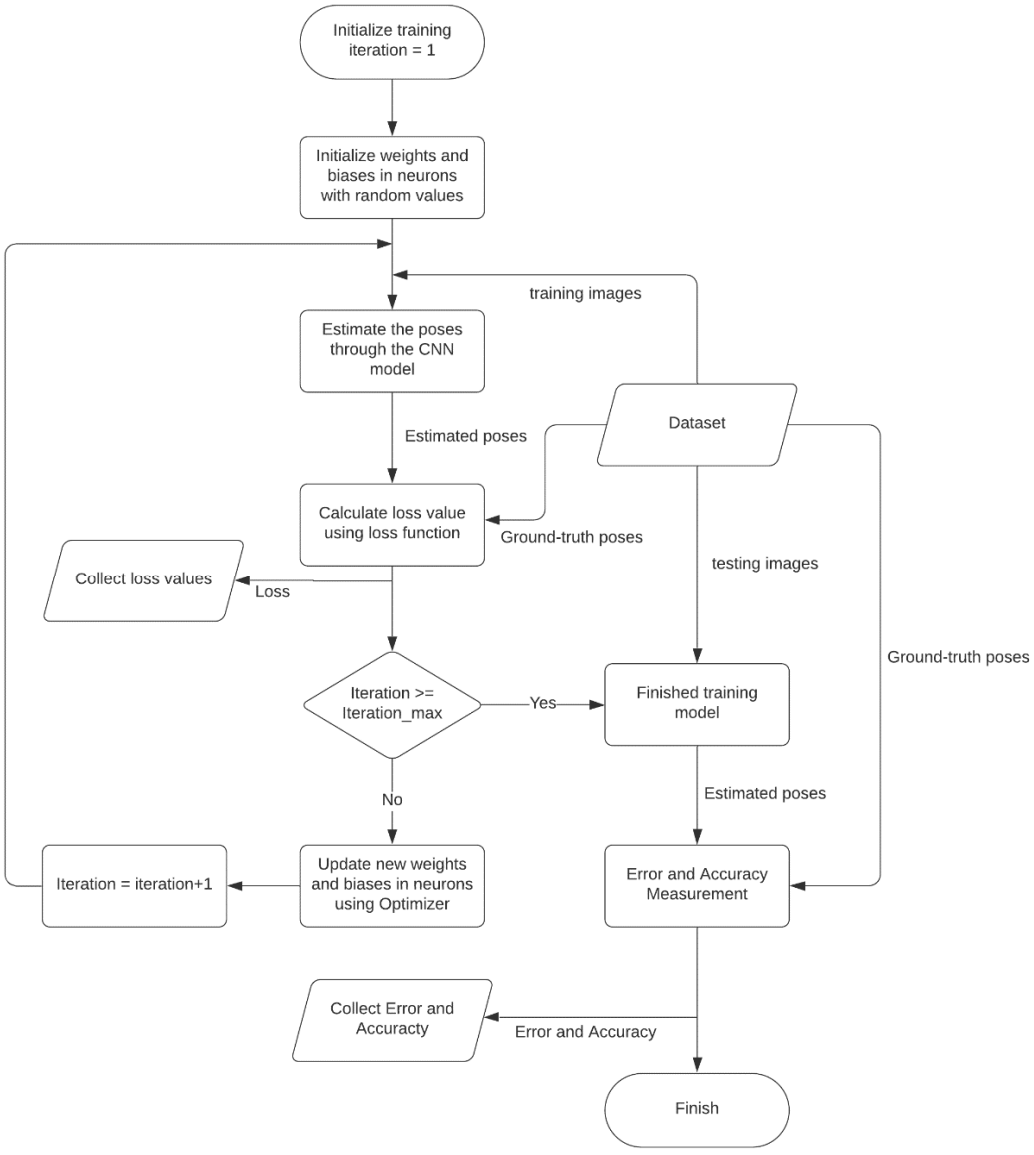


Figure 1. The process of pose estimation model construction.

3.1. Dataset Preparation

Before repurposing a pretrained model, a dataset must be prepared. The dataset has a critical role in the learning process. A dataset generated solely by the researcher might lead to inaccurate and invalid dataset. For this reason, this experiment collected data from public datasets. Proença and Gao [15] provided a publicly available spacecraft dataset suitable for practicing pose estimation with supervised learning algorithms. In the Unreal Engine 4 simulator, the object moving frame was mounted on the spacecraft, and the spacecraft images were acquired by the camera, which was considered the reference frame. Therefore, the relative parameters of position and attitude were described by the position and attitude of the spacecraft with respect to the camera. In the provided dataset (refer to the example in Figure 2), the background image features include the Earth's surface, outer space, and simulated directional light reflections. For entire image processing, the lighting condition is a significant challenge for the feature detection of convolutional layers. As has been proved by Volpe et al. [25], there are differences between the reconstructed shape and the CAD model, which indicates that the lighting condition affects the difficulty in feature identification. By capitalizing on the capabilities of Unreal Engine 4, the images dataset was generated from the simulation of realistic scenes. Information about the relative position and relative attitude and the labeled numerical ground-truth data were collected in the training stage and served as the testing reference. Moreover, the transformation values include relative position and relative attitude, which consists of Euler angles, can be converted to quaternions. The attitude conversion object overcomes the problems of singularity and increases the computational performance.

Figure 2. Sample images from the dataset.

The base model in this research was trained on a dataset for the Soyuz spacecraft. Then, the fine-tuning training process repurposed the pretrained model. The dataset includes 5000 images with position and attitude. Conveniently, the attitude representation in the public dataset was already in the form of quaternions. Each individual image is labeled with the file name, position, and attitude. The dataset supplies both training and testing sets, with 4500 and 500 images, respectively. The data consists of 1280 × 960 pixel .png images labeled with the relative position and the relative attitude in quaternions, represented in a .csv file. The axis notations for spacecraft motion and camera reference position are illustrated in Figure 3. These data were prepared for the training step. Thus, the learning process took the images as input data and the output parameters specified the pose of the spacecraft in the images.

3.2. Pose Estimation Algorithm and Preprocessing

From the state-of-the-art of the convolutional neural network, the model contains the local receptive field, which slides through the whole image to detect the features of the spacecraft. The process converts the image pixels into the numerical data by taking the attributes of pixels in the local receptive field to the corresponding neuron of the first hidden layer. The information from the images passes through the layers of neurons until the poses is determined at the final layer (Figure 4). For the application of spacecraft pose estimation, the trained CNN model contains the direct empirical correlation of the images and estimated poses [26]. In this study, the pretrained convolutional neural model GoogLeNet

was employed as the base architecture for pose estimation. In [17], the authors modified the 23 layers of the CNN in the original GoogLeNet architecture; this model was also adopted in this study instead of the original 22-layer version. GoogLeNet was selected based on two factors: its accuracy and limited computer resource consumption. GoogLeNet has provided accurate results in many prior pose estimation studies and consumes only moderate computer resources.

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

Figure 3. (**a**) Axis notation for a moving spacecraft frame, and (**b**) axis notation of the camera frame.

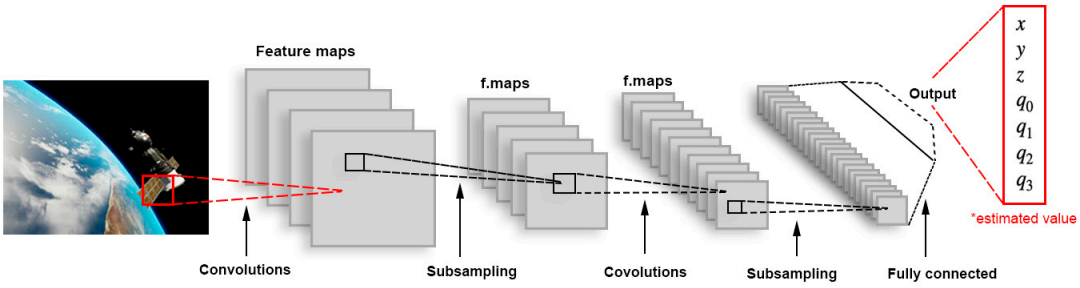


Figure 4. Vision-based pose estimation through the convolutional neural network (CNN) model.

The Soyuz images were used as model input. Nevertheless, the spacecraft images must be reproduced in a suitable form. This reproduction stage is called image preprocessing. The image format GoogLeNet uses as the input consists of 224 × 224 pixel images. Thus, the original image files were transformed into the smallest possible resolution. First, the original images (which were 1280 × 960 pixels) were resized to be four times smaller (to 320 × 240 pixels). To satisfy GoogLeNet's format, these small images were then center-cropped to 224 × 224 pixels (Figure 5). If the images were resized to an even smaller resolution, image details would be lost because the cropping process removes details at the edges of the images. The numerical output for the spacecraft pose is given by a seven-column dataset. In this research, the mathematical pose expression is defined as follows:

$$\text{Poses} = \left[\text{x, y, z}, q_1, q_2, q_3, q_0\right]. \tag{4}$$

This seven-dimensional vector must be sliced into two parts:

$$\text{Position} = [\text{x, y, z}], \tag{5}$$

and

$$\text{Attitude} = \left[q_0, q_1, q_2, q_3\right], \tag{6}$$

where x, y, and z are the magnitude of the relative distance in the X, Y, and Z-axes, respectively, and q_0 represents the real part of the quaternions, while q_1, q_2, and q_3 are the components in the vector part.

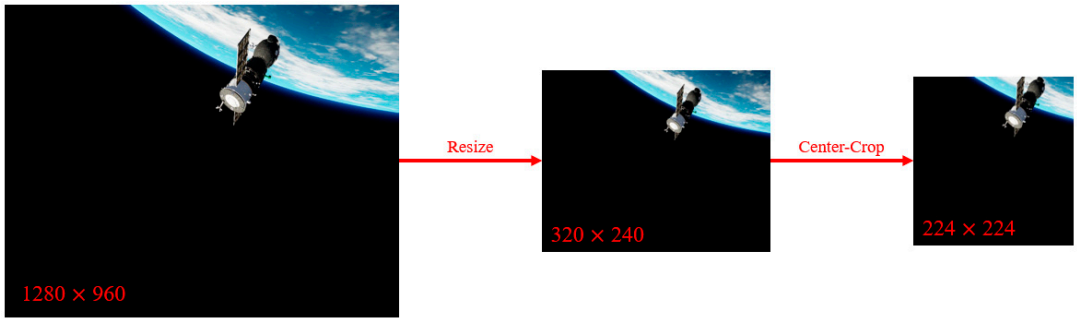


Figure 5. Resizing the image to fit the study format.

3.3. Exponential-Based Pose Estimation

3.3.1. Exponential Loss Function

Another consideration for pose estimation algorithm development is to choose a suitable mathematical expression for the loss function. The loss function is the function that measures the difference between the estimated value and the ground-truth value of the positions and attitude during the training stage, as shown in Figure 1. In this experiment, the Adam optimizer was adopted to identify and develop the most suitable weights for the CNN neurons during iteration to achieve the lowest loss value. The experiment applied Equation (3), which is the combination of positional and orientation loss with the Adam optimizer. Equation (3) consists of two terms: position loss and orientation loss. Equation (1) was considered as the term for position loss because Euclidean distance is suitable for measuring the magnitude of the distance between two objects.

It was difficult to find a suitable mathematical model for the orientation loss function. The first trial was performed using Equation (2) as the orientation term. There is a mathematical conflict with the loss function because the loss value becomes infinite when $\left|q_i^T q_{gt}\right| = 1$. To avoid this problem, the experiment was next implemented using Equation (7), which is the cosine of geodesic loss. Therefore, the loss function is completely algebraic and does not involve trigonometry expressions.

$$\text{Loss}_{\text{orientation}} = \beta_q\left(1 - \left|q_i^T q_{gt}\right|\right). \tag{7}$$

However, the second implementation with Equation (7) resulted in the loss value diverging to infinity. Therefore, the third trial used the natural exponent of the cosine of geodesic loss, which is expressed in Equation (8)

$$\text{Loss}_{\text{orientation}} = \beta_q \exp\left(1 - \left|q_i^T q_{gt}\right|\right). \tag{8}$$

For the third trial, the total loss value tended to converge to zero. Thus, this function can be taken as the orientation term in Equation (3). Then, the mathematical model of the total loss is defined as follows:

$$\text{Loss}_{\text{total}} = \beta_x\left\|x_i - x_{gt}\right\| + \beta_q \exp\left(1 - \left|q_i^T q_{gt}\right|\right). \tag{9}$$

The result of this mathematical expression was minimized using the Adam optimizer. In each iteration, the optimizer changed the weights and biases according to the descending gradient during artificial neural model development. The most acceptable values of each scaling coefficient, β_x and β_q, can be found by trial and error. The values of these scaling coefficients are dataset dependent; therefore, models trained on different datasets may have different scaling coefficients.

3.3.2. Experimental Methods on the Exponential Loss Function

From the mathematical expression in Equation (9), the fine-tuning process of β_x and β_q was performed by constructing the model on 3000 repeated training iterations with a learning rate of 0.001.

Using this number of iterations, the model loss remained approximately unchanged after 2500 iterations. The slope at higher iterations is approximately constant in the first observation. Therefore, the number of iteration loops was set to 3000 in this experiment. After the training stage, the accuracy of the trained model was measured in the testing stage. If the test result is unacceptable, the model was reconstructed by changing the values of the scaling coefficients. In the worst case, the model would need to be constructed by modifying the mathematical model of the loss functions. The model implementations in this section were conducted on a laptop equipped with an NVIDIA GeForce RTX 2080 GPU with 8 GB of memory. Clearly, the computer resources are severely restricted. Consequently, the computation time per iteration is longer than they would be on a higher-performance processor.

The accuracy of the model was measured during the testing stage. Errors measured during the testing stage reflect model performance. Generally, the error involves a comparison of the estimated value and the ground-truth value. To avoid confusion between the concepts of loss and error, in this study, loss is the difference between the estimated value and the ground-truth value during the training stage, while error is the difference between the estimated value and the ground-truth value at the testing stage. In addition, the loss is measured from the training set, but the error is measured from the testing set.

As a result that the artificial neural model is constructed during the training process, its performance is usually optimized when the model is built using the most suitable mathematical expressions. Therefore, all the mathematical functions and scaling coefficients must be fine-tuned. This experiment focuses on fine-tuning the scaling coefficients β_x and β_q by trial and error. For a deep neural model, the performance of the model can be determined only at the testing stage. Thus, the testing was done alongside the process of fine-tuning the scaling coefficients.

A model trained with different scaling coefficients always yields different levels of performance. Therefore, the goal is to determine the most suitable scaling coefficients. When a test result is unsatisfactory, the model must be reconstructed under the new training conditions until the error is acceptable. This research considered the error as the magnitude of the distance vector to reflect the model performance. For the positions and attitudes, the distances between the estimated values and the ground-truth values were calculated with the Euclidean distance. Equations (10) and (11) describe the Euclidean distance, which is the norm of the position vector and orientation in the quaternion components.

$$\text{Error}_{\text{position}} = \left\| x_i - x_{gt} \right\|, \tag{10}$$

and similarly,

$$\text{Error}_{\text{orientation}} = \left\| q_i - q_{gt} \right\|. \tag{11}$$

By comparing the behaviors of Equations (1) and (8), we found that when the domain of the exponential function is less than one, the value of the function is minimal compared to the value of Equation (1). Therefore, in this experiment, the scaling coefficient of position β_x were fixed to one and the scaling coefficient of orientation β_q was varied. The experiment started testing at low values of β_q and then increased to higher values by multiplying it by 100 to find the most acceptable range for β_q. Subsequently, this experiment investigated the artificial neural model with Equation (9), set maximum number of iterations as 3000, and set β_x as 1 to control the experimental environment. The values of the independent variable β_q were 1000, 100,000, and 10,000,000 in the training stage. From the results under those training conditions, we assume that there might be some value that can reduce the orientation error. Then, the experiment further adopted a random value between 100,000 and 10,000,000 when training the model, and the best training value for β_q was 2,547,500. The testing stage acquired the results from the 500 test samples. Then, a representation of those values must be selected. In this study, the median value was selected among the 500 testing sample results under the assumption that some overestimated values and underestimated values likely exist that would cause a change in the distribution. Using the median ignores these outlier values and expresses only the middle position of all the results.

Next, the most acceptable model was selected. From Equations (10) and (11), the position error is calculated in the unit of meters, while the orientation error implicitly appears unclear under the representation of quaternions. Therefore, to visualize the result using a more familiar representation, the results of this model were converted to Euler angles. From [27], the formulas to transform the quaternions in the aerospace sequence to Euler angles are

$$\phi = \tan^{-1}\left[\frac{2\left(q_0q_1 + q_2q_3\right)}{2\left(q_0^2 + q_3^2\right) - 1}\right], \tag{12}$$

$$\theta = \sin^{-1}\left[2\left(q_0q_2 - q_1q_3\right)\right], \tag{13}$$

and

$$\psi = \tan^{-1}\left[\frac{2\left(q_0q_3 + q_1q_2\right)}{2\left(q_0^2 + q_1^2\right) - 1}\right], \tag{14}$$

where ψ is the heading or yaw angle around the Z-axis, θ is the elevation or pitch angle around the Y-axis, and φ is the bank or roll angle around the X-axis. Then, the orientation error was separated into 3 parameters defined as the absolute errors of angles. Similar to the example in Equation (15), the angle for Y-axis, and Z-axis are calculated by replacing ϕ with θ, and φ, respectively.

$$\text{Error}_{\text{angle (x-axis)}} = \left|\phi_i - \phi_{gt}\right|, \tag{15}$$

For implementation in spacecraft dynamics control, a quaternion representation is more general because there are no conflicts with the rotation sequence or with trigonometry functions. Moreover, the quaternion representation is completely algebraic, which improves the computational performance. However, the quaternion representation is quite abstract, while Euler angles are more physically comprehensible. When comparing Euler angles, they must follow the same rotation sequence. In this experiment, the Euler angles follow the aerospace sequence, making Equations (15) valid in all axes.

The experiment identified the most acceptable result from all the results of the approximate fine-tuning to convert the single attitude error measurement to errors with a Euler angle representation. The predicted quaternions and ground-truth quaternions were converted to the Euler angle representation through Equations (12) and (13). After the individual manipulations of Equations (15) in all axes, the median of the three angles from 500 samples was selected to represent the orientation error from the Euler angle representations. Another performance indicator is accuracy; the error percentage is applied to determine an accuracy measurement for each model. Therefore, the measurements for the model's position and orientation accuracy are given by Equations (16) and (17), respectively. Similar to the other performance indicators, we adopted the median to represent the overall accuracy.

$$\text{Accuracy}_{\text{position}} = \left(1 - \frac{\left\|x_i - x_{gt}\right\|}{\left\|x_{gt}\right\|}\right) \times 100, \tag{16}$$

$$\text{Accuracy}_{\text{orientation}} = \left(1 - \frac{\left\|q_i - q_{gt}\right\|}{\left\|q_{gt}\right\|}\right) \times 100. \tag{17}$$

3.4. Weighted Euclidean-Based Pose Estimation

3.4.1. Weighted Euclidean Loss Function

Next, we formulated an alternative mathematical model of the loss function. The experiment for the weighted Euclidean loss function relied on the hypothesis that a significant error may occur when the model is constructed on the exponential function. Moreover, the previous loss function

did not consider the characteristics of GoogLeNet. Similar to the original version, the modified version of GoogLeNet has three regressors. Thus, the new loss function needs to be constructed by considering the architecture. Equation (9) reflects the equal importance of the prediction values from the three regressors. However, the testing stage generally considers only the final regressor. Thus, the importance of the other two prediction results needs to be scaled down. The regressor coefficients were multiplied by the loss function from each n regression layer to reflect the importance of the prediction values. The resulting weighted Euclidean loss function considers the position loss, as shown in Equations (18), where μ is the regressor coefficient.

$$\text{Loss}_{\text{position,n}} = \mu_{x,n}\beta_x\left\|x_{i,n} - x_{gt}\right\|. \tag{18}$$

To investigate the hypothesis, this experiment employed a simpler mathematical model of the orientation loss function by implementing Euclidean distance for the orientation loss based on the assumption that position and orientation parameters behave identically from a data processing perspective. Therefore, similar to the regressor coefficients, the orientation loss functions are defined in Equations (19).

$$\text{Loss}_{\text{orientation,n}} = \mu_{q,n}\beta_q\left\|q_{i,n} - q_{gt}\right\|. \tag{19}$$

This experiment controlled the importance of position loss from the first regressor. The second regressor is 30% of the position loss from the final regressor, the orientation loss from the first regressor, and one-third of the orientation loss from the final regressor. The controlled values of the regressor coefficients are listed in Table 1.

Table 1. Controlled regressor coefficients.

Regressor Coefficients	Controlled Values
$\mu_{x,1}$	0.3
$\mu_{x,2}$	0.3
$\mu_{x,3}$	1
$\mu_{q,1}$	1/3
$\mu_{q,2}$	1/3
$\mu_{q,3}$	1

The total loss function is the sum of all the position loss functions and orientation loss functions in each layer, n, as defined in Equation (20). Then, the final total loss equation can be derived by substituting the regressor coefficients in Table 1.

$$\text{Loss}_{total} = \sum_{n=1}^{3}\text{Loss}_{\text{position,n}} + \sum_{n=1}^{3}\text{Loss}_{\text{orientation,n}}. \tag{20}$$

Equation (20) can be written in a more detailed form as

$$\begin{aligned}\text{Loss}_{\text{total}} = {} & \mu_{x,1}\beta_x\left\|x_{i,1} - x_{gt}\right\| + \mu_{x,2}\beta_x\left\|x_{i,2} - x_{gt}\right\| + \mu_{x,3}\beta_x\left\|x_{i,3} - x_{gt}\right\| + \\ & \mu_{q,1}\beta_q\left\|q_{i,1} - q_{gt}\right\| + \mu_{q,2}\beta_q\left\|q_{i,2} - q_{gt}\right\| + \mu_{q,3}\beta_q\left\|q_{i,3} - q_{gt}\right\|.\end{aligned} \tag{21}$$

Then, substitute the controlled regressor coefficients and take the common factor:

$$\begin{aligned}\text{Loss}_{\text{total}} = {} & \beta_x\left[0.3\left\|x_{i,1} - x_{gt}\right\| + 0.3\left\|x_{i,2} - x_{gt}\right\| + \left\|x_{i,3} - x_{gt}\right\|\right] + \\ & \beta_q\left[\tfrac{1}{3}\left\|q_{i,1} - q_{gt}\right\| + \tfrac{1}{3}\left\|q_{i,2} - q_{gt}\right\| + \left\|q_{i,3} - q_{gt}\right\|\right].\end{aligned} \tag{22}$$

The model was trained for 30,000 iterations at a learning rate of 0.001 with the Adam optimizer to optimize the weights and bias values. Similar to the previous experiment, the scaling coefficients

β_x and β_q must be fine-tuned to achieve the highest accuracy. The model is trained ten times the number of iterations in the previous experiment because the computational performance increased. With the support of the National Astronomical Research Institute (NARIT), the training algorithm was submitted to and executed on a high-performance computational unit. Chalawan, a high-performance computer (HPC), was built for research on data processing, simulation, and optimization, which are crucial for astronomy and astrophysics. The training stage for this study was implemented on the GPU compute node. After that, the pose estimation models were tested on the CPU head node. With more powerful processors in that system, the pretrained model was trained for 30,000 iterations. The iteration loss becomes approximately constant just before the 30,000th iteration in the first trial with the scaling coefficients β_x and β_q set to 1 and 300, respectively.

3.4.2. Experimental Methods on the Weighted Euclidean Loss Function

In this experiment, fine-tuning was performed by training the model with the loss function in Equation (22) for 30,000 iterations. The scaling coefficient of position, β_x, was fixed at 1 while the scaling coefficient of orientation, β_q, was varied, similar to the previous experiment. However, the error measurement was slightly changed; the positional calculation was retained from Equation (10), but the prediction error for orientation was changed to Equation (23) because the norm of quaternions has a disadvantage for visualization. This alternative error measurement was slightly modified the scoring from the Kelvins Pose Estimation Challenge 2019 into Equation (23), which is twice the dot product between two quaternions unit vectors.

$$\text{Error}_{\text{orientation}} = 2\arccos\left(\frac{q_i}{\|q_i\|}^T \frac{q_{gt}}{\|q_{gt}\|}\right). \tag{23}$$

The value of the orientation scaling coefficient β_q was varied. The fine-tuning process started from 300 and increased by 300 each time (to 600, 900, 1200, 1500, and 1800). The model error when β_q is equal to 1500 was lower than that of the values. We assumed that there might be some value of β_q that resulted in lower error between 1500 and 1800. Therefore, we conducted an experiment with β_q set to 1650 because that value is in the midpoint between 1500 and 1800. The iteration loss during the training process of each trial model was plotted to inspect the learning process behavior due to the variation of the orientation scaling coefficient β_q.

The result that achieved the smallest error in attitude estimation was considered the most acceptable result because there are many difficulties involved in reducing the prediction error for orientation. In addition, achieving good attitude estimation was considered to be more important than position estimation. The position error is represented in units of meters, while a single hypothetical parameter represents the orientation error. Based on the visualization advantages of Euler angles, the result from the most efficient model was transformed into the error of three Euler angles. The estimated quaternions and the ground-truth quaternions in 500 test samples were transformed into three angles by the aerospace Euler sequence in Equations (12)–(14). Then, angles in the same sequence can be compared individually for each sample using Equations (15) in all axes. Additionally, the representative errors in the three Euler angles were taken as the median among the 500 test samples in the same way as the previous procedure. The formulas to obtain the position and orientation accuracies are described in Equations (16) and (17), respectively. The representative value of all the results was taken as the median among the 500 testing samples.

Further experiments were then conducted using the most efficient scaling factors in the same training conditions (except the number of iterations). The number of iterations was increased to five times the former number (which is 150,000). We assume that increasing the number of iterations might result in better model performance.

4. Results and Discussion

Based on the research methodology, the results include the fine-tuning errors, the position errors in meters, the orientation errors in Euler angle representations, and the accuracy of both position and orientation. Moreover, the comparison between the exponential-based model [27] and the weighted Euclidean-based model includes errors, accuracy, and the experimental condition, allowing visualizations of the impacts of all the factors during the model learning process.

4.1. Exponential-Based Pose Estimation

4.1.1. Results of the Exponential-Based Pose Estimation Model

The exponential-based model was constructed based on a modified version of GoogLeNet. This model was developed in Python with 4500 Unreal Engine 4-rendered training data of the Soyuz spacecraft. The model was trained with the exponential of the cosine of the geodesic loss function in Equation (9) for 3000 iterations on a laptop equipped with an RTX 2080 8 GB. The Adam optimizer was used to minimize the loss value during the learning process. The learning behavior, as illustrated in Figure 6, was plotted by monitoring the total loss at each iteration loop during the training stage until the model reached the maximum number of iterations.

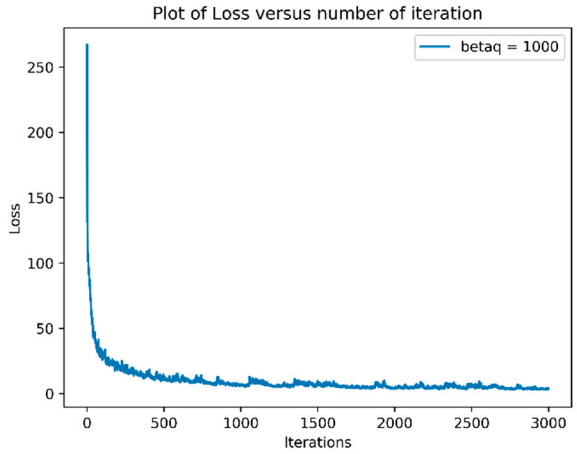


Figure 6. The relationship of the total loss of the exponential-based model at different iteration loops during the training process.

The fine-tuning process was performed by varying the orientation scaling coefficients to find the most acceptable result. Table 2 was recorded when the trained models in each condition were tested on 500 image samples. The position and quaternion outputs were individually compared to the ground-truth values with Equations (10) and (11), respectively. In addition, the results in Table 2 are plotted in Figure 7. After fine-tuning operation, the most acceptable result occurred when the position scaling coefficient β_x was set to 1 and the orientation scaling coefficient β_q was set to 100,000. After testing on the 500 test-image samples, the position and orientation estimation errors of this model are tabulated in Tables 3 and 4, while the prediction accuracy is shown in Table 5.

Table 2. Fine-tuning error of the exponential-based model with various orientation scaling coefficients, β_q.

β_x	β_q	Med. Position Error (m)	Med. Orientation Error
1	1000	0.82710419	0.92926225
1	100,000	0.9316442	0.90732103
1	2,547,500	1.1937147	0.94482738
1	10,000,000	1.3219991	1.11328518

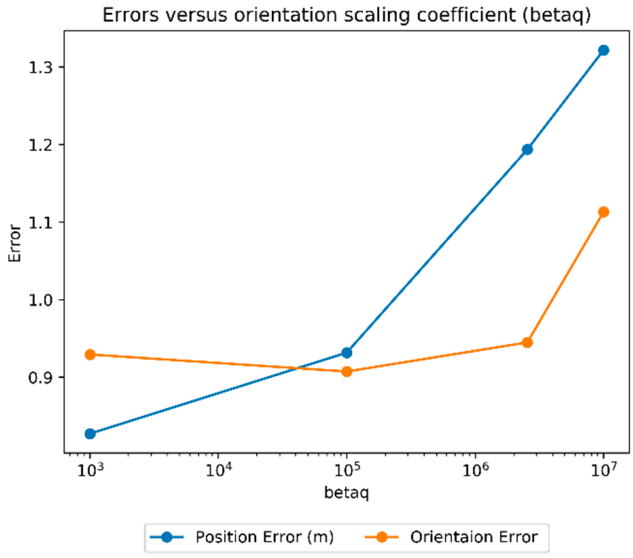


Figure 7. Experimental results from varying the orientation scaling coefficient in the exponential-based model.

Table 3. Fine-tuning errors of the exponential-based model.

β_x	β_q	Med. Position Error (m)	Med. Orientation Error
1	100,000	0.9316442	0.90732103

Table 4. Euler angle errors of the exponential-based model.

β_x	β_q	Med. Φ Error (deg)	Med. θ Error (deg)	Med. Ψ Error (deg)
1	100,000	91.74883037	30.10585343	84.42547416

Table 5. Accuracy of the exponential-based model.

β_x	β_q	Med. Position Accuracy	Med. Orientation Accuracy
1	100,000	93.92803771%	22.73989517%

4.1.2. Discussion on the Exponential-Based Pose Estimation Model

The most acceptable results in Tables 3–5 show that significant orientation error occurs in both the quaternions and the Euler angles systems from [27]. The model achieves a high performance for position estimation but performs poorly for attitude estimation. The large error in orientation estimation might be caused by the minimization of the orientation loss during the training process. Under the applied research methodology, the model was trained for 3000 iterations because the value of the total loss was approximately constant after approximately 2500 iterations as revealed by monitoring the numerical training loss. Figure 6 shows that the total loss value decreased very quickly. Moreover, the orientation error measurement in the fine-tuning process suffered from poor visualization—it looked small compared to the position error but expanded after being converted to the aerospace Euler angle sequence.

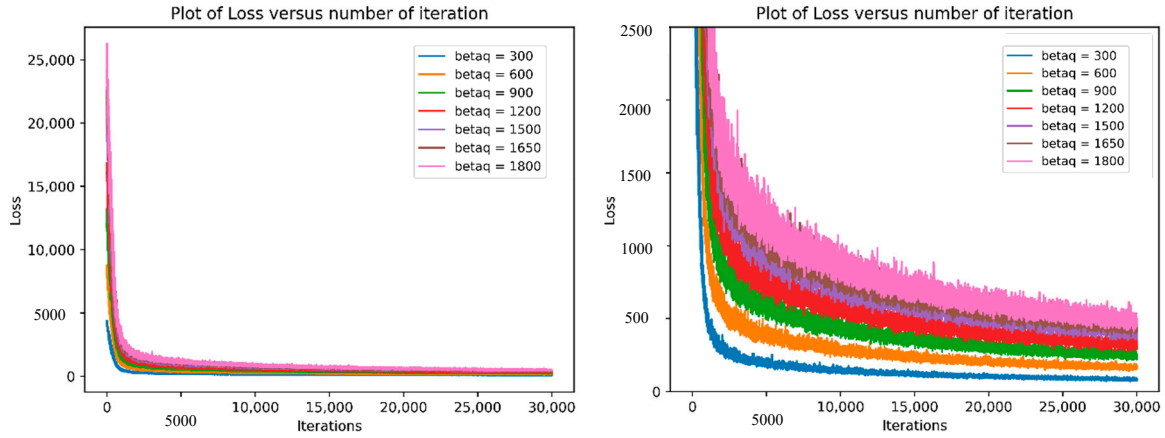
According to the cosine of the geodesic loss function, the exponent of Equation (9) must be minimized to zero. Therefore, the result of Equation (9) is expected to be equal to, β_q, but the Adam optimizer always reduces the iteration loss by expecting the loss to be equal to 0 at the global minimum in the weight spaces. From this mathematical mistake, the model might be trained by ignoring the minimization in the orientation loss, causing the loss to converge to a constant value very quickly. Moreover, the accuracy of the position prediction is extremely high compared to the

orientation estimation accuracy because the importance of position learning was scaled up to reduce the importance orientation learning. This event proved that a mistake in the formulation might be the cause of the failure in orientation estimation. In the mathematical formulation of the loss function, it is possible to train the model with the correction in the orientation loss function by subtracting 1 from the exponential term of the orientation loss and taking the result as the absolute value. This corrected loss function may solve the conflict of the exponential-based loss function and satisfy the principle of the Adam optimizer. After the correction, the model learning behavior might learn through the minimization of both the position and orientation loss values. However, this experiment did not support this assumption. Therefore, it is the hypothesis for further research on pose estimation.

4.2. Weighted Euclidean-Based Pose Estimation

4.2.1. Results of Weighted Euclidean-Based Pose Estimation Model

The final model construction was based on a modified version of GoogLeNet and implemented in a Python environment with 4500 generated images in the training dataset of the Soyuz spacecraft. The model was trained for 30,000 iterations using Equation (22) as the loss function and optimized by the Adam optimizer. The model was trained on the GPUs of the compute nodes at the National Astronomical Research Institute (NARIT). The experiment on weighted Euclidean loss function was studied by evaluating the differences in the learning behavior and the performance of each model under different orientation scaling coefficients. The learning behavior is considered as the history of the iteration loss during the training process. Figure 8a shows the overall learning behavior of each model with different orientation scaling coefficients, while the illustration in Figure 8b focuses on the events after the loss was reduced to less than 2500 to more closely inspect the learning behavior. In the fine-tuning stage, the position error and the orientation error of models that were constructed with different orientation scaling coefficients were manipulated with Equations (10) and (23). The results of all the models in the fine-tuning stage are tabulated in Table 6 and plotted in Figure 9. According to the fine-tuning method, the most efficient scaling coefficients occur when the position scaling coefficient β_x is set to 1, and the orientation scaling coefficient β_q is set to 1500. After testing on the 500 test-set samples, the model performance is reflected by the values shown in Tables 7–9.



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

Figure 8. (**a**) Behavior of the total loss of the weighted Euclidean-based model at different iterations during the training process; (**b**) behavior of the total loss of the weighted Euclidean-based model during different iterations after the iteration loss decreased below 2500.

Table 6. Fine-tuning error of the weighted Euclidean-based model with various orientation scaling coefficients β_q.

β_x	β_q	Med. Position Error (m)	Med. Orientation Error (deg)
1	300	1.11338711	16.81335914
1	600	1.20315672	16.30207395
1	900	1.28458039	16.59338138
1	1200	1.28057494	16.29479161
1	1500	1.41480502	15.9104448
1	1650	1.32794428	17.39898174
1	1800	1.37533052	16.90322023

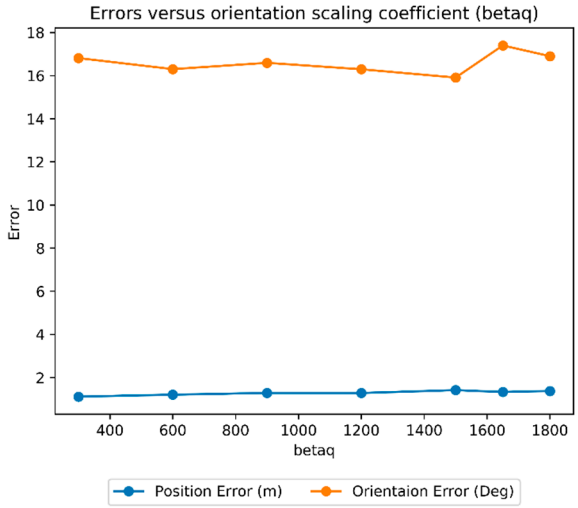


Figure 9. Error plot showing the variations under different orientation scaling coefficient β_q values for the weighted Euclidean-based model.

Table 7. Fine-tuning errors of the weighted Euclidean-based model (30,000 iterations).

β_x	β_q	Med. Position Error (m)	Med. Orientation Error (deg)
1	1500	1.41480502	15.9104448

Table 8. Euler angle errors of the weighted Euclidean-based model (30,000 iterations).

β_x	β_q	Med. Φ Error (deg)	Med. θ Error (deg)	Med. Ψ Error (deg)
1	1500	8.51153642	8.75737224	8.42390433

Table 9. Accuracy of the weighted Euclidean-based model (30,000 iterations).

β_x	β_q	Med. Position Accuracy	Med. Orientation Accuracy
1	1500	91.55210137%	85.84760674%

Subsequent experiments were conducted with that same loss function and the most efficient scaling coefficients (β_x = 1 and β_q = 1500) by increasing the maximum number of iterations to 150,000. The learning behavior is shown in Figure 10, and the performance indicators are tabulated in Tables 10–12.

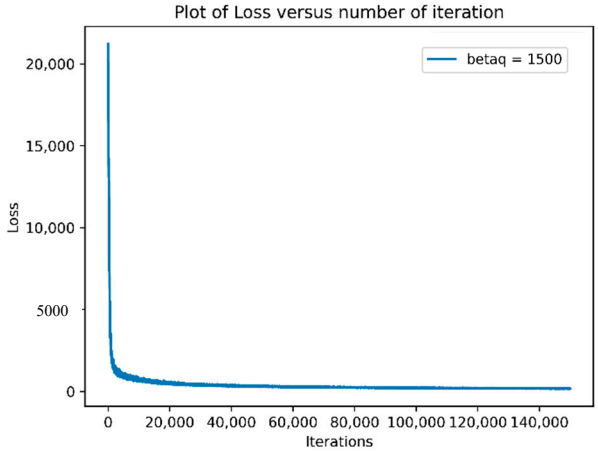


Figure 10. Behavior of the total model loss at various iterations during the training process (trained on weighted Euclidean loss with 150,000 iterations).

Table 10. Fine-tuning errors of the weighted Euclidean-based model (150,000 iterations).

β_x	β_q	Med. Position Error (m)	Med. Orientation Error (deg)
1	1500	1.19148937	13.70432697

Table 11. Euler angle errors of the weighted Euclidean-based model (150,000 iterations).

β_x	β_q	Med. Φ Error (deg)	Med. θ Error (deg)	Med. Ψ Error (deg)
1	1500	7.14243874	7.53373215	6.73018005

Table 12. Accuracy of the weighted Euclidean-based model (150,000 iterations).

β_x	β_q	Med. Position Accuracy	Med. Orientation Accuracy
1	1500	92.53104042%	87.9304795%

4.2.2. Discussion of the Weighted Euclidean-Based Pose Estimation Model

According to Figure 8a, the iteration loss of the model with the larger orientation scaling coefficient β_q began at a higher loss value compared to the model with the smaller orientation scaling coefficients β_q. The iteration loss of the model converged to an approximately constant value. Each model was converted to a different constant because the limit values were scaled up by the corresponding orientation scaling coefficients, β_q. This difference is visualized clearly in Figure 8b. When the learning behavior of models is compared to a vibrated function, the amplitude increases due to the higher orientation scaling coefficient, β_q as in the iteration. Moreover, from Figure 10, the iteration loss of the model with the orientation scaling coefficient equal to 1500 was further reduced after 30,000 iterations. It has been proven that the iteration loss can be further reduced and approach a constant value at approximately the 140,000th iteration. Therefore, the error can be reduced further by increasing the number of iterations.

The results show that this study successfully constructed a moderately high-performance pose estimation model. Although the accuracy results show that it has superior accuracy for predicting the position and moderately high accuracy for estimating the orientation, the errors in position and Euler angles are still prohibitively large to use in an actual docking operation. In advanced computational algorithms such as CNNs, the prediction performance depends strongly on the architecture, the dataset, and the training method. Thus, a pose estimation model constructed from a more complex pretrained model may result in higher accuracy. According to the methodology, these factors were of concern in [27], which constructed a model under limitations of the GPU, an RTX 2080, on a laptop. Therefore,

the modified version of GoogLeNet was chosen because the model achieved successful predictions in [17] while having only low computational resource requirements. Consequently, the pretrained model selected for this study is a controlling factor; the obtained accuracy may be the highest accuracy that this model can achieve for spacecraft position and attitude estimation.

However, there are some ideas for increasing prediction performance. For instance, if the pose estimation model was constructed with a more complex pretrained model as in [15], it might result in higher accuracy and more reliable results. Alternatively, the auxiliary algorithm, such as the line and point manipulations of the detected features, could be applied to construct the pose estimation algorithm [11] as an auxiliary process to the main CNN. Moreover, if a model was specifically constructed for spacecraft pose estimation, such as the SPN from [16], it could result in a more accurate and reliable performance.

4.3. Model Comparisons: The Impact of Training Conditions

There are some significant differences between the exponential-based and weighted Euclidean-based pose estimation models that lead to the different levels of performance. These differences include the loss function, the number of iterations, and the available computational resources. Thus, the comparison includes all the construction factors and the performance indicators for the two construction methods. For the weighted Euclidean-based model, the results of the further trained model are also included in this comparison. Comparative illustrations of the numerical results are shown in Figure 11. From the methodology, the critical differences between the weighted Euclidean-based model and the exponential-based model include the loss function, the number of iterations, and the computational resources. First, the loss function for the weighted Euclidean-based pose estimation model relates to the architecture of the CNN. The architecture of the modified version is slightly changed from the original version, and the changes exclude the number of regressors. There are three regressors in the architecture of the selected neural network. The exponential-based model was not constructed based on the unequal importance of the three regressors. Conversely, the weighted Euclidean-based pose estimation model was developed with a loss function that included the regressor coefficients to mathematically indicate the different importances of the three regressors. In the general testing stage, the prediction occurs at the third regressor; thus, the third regressor is considered to be the main regressor. Moreover, it is impossible to obtain accurate results from the first and second regressors. Therefore, the repurposing of GoogLeNet without considering the multiple regressors hypothetically leads to low prediction performance. The experiment also indicates that the exponential-based loss function is unable to reduce the prediction errors.

The exponential-based model failed for attitude estimation because of a mathematical conflict. Therefore, the impact of the number of iterations can be observed on the two weighted Euclidean-based models. The weighted Euclidean-based model trained for 150,000 iterations resulted in higher performance for both position and attitude prediction than did the model trained for only 30,000 iterations: the prediction errors were reduced and the overall accuracy was increased. Computational resources played no significant role in the model performance. However, the availability of higher-performance computational resources was convenient during the implementation.

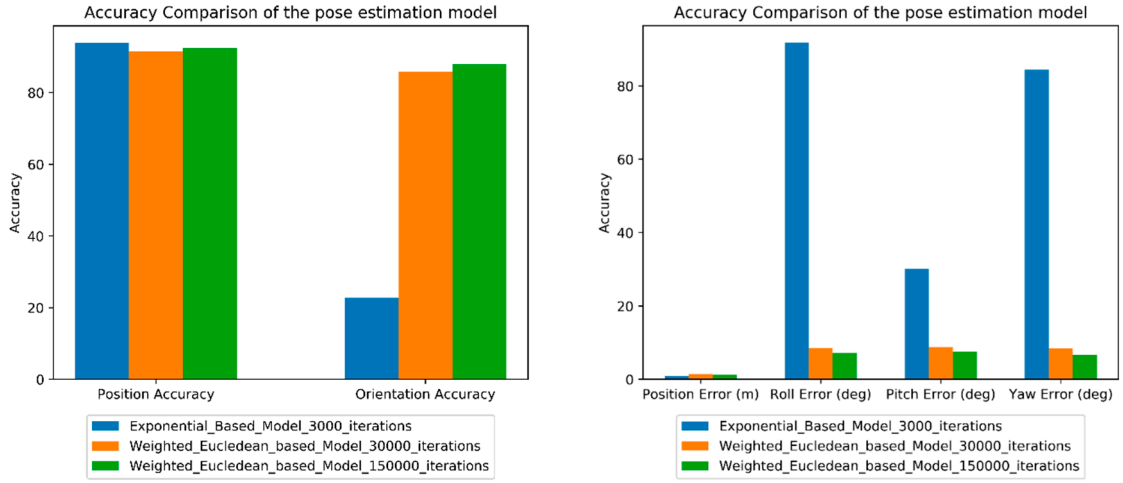


Figure 11. Comparison of the exponential-based pose estimation model and the weighted Euclidean-based pose estimation model: (**a**) Accuracy comparison and (**b**) error comparison.

5. Conclusions

In a spacecraft docking operation, the position and attitude of the target spacecraft must be determined, and a sensor must exist that can obtain those operational parameters. Currently, vision-based algorithms have been developed concurrently with image processing using deep learning algorithms. The goal of this study was to construct a position and attitude estimation model using a deep neural network. The vision-based detection technique has the advantage that it is applicable for both cooperative space objects and non-cooperative space objects. In the implementation, the pose estimation model was constructed based on a state-of-the-art CNN model, which is a modified version of GoogLeNet that forms a general pose estimation model. Then, the model was trained on a simulated public dataset of the Soyuz spacecraft. Subsequently, the model was fine-tuned by repeated training using different mathematical expressions to achieve maximum accuracy. The exponential-based model resulted in high position estimation accuracy but poor orientation estimation accuracy. Thus, the pose estimation model was rebuilt using a different loss function and additional training iterations. With support from the National Astronomical Research Institute (NARIT), we were able to overcome the computational resource limitations. The final weighted Euclidean pose estimation model successfully achieves moderately high predication accuracy.

Under the harsh lighting conditions of outer space, the target spacecraft may not be completely visible in images. Therefore, the vision-based model must detect the target spacecraft with less consideration of the reflection of directional light and the planet surface. A model's performance is strongly dependent on its architecture and on the training procedures. Highly accurate performances are usually obtained from the pose estimation model based on complex pretrained models. However, this study indicated that a convolutional neural model with low complexity can perform at moderately high efficiency when estimating spacecraft position and attitude. Nevertheless, although the complete model of this research resulted in high efficiency, a real-world spacecraft docking operation requires greater position and attitude accuracy and reliability from an estimation system.

Future research should target achieving higher prediction accuracy. Such a model could be constructed using a high-performance pretrained model such as VGG, Inception, DenseNet, or ResNet, whose architectures include deeper layers of neurons. When computational resource are unlimited, a position and attitude estimation model can be constructed by repurposing a high complexity pretrained model. Currently, cluster computing and cloud computing are excellent choices for reducing the computation time during model construction, but accessing the compute nodes may be costly.

The auxiliary algorithms are also an excellent choice for reducing the prediction error of a pose estimation model. Many studies have manipulated the detected points and lines of interest to extract position and attitude parameters from input images [3,9–12]. These contributions provide ideas for future work. Point and line detection can be performed with lower complexity than vision-based CNN algorithms such as OpenCV; then, feature detection can be conducted using an auxiliary deep learning model. The outputs of those algorithms are numerical data that can be combined with image data into a mixed form of input.

In an actual spacecraft docking operation, the spacecraft is in dynamic motion rather than static as in a single 2D image. Moreover, the operation involves both detecting and tracking the spacecraft. Therefore, a vision-based position and attitude estimation model can be applied to the state estimation algorithm or available techniques [12]. The principle of state estimation has been widely applied in the field of spacecraft dynamics and control. For example, the Kalman filter is an elementary state estimation algorithm that combines state prediction using a physics-based model with measurement during the update stage. A vision-based estimation system could be used as the measurement model for spacecraft tracking. Hypothetically, the error of the position and attitude estimation model could be corrected by the physics-based model during an actual docking operation with spacecraft in motion.

In summary, to satisfy a real docking operation, many auxiliary algorithms are recommended for future research to increase the performance of the vision-based position and attitude estimation model. The particular characteristic of the vision-based CNN model is that it is very specific to the environment of the dataset. For example, the model that was trained with the simulation data will perform a satisfactory estimation of this synthetic dataset. However, to address the real docking operation, the constructed model with the knowledge of spacecraft pose estimation can be hypothetically trained further with the data from actual operation [27]. With this feasibility, the vision-based CNN pose estimation model could be trained with real photos to be practical and provide reliability to the actual spacecraft docking in the near future. Moreover, the advanced state estimation algorithm combined with vision-based detection could be a critical factor in achieving higher efficiency in spacecraft motion prediction with regard to actual space interactions.

Author Contributions: Conceptualization, T.P., P.K., and P.T.; methodology, T.P., P.K., and P.T.; software, T.P., P.K., and U.S.; validation, T.P., P.K., and P.T.; formal analysis, T.P., P.K., and P.T.; investigation, T.P., P.K., and P.T.; resources, T.P., P.K., and P.T.; data curation, T.P.; writing—original draft preparation, T.P., W.H., T.J., and T.S.; writing—review and editing, T.P., P.K., P.T., and S.C.; visualization, T.P., P.K., P.T., and S.C.; supervision, P.K., P.T., S.C., and U.S.; project administration, S.Y., D.D. and P.B.; funding acquisition, P.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received financial support from the Air-Space Control, Optimization, and Management Laboratory (ASCOM-LAB), the International Academy of Aviation Industry (IAAI) and Academic Melting Pot Program, and the KMITL Research Fund, King Mongkut's Institute of Technology Ladkrabang and Science, Research and Innovation Promotion Fund (Grant No: 1383848).

Acknowledgments: The authors would like to thank all the lecturers, researchers, and staffs from the International Academy of Aviation Industry, King Mongkut's Institute of Technology Ladkrabang, and the National Astronomical Research Institute of Thailand (NARIT) for their support during this study. Moreover, the authors appreciate the opportunity to use the Chalawan HPC cluster, which is operated and maintained by the National Astronomical Research Institute of Thailand (NARIT) under the Ministry of Higher Education, Science, Research and Innovation of Royal Thai government.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Donahoe, S.; Lewis, J.; Carroll, M.; Le, T. International Docking Standardization NASA. In Proceedings of the Interoperability of Future Docking Systems, Noordwijk, The Netherlands, 31 March–3 April 2009.
2. Ma, Z.; Ma, O.; Shashikanth, B.N. Optimal Control for Spacecraft to Rendezvous with a Tumbling Satellite in a Close Range. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 4109–4114.

3. Casonato, G.; Palmerini, G.B. Visual Techniques Applied to the ATV/1SS Rendezvous Monitoring. In Proceedings of the 2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720), Big Sky, MT, USA, 6–13 March 2004; pp. 1–625.
4. Heorhii, K.; Andrii, P.; Daria, H.; Vladyslav, Y. Application of Deep Learning in the Processing of the Aerospace System's Multispectral Images. In *Handbook of Research on Artificial Intelligence Applications in the Aviation and Aerospace Industries*; Tetiana, S., Yuliya, S., Arnold, S., Eds.; IGI Global: Hershey, PA, USA, 2020; pp. 134–147.
5. Maheshwari, A.; Davendralingam, N.; DeLaurentis, D.A. A Comparative Study of Machine Learning Techniques for Aviation Applications. In Proceedings of the 2018 Aviation Technology, Integration, and Operations Conference, Atlanta, Georgia, 25–29 June 2018.
6. Ewald, V.; Goby, X.; Jansen, H.; Groves, R.M.; Benedictus, R. Incorporating Inductive Bias into Deep Learning: A Perspective from Automated Visual Inspection in Aircraft Maintenance. In Proceedings of the 10th International Symposium on NDT in Aerospace, Dresden, Germany, 24–26 October 2018.
7. Rengasamy, D.; Morvan, H.P.; Figueredo, G.P. Deep Learning Approaches to Aircraft Maintenance, Repair and Overhaul: A Review. In Proceedings of the 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 150–156.
8. Bamber, D.; Palmer, P.; Mackin, S. Attitude Determination Through Image Registration Model and Test-Case for Novel Attitude System in Low Earth Orbit. In Proceedings of the AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Keystone, Colorado, 21–24 August 2006.
9. Liu, C.; Hu, W. Relative pose estimation for cylinder-shaped spacecrafts using single image. *Int. J. Geogr. Trans. Aerosp. Electron. Syst. Inf. Sci.* **2014**, *50*, 3036–3056. [CrossRef]
10. Aumann, A. *Distributed Spacecraft 3D Pose and Position Estimation based on Camera Images*; Julius-Maximilians -Universität: Würzburg, Germany, 2017.
11. Sharma, S.; Ventura, J.; D'Amico, S. Robust model-based monocular pose initialization for noncooperative spacecraft rendezvous. *J. Spacecr. Rocket.* **2018**, *55*, 1414–1429. [CrossRef]
12. Kelsey, J.M.; Byrne, J.; Cosgrove, M.; Seereeram, S.; Mehra, R.K. Vision-Based Relative Pose Estimation for Autonomous Rendezvous and Docking. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 4–11 March 2006; p. 20.
13. George Karimpanal, T.; Bouffanais, R. Self-organizing maps for storage and transfer of knowledge in reinforcement learning. *Adapt. Behav.* **2018**, *27*, 111–126. [CrossRef]
14. Liao, S.; Gavves, E.; Snoek, C.G.M. Spherical Regression: Learning Viewpoints, Surface Normals and 3D Rotations on N-Spheres. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–18 June 2019; pp. 9751–9759.
15. Proença, P.F.; Gao, Y. Deep learning for spacecraft pose estimation from photorealistic rendering. *arXiv* **2019**, arXiv:1907.04298.
16. Sharma, S.; Beierle, C.; Amico, S.D. Pose Estimation for Non-Cooperative Spacecraft Rendezvous Using Convolutional Neural Networks. In Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2018; pp. 1–12.
17. Kendall, A.; Grimes, M.; Cipolla, R. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2938–2946.
18. Mahendran, S.; Ali, H.; Vidal, R. 3D Pose Regression Using Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 494–495.
19. Mahendran, S.; Ali, H.; Vidal, R. A Mixed Classification-Regression Framework for 3D Pose Estimation from 2D Images. *arXiv* **2018**, arXiv:1805.03225.
20. Salehi, S.S.M.; Khan, S.; Erdogmus, D.; Gholipour, A. Real-time deep pose estimation with geodesic loss for image-to-template rigid registration. *IEEE Trans. Med. Imaging* **2019**, *38*, 470–481. [CrossRef] [PubMed]
21. Do, T.-T.; Cai, M.; Pham, T.; Reid, I.D. Deep-6DPose: Recovering 6D object pose from a single RGB image. *arXiv* **2018**, arXiv:1802.10367.
22. Tekin, B.; Sinha, S.N.; Fua, P. Real-Time Seamless Single Shot 6D Object Pose Prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 292–301.

23. Wu, J.; Zhou, B.; Russell, R.; Kee, V.; Wagner, S.; Hebert, M.; Torralba, A.; Johnson, D.M.S. Real-Time Object Pose Estimation with Pose Interpreter Networks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 6798–6805.
24. Xiang, Y.; Schmidt, T.; Narayanan, V.; Fox, D. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *arXiv* **2017**, arXiv:1711.00199.
25. Volpe, R.; Sabatini, M.; Palmerini, G. Shape reconstruction of a tumbling unknown orbital target by passive imaging. *Adv. Astronaut. Sci.* **2020**, *170*, 15–29.
26. Phisannupawong, T.; Kamsing, P.; Tortceka, P.; Yooyen, S. Vision-Based Attitude Estimation for Spacecraft Docking Operation Rhrough Deep Learning Algorithm. In Proceedings of the 22nd International Conference on Advanced Communication Technology (ICACT), Phoenix Park, PyeongChang, Korea, 16–19 February 2020; pp. 280–284.
27. Kuipers, J.B. *Quaternions and Rotation Sequences A Primer with Applications to Orbits, Aerospace and Virtual Reality*; Princeton University Press: Princeton, NJ, USA, 1999.