AIAA SciTech Forum
6-10 January 2025, Orlando, FL
AIAA SCITECH 2025 Forum

10.2514/6.2025-1701

Leveraging Gated Recurrent Units for Iterative Online Precise Attitude Control for Geodetic Missions

Vrushabh Zinage*[1], Shrenik Zinage†[2], Srinivas Bettadpur‡[1], and Efstathios Bakolas§[1]
[1]*The University of Texas at Austin, Austin, Texas, 78712*
[2]*Purdue University, West Lafayette, Indiana, 47906*

In this paper, we consider the problem of precise attitude control for geodetic missions, such as the GRACE Follow-on (GRACE-FO) mission. Traditional and well-established control methods, such as Proportional-Integral-Derivative (PID) controllers, have been the standard in attitude control for most space missions, including the GRACE-FO mission. Instead of significantly modifying (or replacing) the original PID controllers that are being used for these missions, we introduce an iterative modification to the PID controller that ensures improved attitude control precision (i.e., reduction in attitude error). The proposed modification leverages Gated Recurrent Units (GRU) to learn and predict external disturbance trends derived from incoming attitude measurements from the GRACE satellites. Our analysis has revealed a distinct trend in the external disturbance time-series data, suggesting the potential utility of GRU's to predict future disturbances acting on the system. The learned GRU model compensates for these disturbances within the standard PID control loop in real time via an additive correction term which is updated at regular time intervals. The simulation results verify the significant reduction in attitude error, verifying the efficacy of our proposed approach.

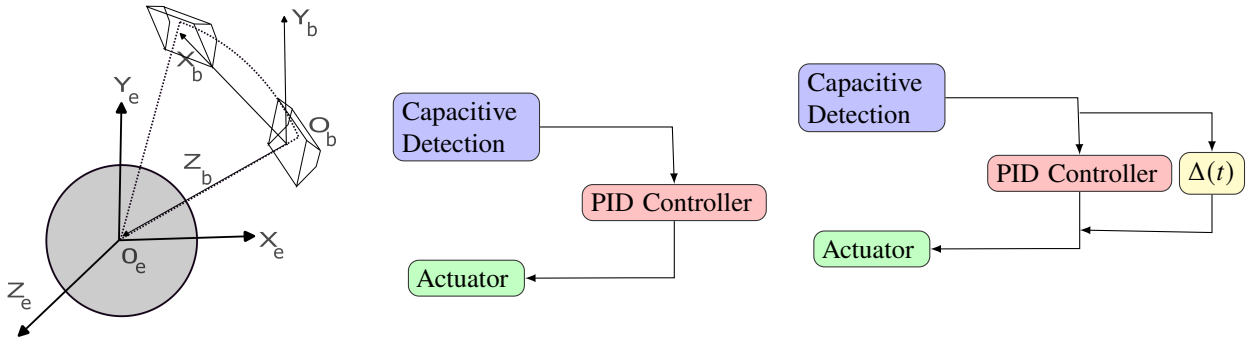
I. Introduction

The GRACE Follow-On mission is a joint scientific project between the United States and Germany, continuing the approach initiated by the earlier GRACE mission [1, 2]. The missions were implemented for NASA by the Caltech/Jet Propulsion Laboratory in cooperation with the German Research Centre for Geosciences (GFZ), with mission operations managed by the German Space Operations Centre (DLR-GSOC). The primary aim of this mission is to gather data to produce both constant and time-varying maps of the Earth's gravity field. In May 2018, the GRACE Follow-on mission successfully launched two satellites into a polar orbit approximately 491 km above the Earth. This mission continues to measure the Earth's gravity field, focusing on how it changes over time, and provides radio occultation measurements as well. The two satellites maintained a distance between each other of 170 to 270 km, working as detectors in the Earth's gravity field. They use a microwave tracking system to measure their distance apart very precisely, up to 1μm. Furthermore, a laser ranging interferometer [3, 4], is included to test new technology. To achieve the best scientific results, the satellites need to point at each other very steadily and accurately while minimizing the effect of any disturbances. This requires highly precise attitude control of their orientations with respect to each other.

One of the critical factors that can affect the accuracy of measurements and consequently the derived data product is the spacecraft attitude error. Current state-of-the-art methods to control and correct this error primarily involve the use of Proportional-Integral-Derivative (PID) controllers. Although PID controllers have been effective and theoretically well established, future missions may have higher precision requirements than what PID controllers can achieve at present. Specifically, geodetic missions, such as GRACE or satellite gravity gradiometers, have stringent requirements for attitude control due to several critical factors discussed in [5]. Furthermore, these missions require precise information about the attitude for accurate scientific analysis, which can be achieved by minimizing the effects of angular rates on the inertial sensors and external disturbances acting on these satellites. To achieve this goal, one must be able to take into account these disturbances and actively compensate them in the designed controllers.

Traditional and well established disturbance observers such as Active Disturbance Rejection Control (ADRC) [6, 7] that can estimate the external disturbance online are restricted to systems that have a certain structure in their governing

*PhD Candidate, University of Texas at Austin, Austin, TX, USA, Student Member AIAA
†Graduate Student, School of Mechanical Engineering, Purdue University, West Lafayette
‡Professor, Center for Space Research and Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Austin, TX, USA
§Associate Professor, Department of Aerospace Engineering and Engineering Mechanics, Senior Member AIAA.



(a) Coordinate system **(b) PID controller [2]** **(c) Proposed minimally modified PID controller**

Figure 1 (a) The coordinate system. Figs. (b) and (c) represent the actual attitude PID control block in GRACE-FO satellites [2] and the proposed minimally modified PID controller, respectively, with GRU based time varying disturbance compensator $\Delta(t)$.

dynamics. Furthermore, they are not able to fully exploit the observed trend in the disturbances if there are any. Toward this goal, neural networks have recently been used successfully for modeling engineered systems for a considerable period [8–12]. Among the various forms of neural networks, recurrent neural networks (RNNs) stand out for their ability to incorporate feedback loops, allowing the output of one neuron (as shown in Fig. 2b) to influence its subsequent input, which mirrors temporal dynamics. This characteristic makes RNNs particularly suitable for modeling sequential processes and has led to their frequent application in the forecasting of time series [13–16]. A common challenge with traditional RNNs is their struggle with short-term memory, often attributed to the problem of vanishing gradients [17]. The problem of vanishing gradients arises when the gradient of the loss function approaches zero, making it almost impossible to update the weights of the network. This prevents the network from learning effectively from the training data. An effective solution to this problem is the use of long-short-term memory (LSTM) networks [18], a specialized form of RNN designed to capture long-term dependencies. LSTMs are distinguished by their use of three gates, which, while improving their capability, also result in greater memory requirements due to an increased number of training parameters. An alternative that addresses this issue is the gated recurrent unit (GRU) [19], which simplifies the structure by using only two gates, thus offering faster computations. Numerous studies have been conducted to evaluate the predictive performance of these RNN variants and their effectiveness in the modeling of complex engineered systems [20, 21]. For example, [21] conducted a comparative analysis of different RNN architectures, including traditional RNNs, LSTMs and GRUs, in the context of predicting time series data. They showed superior performance of LSTM and GRU networks over conventional RNNs, with the GRU model, in particular, demonstrating significant advantages over both traditional RNN and LSTM models.

The main contributions of this paper are as follows. First, using the GRACE data product that consists of the time series data of the attitude error, we estimate the external additive disturbances acting on the spacecraft. Second, leveraging the observed trend in the disturbance time series data, we design a GRU based model to design a minimally modified PID controller that actively compensates for the external disturbances.

The outline of the paper is as follows. Section II delves into the preliminaries followed by the proposed approach. Section III discusses the numerical simulations, and the concluding remarks are succinctly presented in Section IV.

II. Preliminaries and Proposed Approach

A. Coordinate system

In this paper, we use the following right-handed Cartesian coordinate system. First, the origin of the Inertial Frame (IF) $OX_eY_eZ_e$ is at the Earth's center of mass. As shown in Fig. 1a, the OZ_e axis aligns with the Earth's rotation axis, and the OX_e axis points towards the vernal equinox of the J2000 epoch. OY_e is obtained from the right-hand rule. Second, the Body Frame (BF) $O_bX_bY_bZ_b$. The axes of the BF correspond to the satellite's principal axes of inertia, with O_bX_b aligning with the line of sight of the ranging device, and O_bZ_b orthogonal to the satellite's bottom. Finally,

the Reference Frame (RF) $O_r X_r Y_r Z_r$ is the frame that defines the desired attitude of the satellite with respect to the other satellite. First, we construct the reference motion for the second satellite, which is almost identical to that of the first. The basis vectors of the RF are defined as follows; $\boldsymbol{e}_1$ aligns with the line of sight between the two satellites, $\boldsymbol{e}_2$ is orthogonal to both the line of sight and the satellite's radius vector, and $\boldsymbol{e}_3$ completes the system as a right orthogonal vector. Let $\boldsymbol{r}_1, \boldsymbol{r}_2$ and $\boldsymbol{v}_1, \boldsymbol{v}_2$ represent the positions and velocities of the first and second satellites, respectively, which are assumed to be known a priori. Then, the basis vectors are defined as:

$$\boldsymbol{e}_1 = \frac{\boldsymbol{r}_1 - \boldsymbol{r}_2}{\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|}, \quad \boldsymbol{e}_3 = -\frac{\boldsymbol{r}_2 - \boldsymbol{e}_1\langle \boldsymbol{r}_2, \boldsymbol{e}_1\rangle}{\|\boldsymbol{r}_2 - \boldsymbol{e}_1\langle \boldsymbol{r}_2, \boldsymbol{e}_1\rangle\|}, \quad \boldsymbol{e}_2 = \boldsymbol{e}_3 \times \boldsymbol{e}_1.$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product, and $\times$ denotes the cross product between two vectors.

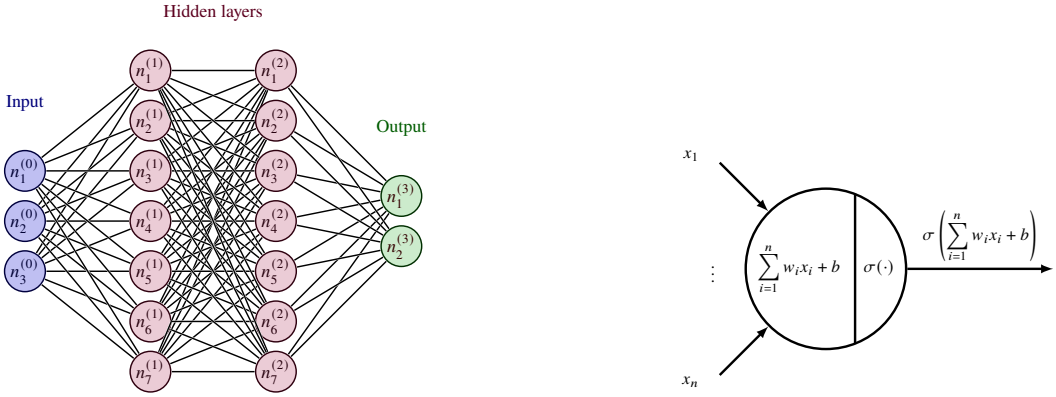
B. Nonlinear attitude dynamics

The kinematics and the dynamics of the spacecraft attitude system are modeled as

$$\dot{R} = R\omega, \qquad I\dot{\omega} = -\omega \times (I\omega) + \boldsymbol{\tau} + \boldsymbol{d} \tag{1}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity of the spacecraft, $R \in SO(3)$ ($SO(3)$ denotes the 3D rotation group) is the rotation matrix, $I \in \mathbb{S}^3_+$ ($\mathbb{S}^3_+$ denotes the set of symmetric positive definite 3×3 matrices) is the inertia, $\boldsymbol{\tau} \in \mathbb{R}^3$ is the control torque and $\boldsymbol{d} \in \mathbb{R}^3$ is the time-varying net external disturbances acting on the spacecraft. Note that the attitude and the angular rate are with respect to the reference frame (RF) that is estimated from the ground-based station. We assume that this RF is known precisely for our simulations. Furthermore, the main objective of this paper is two-fold (Section II.F). The first is to estimate the disturbances $\boldsymbol{d}$, given the time series data of the relative attitude and angular rates. Second, is to modify the traditionally PID controller minimally so as to actively compensate for the additive external disturbances acting on the satellites.

C. Feedforward neural network (FNN)



(a) Schematic of a neural network (NN) **(b) Schematic of a single neuron**

Figure 2 Feedforward neural network (FNN)

Neural networks (NN) are algorithms used for approximating functions based on data, structured through layers that start with the input and end with the output. These layers are organized into a hierarchy, where each level, except the first and last layers, is called a *hidden* layer. By adjusting the number and size of these hidden layers, NNs can handle functions of different complexities. A typical NN configuration, featuring two hidden layers, is depicted in Fig. 2a. Within this diagram, each circle symbolizes the basic computational element of NN, known as a neuron (Fig.2b) which involves a linear transformation followed by a nonlinear operation.

A feedforward neural network (FNN) [22] is essentially a NN characterized by a series of hidden layers. The term "activation" refers to the output generated by each layer. In an FNN, the activation output from a preceding layer becomes the input for the subsequent layer. Assuming L is the total number of layers, the activation generated by the j^{th} hidden

layer can be recursively expressed as:

$$\mathbf{z}^{(j)} = \sigma\left(W^{(j)}\mathbf{z}^{(j-1)} + \mathbf{b}^{(j)}\right), \forall j \in \{1, 2, \cdots, L\}, \quad \mathbf{z}^0 = \boldsymbol{\xi}, \quad d_0 = D$$

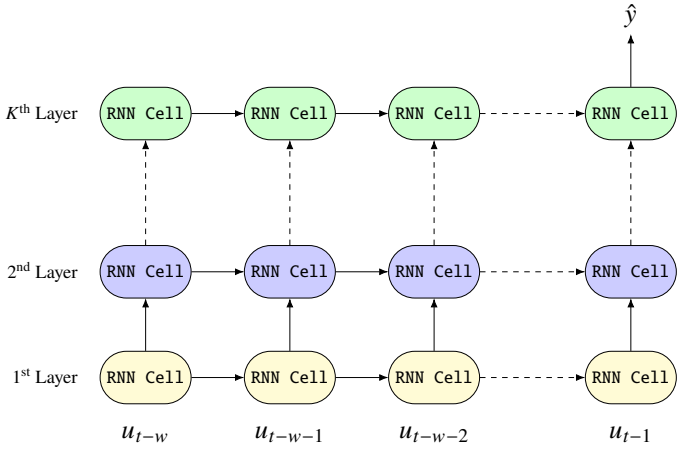
where ξ is the input vector, $W^{(j)}$ represents the weight matrix and $\mathbf{b}^{(j)}$ the bias vector for the j^{th} layer, with d_j indicating the number of neurons in that layer. Note that the activation function σ denotes an element-wise applied nonlinear function, with common choices being the logistic, hyperbolic tangent, or ReLU functions [23, 24]. The weight matrix $W^{(j)}$ and the biases $\mathbf{b}^{(j)}$ (for $j \in [1, L+1]_d$), are termed its parameters, collectively denoted as $\boldsymbol{\theta} = \left\{W^{(j)}, \mathbf{b}^{(j)}\right\}_{j=1}^{L+1} \in \boldsymbol{\Theta}$. These parameters, which consist of the weights and biases, fully describe the network structure.

D. Recurrent neural network (RNN)

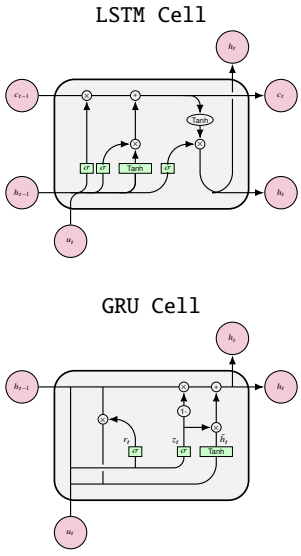
RNNs [25] are a class of artificial neural networks in which connections between neurons form a directed graph along a temporal sequence. This allows them to exhibit temporal dynamic behavior and process sequences of inputs, unlike FNN which is a static input-output mapping where the flow of information is unidirectional from input to output layers without any cycles. The core of an RNN is its cell, which processes one input at a time in a sequence, maintaining a hidden state $\mathbf{h}_t$ that captures information about the past elements of the sequence. Assuming $\mathbf{u}_t$, $\mathbf{h}_t$, $\mathbf{y}_t$ represent the input, hidden state, and output at time t respectively, the governing equations of a conventional RNN cell can be described as follows:

$$\mathbf{h}_t = \sigma\left(W_{hh}\mathbf{h}_{t-1} + W_{uh}\mathbf{u}_t + \mathbf{b}_h\right), \qquad \mathbf{y}_t = W_{hy}\mathbf{h}_t + \mathbf{b}_y,$$

where W_{hh}, W_{uh}, and W_{hy} represent the weight matrices, $\mathbf{b}_h$ and $\mathbf{b}_y$ are bias vectors, and σ being a nonlinear activation function. RNN can predict an output given a window of inputs by processing each element of the input sequence one at a time, updating its hidden state based on the current input and the previous hidden state, as shown in Fig. 3a. This allows the network to make predictions based on the information it has seen up to the last w time steps (Fig. 3) of a sequence. LSTM and GRU are two popular RNN variants designed to address the problem of vanishing gradients [17, 26] (which occurs in neural networks when the gradients of the loss function become exceedingly small during backpropagation, causing the weights in the previous layers to update minimally and hindering effective learning) in conventional RNNs, enabling them to learn long-range dependencies more effectively.



(a) Schematic of RNN (many to one) with K layers



(b) Popular variants of RNN cell

Figure 3 Recurrent neural network (RNN)

1. Long short term memory (LSTM)

LSTM networks [18] are a specialized form of RNN designed for the prediction of sequential data. The core of LSTM is a memory cell that maintains its state over time, managed by three gates: input ($\mathbf{i}_t$), forget ($\mathbf{f}_t$), and output ($\mathbf{o}_t$). These gates, along with a candidate value, regulate the flow and updating of information within the cell, effectively addressing the vanishing gradient problem seen in conventional RNNs. By selectively adding or removing information, LSTMs can preserve relevant information over longer time periods, significantly improving the flow and retention of information compared to traditional RNNs. Furthermore, the forget gate determines the extent of information to be discarded from the cell state (Fig. 3b):

$$\mathbf{f}_t = \sigma(W_{u_f}\mathbf{u}_t + W_{h_f}\mathbf{h}_{t-1} + \mathbf{b}_f),$$

where $\mathbf{f}_t$ denotes the output of the forget gate at time t, W_{u_f} and W_{h_f} are the input and recurrent weights of the forget gate, $\mathbf{b}_f$ is the bias, and $\sigma(.)$ represents the sigmoid activation function. Following, the decision of the input gate to update the state is captured by:

$$\mathbf{i}_t = \sigma(W_{u_i}\mathbf{u}_t + W_{h_i}\mathbf{h}_{t-1} + \mathbf{b}_i).$$

A hyperbolic tangent function then generates a vector of candidate values $\mathbf{z}_t$ for state addition:

$$\mathbf{z}_t = \tanh(W_{u_z}\mathbf{u}_t + W_{h_z}\mathbf{h}_{t-1} + \mathbf{b}_z),$$

with $(W_{u_i}, W_{h_i}, \mathbf{b}_i)$ and $(W_{u_z}, W_{h_z}, \mathbf{b}_z)$ denoting the weights and biases associated with the input gate and cell update mechanism, respectively. The cell state update is formulated as follows:

$$\mathbf{c}_t = \mathbf{f}_t \star \mathbf{c}_{t-1} + \mathbf{i}_t \star \mathbf{z}_t,$$

where $\star$ signifies element-wise multiplication. Lastly, based on the output gate, certain parts of the cell state are outputted. The output of LSTM cell $\mathbf{h}_t$ is then calculated by applying a hyperbolic tangent function to the cell state and multiplying the result by the output gate, as shown below:

$$\mathbf{o}_t = \sigma(W_{u_o}\mathbf{u}_t + W_{h_o}\mathbf{h}_{t-1} + \mathbf{b}_o), \qquad \mathbf{h}_t = \mathbf{o}_t \star \tanh(\mathbf{c}_t),$$

where $(W_{u_o}, W_{h_o}, \mathbf{b}_o)$ represents the recurrent weights, input weights, and bias of the output gate respectively.

2. Gated recurrent unit (GRU)

Gated Recurrent Units (GRUs) are a streamlined variant of LSTM networks, introduced in [27] to simplify the LSTM architecture by eliminating the cell state and reducing the number of gates from three to two: the update gate ($\mathbf{z}_t$) and the reset gate ($\boldsymbol{r}_t$). These modifications allow for fewer computations during training, improving efficiency without compromising predictive performance. The update gate determines the extent of information transition from the previous state, while the reset gate integrates the current and previous states, adjusting the influence of past information. Fig. 3b illustrates the structural design of a GRU cell. The update law for GRU is given by:

$$\mathbf{z}_t = \sigma\left(W_{u_z}\mathbf{u}_t + W_{h_z}\mathbf{h}_{t-1} + \mathbf{b}_z\right).$$

This update gate $\mathbf{z}_t$ helps in deciding the extent to which information from past time steps is forwarded into the future time steps. Consequently, the reset gate $\boldsymbol{r}_t$ is computed, which is important in determining the level of past information to forget. Mathematically,

$$\boldsymbol{r}_t = \sigma\left(W_{u_r}\mathbf{u}_t + W_{h_r}\mathbf{h}_{t-1} + \mathbf{b}_r\right),$$

where W_{u_r}, W_{h_r}, and $\mathbf{b}_r$ denote the input weights, recurrent weights, and bias associated with the reset gate, respectively. Subsequently, the GRU formulates a new memory content, leveraging the reset gate to retain pertinent past information, expressed as:

$$\tilde{\mathbf{h}}_t = \tanh\left(W_{u_h}\mathbf{u}_t + W_{h_h}\mathbf{h}_{t-1} + \mathbf{b}_h\right),$$

where the $\tanh$ is the hyperbolic tangent function, and W_{u_h}, W_{h_h}, and $\mathbf{b}_h$ are the input weights, recurrent weights, and bias for the new memory content $\tilde{h}_t$ respectively. Finally, the update law for the current hidden state $\mathbf{h}_t$ is given by

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \star \mathbf{h}_{t-1} + \mathbf{z}_t \star \tilde{\mathbf{h}}_t,$$

with $\star$ denoting the element-wise (Hadamard) multiplication.

E. Attitude and orbit control system

The control torque $\boldsymbol{\tau}$ generated by the magnetorquers from the GRACE-FO satellites is given by [28]

$$\boldsymbol{\tau} = \mathbf{m} \times \mathbf{B}, \tag{2}$$

where $\mathbf{m}$ is the magnetic dipole vector created by three perpendicular magnetorquers, and $\mathbf{B}$ is the external magnetic field (for instance earth's magnetic field and due to other external sources which can be measured). Since the control torque is always orthogonal to the magnetic field's direction, there is always a direction at any given moment where the control torque cannot be generated. Various methods exist for using magnetorquers for orbital or inertial stabilization, most of which are based on PID controllers [29–31]. In these methods, the dipole vector $\mathbf{m}$ is selected as follows:

$$\mathbf{m} = \mathbf{B} \times \underbrace{\left(-k_p \boldsymbol{e} - k_d \dot{\boldsymbol{e}} - k_i \int_0^t \boldsymbol{e}(s)\mathrm{d}s\right)}_{\boldsymbol{u}_{\mathrm{PID}}} \tag{3}$$

where $\boldsymbol{e}$ is the relative attitude error between the two GRACE-FO satellites, $k_p > 0$, $k_d > 0$, and k_i are the PID gains. Consequently, when the magnetic field is parallel to the applied control input (which usually occurs near the equator), the net dipole moment $\mathbf{m}$ becomes zero, resulting in zero net torque. When this occurs, the torque for the satellites is generated using the thrusters present on the satellites.

F. Proposed iterative approach

In this section, we present our proposed approach that minimally modifies the traditional PID controller $\boldsymbol{u}_{\mathrm{PID}}$ to mitigate the relative attitude error (or improve the pointing accuracy) between the two satellites. We assume that the post-facto ground processed relative attitude estimate (or the desired targeted pointing) between the satellites is known a priori. This ground-processed relative attitude estimate is used for computing the actual external disturbances acting on the satellites and compensating them via a modified PID controller as discussed in the following. The proposed approach consists mainly of two steps. In the first step, the time series data of the relative attitude error from the GRACE-FO data product (for a particular time period, say $T > 0$) are used to estimate the additive disturbances acting on the GRACE-FO satellites. The disturbance values are then fed to the GRU network to predict the disturbances for the next time period T. Using these predictions, $\boldsymbol{u}_{\mathrm{PID}}$ is modified as follows:

$$\boldsymbol{u}^1_{\mathrm{PID}} = \boldsymbol{u}_{\mathrm{PID}} - \Delta_1(t) \quad \forall\ t \in [T,\ 2T],$$

where the superscript in $\boldsymbol{u}^1_{\mathrm{PID}}$ denotes the first iteration (or first modification) in the original $\boldsymbol{u}_{\mathrm{PID}}$ and $\Delta_1(t)$ is the GRU based prediction for the disturbances for the time period T. Substituting the updated PID controller in (1) gives

$$\dot{R} = R\omega, \qquad I\dot{\omega} = -\omega \times (I\omega) + \boldsymbol{u}_{\mathrm{PID}} + \underbrace{\boldsymbol{d} - \Delta_1(t)}_{\boldsymbol{d}_1} \quad \forall\ t \in [T,\ 2T].$$

The additional Δ_1 (from the trained GRU network using estimated disturbance data from time $[0,\ T]$) term tries to compensate for the disturbances $\boldsymbol{d}$ as much as possible). Let the virtual disturbances that are acting on the satellites be given by $\boldsymbol{d}_1 := \boldsymbol{d} - \Delta_1(t)$. These two steps are repeated again. In particular, using the modified PID controller $\boldsymbol{u}^1_{\mathrm{PID}}$ (4), the states of the satellites are propagated for a time period T i.e. for $t \in [T,\ 2T]$. Subsequently, these states are used to estimate the virtual disturbances (i.e., $\boldsymbol{d}_1$ for the first iteration) and thereafter update the trained GRU model. Note that the GRU model must be updated as the disturbance values $\boldsymbol{d}$ for the time period $t \in [0,\ T]$ might be different from the disturbance values during the time period $t \in [T,\ 2T]$. Furthermore, $\boldsymbol{d}_1$ is termed as the virtual disturbance because they are the unknown residue obtained from subtracting the GRU predicted values and the actual disturbances from the previous iteration. Consequently, $\boldsymbol{u}^N_{\mathrm{PID}}$ after N iterations can be written as follows:

$$\boldsymbol{u}^N_{\mathrm{PID}} = \boldsymbol{u}_{\mathrm{PID}} - \sum_{i=1}^{N} \Delta_i(t), \quad \forall\ t \in [NT,\ (N+1)T]. \tag{4}$$

The dynamics are then given by

$$\dot{R} = R\omega, \qquad I\dot{\omega} = -\omega \times (I\omega) + \boldsymbol{u}_{\mathrm{PID}} + \underbrace{\boldsymbol{d} - \sum_{i=1}^{N} \Delta_i(t)}_{\boldsymbol{d}_N}, \quad \forall\ t \in [NT,\ (N+1)T],$$

where Δ_k is the trained GRU model trained over the time series data of $\boldsymbol{d} - \sum_{i=1}^{k-1} \Delta_i(t)$ (virtual disturbances) where $k = \{1, \ldots, N\}$. Algorithm 1 summarizes the proposed approach. The training methodology for the GRU is described in Algorithm 2. Given the susceptibility of neural networks to overfitting, we use early stopping as a regularization strategy, characterized by patience of p epochs. This implies that the training process is terminated if there is no improvement in loss over a consecutive span of p epochs. Note that depending on the computational bandwidth of the satellites, it is always possible to increase/decrease the time period T to estimate (from the relative error in the previous iteration) and predict disturbances (for the next iteration).

Algorithm 1 Iterative approach

Initialize the relative attitude error $\mathcal{A}_0$ from the GRACE-FO data product and set $i = 0$
Define $\boldsymbol{u}_{\text{PID}}^0 := \boldsymbol{u}_{\text{PID}}$ as PID controller (3)
repeat
    Extract external disturbance time series data obtained from $\mathcal{A}_i$, attitude dynamics (1), and $\boldsymbol{u}_{\text{PID}}^i$
    Store these disturbances in set $\mathcal{D}_i^e$
    Learn time series trend in data $\mathcal{D}_i^e$ using Algorithm 2
    Modify control input: $\boldsymbol{u}_{\text{PID}}^{i+1} \leftarrow \boldsymbol{u}_{\text{PID}}^i - \Delta_i(t)$ ▷ $\Delta_i(t)$ is a GRU based model
    Propagate the attitude dynamics (1) using the modified PID controller $\boldsymbol{u}_{\text{PID}}^{i+1}$ for time $t \in [(i+1)T,\ (i+2)T]$
    $i \leftarrow i + 1$
until a stopping criterion is met

Algorithm 2 Training GRU network via backpropagation

Dataset $\mathcal{D}_i^e = \{d_1, d_2, ..., d_T\}$
Parameters: Loss function $L(\theta)$, θ: parameters of the GRU network, initial parameter θ_0, learning rate α, batch size b, number of epochs E, input window size w, number of layers L, number of hidden units in each layer h, patience p.
Initialize best loss $L_{\text{best}} \approx \infty$
Initialize patience counter $p_{\text{counter}} = 0$
for $i = 1 \ldots E$ **do**
    $(d_t, \{d_{t-k}, ..., d_{t-1}\})$ = Sample training batch
    $\theta := \theta_0 \sim p\,(\theta_0)$ ▷ Initialize the weights from a Glorot uniform distribution
    for $j = 1 \ldots b$ **do**
        $\hat{d}_t = \texttt{GRU}\,(\{d_{t-w}, ..., d_{t-1}\}; \theta, L, h)$
        $L_{\text{total}} = \sum_{j=1}^{b} L\left(d_j, \hat{d}_j\right)$ ▷ L is Huber loss between the actual and the GRU based predicted disturbances
        $\nabla L(\theta) = \frac{\partial L_{\text{total}}}{\partial \theta}$;
        $\theta = \theta - \alpha \nabla L(\theta)$ ▷ Adam optimizer for updating parameters θ
    end for
    if $L_{\text{total}} < L_{\text{best}}$ **then**
        $L_{\text{best}} = L_{\text{total}}$
        $p_{\text{counter}} = 0$
    else
        $p_{\text{counter}} = p_{\text{counter}} + 1$
    end if
    if $p_{\text{counter}} > p$ **then**
        break
    end if
end for
return θ

III. Results

In this section, we discuss the efficacy of the proposed modified PID controller in attenuating the effects of external disturbances on the GRACE satellites. The implementation of the training and testing phases of this approach is programmed using the `equinox` and `optax` library with a JAX [32] backend. The advantages of using JAX over other popular open source machine learning libraries, such as PyTorch [33] and TensorFlow [34], lie in its functional programming paradigm and automatic differentiation capabilities, which are particularly advantageous for high-performance computation tasks and real-time implementability (10-100x speedup compared to PyTorch or TensorFlow). The code for replicating the results of this paper is available from https://github.com/shrenikvz/gru-attitude-control-geodetic-missions.

Table 1 Tuned hyperparameters for GRU network

Sampling Time	1 s
No. of Layers	3
No. of hidden units	128
Loss Function	Huber
Learning Rate	0.005
Regularization	Early Stop. of Loss (Pat. 50)
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Weight Initialisation	Glorot Uniform
Batch Size	64
Window Size	5 s
Max Epochs	500
No. of times trained	5

Table 2 Average computational training time for GRU network (when the network is trained 5 times)

Iterations N	**Time**
1	2.17 s
2	2.29 s
3	1.31 s
4	1.63 s

Note that the neural networks were trained five times for each iteration to ensure consistency and reliability in performance, thus mitigating issues associated with random initializations, hyperparameter sensitivity, and the stochastic nature of the training process. In this work, a Huber loss is used because outliers in the training data are usually not handled properly by mean squared error (MSE). Furthermore, the mean absolute error (MAE) is computationally expensive as it uses the modulus operator function and may lead to local minima issues. The Huber loss is a hybrid of the MSE and MAE, operating as the MSE when the error is less than a specific threshold $\delta > 0$, and as the MAE when the error exceeds this threshold. This is mathematically represented as follows:

$$L_\delta(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \begin{cases} \frac{1}{2}(\boldsymbol{y} - \hat{\boldsymbol{y}})^2 & \text{for } \mid \boldsymbol{y} - \hat{\boldsymbol{y}}) \mid \leq \delta \\ \delta \cdot \left(\mid \boldsymbol{y} - \hat{\boldsymbol{y}}) \mid -\frac{1}{2}\delta \right) & \text{otherwise} \end{cases}.$$

The above loss function is shown to be computationally efficient and is also able to handle outliers in the data. However, a notable limitation is that the hyperparameters need to be optimized to maximize model accuracy. In our algorithm, we use the Adam optimizer [35], with the hyperparameters tuned for optimal performance (see Table 1).

The training has been performed on a Macbook Pro with an Apple M1 pro chip and 16 GB of memory. Table 2 shows the average computational time (in s) to train the GRU network for different number of total iterations N. As seen from this table, the average computational time has an overall decreasing trend with N. This is mainly because, as N increases, the amount of training data available to train the GRU network is less. Furthermore, it must be noted that depending on the computational budget on the GRACE-FO satellites or the ground based station, the amount of training data (related to iterations N) fed to the GRU network can be increased or decreased.

Fig. 4 shows the evolution of angular rates and Euler angles across four iterations (i.e, $N = 4$). In the first iteration, both angular rates and Euler angles exhibit significant fluctuations over time. As the iterations progress, there is a notable reduction in the amplitude of these fluctuations, particularly from the second iteration onward. This is mainly because the GRU network is able to effectively capture the trend in the external disturbances acting on the satellites. Finally, by the fourth iteration, it is observed that the relative attitude and angular rate errors have reached a steady-state value. This trend verifies the efficacy of our proposed approach in mitigating the external disturbances acting on the

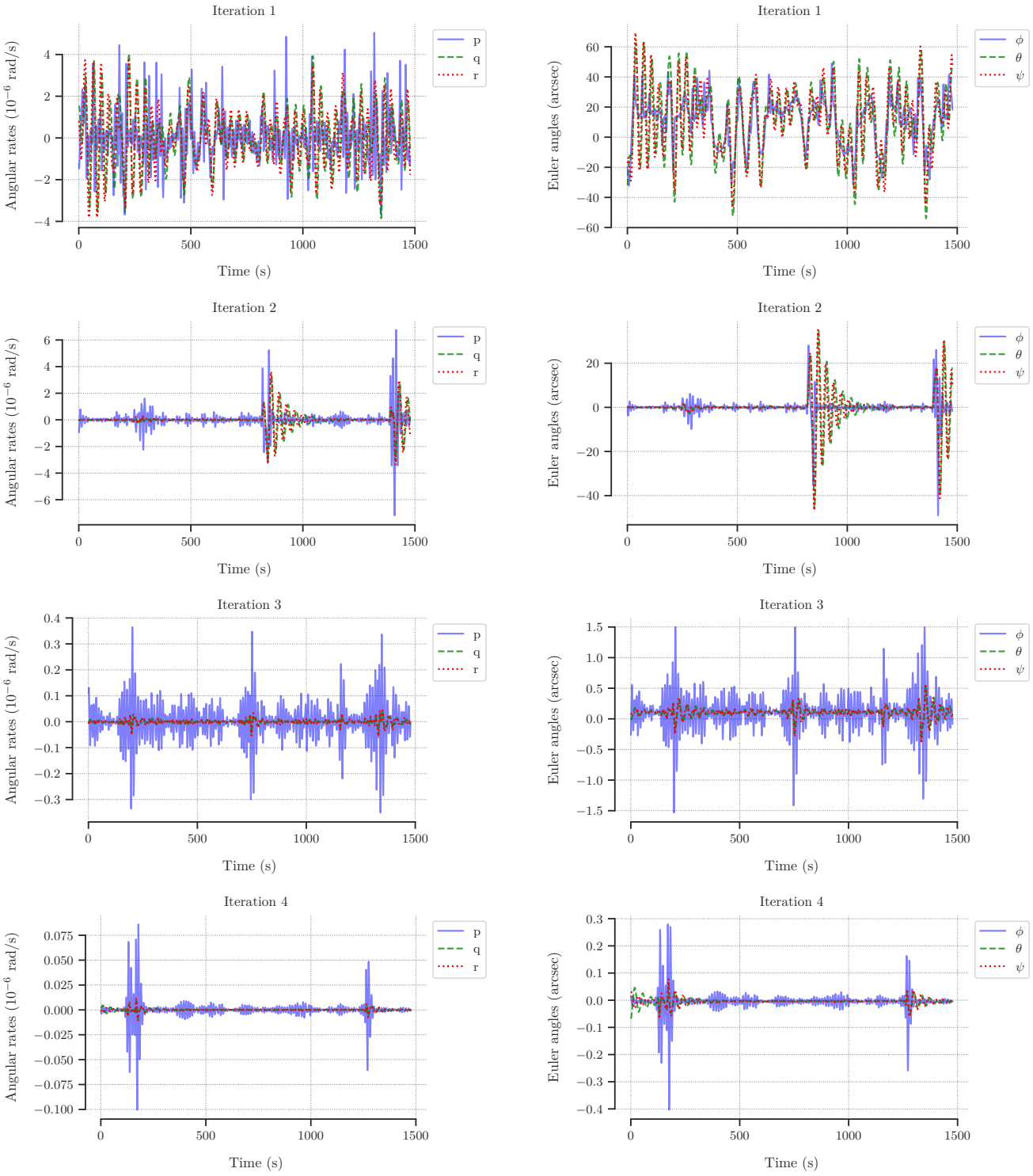


Figure 4 Evolution of angular rates and relative attitude with time across different iterations using the modified PID controller u^i_{PID} ($i \in \{1, \ldots, 4\}$)

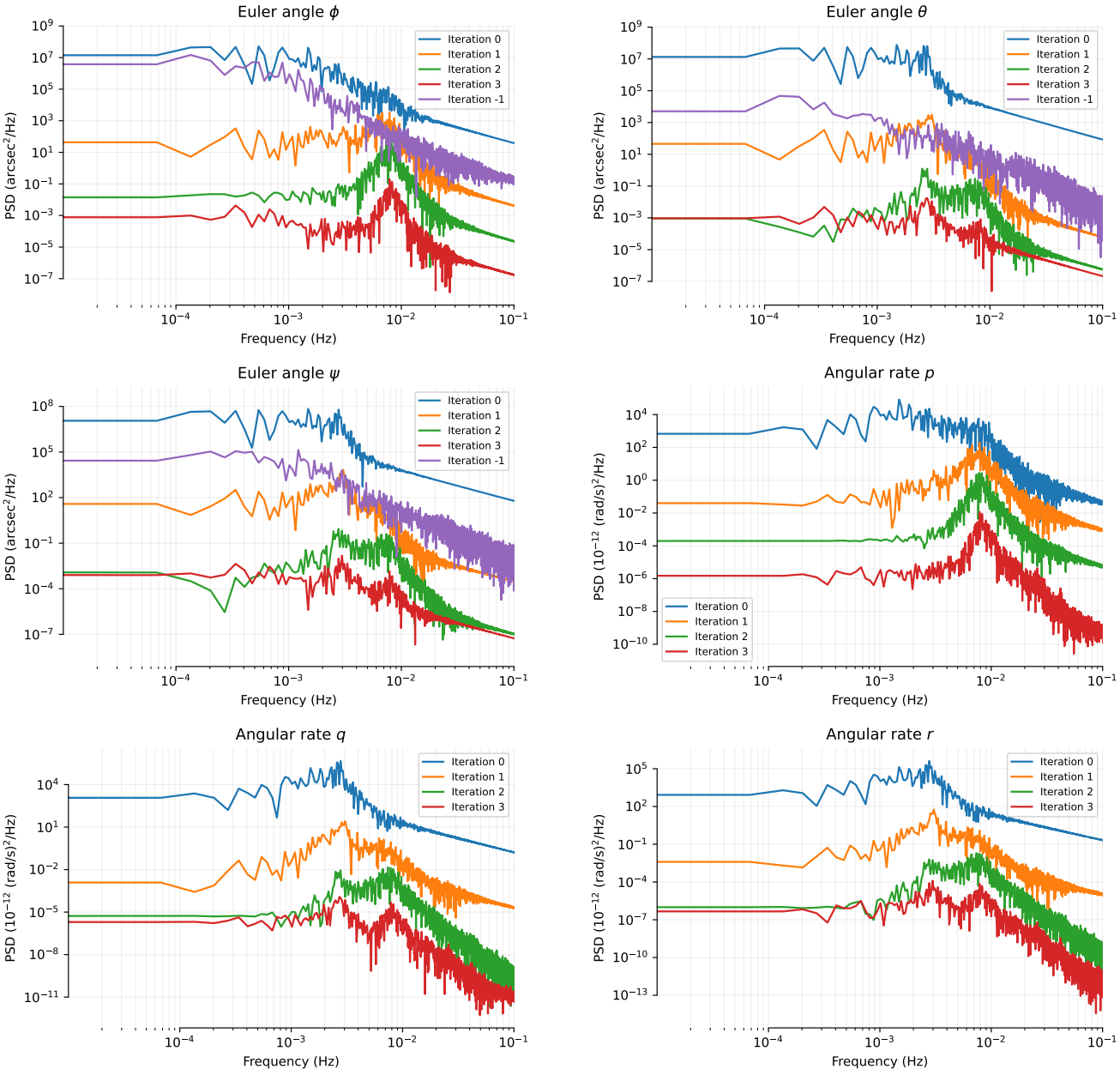


Figure 5 The Power Spectral Density (PSD) plots of the Euler and angular rates over various iterations

GRACE satellites, resulting in a more precise stabilization of the relative attitude and angular rate errors over time compared to traditional PID controllers. The presence of a few spikes in the later iterations might be mainly due to attitude changes caused by the activation of the thrusters on these satellites (i.e, when the magnetic field **B** is parallel to the applied control input $\boldsymbol{u}_{\text{PID}}$), but they have a minimal impact compared to the first iteration, highlighting the improved robustness and reliability of our proposed approach. Fig. 5 shows the power spectral density plots of the relative attitude and angular rates at different iteration steps where iteration equal to -1 denotes the PSD plots for the true attitude and angular rate data at the start (i.e., even before the proposed PID controller is applied). This plot computes the power spectral density (PSD) via the fast fourier transform method. As seen from this figure, the maximum peak in these plots decreases with increase in the number of iterations.

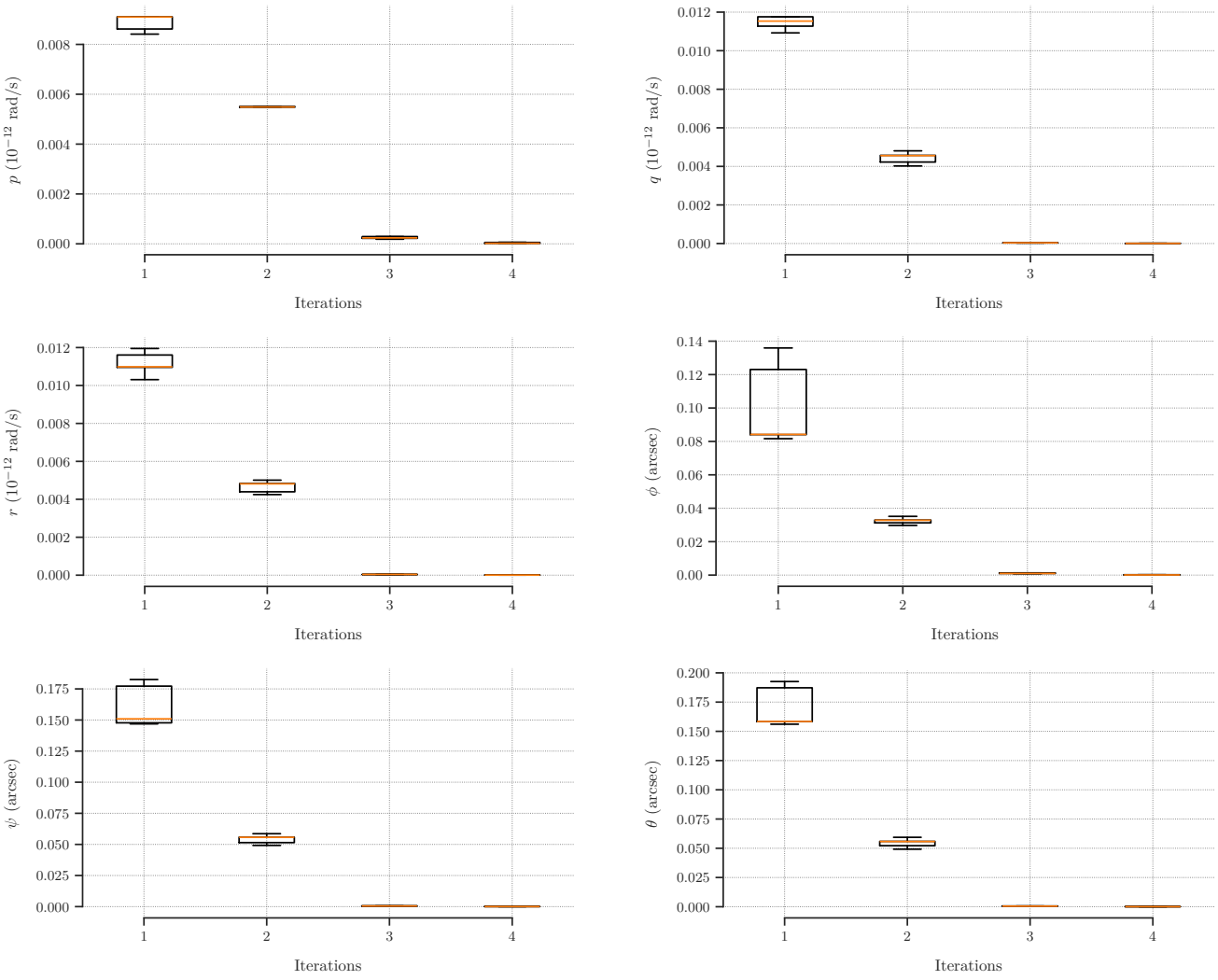


Figure 6 Box plots of relative attitude ($[\phi,\ \theta,\ \psi]$) and angular rate $\omega = [p,\ q,\ r]^{\mathrm{T}}$ error across four iterations for GRACE satellites

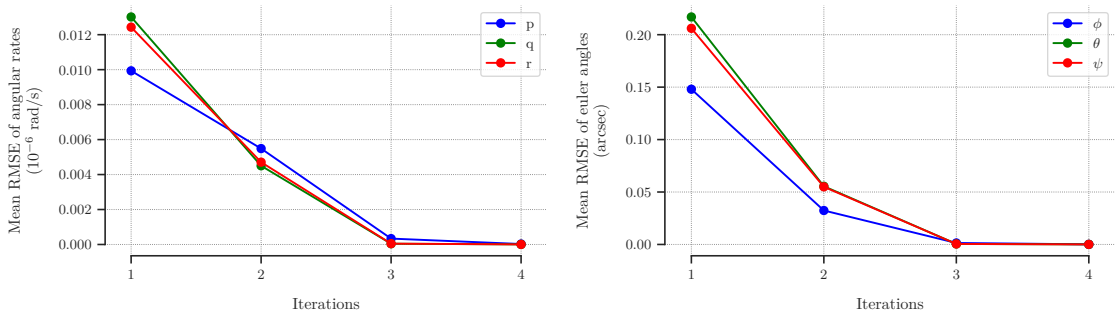


Figure 7 Evolution of mean RMSE with iterations

Finally, the box plots in Fig. 6 show the error distributions for the attitude angles ($\phi,\ \theta,\ \psi$) and angular rates ($p,\ q,\ r$) when the network has been trained multiple times over four iterations which are part of the proposed algorithm (Algorithm 1) for the GRACE-FO satellites. For attitude angles ($\phi,\ \theta,\ \psi$), the median values appear to stabilize or slightly decrease across iterations, with the interquartile range (IQR) narrowing, indicating a convergence towards a more precise estimate of the attitude angles. Fig. 7 depicts the evolution of the mean RMSE values for these variables with iterations. We can observe that these values continue to decrease with iterations and finally converge.

IV. Conclusion

In this paper, we explore the potential of Gated Recurrent Unit (GRU) networks for precise attitude and angular rate control, particularly for geodetic missions such as the GRACE-FO mission. Leveraging the relative attitude and angular rate time series data from the GRACE-FO data product, we estimated the additive time-varying disturbances acting on the satellites. Next, we observed a periodic trend in these disturbances data, prompting the utilization of GRUs to capture and predict these disturbance values. Our proposed modified PID controller minimally modifies the traditional PID controller by augmenting it with a GRU network that compensates for the disturbances acting on the satellites. Future work includes exploration of angular rate spectrum control and stability margins for the proposed controllers. Additionally, it would be interesting to investigate the susceptibility to sensor/actuator imperfections on the proposed controller.

V. Acknowledgments

The research has been supported by NASA Grant 80NSSC22K0287.

References

[1] Tapley, B. D., Bettadpur, S., Watkins, M., and Reigber, C., "The gravity recovery and climate experiment: Mission overview and early results," *Geophysical research letters*, Vol. 31, No. 9, 2004.

[2] Kornfeld, R. P., Arnold, B. W., Gross, M. A., Dahya, N. T., Klipstein, W. M., Gath, P. F., and Bettadpur, S., "GRACE-FO: the gravity recovery and climate experiment follow-on mission," *Journal of spacecraft and rockets*, Vol. 56, No. 3, 2019, pp. 931–951.

[3] Landerer, F. W., Flechtner, F. M., Save, H., Webb, F. H., Bandikova, T., Bertiger, W. I., Bettadpur, S. V., Byun, S. H., Dahle, C., Dobslaw, H., et al., "Extending the global mass change data record: GRACE Follow-On instrument and science data performance," *Geophysical Research Letters*, Vol. 47, No. 12, 2020, p. e2020GL088306.

[4] Abich, K., Abramovici, A., Amparan, B., Baatzsch, A., Okihiro, B. B., Barr, D. C., Bize, M. P., Bogan, C., Braxmaier, C., Burke, M. J., et al., "In-orbit performance of the GRACE follow-on laser ranging interferometer," *Physical review letters*, Vol. 123, No. 3, 2019, p. 031101.

[5] Rosen, M. D., "Analysis of hybrid satellite-to-satellite tracking and quantum gravity gradiometry architecture for time-variable gravity sensing missions," Ph.D. thesis, 2021.

[6] Han, J., "From PID to active disturbance rejection control," *IEEE transactions on Industrial Electronics*, Vol. 56, No. 3, 2009, pp. 900–906.

[7] Zhang, C., He, J., Duan, L., and Kang, Q., "Design of an active disturbance rejection control for drag-free satellite," *Microgravity Science and Technology*, Vol. 31, 2019, pp. 31–48.

[8] Narendra, K. S., and Parthasarathy, K., "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, Vol. 6, No. 2, 1992, pp. 109–131.

[9] González-García, R., Rico-Martìnez, R., and Kevrekidis, I. G., "Identification of distributed parameter systems: A neural net based approach," *Computers & chemical engineering*, Vol. 22, 1998, pp. S965–S968.

[10] Chen, S., Billings, S. A., and Grant, P., "Non-linear system identification using neural networks," *International journal of control*, Vol. 51, No. 6, 1990, pp. 1191–1214.

[11] Milano, M., and Koumoutsakos, P., "Neural network modeling for near wall turbulent flow," *Journal of Computational Physics*, Vol. 182, No. 1, 2002, pp. 1–26.

[12] Rico-Martinez, R., and Kevrekidis, I. G., "Continuous time modeling of nonlinear systems: A neural network-based approach," *IEEE International Conference on Neural Networks*, IEEE, 1993, pp. 1522–1525.

[13] Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., and Koumoutsakos, P., "Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 474, No. 2213, 2018, p. 20170844.

[14] Pan, S., and Duraisamy, K., "Long-time predictive modeling of nonlinear dynamical systems using neural networks," *Complexity*, Vol. 2018, 2018.

[15] Pathak, J., Wikner, A., Fussell, R., Chandra, S., Hunt, B. R., Girvan, M., and Ott, E., "Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, Vol. 28, No. 4, 2018, p. 041101.

[16] Lu, Z., Hunt, B. R., and Ott, E., "Attractor reconstruction by machine learning," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, Vol. 28, No. 6, 2018, p. 061104.

[17] Pascanu, R., Mikolov, T., and Bengio, Y., "On the difficulty of training recurrent neural networks," *International conference on machine learning*, Pmlr, 2013, pp. 1310–1318.

[18] Hochreiter, S., and Schmidhuber, J., "Long short-term memory," *Neural computation*, Vol. 9, No. 8, 1997, pp. 1735–1780.

[19] Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y., "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.

[20] Zinage, S. V., "INVESTIGATION OF DIFFERENT DATA DRIVEN APPROACHES FOR MODELING ENGINEERED SYSTEMS," , 2022. https://doi.org/10.25394/PGS.21671480.v1.

[21] Shopov, V., and Markova, V., "Identification of non-linear dynamic system," *2019 International Conference on Information Technologies (InfoTech)*, IEEE, 2019, pp. 1–3.

[22] Bebis, G., and Georgiopoulos, M., "Feed-forward neural networks," *Ieee Potentials*, Vol. 13, No. 4, 1994, pp. 27–31.

[23] Goodfellow, I., Bengio, Y., and Courville, A., *Deep learning*, MIT press, 2016.

[24] Dubey, S. R., Singh, S. K., and Chaudhuri, B. B., "Activation functions in deep learning: A comprehensive survey and benchmark," *Neurocomputing*, Vol. 503, 2022, pp. 92–108.

[25] Schmidt, R. M., "Recurrent neural networks (rnns): A gentle introduction and overview," *arXiv preprint arXiv:1912.05911*, 2019.

[26] Bengio, Y., Simard, P., and Frasconi, P., "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, Vol. 5, No. 2, 1994, pp. 157–166.

[27] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[28] Kinzie, R., Bevilacqua, R., Seo, D., Conklin, J. W., and Wass, P. J., "Dual quaternion-based dynamics and control for gravity recovery missions," *Acta Astronautica*, Vol. 213, 2023, pp. 764–776.

[29] Celani, F., "Robust three-axis attitude stabilization for inertial pointing spacecraft using magnetorquers," *Acta Astronautica*, Vol. 107, 2015, pp. 87–96.

[30] Ovchinnikov, M. Y., Roldugin, D., and Penkov, V., "Three-axis active magnetic attitude control asymptotical study," *Acta Astronautica*, Vol. 110, 2015, pp. 279–286.

[31] Lovera, M., and Astolfi, A., "Global magnetic attitude control of inertially pointing spacecraft," *Journal of guidance, control, and dynamics*, Vol. 28, No. 5, 2005, pp. 1065–1072.

[32] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q., "JAX: composable transformations of Python+NumPy programs," , 2018. URL http://github.com/google/jax.

[33] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, Vol. 32, 2019.

[34] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al., "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[35] Kingma, D. P., and Ba, J., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.