73rd International Astronautical Congress (IAC), Paris, France, 18-22 September 2022.

IAC-22-C1.1.6

Attitude control of satellites with flexible appendages using inverse simulation

Robert Gordon [a], **Matteo Ceriotti** [b], **Kevin Worrall** [c]

[a] James Watt School of Engineering, University of Glasgow, r.gordon.5@research.gla.ac.uk
[b] James Watt School of Engineering, University of Glasgow, matteo.ceriotti@glasgow.ac.uk
[c] James Watt School of Engineering, University of Glasgow, kevin.worrall@glasgow.ac.uk

Abstract

Many satellites include flexible deploy-able appendages and structures as a method of increasing mission capability with reduced form factors. Consideration of the additional flexible dynamics is important in the design of suitable attitude control methods. This paper will explore the novel use of inverse simulation (InvSim) techniques as a methodology to solve the flexible satellite attitude dynamics, and the practicality of implementing it as a real-time attitude control method.

A simple, but accurate, and computationally efficient mathematical model will be developed, describing the dynamics of a rigid body satellite with an arbitrary number of flexible appendages mounted to it. The InvSim algorithm is fully described and several test cases will show flexibility of the method for satellites with different actuator configurations.

A hybrid control method is then developed, comprising an InvSim offline feed-forward element and an online proportional-derivative (PD) feedback control element. This hybrid controller aims to make use of the superior tracking performance of InvSim while introducing feedback in order to add robustness to model uncertainty and disturbances. The performance of this control system, in terms of tracking error and total control effort, will be compared to a traditional PD controller in the presence of model uncertainty and disturbances, through Monte-Carlo simulations. Additional benefits of InvSim in terms of its practical use during the design process of a flexible satellite will also be explored.

Keywords: Inverse simulation, attitude control, flexible appendages

1. Introduction

The importance of considering the effects of flexible structures and appendages on the dynamics of satellites was first highlighted shortly after Explorer I reached orbit on January 31st 1958. The satellite's attitude was to be stabilised by spinning about its minimum principal axis of inertia. However, unexpectedly the satellite's spin-axis precessed, eventually causing the satellite to spin end over end. The cause for this instability was to be mechanical damping in the four flexible antennas attached to Explorer I's rigid main body, which allowed for energy to be lost from the system. This caused the satellite to move from its initial high energy state to the satellite's minimum energy state, spinning about the maximum inertia principle axis [1, 2]. Other flexible dynamic effects were also discovered after the successful launch of Canada's first satellite Aloutette I in 1962. Non-uniform heating of the satellite's flexible antennas caused an asymmetry in the satellites shape, causing a net de-stabilising aerodynamic torque on the spacecraft [1, 2].

Cubesats, while providing economical access to space [3], commonly include flexible structures and appendages to provide increased capabilities, such as larger solar arrays and instrumentation [1, 4], in a small form factor. It is therefore essential that flexible dynamic effects are considered during the design process of modern satellites, both large and small, and accounted for by the attitude control system.

Mathematical modelling of the attitude dynamics of satellites with flexible appendages is an essential first step when trying to account for their effects during both the satellite's design process and assessing performance of any proposed control solution. This problem reduces down to finding methods that can be used to derive a set of closed form ordinary differential equations (ODEs), which can then be numerically integrated forward through time to find a solution to the dynamics given a specified input.

One such method of deriving the equations of motion is by using the *discrete coordinate* formulation which treats the spacecraft as a series of connected rigid bodies. This approach is very versatile as any satellite configuration can be represented. The more flexible components of the satellite will require a larger number of interconnected rigid bodies to sufficiently represent their flexibility [1, 5], and therefore this method can be computationally limited.

Also commonly used is the *hybrid coordinate* formulation which treats the spacecraft as a combination of rigid bodies, described with discrete coordinates, and flexible appendages, described using modal coordinates [1, 5]. This hybrid method has been used successfully in the design of several attitude control methods [6, 7]. However, use of the hybrid formulation will not result in any closed form solution and instead additional transformations through a discretisation procedure, such as lumped parameter [8], finite element [9], finite differencing [10] or assumed mode methods [11], is needed.

Much of attitude control literature on rigid bodies is applicable to satellites with flexible appendages with some additional considerations to ensure stability and minimise vibration in the appendages. State of the art approaches commonly include robust approaches such as sliding mode control [12, 13] and optimal based methods such as model predictive control [14, 15].

However, easy to implement, and widely understood, proportional-derivative (PD) and proportional-integral-derivative (PID) based control techniques still earn their place among more complex methods [16–18]. A three-axes PD attitude control method is developed by Baghi et al. [18] with novel gain scheduling using Gaussian functions of the satellite's attitude and angular velocity. The gain scheduling allowed for improved performance without requiring any additional adaptive elements or novel piezoelectric *smart materials*, as suggested by Liu et al. [17] and Song and Agrawal [16] respectively.

Inverse simulation (InvSim) is described as an iterative *numerical* process that attempts to map a timeseries/trajectory of desired output states, $\boldsymbol{y}_d(t)$, of a system to the required timeseries of inputs/controls, $\boldsymbol{u}(t)$, needed to follow said trajectory, as can be seen in Fig. 1. This can be thought of as the opposite mapping to a traditional forward simulation, Fig. 2, which instead maps the input/controls, $\boldsymbol{u}(t)$, to a timeseries out the system's output states, $\boldsymbol{y}_d(t)$. This allows for the dynamics for a system to be solved given a prescribed trajectory, which can be especially useful when trying to ascertain the performance characteristics of a proposed vehicle design and has had widespread applications in fields such as fixed wing aircraft [19, 20], and rotor-craft [21–23]. Some applications of InvSim in the space sector have been seen: orbital rendezvous and docking [24], planetary rovers [25], and most recently attitude control of rigid body satellites [26].

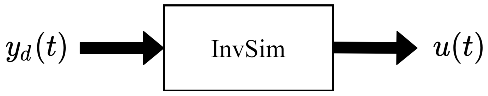


Fig. 1 Inverse Simulation (InvSim) Mapping

$u(t)$
FwdSim
$y(t)$

Fig. 2 Forward Simulation (FwdSim) Mapping

The most popular InvSim method, and the one used in this paper, is the *integral* InvSim algorithm [21, 22] known for its *versatile* and *generalised* solution process. This means that the algorithm is flexible and can be used to solve the inverse dynamics of any model without modifications to the solution process, allowing for accuracy of the results to be increased by simply introducing a more complex model. There exists other InvSim methods, such as the *differential* method [19, 27]. This method however requires the derivation of a new solution for every new model needing to be solved for and suffers significant numerical instabilities [21, 22]. *Analytical* inversion techniques such as nonlinear dynamic inversion (NDI) also exists and have been successfully applied to attitude control of both rigid [28] and flexible [29] satellites in the past. NDI however is limited to control affine systems (dynamics linear to controls) and similar to the *differential* InvSim method requires re-derivation of the solution for each new model.

This paper will explore the feasibility and practicality of InvSim techniques as an attitude control method for satellites with flexible appendages. Several test cases will be presented to show the versatility of the approach to satellite's with different actuator types. Issues related to real-time deployment of InvSim will be addressed through a proposed hybrid control structure, combining an offline InvSim feed-forward element and an online PD feedback element. The robustness and performance of the two control schemes will be tested through extensive Monte-Carlo simulations, varying uncertainties in the model and introducing external disturbance torques. Performance will be quantified through consideration of the attitude tracking error and total control effort. Also discussed will be possible practical uses of InvSim as a design tool/aid for satellites with flexible appendages, and the benefits in terms of the labour saved when considering the time consuming re-tuning required for many traditional control methods.

2. Mathematical Model

The mathematical model employed in this paper will be based on the *discrete coordinate* approach. To further simplify the model only the first order flexing dynamics will be considered through representation of the appendages as point masses attached to a rigid main body by two-degree-of-freedom torsional springs and damper hinges. Fig. 3 shows the specific case of a satellite with two flexible solar panels, however the approach is general for any n_{ap} number of appendages. Different types of appendages can also be modelled: eg. a solar panel will be more stiff about one of its hinges, whereas a long boom with a square cross section would be equally flexible about both hinges.

Several reference frames are defined: an arbitrary inertial frame ($\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, $\hat{\boldsymbol{z}}$), the body fixed frame ($\hat{\boldsymbol{x}}_b$, $\hat{\boldsymbol{y}}_b$, $\hat{\boldsymbol{z}}_b$) and a frame for each appendage ($\hat{\boldsymbol{x}}_i$, $\hat{\boldsymbol{y}}_i$, $\hat{\boldsymbol{z}}_i$) which have their origins defined at the position of the appendage's hinge $\boldsymbol{r}_{h_i/i}$. All of the vectors used herein are measured in the body-fixed frame axes, but velocities and accelerations will be relative to the arbitrary inertial frame. Each appendage is represented through a point mass (m_i), distance of the point mass from its hinge (d_i), spring constants ($k_{y,i}$, $k_{z,i}$), damping constants ($c_{y,i}$, $c_{z,i}$) and the position of the hinge attaching it to the main body ($\boldsymbol{r}_{h_i}$). These properties can be tuned to validate the model to experimental data, providing an accurate, closed form, computationally fast model of 3 degree-of-freedom rotational and $2n_{ap}$ degree-of-freedom flexible dynamics [30, 31] which can be used to rapidly develop and test the InvSim algorithm utilised in this paper. However, more complex models will also work with the same methodologies employed herein.

2.1. Dynamics

The kinetic energy $\mathcal{T}$, potential energy $\mathcal{V}$ and dissipation function $\mathcal{D}$ can be given as:

$$\mathcal{T} = \frac{1}{2}\boldsymbol{\omega}^T \mathbf{I}_{hub}\boldsymbol{\omega} + \frac{1}{2}\sum_{i=1}^{n} m_i \boldsymbol{v}_i^T \boldsymbol{v}_i \tag{1}$$

$$\mathcal{V} = \frac{1}{2}\sum_{i=1}^{n_{ap}} k_{y,i}\theta_{y,i}^2 + k_{z,i}\theta_{z,i}^2 \tag{2}$$

$$\mathcal{D} = \frac{1}{2}\sum_{i=1}^{n_{ap}} c_{y,i}\dot{\theta}_{y,i}^2 + c_{z,i}\dot{\theta}_{z,i}^2 \tag{3}$$

where $\mathbf{I}_{hub}$ is the inertia matrix of the main hub body of the satellite, $\boldsymbol{\omega}$ is the body rate vector of the satellite, $\boldsymbol{v}_i$ is the total inertial velocity of the appendage point mass, $\theta_{y,i}$ and $\theta_{z,i}$ are the appendage's deflection angles, $\dot{\theta}_{y,i}$ and $\dot{\theta}_{z,i}$ are the deflection angle velocities and n_{ap} is the total number of appendages attached to the main hub body.

The Euler-Lagrange equations that form the equations of motion are given as [32, p. 35]:

$$\frac{d}{dt}\left(\frac{\delta\mathcal{T}}{\delta\boldsymbol{\omega}}\right) = \boldsymbol{\tau}_{\boldsymbol{a}} + \boldsymbol{\tau}_d \tag{4}$$

$$\frac{d}{dt}\left(\frac{\delta\mathcal{T}}{\delta\dot{\theta}_{y,i}}\right) - \frac{\delta\mathcal{T}}{\delta\theta_{y,i}} + \frac{\delta\mathcal{V}}{\delta\theta_{y,i}} + \frac{\delta\mathcal{D}}{\delta\dot{\theta}_{y,i}} = 0 \tag{5}$$

$$\frac{d}{dt}\left(\frac{\delta\mathcal{T}}{\delta\dot{\theta}_{z,i}}\right) - \frac{\delta\mathcal{T}}{\delta\theta_{z,i}} + \frac{\delta\mathcal{V}}{\delta\theta_{z,i}} + \frac{\delta\mathcal{D}}{\delta\dot{\theta}_{z,i}} = 0 \tag{6}$$

where $\boldsymbol{\tau}_a$ is the actuator torque, and $\boldsymbol{\tau}_d$ is the external disturbance torque imparted on the main body of the satellite.

To solve the Euler-Lagrange equations, seen in Eqs. (4) to (6), the velocity, $\boldsymbol{v}_i$, and position $\boldsymbol{r}_i$ of the point mass representing each appendage is needed. The position of the i^{th} appendage in body-fixed axes can be given as:

$$\boldsymbol{r}_i = \boldsymbol{r}_{h_i} + \boldsymbol{r}_{h_i/i} \tag{7}$$

where $\boldsymbol{r}_{h_i}$ is the position of the two-degree-of freedom hinge attaching the appendage to the main hub body, and $\boldsymbol{r}_{h_i/i}$ is the vector pointing from the hinge position to the point mass of the appendage. The vector $\boldsymbol{r}_{h_i/i}$ can be defined in terms of the deflection angles and the distance from the hinge d_i:

$$\begin{aligned}\boldsymbol{r}_{h_i/i} = d_i\big[&\cos(\theta_{y,i})\cos(\theta_{z,i})\hat{\boldsymbol{x}}_i \\ &+ \sin(\theta_{i,z})\hat{\boldsymbol{y}}_i - \sin(\theta_{i,y})\hat{\boldsymbol{z}}_i\big]\end{aligned} \tag{8}$$

The total inertial velocity of the appendage can therefore be given as:

$$\boldsymbol{v}_i = \dot{\theta}_{y,i}\hat{\boldsymbol{y}}_i \times \boldsymbol{r}_{h_i/i} + \dot{\theta}_{z,i}\hat{\boldsymbol{z}}_i \times \boldsymbol{r}_{h_i/i} + \boldsymbol{\omega} \times \boldsymbol{r}_i \tag{9}$$

which can be re-written in terms of skew-symmetric matrices:

$$\boldsymbol{v}_i = \left[\boldsymbol{r}_{h_i/i}\right]_\times^T \left(\dot{\theta}_{y,i}\hat{\boldsymbol{y}}_i + \dot{\theta}_{z,i}\hat{\boldsymbol{z}}_i\right) + [\boldsymbol{\omega}]_\times \boldsymbol{r}_i \tag{10}$$

defined as:

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b} \tag{11a}$$

$$[\boldsymbol{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{11b}$$

Several further steps of differentiation and back substituting are then required to get the nonlinear equations of motion in the form of second order, ordinary differentiable equations (ODEs).

2.2. Attitude Kinematics

The attitude kinematics, ie the kinematics of the body fixed axis set ($\hat{\boldsymbol{x}}_b$, $\hat{\boldsymbol{y}}_b$, $\hat{\boldsymbol{z}}_b$) relative to the inertial frame ($\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, $\hat{\boldsymbol{z}}$), is defined through the scalar-first unit quaternions:

$$\boldsymbol{q} = \begin{bmatrix} q_0 \\ \boldsymbol{q}_{1:3} \end{bmatrix} \tag{12}$$

With the quaternion rate being related to the angular body rates thus:

$$\dot{\boldsymbol{q}} = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}_b^T \\ \boldsymbol{\omega} & [\boldsymbol{\omega}_b]_\times \end{bmatrix}\boldsymbol{q} \tag{13}$$

Also of interest within this paper will be the error quaternion, $\boldsymbol{q}_e$, between a desired quaternion attitude, $\boldsymbol{q}_d$, and the actual quaternion attitude $\boldsymbol{q}$, which can be calculated through the following quaternion multiplication:

$$\boldsymbol{q}_e = \boldsymbol{q}_d^* \odot \boldsymbol{q} \tag{14}$$

where $\boldsymbol{q}^*$ is the conjugate of the quaternion $\boldsymbol{q}$ defined as:

$$\boldsymbol{q}^* = \begin{bmatrix} q_0 \\ -\boldsymbol{q}_{1:3} \end{bmatrix} \tag{15}$$

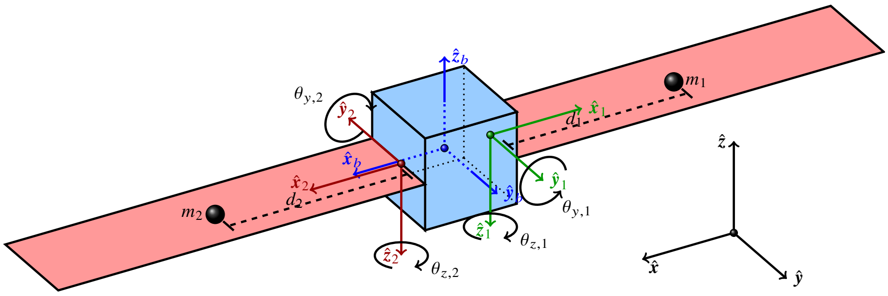


Fig. 3 Mathematical Model: Frames and components

Additionally, the quaternion multiplication used, $\odot$, is the convention shown in [33] with modification to be *scalar first* in order to match the convention used in MATLAB, the software of choice for the work herein:

$$\bar{\boldsymbol{q}} \odot \tilde{\boldsymbol{q}} = \begin{bmatrix} \bar{q}_0 \tilde{q}_0 - \bar{\boldsymbol{q}}_{1:3}^T \tilde{\boldsymbol{q}}_{1:3} \\ \tilde{q}_0 \bar{\boldsymbol{q}}_{1:3} + \bar{q}_0 \tilde{\boldsymbol{q}}_{1:3} + \left[\bar{\boldsymbol{q}}_{1:3}\right]_\times \tilde{\boldsymbol{q}}_{1:3} \end{bmatrix} \tag{16}$$

We can also get the attitude tracking error, or the angle between the desired and actual quaternions by taking the inverse cosine of the scalar element of the quaternion error, $\boldsymbol{q}_e$:

$$\vartheta_e = 2\cos^{-1}(q_{e,0}) \tag{17}$$

This value will be used to quantify the tracking performance of the control methods presented later in this paper.

3. Integral Inverse Simulation Algorithm

The integral InvSim algorithm being employed in this paper is based largely on the original work by Hess and Gao [21] and can be seen detailed in Fig. 4. The first step (step 1 in Fig. 4) is to obtain a discretized time-series of the set of desired output states, $\boldsymbol{y}_d[kT]$, we wish the InvSim to follow. The integral InvSim algorithm will then iteratively solve for each timestep kT, resulting in an input/control solution time-series $\boldsymbol{u}[kT]$, where T is the control timestep duration, and k is the current step in the time-series.

The InvSim problem at each timestep is therefore reduced to the problem of finding $\boldsymbol{u}[kT]$ such that:

$$\boldsymbol{y}[(k+1)T] = \boldsymbol{y}_d[(k+1)T] \tag{18}$$

where $\boldsymbol{y}[(k+1)T]$ is the actual output states of the system T seconds after the input/control, $\boldsymbol{u}[kT]$, is applied, and $\boldsymbol{y}_d[(k+1)T]$ is the desired output states T seconds after the input/control is applied.

To solve for $\boldsymbol{u}[kT]$ for every step k a numerical process, such as the Newton-Raphson (NR) method employed herein, is needed. The NR algorithm (steps 6 to 12 in Fig. 4) will first calculate a Jacobian, $\mathbf{J}_n$, of first order partial differential equations of each output state, y_i, with respect to each input/control, u_j, (step 8 in Fig. 4):

$$\mathbf{J}_n = \begin{bmatrix} \frac{\delta y_1}{\delta u_1} & \frac{\delta y_1}{\delta u_2} & \cdots & \frac{\delta y_1}{\delta u_{n_u}} \\ \frac{\delta y_2}{\delta u_1} & \frac{\delta y_2}{\delta u_2} & \cdots & \frac{\delta y_2}{\delta u_{n_u}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta y_{n_y}}{\delta u_1} & \frac{\delta y_{n_y}}{\delta u_2} & \cdots & \frac{\delta y_{n_y}}{\delta u_{n_u}} \end{bmatrix} \tag{19}$$

where n_u is the total number of inputs/controls and n_y is the total number of output states. The partial differential equations can be calculated numerically through central differencing:

$$\frac{\delta y_i}{\delta u_j} = \frac{y_i[u_j + \Delta u_j]|_{k+1} - y_i[u_j - \Delta u_j]|_{k+1}}{2\Delta u_j}. \tag{20}$$

where Δu_j is a small perturbation applied to the j^{th} element of the control vector. This perturbation size can be determined simply based on the previous input/control solution guess:

$$\Delta \boldsymbol{u} = T\boldsymbol{u}_{n-1} \tag{21}$$

where n is the current NR iteration as seen in Fig. 4. However, on the first iteration of the first time-step, when $k = 1$ and $n = 1$, there will be no previous input/control guess to use. Therefore an initial control guess needs to be defined to start the InvSim solution process (step 2 in Fig. 4). Additionally a minimum perturbation size Δu_{min} should be chosen, based on the speed of the system dynamics, in order to ensure sufficient perturbation in the output states and avoid numerical issues such as similar differencing:

$$\Delta u_j > \Delta u_{min} \tag{22}$$

The Jacobian, $\mathbf{J}_n$, can therefore be calculated through numerically integrating the perturbed input/controls through the model over the control time-step T and applying Eq. (20). Selection of a smaller integration time-step dt may also be required to ensure that the full dynamics of the system are captured and that the numerical integration is stable. Through inversion of the Jacobian the next best guess at the control solution can be calculated (step 9 in Eq. (23))using:

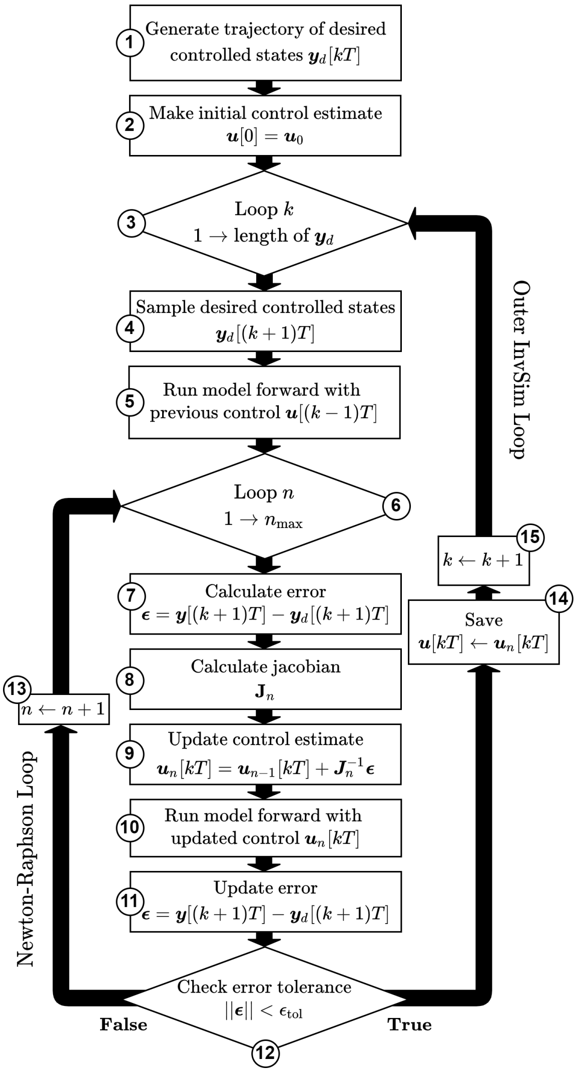

1
Generate trajectory of desired controlled states $\boldsymbol{y}_d[kT]$
2
Make initial control estimate $\boldsymbol{u}[0] = \boldsymbol{u}_0$
3
Loop k
$1 \rightarrow$ length of $\boldsymbol{y}_d$
4
Sample desired controlled states $\boldsymbol{y}_d[(k+1)T]$
5
Run model forward with previous control $\boldsymbol{u}[(k-1)T]$
6
Loop n
$1 \rightarrow n_{\max}$
7
Calculate error $\boldsymbol{\epsilon} = \boldsymbol{y}[(k+1)T] - \boldsymbol{y}_d[(k+1)T]$
8
Calculate jacobian $\mathbf{J}_n$
9
Update control estimate $\boldsymbol{u}_n[kT] = \boldsymbol{u}_{n-1}[kT] + \boldsymbol{J}_n^{-1}\boldsymbol{\epsilon}$
10
Run model forward with updated control $\boldsymbol{u}_n[kT]$
11
Update error $\boldsymbol{\epsilon} = \boldsymbol{y}[(k+1)T] - \boldsymbol{y}_d[(k+1)T]$
12
Check error tolerance $||\boldsymbol{\epsilon}|| < \epsilon_{\text{tol}}$
False
True
13
$n \leftarrow n+1$
14
Save $\boldsymbol{u}[kT] \leftarrow \boldsymbol{u}_n[kT]$
15
$k \leftarrow k+1$
Newton-Raphson Loop
Outer InvSim Loop

Fig. 4 Integral Inverse Simulation (InvSim) Algorithm

$$\boldsymbol{u}_n[kT] = \boldsymbol{u}_{n-1}[kT] + \mathbf{J}_n^{-1}\boldsymbol{\epsilon} \qquad (23)$$

where the error $\boldsymbol{\epsilon}$ is given as the difference between the actual and desired output states:

$$\boldsymbol{\epsilon} = \boldsymbol{y}[(k+1)T] - \boldsymbol{y}_d[(k+1)T] \qquad (24)$$

The error can be obtained by simply integrating forward the model with the previous best guess at the input/control solution, $\boldsymbol{u}_{n-1}[kT]$ (step 5 in Fig. 4). The updated control guess, $\boldsymbol{u}_n[kT]$, can then be integrated forward over T, using Runge-Kutta numerical integration, to obtain updated output states $\boldsymbol{y}[(k+1)T]$ which can be used to update the error $\boldsymbol{\epsilon}$ (step 10 and 11 in Fig. 4). The error is then compared to a prescribed tolerance ϵ_{tol} (step 12 in Fig. 4):

$$||\boldsymbol{\epsilon}|| < \epsilon_{tol} \qquad (25)$$

If the tolerance is met, then the current input/control guess is accepted as the solution and the InvSim loop iterates onto the next timestep that needs to be solved (steps 14 and 15 in Fig. 4). If the tolerance is not met then the NR loop iterates (step 13 in Fig. 4) and continues to update the control solution until either the tolerance is met or the maximum number of NR loops is met, n_{max}.

When the number of inputs/controls is equal to the number of output states, $n_u = n_y$, then the Jacobian, $\mathbf{J}_n$, is square allowing for the inverse to be taken explicitly in Eq. (23). This is known as the *nominal case*. When the number of inputs/controls exceeds the number of output states, $n_u > n_y$, then the Jacobian is non-square and a Moore-Penrose pseudo-inverse [34] must be used [21], this is known as the *redundant case*. The *underactuated* case, when there are less inputs/controls than the output states $n_u < n_y$, can also be solved using the Moore-Penrose pseudo-inverse. However, this will only achieve a least-squares-fit solution to the desired output states trajectory. In some cases this may provide a suitable solution, however, in other cases considerable divergence from the desired output states may occur and performance of the InvSim cannot be guaranteed. An additional check can be added to the InvSim loop that will flag the trajectory as unrealisable in the case where the error $\boldsymbol{\epsilon}$ exceeds some bounds greater than ϵ_{tol}.

It may also be desire-able to ensure that the input/-control solution does not exceed some control saturation, u_{sat}, of the actuators being used. This can simply be achieved through a subroutine within the "Outer InvSim loop" (see Fig. 4) that checks the input/control solution of the current timestep against the saturation which if exceeded, $any(\boldsymbol{u}[kT] > u_{sat})$, will trigger the InvSim algorithm to stop. There also exists the case where the inherent dynamics of the system prevents convergence to a suitable solution, meaning that the given desired output state trajectory is impossible. For the *nominal case* this causes the calculated Jacobian to become singular, determinant becomes zero, meaning that there exist no solution. An modification can be made to the algorithm to check when the Jacobian's determinant becomes zero and trigger the InvSim algorithm to stop. Additionally for the *redundant* and *under-actuated* cases this will simply cause the input/control guess to exponentially approach infinity when executing the Moore-Penrose pseudo-inverse, which can be caught by the inclusion of a control saturation u_{sat}, which can be set arbitrarily high when none really exists.

4. Inverse simulation of satellites with flexible appendages

This section will look at applying the Integral InvSim algorithm to the problem of attitude control of a satellite with flexible appendages. Presented first will be the definition of the quaternion slew paths to be targeted by the InvSim algorithm and the model properties used are detailed. Results for three cases will then be presented:

- *Nominal, control affine*: $n_u = n_y$, and linear to inputs

- *Under-actuated, control affine*: $n_u < n_y$, and linear to inputs
- *Redundant, control non-affine*: $n_u > n_y$, and nonlinear to inputs

This allows for the application of the InvSim algorithm to a wide selection of satellite actuator configurations to be addressed as-well as the flexibility of the method.

4.1. Large angle attitude slew path generation

First, a definition of the desired attitude trajectory used to drive the InvSim solution is required. A rest-to-rest large-angle slew manoeuvre can be defined through spherical linear interpolation (SLERP) [35] giving the desired quaternion path $\boldsymbol{q}_d(t)$:

$$\boldsymbol{q}_d(t) = \frac{\sin((1-\phi(t))\Phi_0)}{\sin(\Phi_0)}\boldsymbol{q}_i + \frac{\sin(\phi(t)\Phi_0)}{\sin(\Phi_0)}\boldsymbol{q}_f \tag{26}$$

where Φ_0 is the angle between the initial and final quaternions, $\boldsymbol{q}_i$ and $\boldsymbol{q}_f$ respectively:

$$\Phi_0 = \cos^{-1}(\boldsymbol{q}_i \cdot \boldsymbol{q}_f) \tag{27}$$

The function $\phi(t)$ returns a value between 0 and 1 which defines the fraction of the quaternion path that has been traversed, with $\phi(t) = 0$ giving $\boldsymbol{q}(t) = \boldsymbol{q}_i$ and $\phi(t) = 0$ giving $\boldsymbol{q}(t) = \boldsymbol{q}_f$. This function can be defined as a 7th order polynomial to ensure a smooth rest-to-rest manouvre:

$$\phi(t) = a_4t^7 + a_3t^6 + a_2t^5 + a_1t^4 \tag{28}$$

where the coefficients can be determined through the time given to complete the slew manoeuvre t_{move}:

$$a_4 = \frac{-20}{t_{move}^7} \tag{29a}$$

$$a_2 = \frac{-84}{t_{move}^5} \tag{29c}$$

$$a_3 = \frac{70}{t_{move}^6} \tag{29b}$$

$$a_1 = \frac{35}{t_{move}^4} \tag{29d}$$

As has been discussed in many previous papers [22, 23, 26], issues of numerical stability and high frequency oscillations in the control solution can be greatly limited by the use of higher-order derivatives as the desired output states $\boldsymbol{y}_d$, or by increasing the integration (dt) and control timesteps (T) to avoid excitation of uncontrolled dynamics. Higher-order derivatives such as velocity and acceleration will be more sensitive to perturbations in the control inputs thus producing a more stable Jacobian. The desired quaternion rate, $\dot{\boldsymbol{q}}_d(t)$, for the large angle slew manoeuvres can therefore be found through simple time differentiation:

$$\dot{\boldsymbol{q}}_d(t) = \frac{-\dot{\phi}(t)\Phi_0\cos((1-\phi(t))\Phi_0)}{\sin(\Phi_0)}\boldsymbol{q}_i + \frac{\dot{\phi}(t)\Phi_0\cos(\phi(t)\Phi_0)}{\sin(\Phi_0)}\boldsymbol{q}_f \tag{30}$$

With the derivative of the polynomial being:

$$\dot{\phi}(t) = 7a_4t^6 + 6a_3 + 5a_2t^4 + 4a_1t^3 \tag{31}$$

The trajectory can therefore be defined through $\boldsymbol{q}_i$, $\boldsymbol{q}_f$, t_{move} and additionally a time-to-hold t_{hold} which is a duration which the InvSim will attempt to hold the final quaternion, i.e $\dot{\boldsymbol{q}} = \boldsymbol{0}$.

4.2. Test scenario

A satellite with two flexible boom-type appendages, equally flexible about both hinges, is represented through the model properties seen in Table 1. In this case the damping in each appendage is also set to zero. The parameters defining the slew manoeuvre for the InvSim to target can be seen in Table 2. The final quaternion $\boldsymbol{q}_f$ is equivalent to two 45° rotations about the body y and z axes.

This manoeuvre and model properties will be used for the rest of this section to provide a demonstration of the InvSim algorithm.

4.3. Nominal, control affine case

The desired output states to be targeted are chosen to be the vector component of the quaternion path:

$$\boldsymbol{y}_d(t) = \boldsymbol{q}_{d,1:3}(t) \tag{32}$$

Here the InvSim problem can be defined nominal, control affine by selecting the input/controls to simply be the equivalent actuator torque $\boldsymbol{\tau}_a$, seen in Section 2:

$$\boldsymbol{\tau}_a = \boldsymbol{u} \tag{33}$$

The InvSim parameters used are detailed in Table 3. The integration timestep, dt, is chosen sufficiently small to capture all of the system dynamics [22] while ensuring that it is not small enough to excite the uncontrolled dynamics of the system which can lead to high frequency oscillations in the control signal [36]. The use of Runge-Kutta integration allows for sufficient propagation of the quaternion rates with the control timestep $T = dt$ and provides a smooth continuous control signal. Using an initial control guess $\boldsymbol{u}_0$ of all zeros is sufficient since we are considering a rest-to-rest manoeuvre.

The minimum perturbation size, Δu_{min}, is chosen to sufficiently perturb the output states of the system and therefore is dependant on the speed of the dynamics, in this case defined primarily by the main hub inertia $\mathbf{I}_{hub}$, appendage mass m_i and distance of the masses from their hinges d_i. Choosing too large a Δu_{min} could reduce the accuracy of the partial derivatives calculated for the Jacobian and result in more NR iterations needed to meet the tolerance required. Too small a Δu_{min} however can result in numerical errors known as *similar differencing* when performing the numerical differentiation. The NR tolerance, ϵ_{tol}, is chosen to be sufficiently small to ensure accuracy of the InvSim solution while minimising the

Table 1 Model Properties

Property	Value	
	Appendage 1	Appendage 2
Mass, m_i | kg	100	100
y-axis spring constant, $k_{y,i}$ | Nm/rad	300	300
z-axis spring constant, $k_{z,i}$ | Nm/rad	300	300
y-axis damping constant, $c_{y,i}$ | Nm/(rad/s)	0	0
z-axis damping constant, $c_{z,i}$ | Nm/(rad/s)	0	0
Mass distance from hinge, d_i | m	1.5	1.5
Appendage hinge position, $\boldsymbol{r}_{h_i}$ | m	$[-1\ 0\ 0]^T$	$[1\ 0\ 0]^T$
Appendage x-axis, $\hat{\boldsymbol{x}}_i$	$[-1\ 0\ 0]^T$	$[1\ 0\ 0]^T$
Appendage y-axis, $\hat{\boldsymbol{y}}_i$	$[0\ 1\ 0]^T$	$[0\ -1\ 0]^T$
Appendage z-axis, $\hat{\boldsymbol{z}}_i$	$[0\ 0\ -1]^T$	$[0\ 0\ -1]^T$
Main hub inertia, $\mathbf{I}_{hub}$ | kgm^2	$\begin{bmatrix} 500 & -10 & -20 \\ -10 & 400 & -30 \\ -20 & -30 & 350 \end{bmatrix}$	
External disturbance torque, $\boldsymbol{\tau}_d$ | Nm	$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$	

Table 2 Slew manoeuvre parameters

Parameter	Value
Initial quaternion, $\boldsymbol{q}_i$	$[1\ 0\ 0\ 0]^T$
Final quaternion, $\boldsymbol{q}_f$	$[0.8536\ \text{-}0.1464\ 0.3536\ 0.3536]^T$
Time-to-move, t_{move}	60 s
Time-to-hold, t_{hold}	30 s

Table 3 Inverse simulation parameters: Nominal, control affine case

Parameter	Value
Control timestep, T	0.01 s
Integration timestep, dt	0.01 s
Initial control guess, $\boldsymbol{u}_0$	$[0\ 0\ 0]^T$ Nm
Minimum control perturbation, Δu_{min}	10 Nm
NR Error tolerance, ϵ_{tol}	1×10^{-16}
Maximum NR iterations, n_{max}	10

number of NR iterations executed. Additionally, n_{max}, is chosen to allow for the error to be minimised without wasting additional NR iterations with no improvement in the solution. Minimising the number of NR iterations while maximising accuracy of the InvSim has important implications when considering real-time deployment as an attitude control system.

The results of the nominal, control affine InvSim case can be seen in Fig. 5. Fig. 5b shows the smooth control solution produced and it can be seen that the desired quaternion path is perfectly followed in Fig. 5a. The deflection angle responses of both the appendages can be seen in Figs. 5c and 5d.

4.4. *Under-actuated, control affine case*

Secondly considered is the under-actuated, control affine case. The desired output states are again selected as the vector component of the quaternion path:

$$\boldsymbol{y}_d(t) = \boldsymbol{q}_{d,1:3}(t) \tag{34}$$

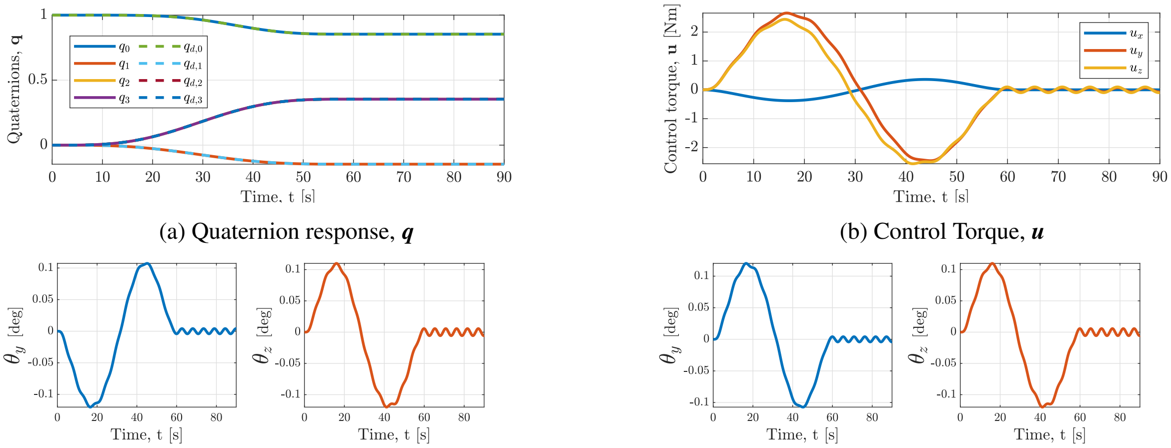
The input/control vector is mapped into the body axes using the actuator distribution matrix **B**:

$$\boldsymbol{\tau}_a = \mathbf{B}\boldsymbol{u} \tag{35}$$

Two inputs/controls are provided, orientated about the body $\hat{\boldsymbol{y}}_b$ and $\hat{\boldsymbol{z}}_b$ axes, giving:

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{36}$$

For the *under-actuated* case, $n_u < n_y$, the integration timestep, dt, and control timestep, T, are both increased, as can be seen in Table 4, to avoid exciting uncontrolled dynamics which leads to instability in the numerical process and/or high frequency oscillations in the control solution. The dimensions of the initial control guess, $\boldsymbol{u}_0$, is altered to match the number of input/controls with the rest of the InvSim parameters being identical to the *nominal case*.



(a) Quaternion response, $\boldsymbol{q}$

(b) Control Torque, $\boldsymbol{u}$

(c) Appendage 1 Deflections

(d) Appendage 2 Deflections

Fig. 5 InvSim solution: Nominal, control affine case

Table 4 Inverse simulation parameters: Under-actuated & control affine case

Parameter	Value
Control timestep, T	0.1 s
Integration timestep, dt	0.1 s
Initial control guess, $\boldsymbol{u}_0$	$[0\ 0]^T$ Nm
Minimum control perturbation, Δu_{min}	10 Nm
NR Error tolerance, ϵ_{tol}	1×10^{-16}
Maximum NR iterations, n_{max}	10

The results of the under-actuated, control affine case can be seen in Fig. 6 with a control solution successfully being converged upon, as can be seen in Fig. 6b. However, the under-actuated system is unable to closely follow the prescribed quaternion path, Fig. 6a, as the under-actuated system doesn't have sufficient control authority about all axes. It could be possible that through reduction of the constraints, eg. only constraining motion about two axes, that a more suitable solution could be found for this under-actuated system. The deflection angle responses can again be seen in Figs. 6c and 6d.

4.5. *Redundant, control non-affine case*

Consideration of actuator dynamics, the motor of a reaction wheel in this case, can simply be taken into account with the addition of the following equations of motion:

$$\boldsymbol{\tau}_a = -\mathbf{B}\dot{\boldsymbol{h}}_w - \boldsymbol{\omega}_b \times (\mathbf{B}\boldsymbol{h}_w) \tag{37}$$

$$\ddot{\boldsymbol{h}}_w = \frac{\boldsymbol{u} - \dot{\boldsymbol{h}}_w}{T_w} \tag{38}$$

where $\boldsymbol{h}_w$ is a vector of reaction wheel angular momenta, $\dot{\boldsymbol{h}}_w$ is a vector of reaction wheel torques, $\ddot{\boldsymbol{h}}_w$ is the rate of change of the reaction wheel torques, and T_w is the time constant of the reaction wheel actuator dynamics. Inclusion of these actuator dynamics makes the system control non-affine. The *redundant* InvSim problem, $n_u > n_y$, can be defined through the actuator distribution matrix $\mathbf{B}$:

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 1/\sqrt{3} \\ 0 & 1 & 0 & 1/\sqrt{3} \\ 0 & 0 & 1 & 1/\sqrt{3} \end{bmatrix} \tag{39}$$

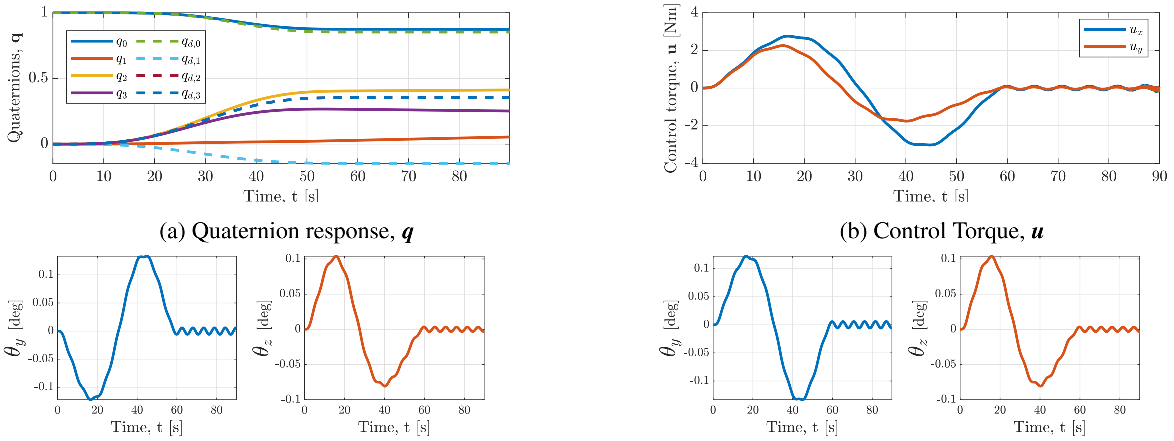
There are therefore four input/controls to the system, and the desired output states are chosen as the vector component of the quaternion rate of the quaternion path:

$$\boldsymbol{y}_d(t) = \dot{\boldsymbol{q}}_{d,1:3}(t) \tag{40}$$

Use of the higher-order derivative output states improves the numerical stability, especially in the case of a control non-affine system due to their greater sensitivity to perturbations in the inputs/controls. The other alternative would be to significantly increase the control timestep, T, to allow propagation of the input/control perturbations into the attitude quaternions which would result in a non-continuous control signal. A requirement for using higher-order derivatives of the quaternion path is that the attitude path *must* be realisable by the attitude dynamics. If they are not, as was seen in the *under-actuated case*, then the quaternions will diverge significantly from the desired path over time.

The InvSim parameters used for the redundant, control non-affine case can be seen in Table 5. The initial control guess $\boldsymbol{u}_0$ is chosen again to match the number of inputs/controls needed by the system.

The results of the redundant, control non-affine case can be seen in Fig. 7. The produced control solution is smooth and continuous, Fig. 7b, with the quaternion path being closely followed, Fig. 7a. The deflection angle responses for both appendages can again be seen in Figs. 7c and 7d.



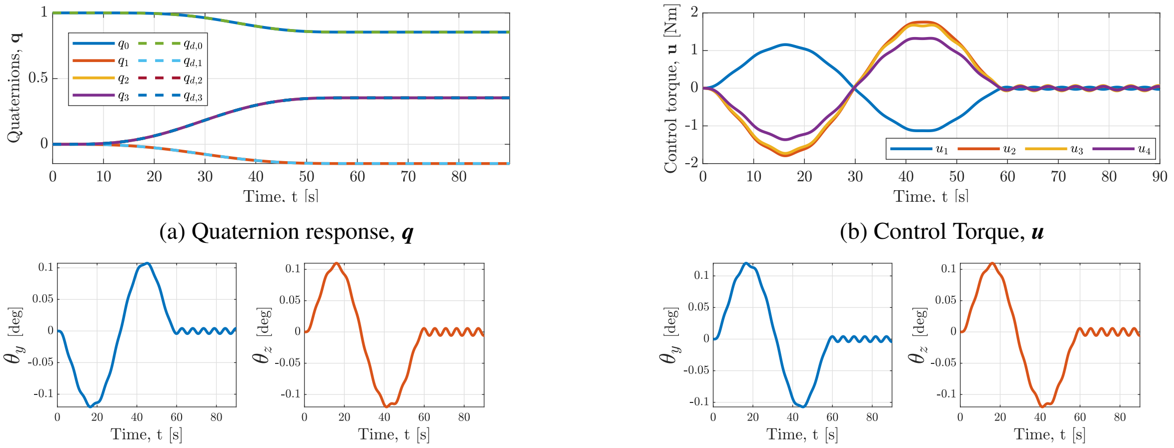
(a) Quaternion response, $\boldsymbol{q}$

(b) Control Torque, $\boldsymbol{u}$

(c) Appendage 1 Deflections

(d) Appendage 2 Deflections

Fig. 6 InvSim solution: Under-actuated, control affine case



(a) Quaternion response, $\boldsymbol{q}$

(b) Control Torque, $\boldsymbol{u}$

(c) Appendage 1 Deflections

(d) Appendage 2 Deflections

Fig. 7 InvSim solution: Redundant, control non-affine case

Table 5 Inverse simulation parameters: Redundant, control non-affine case

Parameter	Value
Control timestep, T	0.01 s
Integration timestep, dt	0.01 s
Initial control guess, $\boldsymbol{u}_0$	$[0\ 0\ 0\ 0]^T$ Nm
Minimum control perturbation, Δu_{min}	10 Nm
NR Error tolerance, ϵ_{tol}	1×10^{-16}
Maximum NR iterations, n_{max}	10

5. Practical inverse simulation attitude control method

As was seen in Section 4 InvSim was able to find a suitable control solution for attitude control of a satellite. However, there are some significant practical considerations when attempting to apply InvSim as an attitude control solution. Primarily the issue is to do with the extensive computational power and time needed to complete the InvSim solution. Even with the simple mathematical model used in this paper the computation was slower than real-time, taking approximately 120 seconds to solve a 90 second manoeuvre *. It can be imagined that if a more complex finite-element model was to be solved for then the computation time would be significantly increased. Therefore, real-time calculation and deployment of the control solution seems highly infeasible with the current algorithm.

The most obvious solution to this issue would be to simply calculate the control solution offline using InvSim and then execute the manoeuvre once computation is completed. This solution is effectively *open-loop* and therefore there is no robustness to uncertainty in the model and disturbances. Proposed in this section is a simple hybrid

*MATLAB 2021a running on: Intel(R) Core(TM) i7-11850H @ 2.50GHz, 16GB RAM, 64-bit Windows 10 controller that combines a traditional linear PD feedback controller with a feedforward InvSim term. This control structure has previously been posed by Lu et al. [37], however has not been yet applied to the control of satellite attitude. The InvSim feedforward will provide the majority of the control effort, dealing with the nonlinear dynamics, the PD feedback will account for the model uncertainty and disturbances experienced by the system. A regular PD controller will be used to compare performance in terms of tracking error and total control effort.

The model parameters seen in Table 1 and additional control non-affine reaction wheel dynamics seen in Section 4.5 are used for the remainder of this section.

5.1. PD controller

The quaternion feedback PD controller used to compare against the hybrid controller can be seen in Fig. 8 with control law:

$$\boldsymbol{u} = \boldsymbol{u}_{PD} = \mathbf{B}^T \left(-\mathbf{K_P}\, \boldsymbol{q}_{e,1:3} - \mathbf{K_D} \frac{d\boldsymbol{q}_{e,1:3}}{dt} \right) \tag{41}$$

where $\mathbf{B}^T$ is the transpose of the actuator distribution matrix seen in Eq. (39), $\boldsymbol{q}_{e,1:3}$ is the vector component of the quaternion error defined in Eq. (14), $\mathbf{K_P}$ is the proportional gain matrix and $\mathbf{K_D}$ is the derivative gain matrix. The gain matrices are tuned using MATLAB through linearisation of the satellite dynamics at rest and placing the poles to achieve a bandwidth of 0.8 rad/s and 60°phase margin. The tuned gains are given in Table 6.

Table 6 PD control gains

Proportional gain, $\mathbf{K_P}$	$diag(320\ 1800\ 1800)$
Derivative gain, $\mathbf{K_D}$	$diag(690\ 3900\ 3800)$

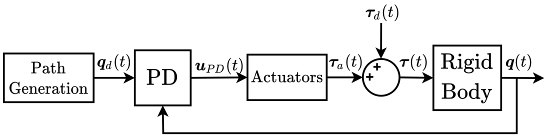


Fig. 8 PD feedback controller

5.2. Hybrid controller

The hybrid controller design can be seen in Fig. 9 which combines the PD feedback control law, $\boldsymbol{u}_{PD}$, seen in Eq. (41) with an offline InvSim feedforward term, $\boldsymbol{u}_{InvSim}$ giving the full control law:

$$\begin{aligned}\boldsymbol{u} &= \boldsymbol{u}_{InvSim} + \boldsymbol{u}_{PD}\\ &= \boldsymbol{u}_{InvSim} + \mathbf{B}^T \left(-\mathbf{K_P}\, \boldsymbol{q}_{e,1:3} - \mathbf{K_D} \frac{d\boldsymbol{q}_{e,1:3}}{dt} \right)\end{aligned} \tag{42}$$

The InvSim term $\boldsymbol{u}_{InvSim}$ is calculated offline before the execution of the manoeuvre, with the PD feedback term $\boldsymbol{u}_{PD}$ being executed online in real-time during the slew manoeuvre.

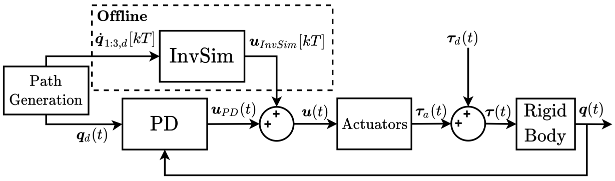


Fig. 9 Hybrid Controller: InvSim feedforward & PD feedback

5.3. Monte-Carlo simulations

In the previous results, seen in Section 4, it was assumed that the model was a perfect representation of the satellite's dynamics and that no disturbances were present. In reality there is always going to exist some uncertainty in the model and external disturbances are commonly present. Monte-Carlo simulations are therefore performed to study the effects of parametric model uncertainties and external disturbance torques on the the tracking error and total actuator effort used during the manoeuvre. The external disturbance torques, $\boldsymbol{\tau}_d$, seen in both Figs. 8 and 9 are given as:

$$\boldsymbol{\tau}_d = \begin{bmatrix} D_x\left(\sin(\Omega_x t + \Psi_x) + N_x(t)\right) \\ D_y\left(\sin(\Omega_y t + \Psi_y) + N_y(t)\right) \\ D_z\left(\sin(\Omega_z t + \Psi_z) + N_z(t)\right) \end{bmatrix} \tag{43}$$

where $(D_x\ D_y\ D_z)$ are the disturbance torque magnitudes, $(\Omega_x\ \Omega_y\ \Omega_z)$ are the angular rates of the sinusoidal disturbances, $(\Psi_x\ \Psi_y\ \Psi_z)$ are the phase shifts, and $(N_x(t)\ N_y(t)\ N_z(t))$ are Gaussian noise signals with standard deviations: $(\sigma_x\ \sigma_y\ \sigma_z)$. For each iteration of the Monte-Carlo simulation the parameters of the disturbances are uniformly randomised within the bounds expressed in Table 7. These parameters are selected relative to the magnitude of the control torque to resemble disturbances similar to those seen in [38, 39]. An example of the external disturbance torques used in the Monte-Carlo simulation can be seen in Fig. 10.

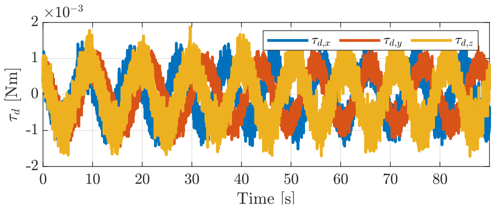


Fig. 10 External disturbance torque example, $\boldsymbol{\tau}_d$

The parametric model uncertainties included in the Monte-Carlo simulations are:

Table 7 Disturbance torque parameters

Parameter	Lower bound	Upper bound
Disturbance torque magnitudes, $(D_x\ D_y\ D_z)$ | Nm	1×10^{-4}	1×10^{-3}
Disturbance angular rates, $(\Omega_x\ \Omega_y\ \Omega_z)$ | rad/s	0.01	1
Disturbance phase shift, $(\Psi_x\ \Psi_y\ \Psi_z)$ | rad	0	π
Noise standard deviation, $(\sigma_x\ \sigma_y\ \sigma_z)$ | Nm	0.1	$0.3\bar{3}$

- ± 10 % uniform uncertainty applied to all elements of the hub inertia, $\mathbf{I}_{hub}$
- ± 10 % uniform uncertainty applied to the reaction wheel time constant, T_w
- ± 10 % uniform uncertainty applied to the appendage mass distances from their hinges, d_i
- ± 10 % uniform uncertainty to the appendage hinge spring constants and damping constants, k_y, k_z, c_y, c_z
- ± 5 °uniform uncertainty applied to alignment of the actuator distribution matrix, **B**
- ± 5 °uniform uncertainty applied to alignment of the appendage axis set, $\hat{\boldsymbol{x}}_i$, $\hat{\boldsymbol{y}}_i$, $\hat{\boldsymbol{z}}_i$
- Uniform uncertainty in the appendage hinge positions, $\boldsymbol{r}_{h_i}$: ± 10 % applied to the distance of the hinge from the body axis origin, and ± 5°rotation about the body axis set

10,000 simulations are run, 1,000 simulations for ten different slew path manoeuvres. The final quaternion $\boldsymbol{q}_f$ for each slew path is equivalent to a set of $Z-Y-X$ Euler rotations about the body axes, by angles $(\psi_z\ \psi_y\ \psi_x)$. The parameters for the ten slew paths can be seen in Table 8 and are labelled (a) through (j).

Table 8 Monte-Carlo simulation: Slew path parameters

Final Euler Angle, $[\psi_z\ \psi_y\ \psi_x]_f$ (°)	Time-to-move, t_{move} (s)	Time-to-hold, t_{hold} (s)
(a) [45 45 0]	60	30
(b) [90 0 0]	60	30
(c) [0 90 0]	60	30
(d) [0 0 90]	60	30
(e) [30 60 90]	60	30
(f) [−50 − 95 30]	60	30
(g) [85 − 10 120]	60	30
(h) [−140 − 170 115]	60	30
(i) [145 − 70 − 95]	60	30
(j) [165 − 145 − 10]	60	30

5.4. Results

The Monte-Carlo results comparing the performance of the PD and Hybrid controllers, subject to model uncertainty and disturbances, for each of the ten manoeuvres are shown in Figs. 11 and 12 with labelling corresponding to that seen in Table 8. The absolute tracking error $|\vartheta_e|$ can be obtained using Eq. (17), then taking the average/mean across the duration of the slew path a *mean absolute tracking error* can be obtained $|\bar{\vartheta_e}|$. The frequency of different *mean absolute tracking errors* can then be plotted on a histogram for each of the ten slew paths, comparing the hybrid and PD controllers as seen in Fig. 11. Similarly the *total control effort/energy* across the duration of each slew path is defined as:

$$\int \left(\sum_{i=1}^{n_u} u_i^2 \right) dt \tag{44}$$

The frequency of different *total control efforts* can then be plotted on a histogram for each of the ten slew paths as seen in Fig. 12.

First it can be seen that both the Hybrid and PD controllers are robust to the bounded disturbances and model uncertainties that they were subjected to. It can be seen clearly that the Hybrid controller produces a significantly smaller tracking error across all the manoeuvres thanks to its additional InvSim feedforward term. The reduction in error achieved by simply adding this InvSim feedforward term ranges from approximately a 91.6% to a 98.2% reduction in the mean tracking error. The Hybrid controller's performance is also more consistent with the mean tracking error rarely exceeding 0.05°, the PD controller's performance varies significantly more between different manoeuvres. For example, it can be seen in Fig. 11g that the PD controller's mean tracking error is reduced with its distribution moved closer to the Hybrid controller and with reduced variance. It is likely that the PD controller's tuned gains are better suited for this particular manoeuvre, and since the gains are static some of the performance will be lost the further from this optimal condition the system deviates. It can therefore be posed that much of the advantage of the Hybrid controller could be compensated for by the PD controller through appropriated gain scheduling, ensuring the gains are always optimal for the

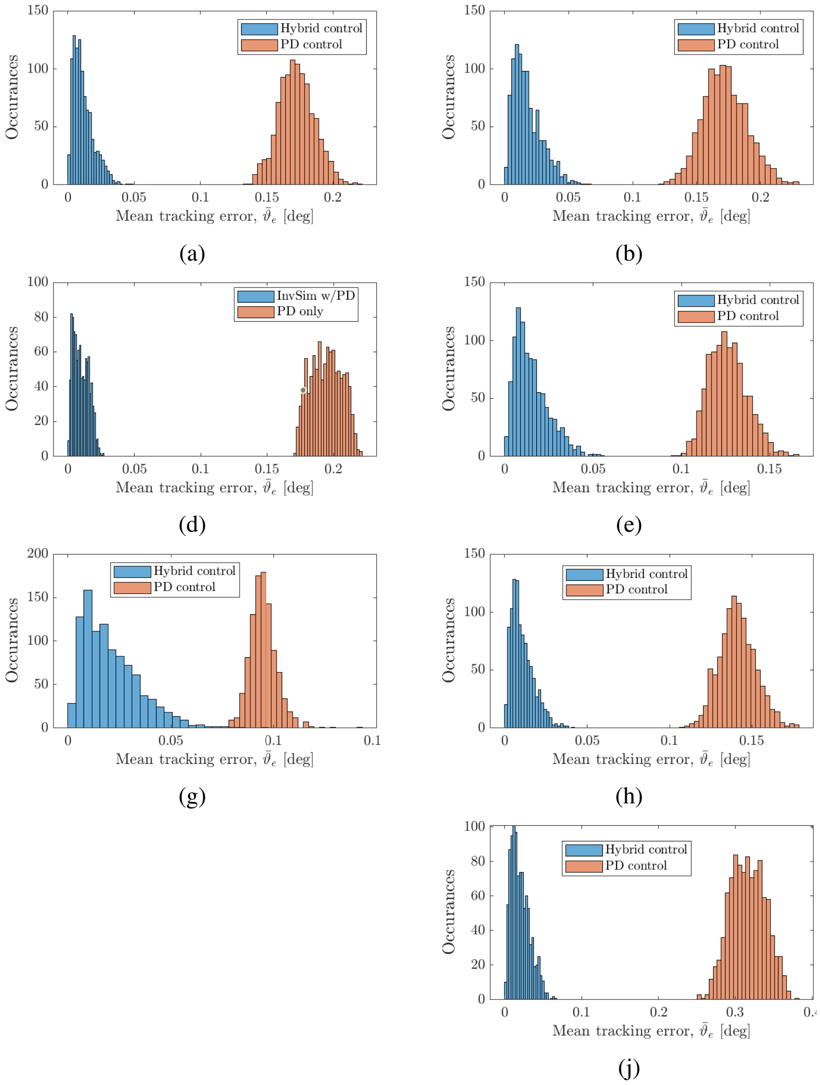


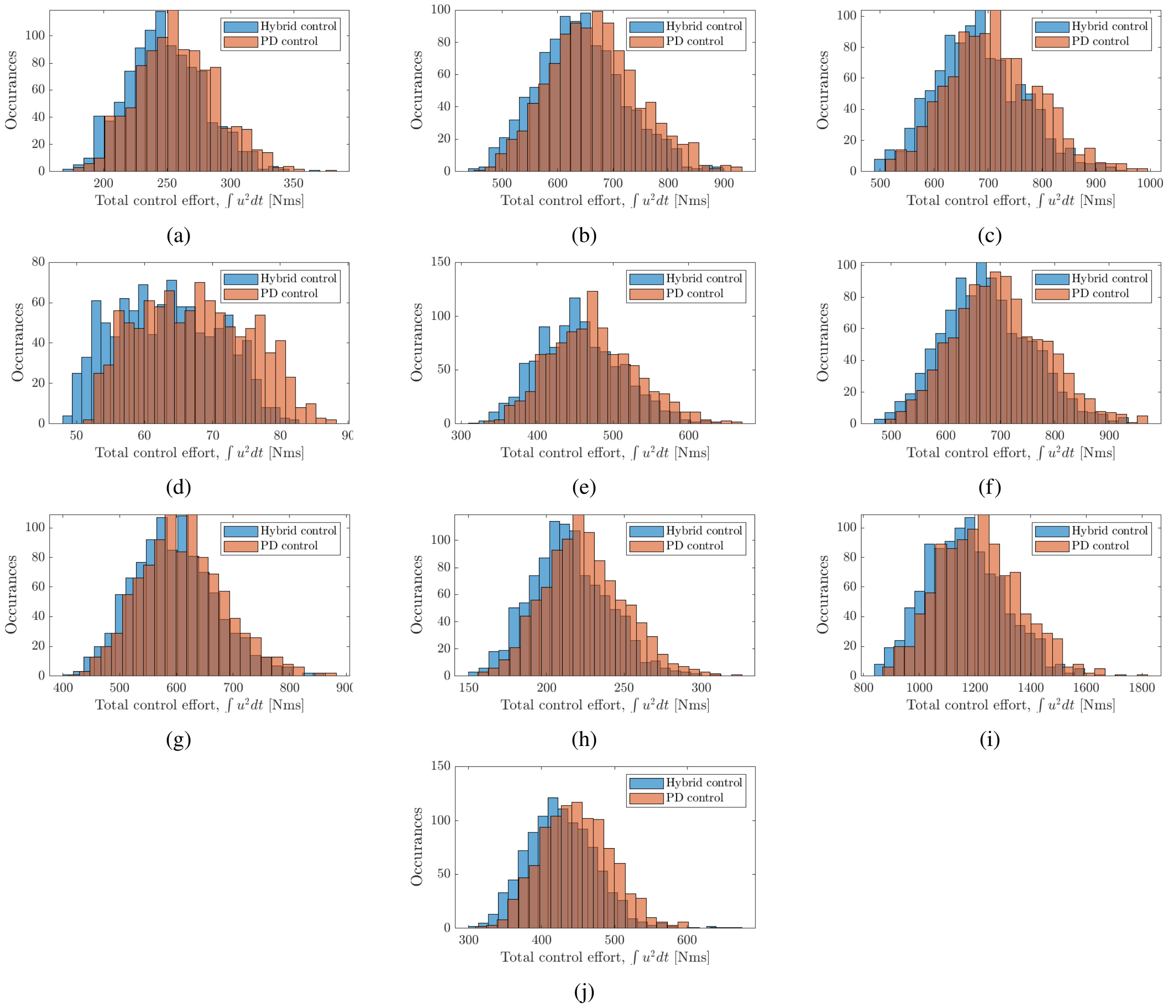
Fig. 11 Mean absolute tracking error across each slew manoeuvre, $|\bar{\vartheta}_e|$

current operating conditions. However, gain scheduling often requires significant effort to tune the controller multiple times, for multiple operating points, or use of complex tuned Gaussian functions [18], and would also require re-tuning when significant are made to the satellite design. Therefore the Hybrid controller developed in this work has the benefit of improving the performance of a PD controller, without laborious tuning for gain scheduling or use of similar complex gain functions such as those used in [18].

It can be seen in Fig. 12 that for the most part the control effort used is identical, with slight reduction for the Hybrid controller compared to PD ranging from 2.5% to 7.8%. Therefore it can be seen that the Hybrid controller provides improved tracking error performance with slightly reduced control effort. The reduction in control effort could be beneficial over the lifetime of the satellite, reducing wear on actuators and extending the effective lifespan of the mission.

6. Conclusions

The work in this paper has successfully shown that the integral InvSim algorithm can be successfully applied to the attitude control problem for satellite's with flexible appendages. InvSim was capable of producing a smooth continuous control signal and follow the quaternion slew path perfectly for both the redundant and nominal cases with control affine and non-affine inputs. For the under-actuated case it was shown that a control solution could still be found that minimised the mean squared error, but was unsuccessful in closely following the quaternion path, proving an illustration that the manoeuvre was not possible with the given actuator configuration. These results have promising implications in the design process of satellites



(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

(i)

(j)

Fig. 12 Total control effort used during each slew manoeuvre, $\int \left(\sum_{i=1}^{n_u} u_i^2\right) dt$

with flexible appendages. The InvSim technique provides a tool for rapidly analysing the attitude tracking performance of different designs with the full dynamics of the system also being available without the need to design and tune a bespoke controller. Design considerations such as the appendage flexing, actuator size, actuator type, and actuator configurations can all be considered.

The practical considerations when attempting to deploy InvSim as a real-time attitude control solution was also explored in terms of the computational cost and robustness. Presented was a hybrid controller consisting of an InvSim pre-computed feedforward term, with PD feedback. The design was shown to be robust to bounded model uncertainty and external disturbances without the need for any gain scheduling or adaptive elements. The mean tracking error of the hybrid controller was also shown to reduce by 91.6% to a 98.2% when compared to PD control only. The total control effort was also reduced by 2.5% to 7.8%. It is noted that the performance of PD could have been improved significantly through appropriate gain scheduling. However, this can be a time consuming process and if multiple iterations of the controller are needed to asses performance during the design process this would be heavily laborious.

Lastly it should be noted that all of the methodologies presented in this work can be applied to more complex mathematical models such as the finite element method. The more complex and accurate the model being used is, the more benefit can be expected when deploying the InvSim solution as an attitude control method. Further investigation into implementation of the methodologies herein with more complex satellite models should therefore be a focus of future work. Additionally further consideration of real-time deployment of the solution, such as the inclusion of adaptive elements to update the model and minimise uncertainties/disturbances, should be explored.

References

[1] V. J. Modi, "Attitude Dynamics of Satellites with Flexible Appendages- A Brief Review," *Journal of Spacecraft and Rockets*, vol. 11, no. 11, pp. 743–751, 1974.

[2] J. Deyst, "A survey of structural flexibility effects on spacecraft control systems," in *7th Aerospace Sciences Meeting*, 1969.

[3] M. N. Sweeting, "Modern Small Satellites-Changing the Economics of Space," *Proceedings of the IEEE*, vol. 106, no. 3, pp. 343–361, 2018.

[4] A. Poghosyan and A. Golkar, "CubeSat evolution: Analyzing CubeSat capabilities for conducting science missions," *Progress in Aerospace Sciences*, vol. 88, pp. 59–83, 2017, ISSN: 0376-0421.

[5] P. W. Likins, "Dynamics and control of flexible space vehicles," 1970.

[6] S. Di Gennaro, "Output stabilization of flexible spacecraft with active vibration suppression," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 3, pp. 747–759, 2003.

[7] L. Cao *et al.*, "Robust fixed-time attitude stabilization control of flexible spacecraft with actuator uncertainty," *Nonlinear Dynamics*, vol. 100, no. 3, pp. 2505–2519, 2020.

[8] L. Meirovitch and H. D. Nelson, "On the high-spin motion of a satellite containing elastic parts.," *Journal of Spacecraft and Rockets*, vol. 3, no. 11, pp. 1597–1602, Nov. 1966, ISSN: 0022-4650.

[9] D. A. Levinson and T. R. Kane, "Spin stability of a satellite equipped with four booms," *Journal of Spacecraft and Rockets*, vol. 13, no. 4, pp. 208–213, Apr. 1976, ISSN: 0022-4650.

[10] W. He and S. S. Ge, "Dynamic modeling and vibration control of a flexible satellite," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 2, pp. 1422–1431, 2015.

[11] V. J. Modi and A. M. Ibrahim, "A general formulation for librational dynamics of spacecraft with deploying appendages," *Journal of Guidance, Control, and Dynamics*, vol. 7, no. 5, pp. 563–569, Sep. 1984.

[12] H. Bang *et al.*, "Flexible spacecraft attitude maneuver by application of sliding mode control," *Acta Astronautica*, vol. 57, no. 11, pp. 841–850, 2005, ISSN: 0094-5765.

[13] Q. Hu and G. Ma, "Variable structure control and active vibration suppression of flexible spacecraft during attitude maneuver," *Aerospace Science and Technology*, vol. 9, no. 4, pp. 307–317, 2005, ISSN: 1270-9638.

[14] M. TayyebTaher and S. M. Esmaeilzadeh, "Model predictive control of attitude maneuver of a geostationary flexible satellite based on genetic algorithm," *Advances in Space Research*, vol. 60, no. 1, pp. 57–64, 2017, ISSN: 0273-1177.

[15] P. Iannelli *et al.*, "A model predictive control for attitude stabilization and spin control of a spacecraft with a flexible rotating payload," *Acta Astronautica*, 2022, ISSN: 0094-5765.

[16] G. Song and B. N. Agrawal, "Vibration suppression of flexible spacecraft during attitude control," *Acta Astronautica*, vol. 49, no. 2, pp. 73–83, 2001, ISSN: 0094-5765.

[17] H. Liu *et al.*, "An anti-disturbance PD control scheme for attitude control and stabilization of flexible spacecrafts," *Nonlinear Dynamics*, vol. 67, no. 3, pp. 2081–2088, 2012.

[18] B. Baghi *et al.*, "Three-axis attitude stabilization of a flexible satellite using non-linear PD controller," *Transactions of the Institute of Measurement and Control*, vol. 40, no. 2, pp. 591–605, 2018.

[19] O. Kato and I. Sugiura, "An interpretation of airplane general motion and contol as inverse problem," *Journal of Guidance, Control, and Dynamics*, vol. 9, no. 2, pp. 198–204, Mar. 1986.

[20] D. P. Boyle and G. E. Chamitoff, "Autonomous Maneuver Tracking for Self-Piloted Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 1, pp. 58–67, Jan. 1999.

[21] R. A. Hess and C. Gao, "Generalized algorithm for inverse simulation applied to helicopter maneuvering flight," *Journal of the American Helicopter Society*, vol. 38, no. 4, pp. 3–15, 1993, ISSN: 00028711.

[22] S. Rutherford and D. G. Thomson, "Improved methodology for inverse simulation," *The Aeronautical Journal (1968)*, vol. 100, no. 993, pp. 79–86, 1996, ISSN: 0001-9240.

[23] D. Thomson and R. Bradley, *Inverse simulation as a tool for flight dynamics research-Principles and applications*, 2006.

[24] W. Zhou *et al.*, "Inverse simulation system for manual-controlled rendezvous and docking based on artificial neural network," *Advances in Space Research*, vol. 58, no. 6, pp. 938–949, 2016, ISSN: 0273-1177.

[25] K. Worrall *et al.*, "Application of Inverse Simulation to a wheeled mobile robot," in *ICARA 2015 - Proceedings of the 2015 6th International Conference on Automation, Robotics and Applications*, 2015.

[26] R. Gordon *et al.*, "Attitude Control of a Nanosatellite using Inverse Simulation," in *International Astronautical Congress*, Dubai, UAE, 2021.

[27] D. Thomson and R. Bradley, "Development and verification of an algorithm for helicopter inverse simulation," *Vertica*, vol. 14, no. 2, pp. 185–200, 1990.

[28] C. R. Mclnnes, "Satellite attitude slew manoeuvres using inverse control," *The Aeronautical Journal*, vol. 102, no. 1015, pp. 259–266, 1998, ISSN: 0001-9240.

[29] S. N. Singh, "Flexible spacecraft maneuver: Inverse attitude control and modal stabilization," *Acta Astronautica*, vol. 17, no. 1, pp. 1–9, 1988, ISSN: 0094-5765.

[30] S. Shahriari *et al.*, "An accurate and simple model for flexible satellites for three-dimensional studies," *Journal of Mechanical Science and Technology*, vol. 24, no. 6, pp. 1319–1327, 2010.

[31] C. Allard *et al.*, "General Hinged Rigid-Body Dynamics Approximating First-Order Spacecraft Solar Panel Flexing," *Journal of Spacecraft and Rockets*, vol. 55, no. 5, pp. 1291–1299, 2018.

[32] L. Mazzini, *Flexible Spacecraft Dynamics, Control and Guidance*. Springer, 2016.

[33] F. L. Markley and J. L. Crassidis, *Fundamentals of spacecraft attitude determination and control*. New York, NY: Springer New York, 2014, pp. 1–486, ISBN: 9781493908028.

[34] J. C. A. Barata and M. S. Hussein, "The Moore–Penrose pseudoinverse: A tutorial review of the theory," *Brazilian Journal of Physics*, vol. 42, no. 1-2, pp. 146–165, 2012.

[35] K. Shoemake, "Animating Rotation with Quaternion Curves," in *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '85, New York, NY, USA: Association for Computing Machinery, 1985, pp. 245–254, ISBN: 0897911660.

[36] L. Lu *et al.*, "Issues of numerical accuracy and stability in inverse simulation," *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1350–1364, 2008, ISSN: 1569-190X.

[37] L. Lu *et al.*, "Investigation of inverse simulation for design of feedforward controllers," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 13, no. 5, pp. 437–454, Oct. 2007, ISSN: 1387-3954.

[38] Y. Li *et al.*, "Robust finite time control algorithm for satellite attitude control," *Aerospace Science and Technology*, vol. 68, pp. 46–57, Sep. 2017, ISSN: 1270-9638.

[39] C. Zhong *et al.*, "Attitude Control for Flexible Spacecraft With Disturbance Rejection," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 1, pp. 101–110, 2017.