National Technical University of Athens
SCHOOL OF MECHANICAL ENGINEERING
SECTION OF MECHANICAL DESIGN & AUTOMATIC CONTROL
Control Systems Lab

Thesis

Concurrent state and parameter estimation for satellites with structural flexibilities and fuel sloshing

George Rapakoulias

Supervisor: Evangelos G. Papadopoulos

ATHENS 2021

Εθνικό Μετσόβιο Πολυτεχνείο
ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΟΜΕΑΣ Μ.Κ. & Α.Ε.
Εργαστήριο Αυτομάτου Ελέγχου

Διπλωματική Εργασία

Ταυτόχρονη εκτίμηση κατάστασης και παραμέτρων σε δορυφόρους με δομικές ευκαμψίες και παφλασμό καυσίμων

Γεώργιος Ραπακούλιας

Επιβλέπων Καθηγητής: Ευάγγελος. Γ. Παπαδόπουλος

ΑΘΗΝΑ 2021

Περίληψη

Οι εξελίξεις των τελευταίων ετών έχουν ως αποτέλεσμα οι διαστημικές αποστολές να γίνονται ολοένα και περισσότερο φιλόδοξες. Τυπικές σενάρια αποτελούν αυτά επισκευής ήδη ενεργών δορυφόρων με στόχο την αύξηση του προσδόκιμου ζωής τους ή αποστολές κατάρριψης διαστημικών σκουπιδιών τα οποία καταλαμβάνουν χαμηλές τροχιές γύρω από τη γη και αποτελούν κίνδυνο σύγκρουσης για αστροναύτες και ενεργούς δορυφόρους. Τέτοιου είδους αποστολές απαιτούν δορυφορικά ρομποτικά συστήματα ικανά να εκτελούν κινήσεις υψηλής επιδεξιότητας, ταχύτητας και ακρίβειας που ήταν μέχρι στιγμής αδύνατον να επιτευχθούν με τους κλασσικούς ελεγκτές βασισμένους σε απλά μοντέλα του συστήματος. Ακόμα και στα καλύτερα συστήματα ελέγχου βέβαια, η απόδοση εξαρτάται σε μεγάλο βαθμό από την ακριβής γνώση των παραμέτρων του συστήματος, η οποία δε μπορεί να προέρχεται μόνο προσομοιώσεις και πειράματα σε επίγεια εργαστήρια, αλλά από πιστοποίηση των υπολογισμών με πειράματα σε στο πραγματικό περιβάλλον λειτουργείας του ρομπότ. Έχοντας αυτό υπόψη, ο κλάδος την εκτίμησης παραμέτρων αποτελεί μείζον θέμα για την επίτευξη του στόχου των υψηλών απαιτήσεων ενός σύγχρονου διαστημικού ρομπότ.

Η παρούσα εργασία εστιάζει στο πρόβλημα της εκτίμησης παραμέτρων σε μη γραμμικά συστήματα στα οποία υπάρχουν μη μετρήσιμες μεταβλητές κατάστασης. Αυτή η συγκεκριμένη κατηγορία συστημάτων κάνει την διαδικασία εκτίμησης παραμέτρων να πρέπει να επιτευχθεί βάση γνώσης μόνο δεδομένων εισόδου εξόδου του συστήματος, σε αντίθεση με τα κλασσικά ρομποτικά συστήματα ρομποτικών βραχιόνων στα οποία υπάρχει γνώση του πλήρους διανύσματος κατάστασης. Το πρόβλημα διατυπώνεται στην μορφή ενός προβλήματος διπλής εκτίμησης και αντιμετωπίζεται με κυρίως με χρήση του Unscented Kalman Filter διατυπωμένου για μια επέκταση του συστήματος που περιέχει στο διάνυσμα κατάστασης τις παραμέτρους προς αναγνώριση. Το γενικό αυτό πρόβλημα εξετάζεται μέσω δύο ειδικών περιπτώσεων σχετικών με εφαρμογές διαστημικής. Η πρώτη περίπτωση που εξετάζεται είναι αυτή ενός δορυφόρου με παφλασμό καυσίμου, ενώ η δεύτερη ενός δορυφόρου με εύκαμπτους ηλιακούς φορείς. Σε κάθε περίπτωση, οι παράμετροι των υποσυστημάτων αναγνωρίζονται θεωρώντας γνωστές τις βάσης του δορυφόρου, η οποία μοντελοποιείται σαν στερεό σώμα.

Συγκεκριμένα, για τη αρχική μελέτη του φαινομένου του παφλασμού καυσίμου παρατίθεται μια σειρά από προσομοιώσεις υπολογιστικής ρευστομηχανικής. Αυτές πιστοποιούν ότι τα μηχανικά ανάλογα που χρησιμοποιούνται για την μοντελοποίηση του φαινομένου σε δυναμικά μοντέλα κατάλληλα για έλεγχο έχουν την απαιτούμενη ακρίβεια δεδομένων του μικρού πεδίου επιτάχυνσης στο οποίο εκτίθεται η δεξαμενή καυσίμου σε διαστημικές εφαρμογές σε σχέση με επίγειες αντίστοιχες περιπτώσεις. Επίσης, περιγράφεται μια διαδικασία τόσο για εκτέλεση συζευγμένων και μη συζευγμένων προσομοιώσεων υπολογιστικής ρευστομηχανικής του φαινομένου με τις δυναμικές εξισώσεις του δορυφόρου. Το αποτέλεσμα είναι ότι οι αλγόριθμοι που επιλέχθηκαν από την βιβλιογραφία για την ταυτόχρονη αναγνώριση κατάστασης και παραμέτρων τροφοδοτούνται από ρεαλιστικά δεδομένα εισόδου εξόδου αντί για εξιδανικευμένες μετρήσεις που προέρχονται από τις αναλυτικές εξισώσεις με προσθήκη θορύβου. Με βάση αυτά, βρέθηκε ότι ο αλγόριθμος βασισμένος στο Unscented Kalman Filter είναι ικανός να κάνει εκτίμηση του υποσυστήματος του παφλασμού καυσίμου με σχετικό σφάλμα 15% παρουσία θορύβου στην χειρότερη περίπτωση, υπολογισμένο από κατάλληλες προσομοιώσεις Monte Carlo.

Για την περίπτωση του δορυφόρου με εύκαμπτους ηλιακούς φορείς, μελετώνται δύο υποπεριπτώσεις. Αρχικά η εκτίμηση παραμέτρων γίνεται θεωρώντας γνωστές τις διαστάσεις και την

μάζα των δύο φορέων και άγνωστη μόνο την καμπτική δυσκαμψία των φορέων στην διεύθυνση ταλάντωσης. Αυτό είναι μια λογική υπόθεση, δεδομένου ότι οι παραπάνω παράμετροι μπορούν να μετρηθούν με ακρίβεια πριν ο δορυφόρος τεθεί σε τροχιά. Σε αυτή την περίπτωση το μέγιστο σχετικό σφάλμα παρουσία θορύβου δε ξεπερνά το 4%. Επίσης μελετάται η περίπτωση στην οποία και η μάζα του κάθε εύκαμπτου φορέα είναι άγνωστη. Σε αυτή την περίπτωση το μέγιστο σφάλμα εκτίμησης φτάνει το 15%. Σημειώνεται επίσης ότι η μέθοδος που προτείνεται, η οποία λειτουργεί στο πεδίου του χρόνου έναντι κλασσικών μεθόδων αναγνώρισης παραμέτρων που λειτουργούν στο πεδίο της συχνότητας, δεν έχει δυσκολία στην αναγνώριση ιδιοσυχνοτήτων από πολλούς φορείς στην ίδια κατασκευή οι οποίοι έχουν συχνότητες κοντά ο ένας στον άλλο. Αυτό είναι σε αντίθεση με μεθόδους βασισμένες στην αναγνώριση ιδιοσυχνοτήτων από διαγράμματα φάσματος απόκρισης, στα οποία οι κορυφές που αντιστοιχούν σε ιδιοσυχνότητες πρέπει να αντιστοιχηθούν με τον φορέα ο οποίος τις παράγει.

Abstract

With current advances in space technology, modern space missions are becoming increasingly more ambitious. Typical scenarios including servicing missions for extending the life of already operating satellites, or deorbiting of debris that occupy lower earth orbits posing collision risks for currently active satellites. These missions require space robots with highly precise and agile motion capabilities that were previously impossible using conventional models and control schemes. As with even the best control algorithms though, the performance is highly dependent on accurate knowledge of the various system parameters calculated not only theoretically using simulations or in terrestrial laboratory conditions, but also validated in the actual working environment of the space robot. To this end, parameter estimation algorithms are a pertinent issue in achieving the strict tolerances required in modern space missions.

This thesis focuses on the parameter estimation of nonlinear systems with unmeasurable state variables. This specific branch of systems requires the identification process to use only input-output data rather than the complete state measurements that are typically available in robotic systems like rigid manipulators. The problem is posed as a *dual estimation problem* tackled mainly using the Unscented Kalman Filter in a joint state-parameter configuration. This general problem is thoroughly studied in the context of two specific scenarios related to space applications. The first is a satellite exhibiting fuel sloshing, while the second is a satellite with two flexible solar arrays. In each case the model parameters of each subsystem are identified, considering the inertial parameters of the rest of the spacecraft, modeled as a rigid body, known.

Specifically for studying the phenomenon of fuel sloshing in satellites, a set of CFD simulations is conducted first. These certify that the mechanical equivalent models used to consider the sloshing dynamics are appropriate in the context in which they are used, given the weak acceleration fields created by small satellite thrusters in the specific scenario under consideration. Furthermore, a procedure is described and implemented to create realistic input-output data for the complete spacecraft – sloshing tank system, simulated both in a coupled and decoupled form with respect to the rest of the spacecraft dynamics. The result is that the algorithms selected for parameter estimation from the literature can be tested not only with respect to whether they can be utilized in the ideal scenario, where the structure of the model coincides perfectly with the structure of the true plant but also in the more realistic case where model mismatch exists between them. Using those data, it is found that the algorithm can be utilized for identifying a pendulum mechanical equivalent model with a worst-case relative estimation error of 15% under noise, estimated by Monte Carlo simulations.

For the case of flexible solar arrays, two scenarios are presented. The first assumes all parameters known except from the flexural rigidity in the bending direction. This is a reasonable assumption since all other quantities can be measured directly while the satellite is still in the testing stage. Under those conditions the worst-case relative error in each parameter is less 4% under noise. In a scenario where both the array's flexural stiffness and mass are unknown, the algorithm produces worst-case errors of about 15%. It is also noted that the time domain approach adopted here for identifying the flexible effects can handle multiple structures having natural frequency in the same frequency range, without having the issue of matching each frequency with its source as would be the case in a frequency domain peak-picking method.

Table of Contents

Table of figures

Table of Tables

List of Abbreviations

AMM	Assumed Mode Method
AOCS	Attitude and Orbit Control System
CoM	Center of Mass
CFD	Computational Fluid Dynamics
ESA	European Space Agency
GRV	Gaussian Random Variable
IMU	Inertial Measurement Unit
MSD	Mass Spring Damper
NASA	National Aeronautics and Space Administration
RBF	Radial Basis Function
UKF	Unscented Kalman Filter
UT	Unscented Transformation

List of Symbols

In this thesis lowercase bold letters denote vectors and uppercase bold letters denote matrices. In a vector, left superscript denotes “expressed in frame” while a missing left superscript denotes the vector is expressed in the body-fixed frame of the robot’s base.

$\mathcal{M}$	Generalized mass matrix of a body
$\mathcal{F}$	Symbol to denote a reference Frame
${}^{b}\mathcal{R}_{a}$	Transformation matrix for flexible segment inertia matrix
T	Kinetic energy
V	Potential Energy
L	System’s Lagrangian
$\mathbf{q}$	Generalized coordinates vector
${}^{a}\mathbf{R}_{b}$	Coordinate transformation from frame b to a
$\mathbf{S}$	Matrix relating Euler angle rates with rotational velocities
$\mathbf{v}$	Rigid body’s linear velocity
$\boldsymbol{\omega}$	Rigid body’s angular velocity
$\mathbf{t}$	Twist vector of rigid body
$\mathbf{f}$	External force vector acting on rigid body
$\boldsymbol{\tau}$	External torque vector acting on rigid body
m	mass
ρ	Density
m_f	Fixed fuel mass
m_s	Sloshing mass
k	Spring constant
l_p	Pendulum length in a pendulum mechanical equivalent sloshing model
Bo	Bond number
σ	Surface tension
$\mathbf{I}_b$	Rigid spacecraft moment of inertia
$\mathbf{p}_0$	Sloshing tank center attachment point with respect to spacecraft CoM
$\mathbf{H}$	Generalized system inertial matrix
$\mathbf{c}$	Vector of coriolis centrifugal terms
$\boldsymbol{\varphi}$	Vector containing mode shapes in a bending direction
$\mathbf{p}$	Undeformed position of point in a flexible segment
$\mathbf{p}_c$	Centroidal undeformed position of a point in a flexible segment
$\delta\mathbf{p}$	Transverse undeformed position of a point in a flexible segment
$\mathbf{p}_d$	Deformed position of a point in a flexible segment
$\mathbf{u}$	Deformation vector of a point in a flexible segment
$\mathbf{u}_c$	Deformation of a point in a flexible member’s centerline
$\delta\mathbf{u}$	Transverse part of deformation in a flexible member
$\boldsymbol{\delta}$	Vector containing modal amplitudes in a bending direction
E	Young modulus of elasticity
G	Shear modulus

Σ	Flexural rigidity of an Euler Bernoulli beam
$F(\cdot)$	State transition function of general nonlinear system
$H(\cdot)$	Measurement model of nonlinear system
$\mathbf{P}$	Covariance Matrix
$\boldsymbol{\pi}$	Vector of unknown parameters in a dynamic system
T_s	Sampling time for discretization of dynamic systems
$(\cdot)^{\times}$	Denotes the construction of a skew symmetric matrix from a 3-element vector
$\mathbf{I}_{3x3}$	Unit matrix of dimension 3

1 Introduction

1.1 Motivation

Space exploration is a field that met incredible advances during the past years. Modern space missions are facing tasks requiring increased dexterity and a high level of autonomy. A particularly interesting mission scenario that has been gaining increased attention over the last years is that of repairing or deorbiting inactive, possibly non-cooperative, satellites. This is a very important issue in an overall effort to impede a phenomenon called “Kessler syndrome”, a naturally occurring chain of events that will exponentially increase space junk due to collisions and will ultimately render earth’s lower orbits uninhabitable not only for astronauts but for future satellites as well. Multiple approaches have been proposed over the last decade, ranging from spacecrafts with manipulators for grasping and handling the target satellite to launchable nets for capturing a tumbling debris. The common aspect of all such missions is their increased demands for agile, precise, and highly coordinated motions capabilities. These in generally require the inclusion of a variety of effects in the spacecraft’s control models that were previously ignored because of their minor importance in older missions. These effects include flexibilities of large solar arrays or antenna structures, fuel sloshing and gravity gradient related disturbances among others.

Such phenomena can be predicted through highly complex simulations allowing for a rough tuning of the various parameters participating in the control models. Experimental validation and fine tuning though are difficult, if not impossible, to take place in terrestrial laboratories. The main issue is the emulation of the zero-g environment, playing a key role in almost all of the previously mentioned effects. Two main different approaches are usually adopted in such scenarios. On the one hand, one can design controllers with robustness in mind, acknowledging the inherent uncertainty of the parameters of the system and design them to optimally perform in the presence of it. The other end of the spectrum is adaptive controllers, that actively try to adjust the model parameters and control gains to more accurately describe the actual plant, providing increased performance.

The difficulty of the experimental tuning and testing of space systems can be addressed through the field of system identification. The general concept of identification is to build a dynamic model of a system based on observed data. Although general formulations exist that can identify system dynamics without any prior knowledge of the system, these are generally employed for linear time invariant systems and are of little use in highly non-linear robotic application. A more specific task is the estimation of model parameters, given the structure of the dynamic equations which are generally non-linear. This thesis will focus on a branch of nonlinear models characterized by having state variables that can’t be measured directly.

1.2 Literature review

To give some more context, consider the general classifications made in Figure 1-1. For a linear system, the parameter estimation problem can be tackled in general only from input output data, assuming full state observability [21]. Indeed, a fully observable, discrete time, state-space model can be transformed into an input-output equation fully relating its output with its input eliminating the unmeasurable state

variables. From there, the parameter estimation problem can be solved using linear regression via least squares. On the other hand, in the nonlinear model of a rigid manipulator, where all state variables are directly measurable, the system equations can be transformed into the famous Y-π form, enabling again the employment of least squares algorithms. Linear regression is a convex optimization problem and can be very solved quickly and globally, without the need for an accurate initialization or the convergence issues that are usually pertinent to numerical optimization methods for attacking non-convex problems.

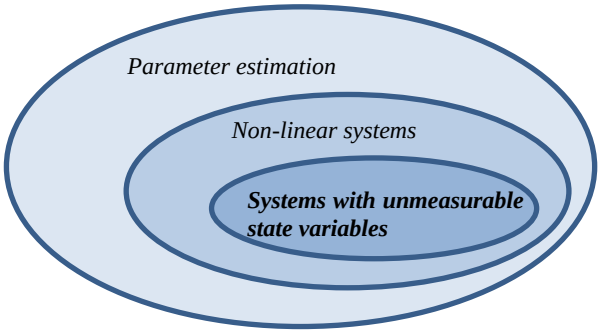


Figure 1-1. Problem definition in the broader context of parameter estimation.

In the case of non-linear systems with unmeasurable state variables, the problem cannot be formulated in a general way as a linear regression problem inheriting all the advantageous attributes of convex problems. The algorithms that are employed in such scenarios are usually iterative, of much higher computational complexity, and sometimes prone to convergence issues. An important distinction one can do to select an appropriate method in these cases is whether the algorithm will be used online or offline. Offline methods are usually posed as nonlinear optimizations problems and are solved iteratively using numerical optimization schemes. They offer the best possible solution in terms of a selected cost function and are usually employed in scenarios when experimental data has been previously gathered and the a *best-fit* parameter vector is sought. On the other hand, online methods sacrifice some accuracy for computational speed and are typically used when tracking of specific model parameters is required to take place in real-time. The uses for these are abundant, ranging from adaptive control schemes, fault detection or disturbance rejection among others.

Focusing on online methods, a typical one employed in the literature is based on state estimation. To provide some context, consider a system described by some dynamic model which has a known structure however unknown values in some of its parameters. Given only input-output data, one possible approach to reconstruct the full state vector in order to proceed with classic parameter estimation algorithms would through Luenberger observers, Kalman Filters or a similar method. These methods however require knowledge of the system's dynamic equations which are unknown. This problem, also known as a dual estimation problem, was originally posed as a nonlinear estimation problem by augmenting the state vector with the unknown parameters of the state space equations. Kopp and Orford [19] proposed the Extended Kalman Filter (EKF) to solve the resulting nonlinear filtering problem. In this thesis, a similar method is used based on the Unscented Kalman Filter (UKF), an improvement of the EKF based on the Unscented Transformation (UT). The specific formulation used can be found in detail in [13],[41][42].

In this thesis the problem is not considered in the general case of non-linear systems but in a more case specific context related to space missions. Two systems are thoroughly studied, the case of a

satellite with a sloshing fuel tank and that of a satellite with flexible solar arrays. These problems, although seem unrelated in terms of structure of dynamic equations, they have one key attribute in common: both have state variables that are either fundamentally impossible to measure, or would require very intricate hardware that is not typically found in a space mission.

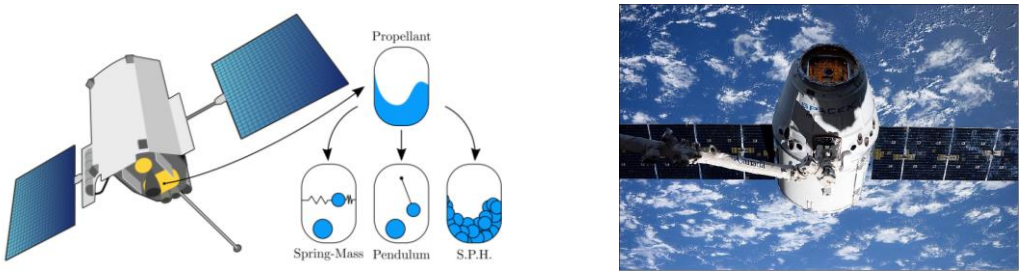


Figure 1-2. Typical cases of space systems with unmeasurable state variables.

The inclusion of sloshing in models appropriate for control usually takes place using the so called simplified mechanical equivalent models [9]. These replace the Navier-Stokes equations that govern the liquid motion with an abstract mechanism that if designed correctly, will reasonably and accurately capture the force interaction between the sloshing tank and the satellite. It becomes quickly obvious that the state variables representing those models do not correspond to a physically measurable quantity and is therefore fundamentally unmeasurable. On the other hand, the case of flexible solar arrays is usually accounted for using a modelling technique called the Assumed Mode Method (AMM). This is typically employed for creating low order models of flexible structures appropriate for control. In the context of AMM, the degrees of freedom of the flexible structured are assumed to be the oscillation amplitudes of a limited number of modes. In this scenario, the amplitude of each mode could be extracted if knowledge of its deformed shape is available, either visually from cameras, from strain sensors on the structure or from accelerometers distributed along it. These sensors add complexity, consume power, and might not be available in already operating spacecrafts. The state variables corresponding to the flexible degrees of freedom are therefore also unmeasurable.

1.3 Thesis organization

This thesis is organized in 4 sections. The 1st chapter is concerned with the derivation of the equations of motion for all the system studied. This includes modelling of a rigid spacecraft with mechanical equivalent models for sloshing and multiple flexible arrays. The 2nd chapter is dedicated to studying the sloshing phenomenon with CFD. This serves the 3 purposes. First, it certifies the validity of the mechanical equivalent models that have been previously employed. Second, a procedure is described for creating realistic data used to evaluate the parameter estimation algorithms that are the main focus of the thesis. Third, a brief literature review is included, along with some simulations conducted exclusively in this thesis, for studying various fuel sloshing scenarios relative to space applications. The 3rd chapter starts with a literature review of the concurrent state-parameter estimation problem in general and then focuses on the space applications presented earlier. Tabulated results regarding the performance of the methods studied are presented for all scenarios studied. In the 4th and final chapter, the outcomes and contributions are summarized along with recommendations for future work.

2 System modelling

2.1 The General formulation for multibody vehicles

The analytic formulation of the equations of motion for the system is the first step towards parameter estimation. It is important that the employed method will yield closed form differential equations with analytic expressions of the systems parameters. There are multiple ways to tackle this problem including the classic Lagrangian approach, Newton – Euler equations, or hybrid methods using quasi-coordinates as in [13][21-22][28]. The main idea of the later, usually applicable to flying robots, is instead of using a global inertial frame to describe the motion of the system, to write the equations of motion in the non-inertial frame of the robot's first link usually called base link or simply base. The orientation and position of the base no longer participate in the equations of motion this way. Furthermore, the inertia of the base link is constant in its own frame. The end result of the method is simpler and more compact equations of motion. In the case where the orientation of the system is not of interest this formulation is clearly beneficial. Given the assumption that the space systems studied here are essentially free of external disturbances like gravity gradient and aerodynamic drag among others, the orientation of the experiment is indeed not relevant and can be excluded from the analysis.

For a general description of the modelling process consider the case of a tree type system that is composed of a rigid base, described by its linear and angular velocity, $\mathbf{v}_0$ and $\boldsymbol{\omega}_0$ respectively, expressed in the body-fixed frame of the base. Multiple appendages are connected to it, with their configuration being described by a vector of generalized coordinates $\mathbf{q}_a$. The appendages might be flexible solar panels, lumped parameter systems etc. The Kinetic energy of the system is

$$T = \frac{1}{2}\mathbf{t}_0^T \mathcal{M}_0 \mathbf{t}_0 + \frac{1}{2}\sum_{i=1}^{n}\left(\frac{1}{2}\mathbf{t}_i^T \mathcal{M}_i \mathbf{t}_i\right) \tag{2-1}$$

where $\mathbf{t}_{0/i} = \begin{bmatrix} \mathbf{v}_{0/i}^T & \boldsymbol{\omega}_{0/i}^T \end{bmatrix}^T$ represents the twist vector of the rigid base and of all other lumped elements respectively. The inertial parameters of each body are concatenated in the generalized mass matrix $\mathcal{M}$ according to the usual convention

$$\mathcal{M} = \begin{bmatrix} m\mathbf{I}_{3\times 3} & 0 \\ 0 & \mathbf{I}_{cm} \end{bmatrix} \tag{2-2}$$

To formulate the system's Lagrangian, further assume that the various appendages connected to the base can store energy in terms of some quadratic potential energy function $V(\mathbf{q}_a)$, while they dissipate energy via a Rayleigh energy dissipation function $R(\dot{\mathbf{q}}_a)$. The Lagrangian under this notation is

$$L = T - V \tag{2-3}$$

To continue with the derivation of the equations of motion, the standard Lagrangian approach would dictate to express the linear and angular velocity of each link in terms of a set of generalized coordinates. For the appendages usually the degrees of freedom of the various joints are used while for the rigid base the inertial position and Euler angles. In the special case the Lagrangian in (2-1) is

orientation and position invariant[1] in the sense that the orientation and position of the system with respect to an inertial frame play no direct role in the its evolution, we would like to avoid adding those to the generalized coordinate vector. Instead, one can use $\mathbf{v}_0$ and $\boldsymbol{\omega}_0$ as quasi-coordinates for directly describing the motion of the base, while expressing $\mathbf{v}_i$ and $\boldsymbol{\omega}_i$ in terms of $\mathbf{q}_a$. For a brief description of how this can be is done, consider the generalized coordinate vector corresponding to the degrees of freedom of the base $\mathbf{q}_b = \begin{bmatrix} p_x & p_y & p_z & \theta_1 & \theta_2 & \theta_3 \end{bmatrix}^{\mathrm{T}}$. This could be composed from the position of the base resolved in an inertial frame and some Euler angle parametrization of its attitude. The relations between the generalized speeds $\dot{\mathbf{q}}_b$ and the quasicoordinates $\mathbf{v}_0$ and $\boldsymbol{\omega}_0$ is given by

$$\begin{bmatrix} \mathbf{v}_0 \\ \boldsymbol{\omega}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_0^T & 0 \\ 0 & \mathbf{S} \end{bmatrix} \dot{\mathbf{q}}_b \tag{2-4}$$

where $\mathbf{R}_0^T$ is the rotation matrix simply transforming the velocity vector from the inertial to the body fixed frame and $\mathbf{S}$ is the matrix converting Euler angle rates to body fixed rotational velocity.

The classic Lagrangian formulation expressed in terms of $\mathbf{q} = \begin{bmatrix} \mathbf{q}_b^T & \mathbf{q}_a^T \end{bmatrix}^T$ using matrix calculus notation is

$$\begin{aligned} &\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}_b}\right)^T = \mathbf{Q}_b \\ &\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_{a,j}}\right) - \frac{\partial L}{\partial q_{a,j}} + \frac{\partial R}{\partial \dot{q}_{a,j}} = Q_{b,j} \qquad j = 1,2,...n \end{aligned} \tag{2-5}$$

Notice that the term $\partial L / \partial \mathbf{q}_b$ in the first equation is omitted because the kinetic energy in (2-1), the potential energy function $V(\mathbf{q}_a)$ and by extension the Lagrangian do not depend explicitly on $\mathbf{q}_b$. The term should only be included if the change of variables of (2-4) is performed on the Lagrangian, which is exactly what we want to avoid. The objective now is using the chain rule, to convert the differentiation with respect to $\dot{\mathbf{q}}_b$ in the first 6 equations of (2-5) using (2-4). This process can be found in detail in [24] for the general case while for a system similar to this one in [23]. The resulting the equations in vector form will be used here directly. These are

$$\begin{aligned} &\frac{d}{dt}\left(\frac{\partial L}{\partial \mathbf{v}_0}\right) + \boldsymbol{\omega}_0^{\times} \frac{\partial L}{\partial \mathbf{v}_0} = \mathbf{f} \\ &\frac{d}{dt}\left(\frac{\partial L}{\partial \boldsymbol{\omega}_0}\right) + \boldsymbol{\omega}_0^{\times} \frac{\partial L}{\partial \boldsymbol{\omega}_0} + \mathbf{v}_0^{\times} \frac{\partial L}{\partial \mathbf{v}_0} = \boldsymbol{\tau} \\ &\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} + \frac{\partial R}{\partial \dot{q}_i} = F_i \end{aligned} \tag{2-6}$$

Where the skew symmetric matrix of a 3-element vector $\mathbf{a}$ denoted by $\mathbf{a}^{\times}$ is defined as

[1] The Lagrangian is considered invariant with respect to some degree of freedom q_i when $\frac{\partial L}{\partial q_i} = 0$. In this case, q_i is also referred to as a cyclic variable.

$$\mathbf{a}^{\times} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & x_1 & 0 \end{bmatrix}$$

fulfilling the property $\mathbf{a} \times \mathbf{b} = \mathbf{a}^{\times}\mathbf{b}$.

Although the formulation of (2-6) is typically used among roboticists to eliminate the Euler angles vector from the equations of motion in order to use some more powerful attitude representation like quaternions, the motive here is to write the equations of motion without using a global reference frame, eliminating therefore the need for orientation measurements.

2.2 Modelling of a 1D fuel sloshing tank

Forgetting for the moment the powerful tools presented above for studying dynamics in 3D, the first model developed in this thesis is a simple one. Because of its extensive use in the following chapters though, it will be developed here explicitly. The model represents the dynamics of a fuel tank exhibiting fuel sloshing. Typically, when this phenomenon has to be taken into consideration in the control system model some mechanical equivalent model is used. In this one-dimensional case the mass spring damper (MSD) equivalent model will be considered.

A schematic of the model can be viewed in Figure 2.1. It is composed of a massless structure representing the tank, constrained to move in one axis only. Two point masses are attached to it, one rigidly and one via a spring and a damper. The fixed mass, denoted by m_f, has the same velocity as the container, denoted by v_t. The second mass, denoted by m_s is the sloshing mass. The spring and damper have parameters k and b respectively. The velocity of the sloshing mass with respect to the inertial frame is denoted by v_s.

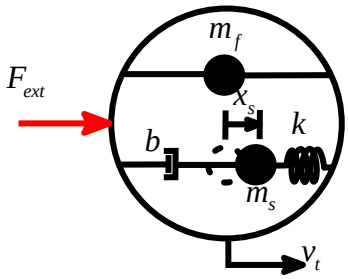


Figure 2-1. Mechanical equivalent model of container with sloshing using a mass, spring and damper

The correlation of each lumped element with the real plant parameters will be explained in detail in Chapter 3.1. Here we are only interested in deriving the equations of motion. Instead of using the deflection of the spring x_s as a state variable, the force stored in it will be used. This is related linearly with its deformation x_s according to

$$f_s = k\,x_s \tag{2-7}$$

By differentiating (2-7) one gets

$$\dot{f}_s = k\dot{x}_s = k\left(v_s - v_t\right) \tag{2-8}$$

Using (2-7) and Newton's second law expressed for each point mass one gets the complete system equations

$$\begin{aligned}\dot{v}_t &= \frac{1}{m_f}\left(F_{ext} - f_s - b\left(v_t - v_s\right)\right)\\ \dot{v}_s &= \frac{1}{m_s}\left(f_s + b\left(v_t - v_s\right)\right)\\ \dot{f}_s &= k\left(v_t - v_s\right)\end{aligned} \tag{2-9}$$

The output of system (2-9) is the tank's velocity v_t .

In the CFD simulations, the motion of the tank is sometimes prescribed and the system's output is the sloshing force exerted to the container walls. To this end, the acceleration of the tank $\dot{v}_t$ might be directly controlled. In such scenario the system equations in (2-9) can be altered to

$$\begin{aligned}\dot{v}_t &= u\\ \dot{v}_s &= \frac{1}{m_s}\left(f_s + b\left(v_t - v_s\right)\right)\\ \dot{f}_s &= k\left(v_t - v_s\right)\end{aligned} \tag{2-10}$$

Usually in such a scenario the system's output is the force

$$F = f_s + b(v_f - v_s) + m_f u \tag{2-11}$$

Which is the essentially the first equation in (2-9) solved for $F = F_{ext}$ and is the required force need to produce the prescribed acceleration, or simply minus the sloshing force.

2.3 Spacecraft with fuel sloshing using a mechanical equivalent model

2.3.1 Mass spring damper equivalent model

Returning to the 3D models, the equations of motion for a satellite with a mass spring damper (MSD) mechanical equivalent sloshing model will be presented, as shown in Figure 2-2. It is composed of a rigid base representing the satellite and a mechanical equivalent model representing the sloshing dynamics. Much like the previous case, the sloshing model consists of a fixed mass m_f rigidly connected to the spacecraft in the location $\mathbf{p}_0$ with respect to its center of mass and a sloshing mass m_s , constrained to move on a plane perpendicular to the main engine's thrust vector, connected to the fixed mass with a spring and a damper. The position of the sloshing mass with respect to the fixed mass expressed in the body frame is denoted by $\mathbf{r}_s = \begin{bmatrix} q_1 & q_2 & 0 \end{bmatrix}^T$ and corresponds to the generalized coordinates of the sloshing subsystem.

To use the Lagrangian approach discussed earlier the kinetic energy of the sloshing subsystem needs to be calculated. The contribution of the rigid base is trivial. The velocities of the two point masses of the MSD sloshing model are

$$\begin{aligned}\mathbf{v}_f &= \mathbf{v}_0 + \boldsymbol{\omega}_0^{\times}\mathbf{p}_0\\ \mathbf{v}_s &= \mathbf{v}_0 + \dot{\mathbf{r}}_s + \boldsymbol{\omega}_0^{\times}\left(\mathbf{p}_0 + \mathbf{r}_s\right)\end{aligned} \tag{2-12}$$

The kinetic energy of the complete system is

$$T = \frac{1}{2}\mathbf{t}_0^T\mathcal{M}_0\mathbf{t}_0 + \frac{1}{2}m_s\mathbf{v}_s^T\mathbf{v}_s + \frac{1}{2}m_f\mathbf{v}_f^T\mathbf{v}_f \tag{2-13}$$

where $\mathcal{M}_0$, $\mathbf{t}_0$ are the generalized mass matrix and twist vector of the rigid base as defined in Chapter 2.1.

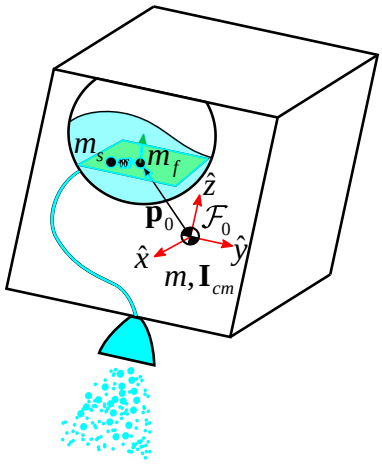


Figure 2-2. Satellite with mass spring damper mechanical equivalent model

If the term $\mathbf{v}_s^T\mathbf{v}_s$ is analyzed it decomposes to

$$\begin{aligned}\mathbf{v}_s^T\mathbf{v}_s = \; & \mathbf{v}_0^T\mathbf{v}_0 + \dot{\mathbf{r}}_s^T\dot{\mathbf{r}}_s + \boldsymbol{\omega}_0^T\left[\left(\mathbf{p}_0+\mathbf{r}_s\right)^{\times}\right]^2\boldsymbol{\omega}_0 \\ & + 2\dot{\mathbf{r}}_s^T\mathbf{v}_0 - 2\dot{\mathbf{r}}_s^T\left(\mathbf{p}_0+\mathbf{r}_s\right)^{\times}\boldsymbol{\omega}_0 - 2\mathbf{v}_0^T\left(\mathbf{p}_0+\mathbf{r}_s\right)^{\times}\boldsymbol{\omega}_0\end{aligned} \tag{2-14}$$

while the term $\mathbf{v}_f^T\mathbf{v}_f$ equals

$$\mathbf{v}_f^T\mathbf{v}_f = \mathbf{v}_0^T\mathbf{v}_0 + 2\boldsymbol{\omega}_0^T\left(\mathbf{p}_0^{\times}\right)\mathbf{v}_0 - \boldsymbol{\omega}_0^T\left(\mathbf{p}_0^{\times}\right)^2\boldsymbol{\omega}_0 \tag{2-15}$$

Using (2-14) - (2-15) one can write the system's kinetic energy in matric form as

$$T = \begin{bmatrix}\mathbf{v}_0 & \boldsymbol{\omega}_0 & \mathbf{q}\end{bmatrix}\mathbf{H}\begin{bmatrix}\mathbf{v}_0 \\ \boldsymbol{\omega}_0 \\ \mathbf{q}\end{bmatrix} \tag{2-16}$$

$$\mathbf{H} = \begin{bmatrix}\left(m_s+m_f+m\right)\mathbf{I}_{3x3} & -m_f\left(\mathbf{p}_0\right)^{\times} - m_s\left(\mathbf{p}_0+\mathbf{r}_s\right)^{\times} & m_s\mathbf{D} \\ * & \mathbf{I}_0 - m_s\left(\left[\mathbf{p}_0+\mathbf{r}_s\right]^{\times}\right)^2 - m_f\left(\mathbf{p}_0^{\times}\right)^2 & m_s\left(\mathbf{p}_0+\mathbf{r}_s\right)^{\times}\mathbf{D} \\ * & * & m_s\mathbf{I}_{2x2}\end{bmatrix} \tag{2-17}$$

where the matrix $\mathbf{D} = \begin{bmatrix}\mathbf{I}_{2x2} & \mathbf{0}_{2x1}\end{bmatrix}^T$ essentially selects the first two columns from $(\mathbf{p}_0+\mathbf{r}_s)^{\times}$. The *'s represent symmetry. The potential energy stored in the springs is

$$V = \frac{1}{2}k_x q_1^2 + \frac{1}{2}k_y q_2^2 = \begin{bmatrix}q_1 & q_2\end{bmatrix}\begin{bmatrix}k_x & 0 \\ 0 & k_y\end{bmatrix}\begin{bmatrix}q_1 \\ q_1\end{bmatrix} \tag{2-18}$$

The equations of motion can be easily computed at this point simply by employing (2-3) and (2-6). The resulting equations will be of the form

$$\mathbf{H}(\mathbf{q})\begin{bmatrix}\dot{\mathbf{v}}_0\\ \dot{\boldsymbol{\omega}}_0\\ \ddot{\mathbf{q}}\end{bmatrix}+\mathbf{c}(\mathbf{v}_0,\boldsymbol{\omega}_0,\mathbf{q},\dot{\mathbf{q}})=\begin{bmatrix}\mathbf{f}\\ \boldsymbol{\tau}\\ \mathbf{0}\end{bmatrix} \tag{2-19}$$

where $\mathbf{f}$, $\boldsymbol{\tau}$ are the external forces and torques acting on the spacecraft's Center of Mass by means of thrusters and $\mathbf{c}$ is the vector of non-linear Coriolis terms, containing the potential energy terms as well.

Looking at the inertia matrix of the system in (2-17) the advantages of using this mixed generalized – quasicoordinates formulation are obvious. Indeed, the matrix has much fewer terms compared to how it would be if a classic generalized coordinate formulation based on Euler angles was used, while it is also free of highly nonlinear trigonometric terms coming from the attitude transformation. The vector of nonlinear terms $\mathbf{c}$ is not presented here analytically however it is much more compact when compared to the classic Lagrangian formulation.

2.3.2 Pendulum equivalent model

The procedure followed to model the spacecraft with a pendulum mechanical equivalent sloshing model doesn't differ much from the MSD case. The structure of the model can be viewed in Figure 2-3. The sloshing dynamics are now represented using a 2-DOF pendulum and a fixed mass. The pendulum joint is used parametrized using ZYX Euler angles.

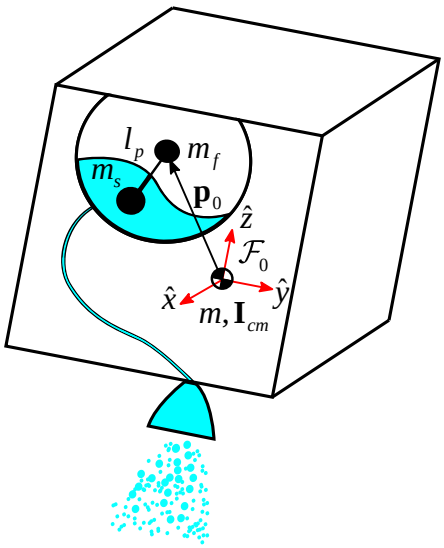


Figure 2-3. Satellite with pendulum mechanical equivalent sloshing model

As with the previous case the objective is to calculate the velocity of the sloshing mass and the fixed mass. Because the mechanism is more complex than in the MSD model, it's kinematics will be derived first. The position of the fixed and the sloshing mass with respect to the spacecraft's center of mass are

$$\begin{aligned}\mathbf{p}_f &= \mathbf{p}_0\\ \mathbf{p}_s &= \mathbf{p}_0+\mathbf{R}_s\begin{bmatrix}0 & 0 & -l_p\end{bmatrix}^T\end{aligned} \tag{2-20}$$

where $\mathbf{R}_s$ is the transformation from the spacecraft frame to the pendulum rod frame and is equal to $\mathbf{R}_s = \mathbf{R}_Z(q_2)\mathbf{R}_Y(q_1)$. The equations won't be computed analytically from this point and forward as with the MSD model given that a symbolic math software can be employed. All operations are done using Matlab's symbolic math toolbox. Differentiating (2-20) to find the velocity of the two masses, taking into consideration that the operation takes place in the rotating reference frame of the spacecraft one gets the desired expressions:

$$\begin{aligned} \mathbf{v}_f &= \boldsymbol{\omega}_0^{\times}\mathbf{p}_0 \\ \mathbf{v}_s &= \boldsymbol{\omega}_0^{\times}\mathbf{p}_s + \dot{\mathbf{R}}_s\begin{bmatrix} 0 & 0 & -l_p \end{bmatrix}^T \end{aligned} \tag{2-21}$$

The system Lagrangian can be easily computed taking into account that the system doesn't store potential energy. Its expression is

$$L = \frac{1}{2}\mathbf{t}_0^T\mathcal{M}_0\mathbf{t}_0 + \frac{1}{2}m_f\mathbf{v}_f^T\mathbf{v}_f + \frac{1}{2}m_s\mathbf{v}_s^T\mathbf{v}_s \tag{2-22}$$

As with the MSD model, the equations of motion of the system can be computed using (2-6) and are of standard form

$$\mathbf{H}(\mathbf{q})\begin{bmatrix} \dot{\mathbf{v}}_0 \\ \dot{\boldsymbol{\omega}}_0 \\ \ddot{\mathbf{q}} \end{bmatrix} + \mathbf{c}(\mathbf{v}_0, \boldsymbol{\omega}_0, \mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\tau} \\ \mathbf{0} \end{bmatrix} \tag{2-23}$$

2.4 Satellite with flexible appendages

Apart from the case of fuel sloshing using a mechanical equivalent model, we are also interested in the case of a spacecraft with flexible appendages. The later could be solar panels, an antenna, or some other beam like structure protruding from the main body and being prone to vibrations. In general, a flexible structure is a continuous system described by PDEs. To avoid infinite dimensional spaces some approximation is needed. One could discretize the system using finite elements and use order reduction to compute a model appropriate for control. This however is not suited for parameter estimation because the model parameters are no longer in an explicit form in the system equations. Another alternative is to use the Assumed Mode Method (AMM) in combination with the assumption of the Euler-Bernoulli beam equations describing the flexible member. The idea of the AMM is to use a finite number of modes of the beam as degrees of freedom and use the amplitude of each one as generalized coordinates. The later will be referred to as modal coordinates. The advantage of this formulation is that reasonably high accuracy can be accomplished by only adding a few modes. The validity of the Euler Bernoulli beam hypothesis is restricting though, especially in the case of solar panels however it has been used extensively in the literature [10] so it will be adopted here as well. Including 2D plate equations in the system would be prohibitive in terms of complexity anyway.

Under these assumptions, the modelling strategy used here is comprised of first building a *segment Lagrangian* containing the kinetic and potential energy of a single flexible appendage calculated in the floating reference frame attached to its one end and then appending it to the system Lagrangian with an appropriate transformation for every flexible member. Before calculating the *segment Lagrangian*, some preliminaries regarding Euler Bernoulli theory will be presented.

Consider the beam of Figure 2-4 with constant cross-section, clamped in one end and free to vibrate on the other. Under small deformations, assume that every cross-section in the beam remains planar, normal to the beam's centroidal axis and undeformed.

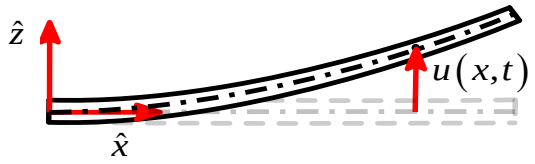


Figure 2-4. One-dimensional Euler-Bernoulli beam.

The PDE describing the motion of the centroidal axis in this case is

$$EI\frac{\partial^4 u}{\partial^4 x}+\mu\frac{\partial^2 u}{\partial^2 t}=0 \tag{2-24}$$

where $u(x,t)$ is the deformation in the bending direction, E is the material's young modulus, I the second moment of area in the direction of bending and μ is the mass per unit length of the beam. If the member deforms in more than one bending directions, equation (2-24) will be written for each one, assuming no coupling between them. Given separability in time and space a general solution for (2-24) of the form

$$u(t,x)=\varphi(x)\delta(t) \tag{2-25}$$

is sought. Plugging harmonically time varying amplitudes $\delta_i(t)=\mathrm{Re}(e^{-i\omega_i t})$ in (2-25) and substituting in (2-24) yields a 4th order ODE with only the spatial variable into play. The general solution is

$$\hat{\varphi}(x)=A\cosh(x)+B\sinh(x)+C\cos(x)+D\sin(x) \tag{2-26}$$

The beam is clamp-loaded in one end and free to vibrate in the other. These boundary conditions are translated into $\hat{\varphi}(0)=0,\hat{\varphi}'(0)=0,\hat{\varphi}''(L)=0,\hat{\varphi}'''(L)=0$ and if imposed in (2-26) yield:

$$\hat{\varphi}(s)=A\left(\cosh(\beta_n s)-\cos(\beta_n s)+\frac{\cos(\beta_n L)+\cosh(\beta_n L)}{\sin(\beta_n L)+\sinh(\beta_n L)}\left(\sin(\beta_n s)-\sinh(\beta_n s)\right)\right) \tag{2-27}$$

where L is the beam's length and β_n are solutions to the algebraic equation

$$\cosh(\beta_n L)\cos(\beta_n L)-1=0 \tag{2-28}$$

The spatial function $\hat{\varphi}(s)$ is called eigenfunction or mode and satisfies (2-27) as well as the boundary conditions. Because (2-24) is linear, solutions of the form of (2-27) can be superimposed. The first few modes can be viewed in Figure 2-5.

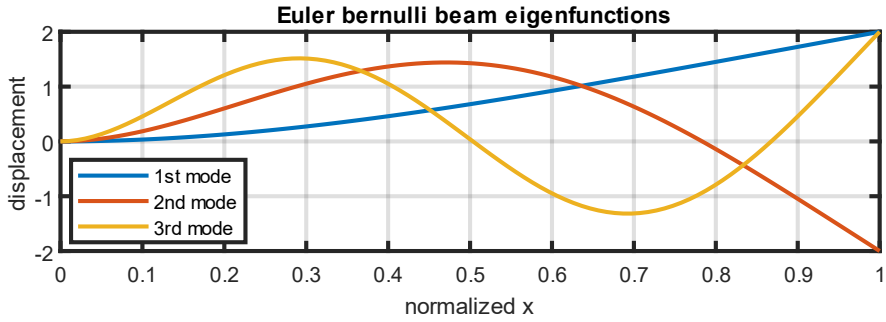


Figure 2-5. First 3 eigenfunctions of an Euler-Bernoulli beam with clamped/free boundary conditions.

The general solution for the beam's centerline is given by an infinite sum of modes

$$u(t,x) = \sum \hat{\varphi}_i(x)\delta_i(t) = \boldsymbol{\varphi}(x)^T \boldsymbol{\delta}(t) \tag{2-29}$$

where all the modes are placed in the vector $\boldsymbol{\varphi}(x)$ and all the mode amplitudes in $\boldsymbol{\delta}(t)$. Furthermore, note that in (2-27) the parameter A in the mode shape undetermined. In the AMM the objective is to use the modes shapes as degrees of freedom for the system and modal amplitudes as generalized coordinates. To this end, there is a freedom for choosing A since it can be absorbed by the corresponding modal amplitude. For practical applications a mode normalization was proposed in [7] that considerably simplifies the equations of motion and therefore will be adopted here as well. According to it one selects A in order to simplify the integral

$$\int_0^L \mu\hat{\varphi}(x)\,dx \tag{2-30}$$

to unity for each mode.

2.4.1 Kinetic energy of a single segment

To formulate the *segment Lagrangian*, assume the beam of Figure 2-6 with a reference frame rigidly attached to its one end. This frame will be called floating frame and denoted by $\mathcal{F}_a$. The left superscript a will denote that a vector is resolved in $\mathcal{F}_a$. The velocity of each point in the beam relative to the floating frame is only due to the beam's deflection. The floating frame is considered moving in inertial space with linear and angular velocities resolved in $\mathcal{F}_a$

$${}^a\mathbf{t}_a = \begin{bmatrix} {}^a\mathbf{v}_a \\ {}^a\boldsymbol{\omega}_\alpha \end{bmatrix} \tag{2-31}$$

Before proceeding with the differential kinematics of each point in the beam, the kinematics will be analyzed first. The goal is to express the position of each point mass in the beam with respect to its generalized modal coordinates.

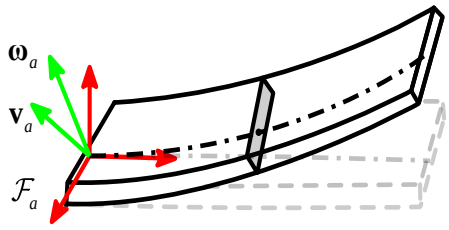


Figure 2-6. Single flexible segment modelled in its own floating frame.

Starting with the undeformed position of each point in the segment's domain, it is split into two components, a centroidal one along the main axis of the beam and a transverse one.

$$ {}^{a}\mathbf{p} = {}^{a}\mathbf{p}_c(x) + {}^{a}\delta\mathbf{p}(y,z) \tag{2-32} $$

The deflection vector corresponding to that point is also split in a similar manner.

$$ {}^{a}\mathbf{u}(\mathbf{p},t) = {}^{a}\mathbf{u}_c(x,t) + {}^{a}\delta\mathbf{u}(\mathbf{p},t) \tag{2-33} $$

According to the above, the deformed position of point $\mathbf{p}$ is

$$ {}^{a}\mathbf{p}_d = {}^{a}\mathbf{p} + {}^{a}\mathbf{u}_c(x,t) + {}^{a}\delta\mathbf{u}\left(\mathbf{p},t\right) \tag{2-34} $$

Notice that according to the Euler-Bernoulli hypothesis all cross-sections of the beam must remain planar and undeformed. Therefore, the transverse component of the deflection $\delta\mathbf{u}(p,t)$ is only due to the cross-section's rotation. This rotation transforms the local frame $\mathcal{F}_p$ in the barycenter of each cross-section, to the floating frame in the clamped origin $\mathcal{F}_a$. The corresponding rotation matrix is ${}^{a}\mathbf{R}_p(\boldsymbol{\psi})$ and is parametrized using the ZYX Euler angles.

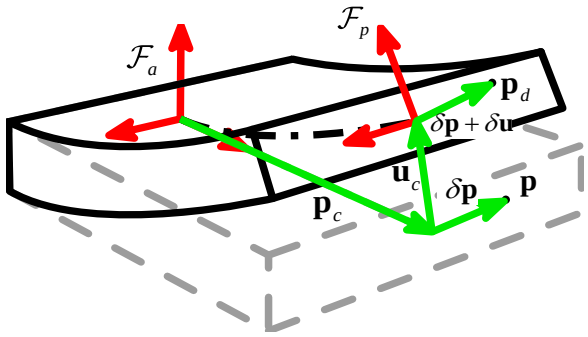


Figure 2-7. Kinematics of the flexible segment.

To continue with the AMM theory, (2-34) needs to be expressed in terms of the modal coordinates. Taking into account bending in two axes, torsion and compression the modal amplitudes or generalized degrees of freedom of the system are

$$ \boldsymbol{\delta} = \left[\boldsymbol{\delta}_x^T \;\; \boldsymbol{\delta}_y^T \;\; \boldsymbol{\delta}_z^T \;\; \boldsymbol{\delta}_a^T\right]^T \tag{2-35} $$

where $\boldsymbol{\delta}_x^T$ accounts for compression, $\boldsymbol{\delta}_y^T$, $\boldsymbol{\delta}_z^T$ for bending and $\boldsymbol{\delta}_a^T$ for torsion. The mode shapes corresponding to each flexible coordinate are stored in the vectors $\boldsymbol{\varphi}_x(s)$, $\boldsymbol{\varphi}_y(s)$, $\boldsymbol{\varphi}_z(s)$, $\boldsymbol{\varphi}_a(s)$. The size of each vector depends on the selected number of modes in each direction. The total number of modes is n_f being composed of n_x , n_y , n_z , n_a entries in each flexible direction. Each point's centroidal deformation can be expressed in terms of the flexible coordinates as:

$$ {}^{a}\mathbf{u}_c\left(x\right) = \left[\boldsymbol{\varphi}_x^T\boldsymbol{\delta}_x \;\; \boldsymbol{\varphi}_y^T\boldsymbol{\delta}_y \;\; \boldsymbol{\varphi}_z^T\boldsymbol{\delta}_z\right]^T \tag{2-36} $$

Using the small angle approximation, the Euler angles vector in ${}^{a}\mathbf{R}_p(\boldsymbol{\psi})$ is

$$ \boldsymbol{\psi} = [\boldsymbol{\varphi}_a^T\boldsymbol{\delta}_a \;\; -\boldsymbol{\varphi}_z'^T\boldsymbol{\delta}_z \;\; \boldsymbol{\varphi}_y'^T\boldsymbol{\delta}_y]^T \tag{2-37} $$

For small angles, the rotation matrix can by approximated by ${}^{a}\mathbf{R}_{p}(\boldsymbol{\psi}) = \mathbf{I}_{3\times3} + \boldsymbol{\psi}^{\times}$ and the matrix relating Euler angle rates to rotational velocities becomes unity [14]. The transverse part of the deflection $\delta\mathbf{u}$ is only due to the frame's rotation and under the above assumption becomes

$$ {}^{a}\delta\mathbf{p}(y,z) + {}^{a}\delta\mathbf{u}(\mathbf{p},t) = \left(\mathbf{I}_{3\times3} + \boldsymbol{\psi}^{\times}\right){}^{a}\delta\mathbf{p} \Rightarrow $$

$$ {}^{a}\delta\mathbf{u}(\mathbf{p},t) = \boldsymbol{\psi}^{\times}\,{}^{a}\delta\mathbf{p} = -\left({}^{a}\delta\mathbf{p}\right)^{\times}\boldsymbol{\psi} \tag{2-38} $$

Substituting (2-36) - (2-38) in (2-34) one gets the final expression for the position of each point in the beam with respect to the modal coordinates. Differentiating (2-34) and taking into consideration the that it takes place on a rotating frame the velocity of each point in the beam is

$$ {}^{a}\mathbf{v}(\mathbf{p}) = {}^{a}\boldsymbol{\omega}_{\alpha}^{\times}\,{}^{a}\mathbf{p}_{d} + \frac{d}{dt}\left({}^{a}\mathbf{p}_{d}\right) \tag{2-39} $$

Given the velocity of each point, its corresponding infinitesimal kinetic energy is

$$ dT({}^{a}\mathbf{p}) = \frac{1}{2}\,{}^{a}\mathbf{v}^{T}\,{}^{a}\mathbf{v}dm \tag{2-40} $$

In order to find the total kinetic energy of the segment, one needs to integrate over its hole domain. Given its constant cross-section the calculation reduces to a single integral along the x axis. Substituting (2-39) in (2-40) and integrating directly though, is found to be a cumbersome process, even when utilizing a symbolic computation tool. One way of simplifying the process is to omit the second order modal terms in (2-39) sacrificing some accuracy under the assumptions that the modal DOFs will remain small. In [10] each term in the infinitesimal kinetic energy is calculated and integrated explicitly while identifying common terms and calculating them only once, massively reducing the computational cost of the process and allowing for the exact computation of the integral of (2-40). This method is followed here as well. The end result is that the inertia matrix is expressed analytically in vector notation and sorted with respect to zero order, linear and quadratic dependence with the flexible coordinates. The resulting kinetic energy is

$$ T_{a} = \frac{1}{2}\,{}^{a}\mathbf{t}_{f}^{T}\mathcal{M}_{a}\,{}^{a}\mathbf{t}_{f} \tag{2-41} $$

were

$$ {}^{a}\mathbf{t}_{f} = \left[\,{}^{a}\mathbf{v}_{a}^{T} \quad {}^{a}\boldsymbol{\omega}_{a}^{T} \quad \dot{\boldsymbol{\delta}}^{T}\,\right]^{T} $$

Because of the detailed analysis in [10] one can easily choose which terms to include in the appendage's generalized mass matrix $\mathcal{M}_{a}$ in (2-41). Including all terms adds computational complexity and is not necessary in the case of small deflections. In this thesis, only the terms with up to linear dependance are included. Finally, note that for a single flexible segment, the twist of its floating frame will be augmented with its modal DOFs, completely describing its form. The new twist vector will be referred to as augmented twist and denoted by ${}^{a}\mathbf{t}_{f}$.

2.4.2 Potential Energy of a single segment

Calculating the potential energy of the segment is simpler. According to the Euler-Bernoulli theory the potential energy stored in the beam due to strain is

$$U = \frac{1}{2}\int \left(ES\left(u_x'\right)^2 + EI_y\left(u_y''\right)^2 + EI_y\left(u_y''\right)^2 + GI_p\left(u_a'\right)^2 \right) ds \tag{2-42}$$

where E is the material's Young modulus, G is the Poisson modulus, S the area, I_y and I_z the second moments of area and I_p the second polar moment of area of the cross-section. In terms of the flexible coordinates vector in (2-30), it can be expressed as

$$U = \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}} \int_0^L \begin{bmatrix} ES\boldsymbol{\varphi}_x'\boldsymbol{\varphi}_x'^T & 0_{n_x \times n_y} & 0_{n_x \times n_z} & 0_{n_x \times n_a} \\ * & EI_y\boldsymbol{\varphi}_y''\boldsymbol{\varphi}_y''^T & 0_{n_y \times n_z} & 0_{n_y \times n_a} \\ * & * & EI_z\boldsymbol{\varphi}_z''\boldsymbol{\varphi}_z''^T & 0_{n_z \times n_a} \\ * & * & * & GI_p\boldsymbol{\varphi}_a'\boldsymbol{\varphi}_a'^T \end{bmatrix} dx\,\boldsymbol{\delta} \tag{2-43}$$

giving rise to the segment's stiffness matrix K_p. Notice that this approach results in a linear model of elasticity with the stiffness matrix having a zero-order dependence with the flexible coordinates. The products EI_y and EI_z can be combined into one parameter called the beams flexural rigidity in the y and z direction. The new parameter denoted $\Sigma_{y/z}$ will be the parameter towards estimation in the following chapters.

2.4.3 The complete satellite model

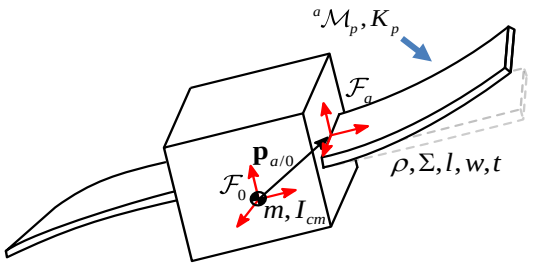


Figure 2-8. Spacecraft with multiple flexible appendages.

Up to this point, the Lagrangian of a single segment can be formulated from (2-41) and (2-43) as

$$L_a = T_a - V_a \tag{2-44}$$

and is a function of the it's augmented twist vector ${}^a\mathbf{t}_f$ resolved in its own floating frame and the various model parameters like panel dimensions, material density etc.

For an appendage that is rigidly fixed in the spacecraft, adding its contribution to the system's Lagrangian is simple. One only has to express the twist vector of the appendage's frame as a function

of the spacecraft's twist. The attachment point of the i[th] appendage in the spacecraft's frame is denoted by $\mathbf{p}_{a_i/0}$. The twist vector of the attachment point in the spacecraft frame can be found by the twist propagation matrix [10] as

$$\begin{bmatrix} \mathbf{v}_a \\ \boldsymbol{\omega}_a \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3} & -\left(\mathbf{p}_{a_i/0}\right)^{\times} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix} \begin{bmatrix} \mathbf{v}_0 \\ \boldsymbol{\omega}_0 \end{bmatrix} \tag{2-45}$$

The twist now needs to be transformed into the appendage's floating frame under the following transformation

$$\begin{bmatrix} {}^{a}\mathbf{v}_a \\ {}^{a}\boldsymbol{\omega}_a \end{bmatrix} = \begin{bmatrix} {}^{a}\mathbf{R}_0 & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{a}\mathbf{R}_0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_a \\ \boldsymbol{\omega}_a \end{bmatrix} \tag{2-46}$$

where ${}^{a}\mathbf{R}_0$ is the transformation from the spacecraft's frame to each flexible's member floating frame. Using (2-45) and (2-46) one can define the transformation

$${}^{a}\mathcal{R}_0 = \begin{bmatrix} {}^{a}\mathbf{R}_0 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times n_f} \\ \mathbf{0}_{3\times3} & {}^{a}\mathbf{R}_0 & \mathbf{0}_{3\times n_f} \\ \mathbf{0}_{n_f\times3} & \mathbf{0}_{n_f\times3} & \mathbf{I}_{n_f\times n_f} \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times3} & -\left(\mathbf{p}_{a_i/0}\right)^{\times} & \mathbf{0}_{3\times n_f} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times n_f} \\ \mathbf{0}_{n_f\times3} & \mathbf{0}_{n_f\times3} & \mathbf{I}_{n_f\times n_f} \end{bmatrix} \tag{2-47}$$

and noting that

$${}^{a}\mathbf{t}_f = {}^{a}\mathcal{R}_0 \begin{bmatrix} \mathbf{v}_0^T & \boldsymbol{\omega}_0^T & \dot{\boldsymbol{\delta}}^T \end{bmatrix}^T \tag{2-48}$$

one can directly transform the mass matrix in (2-41) to the equivalent mass matrix of the appendage in its attached point, expressed for the spacecraft's twist vector, augmented with the flexible coordinates of the segment.

$$\mathcal{M}_a = \left({}^{a}\mathcal{R}_0\right)^T {}^{a}\mathcal{M}_a {}^{a}\mathcal{R}_0 \tag{2-49}$$

The system's Lagrangian can now be calculated as

$$T = \frac{1}{2}\mathbf{t}_0^T \mathcal{M}_0 \mathbf{t}_0 + \sum\left(\frac{1}{2}\mathbf{t}_{f_i}^T \left({}^{a_i}\mathcal{R}_0^T {}^{a_i}\mathcal{M}_{a_i} {}^{a_i}\mathcal{R}_0\right)\mathbf{t}_{f_i} - \frac{1}{2}\boldsymbol{\delta}_i^T K_a \boldsymbol{\delta}_i\right) \tag{2-50}$$

summing to include the contributions from all the appendages. The augmented twist $\mathbf{t}_{f_i}$ for each flexible appendage in (2-50) is composed of $\mathbf{v}_0$, $\boldsymbol{\omega}_0$ and the its modal coordinates. Notice that the stiffness matrix doesn't require a transformation. Given (2-50) one can calculate analytically the systems inertia matrix or employ (2-6) to get the system's equations of motion.

2.5 Discretization and conversion to state space

All equations presented up to now are first order continuous-time differential equations for the linear and angular velocity of the rigid base and second order for the rest of the generalized coordinates. Their general form is

$$\mathbf{H}(\mathbf{q})\begin{bmatrix}\dot{\mathbf{v}}_0\\ \dot{\boldsymbol{\omega}}_0\\ \ddot{\mathbf{q}}\end{bmatrix}+\mathbf{c}(\mathbf{v}_0,\boldsymbol{\omega}_0,\mathbf{q},\dot{\mathbf{q}})=\begin{bmatrix}\mathbf{f}\\ \boldsymbol{\tau}\\ \mathbf{0}\end{bmatrix} \tag{2-51}$$

They can be easily converted a non-linear state space system of the form

$$\dot{\mathbf{x}}=f(\mathbf{x},\mathbf{u}) \tag{2-52}$$

where

$$\mathbf{x}=\begin{bmatrix}\mathbf{v}_0^T & \boldsymbol{\omega}_0^T & \dot{\mathbf{q}} & \mathbf{q}\end{bmatrix}^T \tag{2-53}$$

$$\mathbf{u}=\begin{bmatrix}\mathbf{f}^{\mathrm{T}} & \boldsymbol{\tau}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \tag{2-54}$$

$$f(\mathbf{x},\mathbf{u})=\begin{bmatrix}\mathbf{H}(\mathbf{q})^{-1}\left(\mathbf{u}-\mathbf{c}(\mathbf{v}_0,\boldsymbol{\omega}_0,\mathbf{q},\dot{\mathbf{q}})\right)\\ \dot{\mathbf{q}}\end{bmatrix} \tag{2-55}$$

Kalman Filtering techniques require discrete time equations. To this end (2-55) is linearized using a first order discretization. The discrete form of (2-52) is

$$\mathbf{x}_{k+1}=\mathbf{x}_k+T_s f(\mathbf{x}_k,\mathbf{u}_k) \tag{2-56}$$

2.6 Validation of equations in simscape

Simscape is an environment for building physical systems in Simulink. It allows to construct lumped element multidomain models using fundamental building blocks while maintaining full compatibility with the rest of Simulink's functionalities. It will be used in this thesis to validate the analytic equations of motion that were previously created. Only the rigid body dynamics toolbox of simscape will be utilized. The model for the 3D fuel sloshing case will be used for the following discussion. The plant's can be viewed in Figure 2-9.

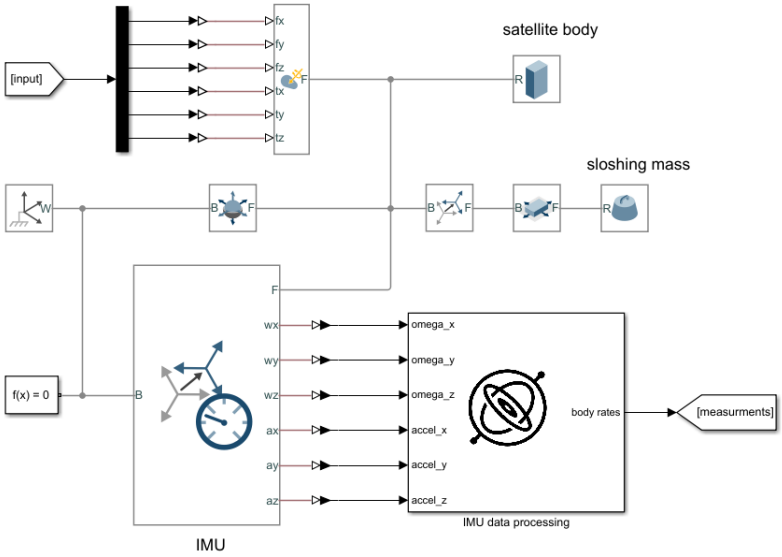


Figure 2-9. Simscape plant for a satellite with a mass spring damper equivalent sloshing model.

The models starts from an inertial "world" frame connected via a 6 DOF joint to the satellite's body center of mass. There, through a planar joint with stiffness and damping, the sloshing mass is connected. Detailed thruster and reaction wheels modelling is omitted for simplicity and is replaced with a force and torque block connected to the satellite's center of mass. Measurements are extracted from the model using a transformation sensor which is set to output linear acceleration and angular velocity measurements from the satellite's CM as if it were an ideal Inertial Measurement Unit (IMU). To make this as realistic as possible Gaussian white noise is added to the IMU signals based on the ADIS16490 sensor datasheet. The noise in the datasheet is given as noise amplitude density which is the square root of noise power spectral density (PSD). To convert this to standard deviation one simply multiplies with the sampling frequency. The dependence of the noise's standard deviation on the sampling frequency essentially corresponds to the fact that the noise contained in the spectrum above the sampling frequency is filtered off. The noise levels in terms of standard deviation will be calculated for sampling frequencies of 100 and 1000 Hz.

Table 2-1. Noise characteristics based on the ADIS16490

Sensor	**Noise Amplitude Density [Custom Units]**	**Power Spectral Density [SI]**	σ_{100}^2	σ_{1000}^2
Accelerometer	$16\ \mu g/\sqrt{\text{Hz}}$	$2.46\cdot10^{-8}\ \text{m}^2/\text{s}^4/\text{Hz}$	$2.46\cdot10^{-6}\text{m}^2/\text{s}^4$	$2.46\cdot10^{-5}\text{m}^2/\text{s}^4$
Gyroscope	$0.002\ ^\circ/\text{s}/\sqrt{Hz}$	$1.22\cdot10^{-9}\ \text{rad}^2/\text{s}^2/\text{Hz}$	$1.22\cdot10^{-7}\ \text{rad}^2/\text{s}^2$	$1.22\cdot10^{-6}\ \text{rad}^2/\text{s}^2$

Finally, it is noted that more uncertainty sources could be modelled to capture the actual noise distribution more accurately like bias instability, cross-correlation due to poor axis orthogonality or even temperature dependance with the above parameters, however these are assumed ideal for simplification and based on the fact that real space missions have extremely precise and well calibrated instruments.

In order for the identification algorithm to operate correctly, velocity and angular velocity measurements resolved in the local frame of the satellite should be provided. The angular velocity vector is directly available from the gyroscope. To construct the linear velocity vector, an integration process must take place to avoid the need for more sensors like GPS. The IMU outputs inertial acceleration data resolved in the local frame of the satellite. This signal can't be integrated directly because while the orientation of the measurement frame is changing the past measurement's frame is no longer aligned with the present one. One way to overcome the issue would be to resolve all measurements, provided attitude knowledge, to a common frame for the integration process to take place and then project the integrated acceleration back to the body frame. Attitude measurements however are hard to obtain in space and would introduce a new source of noise. A better alternative is to use the angular velocity vector from the gyroscope to convert the inertial acceleration measurement to the rate of change of the velocity vector in the spacecraft reference frame. This can be done by relating the inertial and relative rates of change of a vector in a rotating and a non-rotating frame. The required formula is

$$\underbrace{\left.\frac{d\mathbf{v}_0}{dt}\right|_i}_{\text{IMU meas.}} = \underbrace{\left.\frac{d\mathbf{v}_0}{dt}\right|_r}_{\text{relative rate}} + \boldsymbol{\omega}_0^{\times}\mathbf{v}_0 \tag{2-57}$$

By solving for the relative rate, one gets

$$\underbrace{\left.\frac{d\mathbf{v}_0}{dt}\right|_r}_{\text{relative rate}} = \underbrace{\left.\frac{d\mathbf{v}_0}{dt}\right|_i}_{\text{IMU meas.}} - \boldsymbol{\omega}_0^{\times}\mathbf{v}_0 \tag{2-58}$$

witch essentially removes the centrifugal component from the acceleration measurements since those are due to rotation and not actual velocity changes in the body frame. In order to calculate the velocity in the body frame (2-58) must be integrated in time as new measurements arrive. To make the case realistic, this will be done in a discrete manner just as it would be done in a real scenario. To this end a recursive relation is formulated, based on a first order difference scheme, linking the present velocity estimations with the past ones and the signals from the IMU. The formula is

$$\begin{pmatrix} v_1^k \\ v_2^k \\ v_3^k \end{pmatrix} = \begin{pmatrix} v_1^{k-1} \\ v_2^{k-1} \\ v_3^{k-1} \end{pmatrix} + T \begin{pmatrix} a_1^{k-1} - \omega_2^{k-1} v_3^{k-1} + \omega_3^{k-1} v_2^{k-1} \\ a_2^{k-1} - \omega_1^{k-1} v_3^{k-1} + \omega_3^{k-1} v_1^{k-1} \\ a_3^{k-1} - \omega_1^{k-1} v_2^{k-1} + \omega_2^{k-1} v_1^{k-1} \end{pmatrix} \tag{2-59}$$

where a_i^k and ω_i^k are the components of the accelerometer and gyroscope vector measurements respectively and v_i^k the components of the linear velocity vector $\mathbf{v}_0$. The exponent k denotes the time instance of the measurement. The initialization of the recursive relation is *zero initial velocity* and corresponds to the known initial condition for the integration process. Coming back to the simscape model, all above is encapsulated in the *IMU data processing* block (see Figure 2-9).

The purpose of all the above is to feed the identification algorithm with as much realistic data as possible. A result of this approach is that the noise in the velocity measurements is not Gaussian because of the integration process, the nonlinear combination of variables and the inexact integration scheme in (2-59). This introduces higher errors in the Kalman Filter based identification algorithms developed in the next chapters, however it is adopted since it makes the scenario more realistic. A clear picture of the noise levels, depicted as absolute errors between the true and the noise corrupted signal, can be seen in Figure 2-10 for a sample simulation. Given an average velocity RMS of 1m/s and an average rotational velocity RMS of 0.01 rad/s, the signal to noise ratio is about 1000 for the velocity and 10 for the rotational velocity. Notice however that the velocity error deviates significantly from the Gaussian assumption.

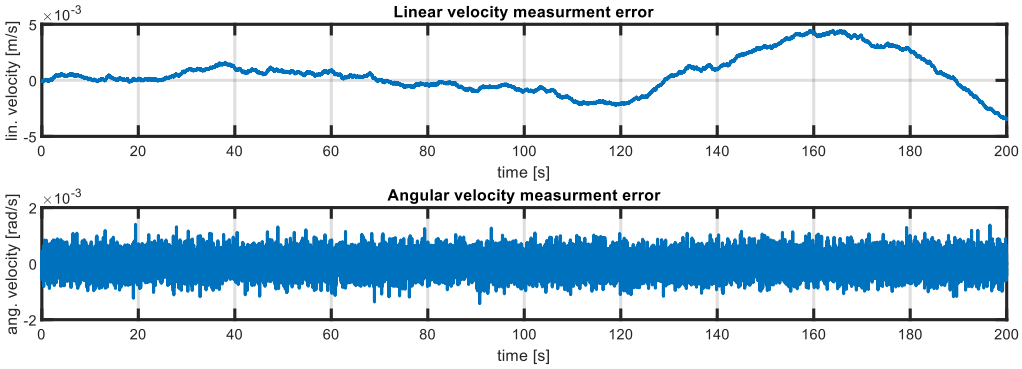


Figure 2-10. Absolute errors due to noise in the linear and angular velocities calculated from the IMU.

Finally, some details will be discussed with regards to the flexible appendage modelling in Simscape. Two alternative methods can be implemented. The simplest way is to use Simscape's built in block for flexible slender beams. According to Matlab's documentation the equations behind this block are finite beam elements based on Euler-Bernoulli theory as well. This is convenient since the analytic equations are also based on the later assumption as well, however it doesn't allow for testing the hypothesis of whether the Euler-Bernoulli approximation is indeed sufficient for representing the arguably more complex dynamics of the 2D shell-like structure of the actual solar panel. Overcoming this problem pushes the capabilities of Simscape in the current (2020b) Matlab version to its boundaries. Although plug and play blocks for more complex flexible structures are not available, Simscape offers the capability to insert custom, constant, reduced order system M,C,K[2] matrices for a general dynamic system and couple it with the rest of the model. Exporting those matrices for a flexible panel structure is possible using ANSYS® or some other custom Finite Element Analysis (FEA) code. Due to the complexity of this approach, it has not been tried in this thesis. It is suggested though as future work, given the concerns that exist in the literature that an Euler-Bernoulli beam model might not be adequate for high accuracy simulation of flexible solar panels.

[2] Mass [M], Damping[C], Stiffness[K] matrices resulting from order reduction of a linear finite element analysis system.

3 Sloshing model validation using CFD

In fluid dynamics slosh refers to the movement of liquids inside a container. Even using advanced CFD models, the prediction of the phenomenon is not easy, while their inclusion in real time control systems is practically impossible due to their immense computational cost compared to typical lumped element models. To overcome this issue, mechanical equivalent models are usually employed in missions where sloshing poses a potentially critical disturbance [4][11][33]. The most typical ones are the pendulum and mass spring damper equivalent models. It is important to note that these models do not describe the satellite – liquid force interaction in general, they are only valid under certain conditions. To use these models correctly, their origins must be briefly discussed.

3.1 Lateral sloshing overview

The most common type of fuel sloshing is the *lateral sloshing*. Consider the partially filled tank of Figure 3-1 under some uniform acceleration field, either caused by gravity or by the acceleration of the thrusters on a satellite. The tank oscillates laterally with respect to that acceleration field and an oscillating force is generated in this lateral direction. Further assume that the excitation is small enough in order for the free surface not to have extreme distortions. This type of sloshing force can be predicted accurately by the mechanical equivalent models in Figure 3-1. Although rigorous methods exist to analytically derive these models in the case of simple container shapes based on the linearization of the flow equations and modal analysis, they are beyond the scope of this thesis. The interested reader is referenced to [9]. Qualitatively though, it should be expected that a mass spring damper can capture the basic phenomenon happening here. The sloshing tank can store potential energy due to the gravitational or acceleration field in terms of liquid level and kinetic energy due to the motion of the liquid. The linear oscillator therefore seems like a reasonable mechanism to capture the basic physics of the phenomenon.

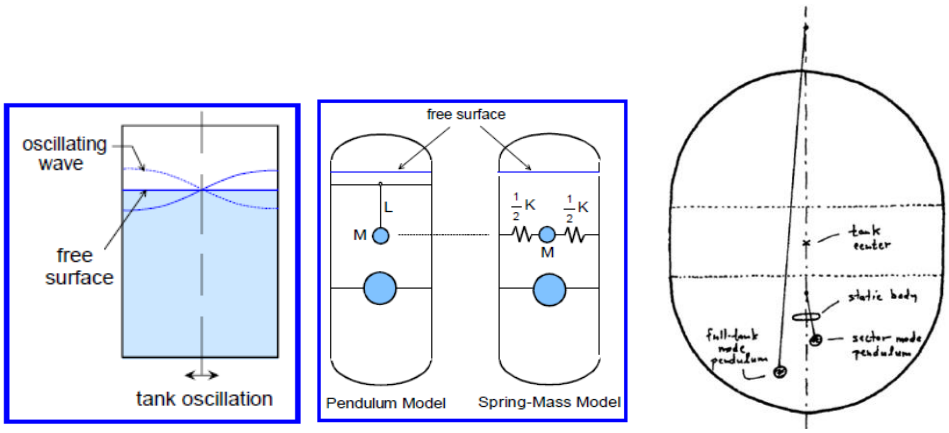


Figure 3-1. (a) actual sloshing plant (b) to proposed mechanical equivalents (c) sloshing model used in Cassini spacecraft [11].

While in laboratory conditions and in terrestrial applications the gravitational acceleration is large enough to dominate all other forces in the flow, this is not always the case in space applications [9][20]. In satellites or rockets the vertical acceleration field generated by the main engine or some other smaller thruster will be significantly weaker than the gravitational field force. In such scenarios surface tension forces in the liquid-gas interface, that are neglected by the linearized flow models, might have a large

impact in the sloshing force response. The number that quantifies their importance is called the Bond number. It is defined as the ratio between acceleration related to surface tension forces. Its formula is

$$Bo = \frac{\rho a R^2}{\sigma} \tag{3-1}$$

where ρ is the fluid density, a the vertical acceleration the container is subjected to, R a characteristic dimension of the container and σ the surface tension in the interface separating the fluid and gas phase. When the Bond number is well above 1 the flow is dominated by the acceleration while when its value is below 1 the flow is capillary dominated. The mechanical equivalent models work better for large bond numbers or high-g sloshing as it is usually referred to while ad-hock modifications exist to extend their use in low-g scenarios [11].

3.1.1 Lateral sloshing benchmark

All the above can be viewed in a simple benchmark case, that will also work for validating the CFD code that will be later used to generate realistic sloshing force data. The objective will be to see how well a mass spring damper model agrees with the CFD data. Consider a simple 3D rectangular container like the one in Figure 3-2. The gravity field is aligned with the z axis, while the container is constrained to move only in the x axis. To simplify the CFD simulation by eliminating the coupling with the rigid body dynamics governing its container, the motion of the later will be prescribed. The solver used is openFOAM's interFOAM solver implementing the Volume of Fluid (VOF) method. Details regarding the exact setup and parameters of the simulation can be found on Appendix A.

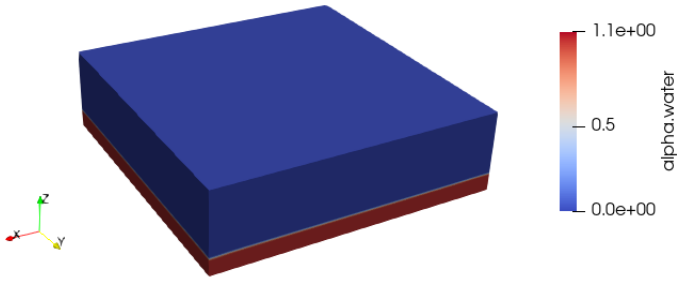


Figure 3-2. Sloshing benchmark for evaluating the CFD simulation.

Table 3-1. Lateral sloshing benchmark case parameters.

Variable description	Symbol [Units]	Value
Container x length	b [m]	0.25
Container y length	a [m]	0.25
Liquid level height	h [m]	0.7
Gravitational acceleration	g [m/s^2]	9.81
Liquid(water)density	ρ_w [kg/m^3]	998.2
Liquid kinematic viscosity	ν_w[m^2/s]	1e-06
Total liquid mass	m_w [kg]	0.98
Air density	ρ_a [kg/m^3]	1
Air viscosity	ν_a [m^2/s]	1.48e-5
Bond number	Bo [-]	8000

Liquid surface tension	σ [N/m]	Neglected (0)
Flow type	-	Laminar

Analytic formulas exist for determining the parameters for the mechanical equivalent model based on the solutions of the linearized flow equations. These relate the sloshing mass and the natural frequencies of the mechanical equivalent model with the size of the container. According to [9] the formulas are:

$$\omega_n^2 = (2n-1)\pi(g/a)\tanh\left(\pi(2n-1)(h/a)\right) \qquad n = 1, 2... \tag{3-2}$$

$$m_i = m_{liq}\left(\frac{8a\tanh\left((2n-1)\pi h/a\right)}{h(2n-1)^3\pi^3}\right) \qquad n = 1, 2... \tag{3-3}$$

where m_{liq} is the total liquid mass, *a*, *b* are the container's dimensions as noted in Table 3-1, *h* is the liquid level height in rest, *g* is the acceleration field intensity the container is subjected to, in this case gravity, and *n* the mode number.

Note that based on (3-2) there is a linear relation between the acceleration the container is subjected to and the square of the natural frequency of the sloshing force. Using a pendulum model has the benefit that this adjustment happens automatically as its natural frequency is given by $\omega_{n,pend}^2 = g/L$ while in the mass spring damper model one has to tune the spring constant accordingly. Furthermore, to preserve the static properties of the liquid [9], the sum of all masses must equal the total liquid mass. This condition yield:

$$m_0 + \sum m_i = m_{liq} \tag{3-4}$$

If one in is interested in the pitching moment created by the sloshing liquid, the vertical position of the oscillating and fixed masses can be adjusted as well, but this will be neglected in this case for simplicity. Equations (3-2) - (3-4) are enough for evaluating all the mechanical equivalent model parameters except from damping. This can easily be set empirically or in a case-specific manner to match the CFD response.

In order to validate the CFD solver, one would be also interested in a "best-fit" model calculated by minimizing the error between the mass spring damper predicted and the CFD response. This will be done here using nonlinear numerical optimization methods in Matlab. To this end, consider the simplified mechanical equivalent system model in (2-9), (2-10). The equations are in state space form and can easily be integrated numerically. Then, the difference between the mass spring damper and the CFD response will be calculated. The optimization cost function will be

$$J = \int_0^{t_f} (F_{CFD} - F_{MSD})^2 \, dt \tag{3-5}$$

The F_{MSD} will be calculated using (2-10), (2-11), with an input u = 0.1 m/s^2 representing the lateral acceleration of the container. For the CFD simulation an appropriate motion profile is used to describe the motion of the container walls. The optimization results can be viewed in Figure 3-3 and Table 3-2.

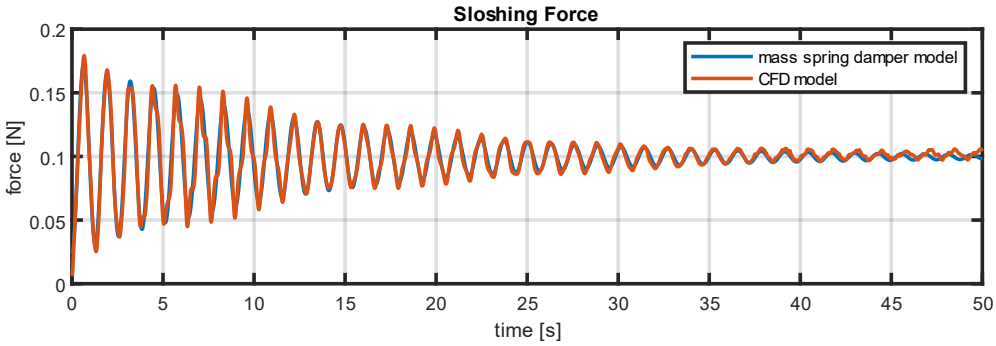


Figure 3-3. CFD force response vs best-fit MSD model.

Table 3-2. Analytic model parameters vs CFD best-fit for the sloshing benchmark case.

Parameter [Units]	Analytic formulas	Best fit model
m_s [kg]	0.79	0.76
m_f [kg]	0.21	0.2
k [N/m]	39	38.8
b [Ns/m]	-	0.24

Concluding remarks:

First of all, indeed the best-fit mechanical equivalent model predicts the true plant, assumed here to be the CFD response, quite well. This is expected since the mechanical equivalent models for lateral sloshing are well established, while the rest parameters of the simulation like the rectangular container shape, the high bond number and the relatively small excitation make the case almost ideal. As seen later in the thesis, the case of fuel sloshing in the spherical container of a satellite in low-g conditions will not be predicted as well.

Furthermore, the analytic formulas and the best fit model seem to closely agree. This is expected as well since the simulation performed is well within the vicinity of the accuracy of the equivalent model in all aspects mentioned earlier. Apart from the parameters of Table 3-2, the 1st natural frequency of the sloshing agrees to a relative error of less than 0.01% with the analytically calculated one from (3-2). These results of course are not expected to be as good for the low-g cases in spacecrafts, however validating the solver for such scenarios goes beyond the scope of the thesis. To this end, the bond numbers selected to be studied here are in generally in the high-g regime.

3.1.2 Satellite lateral sloshing data generation

The inclusion of the CFD analysis in this thesis serves two purposes. The first is to validate the mechanical equivalent sloshing models while making sure they are used in the right context and the second is to generate realistic sloshing force data to test the onboard identification algorithm. The sloshing data generated will be exclusively for lateral sloshing since this is the only case where validated models exist that can capture the physics of the problem. Consider a case where a satellite, as in Figure 3-4, is accelerating in the thrust direction by its main engine or some orbital maneuvering thruster. A lateral force is applied either to chance attitude or to change trajectory. This excites the partially full fuel tank and causes the liquid to slosh. This scenario corresponds to a Bond number much lower that the terrestrial application presented in the benchmark above. For a satellite of mass in the vicinity of

1000 kg and a thrust force of 100N the resulting bond number is 64, still placing it in the high-g regime, although significantly closer a low-g scenario.

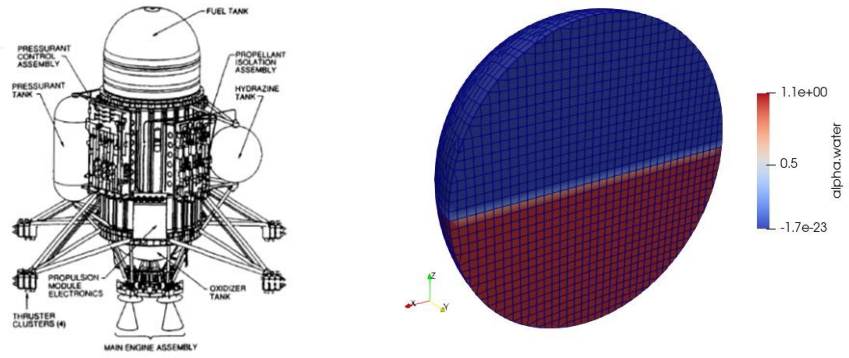


Figure 3-4. Left: Cassini Huygens probe. Right: Spherical liquid fuel tank modelled using the volume of fluid method in interFOAM.

Studying the motion of the spacecraft in the presence of sloshing is a coupled fluid-structure interaction problem since the sloshing forces generated by the fluid interact with the rigid body dynamics affecting its trajectory and vice-versa. To address the problem in a coupled way multiple algorithms have been developed that solve the fluid and rigid body dynamics problem concurrently like the Newmark scheme for integrating the rigid body dynamic equations. This is implemented in the openFOAM environment however it has limited capabilities mainly because it doesn't allow for time varying excitation forces. Even in the case of a constant thrust force, this would be applied in the moving reference frame of the container and therefore would be time varying and dependent on the evolution of the attitude, if expressed in the inertial frame. This essentially restricts the use of coupled simulations with the existing Neumark scheme for very simple scenarios like the 1D sloshing described above where since the orientation remains constant the lateral excitation force doesn't depend on time. To this end, a simulation of such scenario will be performed since it is the simplest possible sloshing experiment perceivable. Note that compared to the benchmark case, apart from the simulation here being coupled, the container is spherical and the conditions are much closer to fuel sloshing in space applications. The thrust wise force though is still replaced with a global acceleration field of appropriate magnitude.

The tank is constrained to move in the x axis only, excited by a 30N lateral disturbance. The initial configuration of the fluid will be assumed to be the steady state, having settled in the rear side of the container due to the thrust force. The complete parameters of the problem can be viewed in Table 3-3.

Table 3-3. Parameters of the spherical sloshing tank used in CFD simulations.

Variable description	Symbol [Units]	Value
Container radius	R [m]	0.5
Liquid(water)density	ρ_w [kg/m^3]	998.2
Liquid kinematic viscosity	ν_w[m^2/s]	1e-06
Total liquid mass	m_w [kg]	264
Total container mass (inc. spacecraft)	m_c [kg]	800
Air density	ρ_a [kg/m^3]	1
Air viscosity	ν_a [m^2/s]	1.48e-5
Thrust-wise acceleration field	a [m/s^2]	0.1 m/s^2
Bond number	Bo [-]	64
Liquid surface tension	σ [N/m]	7e-3
Flow type	-	Laminar

As it can be seen clearly in Figure 3-5, the acceleration calculated from the CFD differs slightly from the ideal sinusoidal response of the lumped parameter system, however the results are still close. It has an average value of approximately 0.029 m/s^2 which is reasonable since the container's combined liquid/rigid mass is approximately 1000 kg and the lateral excitation force 30N.

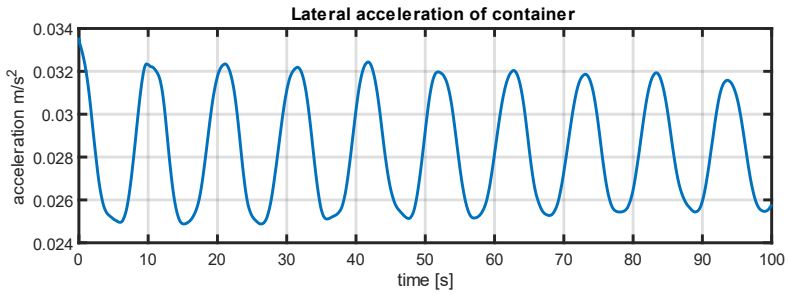


Figure 3-5. Container's lateral acceleration calculated using a coupled CFD/rigid body dynamics solver for a Bond number of 63.

A case similar to the one described above that is also tested is a scenario resulting from a smaller thrust force coming from a cold gas thruster rather than the main engine. Changing the thrust force from 100 to 10 results in an thrust-wise acceleration field of 0.01m/s^2 and a bond number of 6.3. The lateral excitation in this case is set to 5N.

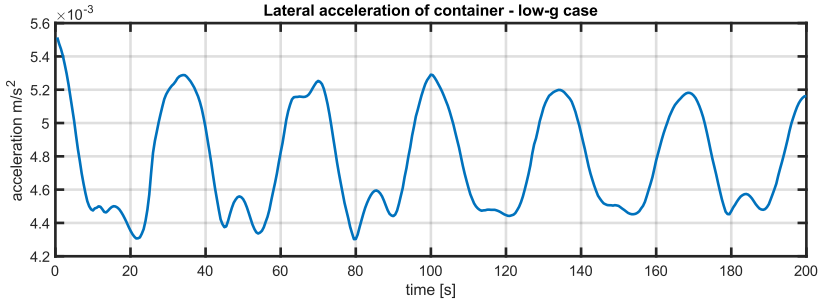


Figure 3-6. Container's lateral acceleration calculated using a coupled CFD/rigid body dynamics solver for a Bond number of 6.3.

By examining the resulting response, one can see that it is qualitatively different from the high-g case. More modes are excited and the non-linear effects of the phenomenon due to surface tension have increased.

The two problems described above are useful in identifying the parameters of a mechanical equivalent models as presented in Chapter 2.2, but are of little use in studying the 3D dynamics systems like those in Chapters 2.3. An interesting point made in this thesis is that the coupled nature of the sloshing satellite problem can be circumvented if one is only interested in creating realistic input-output data for identification purposes. In the true plant, the input is usually the external forces and torques applied on the satellite's rigid body while the output is the readings from an IMU, which is only dependent of the resulting trajectory. Even in the absence of coupled simulation capabilities such an input-output pair could still be generated using prescribed motion, the causality however will have to be reversed. Instead of selecting an input force and find the resulting trajectory we can choose the satellite's trajectory and then calculate the sloshing forces using CFD. By adding the contribution of the satellite's rigid body using its inverse dynamic model one can find the total input forces and torques

that should be applied in order to achieve the prescribed trajectory. A high-level description of the process can be viewed in Figure 3-7.

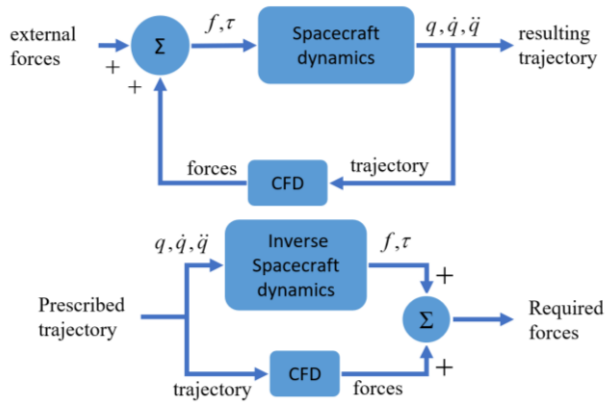


Figure 3-7. Top: Coupled approach in calculating an input-output force-trajectory pair. Lower: Decoupled workaround using prescribed motion.

Although this approach indeed bypasses the coupling problem, caution is advised regarding two thigs. First of all, the prescribed trajectory must be chosen carefully in order for the resulting required inputs to respect the limits of satellite's actuators like thrusters or reaction wheels. Secondly, the resulting input, i.e. the required forces to produce the trajectory, must be sufficiently exciting for the identification algorithm to perform accurately. In parameter estimation experiments, usually step input signals are used that excite all modes or other dynamic properties of the system towards identification. However, since in manner described above, we are only able to choose the output and then find the corresponding input this is not an available option. To address these problems, a rigid body simulation can be performed first with the sloshing replaced with only a rigid equivalent mass for the fuel. A force input can be selected and the resulting trajectory can be used for the process described above.

To complete the above discussion, the equations used in all calculations will be presented. Assume an appropriate trajectory is selected in the manner described above and expressed in terms of, velocity, acceleration, attitude etc. in the spacecraft CoM frame. These will be denoted with the subscript *"p"* from prescribed. The resulting forces and torques will be denoted using *"r"*. Using the Newton-Euler equations for a rigid body one can calculate the required by the spacecraft forces and torques needed to execute this trajectory as

$$\begin{aligned} \mathbf{f}_{r,s/c} &= m\mathbf{a}_p \\ \boldsymbol{\tau}_{r,s/c} &= \mathbf{I}_0\dot{\boldsymbol{\omega}}_p + \boldsymbol{\omega}_p^{\times}\mathbf{I}_0\boldsymbol{\omega}_p \end{aligned} \tag{3-6}$$

where $\mathbf{f}_{r,s/c}$ and $\boldsymbol{\tau}_{r,s/c}$ are the rigid spacecraft's contributions to the required forces and torques. To add the contribution of the sloshing tank, one only needs to take into consideration that the CFD package resolves torques and forces in the inertial frame. These can be easily converted to a frame attached to the tank's center since the trajectory is known. The result is ${}^{t}\mathbf{f}_{r,s}$ and ${}^{t}\boldsymbol{\tau}_{r,s}$. The left superscript *t* denotes the calculations of forces and torques in the tank's reference frame. Then, considering Figure 3-8, the sloshing force can be translated to the spacecraft's CoM using

$$\begin{aligned}\mathbf{f}_{r,s} &= {}^{t}\mathbf{f}_{r,s} \\ \boldsymbol{\tau}_{r,s} &= {}^{t}\boldsymbol{\tau}_{r,s} + \mathbf{p}_{t/0}^{\times}\left({}^{t}\mathbf{f}_{r,s}\right)\end{aligned} \tag{3-7}$$

where $\mathbf{p}_{t/0}$ is the position of the sloshing tank in the spacecraft CoM reference frame. These two frames are considered aligned.

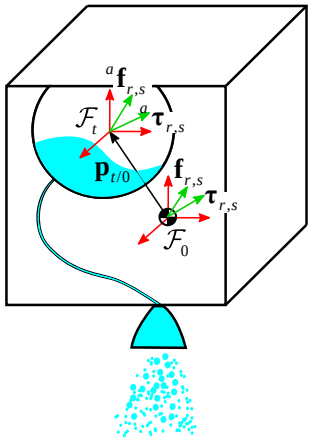


Figure 3-8. Sloshing force transformation from container's frame to Spacecraft's frame

3.2 Other types of fuel sloshing

Apart from the typical case of lateral sloshing, there are different scenarios where a partially full fuel tank in a spacecraft might cause disturbances that the control systems would need to compensate for. Famous cases where these disturbances had a near fatal impact are that of ATS-5 in 1969[35][36], NEAR in 1998 [33] and Falcon 1 in 2007 [3] among others. Due to the complexity of the fluid mechanics equations, it is hard to formulate a simple model that will capture the response of a sloshing tank in the general case. To address this problem sloshing models are created with a specific scenario in mind. When there is a dominating force field like in the case of lateral sloshing the typical mechanical equivalent models already presented can be used.

In the absence of such force, the phenomenon is described as zero-g sloshing and has been an active research field in the last two decades. Dedicated experiments have been launched like the ESA's SLOSHSAT-FLEVO in 2005, a dedicated satellite for gathering experimental data from a container with fluid in zero-g environment and NASA's SPHERES or ESA's Fluidics inside the International Space Station. This data, apart from being useful in studying the phenomenon has been used to fine tune CFD simulators in order to allow for high accuracy virtual experiments that were previously impossible. The outcome of these efforts is the formulation of novel sloshing models that expand the capabilities of the traditional mechanical equivalent models. Some of the resulting models will be presented below.

3.2.1 Zero-g sloshing for spacecraft attitude maneuvers

In the case where the motion of the satellite is only controlled via reaction wheels or disturbed by some appendage like a manipulator the mass spring damper fails to predict the satellite-fuel interaction [29] A series of papers have been published by the *French Aerospace Lab* [3][4] regarding the formulation and potential applications of a novel sloshing model based on a generalized non-linear second order

system. The model refers to a scenario where the satellite is only attitude-controlled via reaction wheel torques. The researchers used the structure of a typical second order dynamic system to describe the sloshing interaction with the satellite. They assumed a general relation between the inertia, coriolis-centrifugal and stiffness matrices with the spacecraft angular velocity and acceleration variables and using optimization methods they matched the predicted and true sloshing force response. In order generate realistic sloshing force data a CFD solver previously developed by the IMFT in particular for this type of multiphase flows was validated first.

Using this model, they created the Robust H-infinity observer-based control system of Figure 3-9 that suppress the sloshing modes and works independently of the main control system of the satellite. Although their approach worked and the sloshing disturbances were mitigated while their model retained a relatively low complexity the method has not been tested neither in a coupled CFD simulation nor in a in a real mission yet.

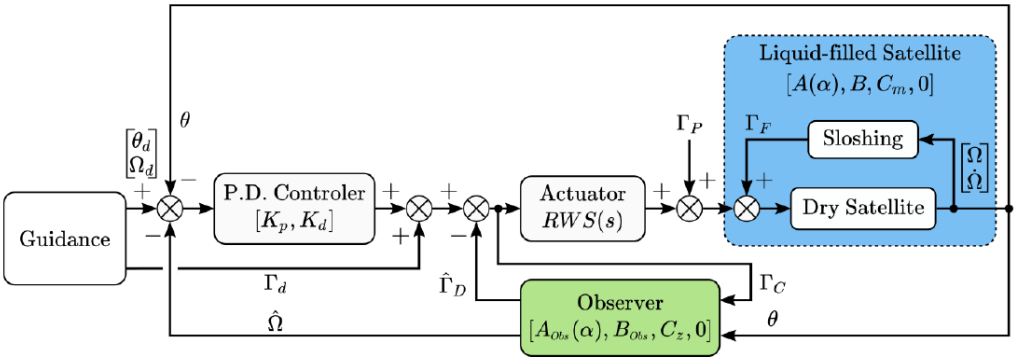


Figure 3-9. Control Scheme proposed by [3] based on a robust observer used to cancel out the sloshing effects

3.2.2 Pulsating ball sloshing model

Using the outcomes of SLOSHSAT-FLEVO the NRL formulated the Sloshsat Motion Simulator, a 22-component state model to allow for a more general sloshing force prediction [35][36]. The liquid was modelled as a spherical slug with variable size but constant density, with its size being controlled by the contact force from the container walls. The model although is stated to be an improvement with regards to the classic mechanical equivalent models has not, to the authors knowledge been used in a real space mission yet.

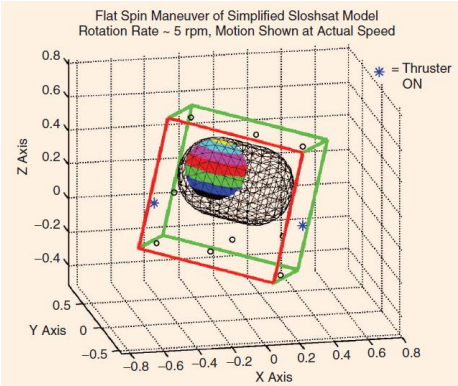


Figure 3-10. Sloshsat Motion Simulator, according to [35],[36].

3.2.3 Sloshing in the direction of acceleration

A case of sloshing that is explored in this thesis is the one of sloshing in the direction of motion. Assume that a satellite is at rest and all liquid in the sloshing tank has obtained a steady state configuration. Assume that the tank starts to accelerate by means of small thrusters in one direction to perform a translational motion. This might be the during a docking maneuver or in a debris capture scenario. The problem investigated here is whether a mass spring damper model can capture the basic physics of the phenomenon.

The parameters for the problem are a spherical container of 0.5 m radius filled with 50% water and 50% air. A reasonable starting configuration for the fluid and water phase can be seen in Figure 3-11. It minimizes the energy stored in surface tension and results in minimum free surface area of the liquid fuel. The liquid phase is concentrated in the outer walls of the container and the gas phase forms a bubble in the center of the container [29].

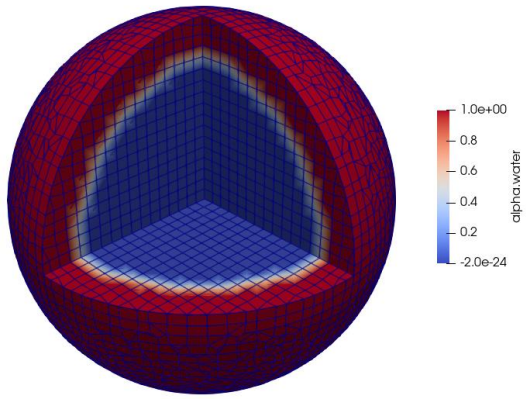
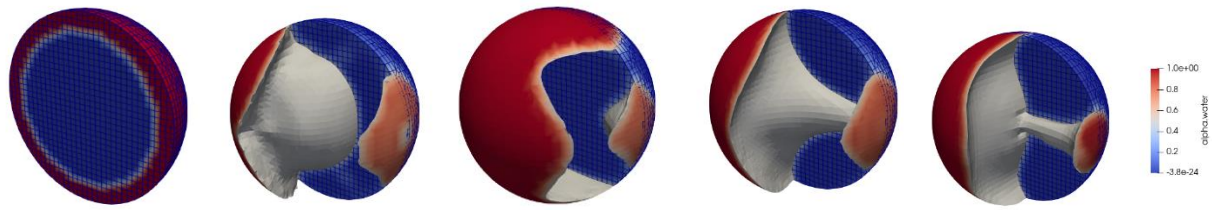


Figure 3-11. Steady state configuration of partially full container. The scalar variable alpha.water represents the fluid phase: 1 translates to 100% liquid and 0 to 100% gas.



The simulation performed here is coupled. The Fluid dynamics solver interacts with the rigid body dynamics and the motion of the tank is affected by the sloshing force. The tank is considered a rigid body with a mass of 800 kg to include the mass of the satellite. The total liquid mass is 261 kg. The tank is constrained to move in one axis and is excited by a constant 50N force. The resulting average acceleration of the system under these conditions is 0.047 m/s^2 and the Bond number is 32, which places the simulation in the high-g regime.

The resulting sloshing force calculated from the simulation can be viewed in Figure 3-13. On a first look the response resembles one from a linear oscillator like the mass spring damper model. On a closer examination however, two issues arise.

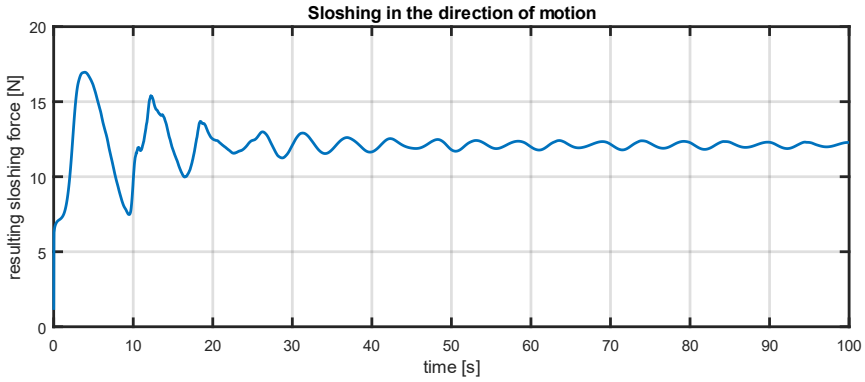


Figure 3-13. Sloshing force in the direction of acceleration

First, the oscillation period in Figure 3-13 is not constant. The first 2 oscillations have a period of about 8.8 s while the lower amplitude oscillations after t = 30 s have a period of 5.3 s. This could be resolved by expanding the mechanical equivalent model to a dual mode system (see Figure 3-14). The equations for the 2-mode lumped parameter system in state space form are

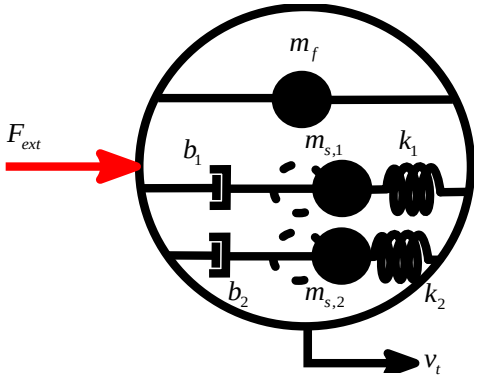


Figure 3-14. Dual mode MSD model for capturing sloshing in the direction of motion

$$
\begin{aligned}
\dot{v}_t &= \frac{1}{m_f}(k_1 x_1 + k_2 x_2 + b_1 v_1 + b_2 v_2) + \frac{1}{m_f} F_{ext} \\
\dot{v}_1 &= -\frac{1}{m_f m_{s,1}}\left(k_1 m_f x_1 + k_1 m_{s,1} x_1 + k_2 m_{s,1} x_2 + b_1 m_f v_1 + b_1 m_{s,1} v_1 + b_2 m_{s,1} v_2\right) - \frac{1}{m_f} F_{ext} \\
\dot{v}_2 &= -\frac{1}{m_f m_{s,2}}\left(k_1 m_{s,2} x_1 + k_2 m_f x_2 + k_2 m_{s,2} x_2 + b_1 m_{s,2} v_1 + b_2 m_f v_2 + b_2 m_{s,2} v_2\right) - \frac{1}{m_f} F_{ext} \\
\dot{x}_1 &= v_1 \\
\dot{x}_2 &= v_2 \\
\dot{x}_3 &= v_3
\end{aligned}
\tag{3-8}
$$

Where v_1, v_2 and x_1, x_2 are the velocities and positions of the 2 sloshing masses $m_{s,1}, m_{s,2}$ with respect to the fixed mass m_f, k_1, k_2 and b_1, b_2 are the spring and damping constants respectively, and v_t, x_t the velocity and position of the tank relative to an inertial frame. Under this notation the sloshing force is equal to

$$F_s = K_1 x_1 + b_1 (v_1 - v_s) + K_2 x_2 + b_2 (v_2 - v_s) + m_1 \dot{v}_1 \tag{3-9}$$

Using optimization methods similarly to Section 3.1.1 and a cost function like (3-5) one can find the best fit model parameters of (3-8) to match the data of Figure 3-13. It was observed however that in order for the response of the mechanical equivalent model to match the CFD response the initial conditions of the lumped parameter system should be included in the optimization variables. This was in order for the phase of the oscillations in the two simulations to match the CFD data. Although the results (see Figure 3-15) look promising, on a close examination in the start of the simulation the effects of non-zero initial conditions in terms of sloshing masses velocities and spring compressions can be seen clearly. The downside of this is that there is no known way to systematically determine the initial conditions of (3-8) the in the mechanical equivalent model, as well as anything other than zero lacks a good physical interpretation given the steady state initial conditions of the CFD, while it also breaks the symmetry of the problem.

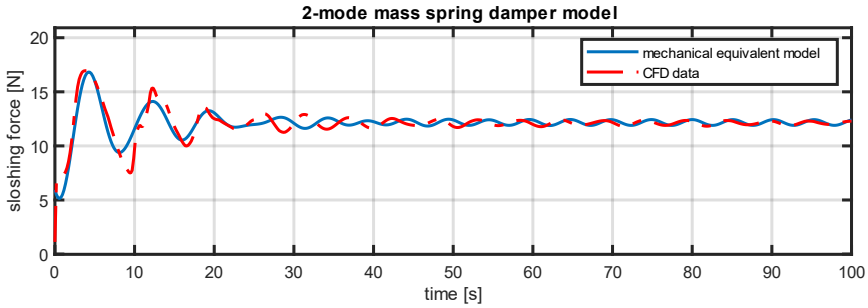


Figure 3-15. Best fit model for the sloshing in direction of acceleration modelled by a 2-mode MSD.

For realizing the second issue of this approach, the simulation must be extended to include the phase where the thruster force stops. Unfortunately, as already mentioned, currently there is no way to have time varying forces in the coupled CFD simulation. This scenario will therefore be studied using prescribed motion. The motion of the tank will be split into an accelerating phase in the beginning corresponding to the thrusters being turned on followed by a constant velocity phase. The question investigated is whether the model identified previously will capture, even roughly, the CFD response.

The issue can be viewed clearly in Figure 3-16. As mentioned earlier, sloshing occurs only in the presence of an acceleration field from thrusters or gravity. In the case of lateral sloshing the thrust force remains activated so the system can store potential energy even if the lateral excitation force stops. In the above case though, the excitation force in the direction of motion is the only one creating the acceleration field, so when it is turned off the system cannot store potential energy and won't oscillate. This is a fundamental difference between the two models and this is why the mass spring damper mechanical equivalent model cannot be used in for describing the phenomenon in a general enough manner.

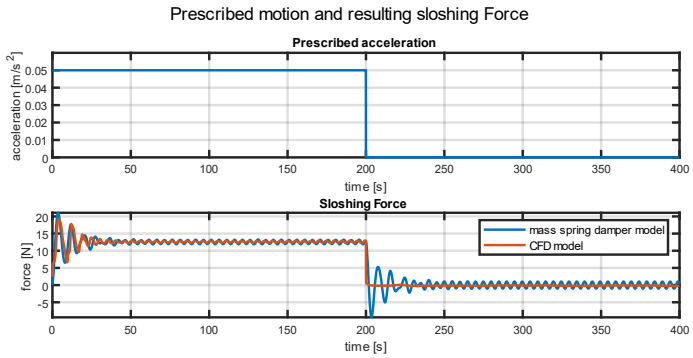


Figure 3-16. Issue with the MSD model for describing sloshing in the direction of acceleration

4 Concurrent state-parameter estimation

4.1 Problem statement and literature review

The combined parameter and state estimation is far from a trivial problem. To provide some context, consider a system described by some dynamic model which has a known structure however unknown values in some of its parameters. If all system state variables are measurable, methods based on linear regression can usually be utilized. In the absence of full state measurements these methods can't be employed directly. If the system is linear, one can transform the state equations into an input-output form, eliminating the unmeasurable state variables. In the case of non-linear state equations though this is not applicable. One possible approach to overcome the issue of unmeasurable state variables is by employing state estimation algorithms like Luenberger observers, Kalman Filters or a similar method. These methods however require knowledge of the system's dynamic equations which are unknown. This problem, also known as a dual estimation problem, was originally posed as a nonlinear estimation problem by augmenting the state vector with the unknown parameters of the state space equations. Kopp and Orford [18] proposed the extended Kalman filter to solve the resulting nonlinear filtering problem.

In general, the algorithms used to solve such problems rely on alternating between using the model to estimate the state, and using the state to estimate the model. According to [12], this process may be either iterative or sequential. Iterative schemes work by repeatedly estimating the state using the current model and all available data, and then estimating the model using the state estimates and all the data. Iterative schemes are necessarily restricted to off-line applications, where a batch of data has been previously collected for processing. In contrast, sequential approaches use each individual measurement as soon as it becomes available to update both the signal and model estimates. This characteristic makes these algorithms useful in either on line or off-line applications.

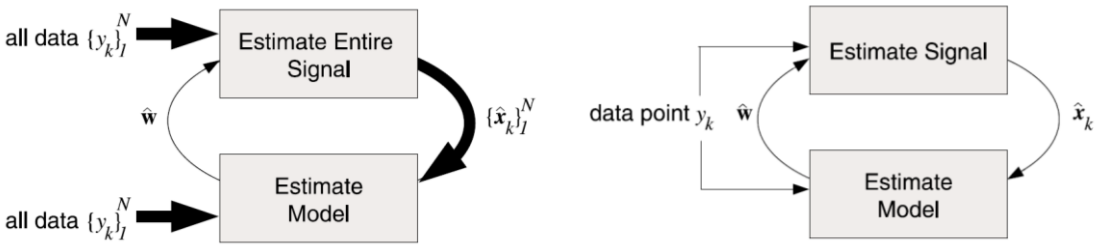


Figure 4-1. Iterative vs Sequential methods for solving the dual estimation problem, taken from [12].

4.2 The unscented Kalman filter

The Kalman Filter is a well-known algorithm for optimal state estimation in discrete time linear systems in the presence of measurements corrupted by Gaussian white noise. Ever since its inception by Rudolf Kalman, it has highly influenced many areas ranging from the aerospace industry to economics. As with all state estimators, it is used to recreate the full state vector of a system based on its input-output data and knowledge of its dynamics. Having met an unprecedented success [20], many researchers have proposed extensions to the classic algorithm to generalize it to nonlinear systems. The unscented

Kalman Filter was originally proposed by Simon J. Julier and Jeffrey K. Uhlmann [16] as an improvement to the widely used Extended Kalman Filter (EKF). The EKF is based on a local linearization of the nonlinear system dynamics in order to propagate the noise covariance matrices and extend the classic linear Kalman Filter equations to nonlinear systems. The unscented Kalman filter is based on the Unscented Transformation (UT), which is founded on the intuition that *"it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation"* (see Figure 4-2). In practice, the algorithm uses an appropriately chosen weighted number of points, called sigma points, to parametrize the probability distribution. Instead of analytically propagating the covariance matrix through the nonlinear system, the sigma points are propagated directly and then their statistical properties are calculated. This prevents the need for analytically calculating Jacobians for linearization, making this technique appropriate for black box models as well. It also yields superior performance of up to third order compared to the first order approximation of the EKF, while maintaining a similar computational burden. This approach was later adopted by Wan and Van der Merwe who greatly expanded its applications to neural network training and dual estimation problems [34][37][38].

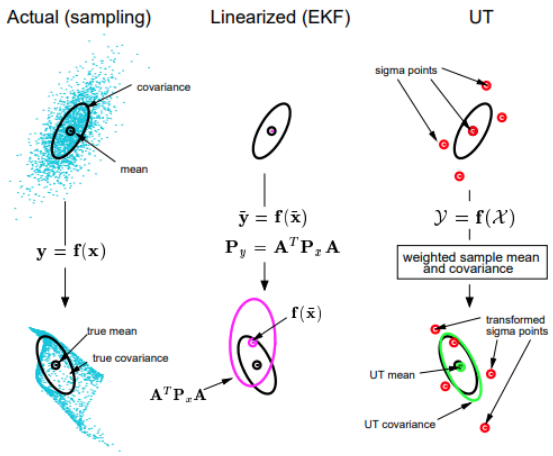


Figure 4-2. Unscented transformation based covariance estimation vs linearized approach [38].

To sum up the process, the Unscented Kalman Filter is composed of a number of discrete steps. Consider a discrete non-linear dynamical system described by

$$\begin{aligned} \mathbf{x}_{k+1} &= F(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k \\ \mathbf{y}_k &= H(\mathbf{x}_k) + \mathbf{v}_k \end{aligned} \tag{4-1}$$

where $\mathbf{x}_k$ is the state vector of dimension n, $\mathbf{u}$ the system's input, $F(\cdot)$ the state transition function and $\mathbf{w}$ the noise input of the system, in this case -however not necessarily- additive. The second equation in (4-1) is usually referred to as the measurement model and extracts the measurements $\mathbf{y}$ from the state, while accounting for measurement noise $\mathbf{v}$. In the general case it is nonlinear, however in the typical case where some of the states are directly observable the linear $\mathbf{C}\mathbf{x}_k$ can be considered. The objective is to progressively construct an estimate of the system's state $\hat{\mathbf{x}}_k$ from the available measurements.

Before proceeding to the main equations of the filter, some preliminaries are defined:

- $\mathbf{w}$ and $\mathbf{v}$ are Gaussian random variables with constant covariances $\mathbf{Q}$ and $\mathbf{R}$ respectively.
- $\mathcal{X}_k$ is a matrix containing the sigma points for propagating the state vector's covariance.

- $\mathbf{x}_{k|k-1}$ is the *a priori* prediction of the state in instance k using the model and measurements up to $k-1$
- $\left(\mathcal{X}_{k|k-1}\right)_i$ represents the i[th] column of the matrix $\mathcal{X}_{k|k-1}$
- α, β are scaling parameters for the UT. They control the spread of the sigma points and are set to values 10^{-3} and 2 respectively while rarely require tuning.
- The weights for the UT are given by

$$\begin{aligned} W_i^{(m)} &= \begin{cases} \lambda/(n+\lambda) & \text{i = 0} \\ 1/\left(2(n+\lambda)\right) & \text{i = 1,2...n} \end{cases} \\ W_i^{(c)} &= \begin{cases} \lambda/(n+\lambda)+\left(1-\alpha^2+\beta\right) & \text{i = 0} \\ 1/\left(2(n+\lambda)\right) & \text{i = 1,2...L} \end{cases} \\ \lambda &= (\alpha^2-1)n \end{aligned} \tag{4-2}$$

The complete UKF can now be presented, as given in [38]:

Initialize using

$$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]$$
$$\mathbf{P}_0 = E[(\mathbf{x}_0-\hat{\mathbf{x}}_0)(\mathbf{x}_0-\hat{\mathbf{x}}_0)^T]$$

For $k=\{1,2...\infty\}$

1. Calculate sigma points[3]

$$\mathcal{X}_k^x = \left[\hat{\mathbf{x}}_{k-1}^+ \;\; \hat{\mathbf{x}}_{k-1}^+ + \left(\sqrt{(n+\lambda)\mathbf{P}_{k-1}}\right) \;\; \hat{\mathbf{x}}_{k-1}^+ - \left(\sqrt{(n+\lambda)\mathbf{P}_{k-1}}\right)\right] \tag{4-3}$$

   Calculate the *a priori* state estimation and its covariance

$$\mathcal{X}_{k|k-1} = F\left(\mathcal{X}_{k-1}, \mathbf{u}_{k-1}\right) \tag{4-4}$$

$$\hat{\mathbf{x}}_k^- = \sum_{i=0}^{2L+1} W_i^{(m)} \left(\mathcal{X}_{k|k-1}\right)_i \tag{4-5}$$

$$\mathbf{P}_k^- = \sum_{i=0}^{2L+1} W_i^{(c)} \left[\left(\mathcal{X}_{k|k-1}\right)_i - \hat{\mathbf{x}}_k^-\right]\left[\left(\mathcal{X}_{k|k-1}\right)_i - \hat{\mathbf{x}}_k^-\right]^T + \mathbf{Q} \tag{4-6}$$

2. Use the measurement function to translate the state prediction and covariance to measurement. If the measurement function is linear the original analytic Kalman Filter equations can be used. In the general case the UT can be employed in the same fashion as in (4-3) - (4-5).

$$\mathcal{Y}_{k|k-1} = H\left(\mathcal{X}_{k|k-1}\right) \tag{4-7}$$

$$\hat{\mathbf{y}}_k^- = \sum_{i=0}^{2L+1} W_i^{(m)} \left(\mathcal{Y}_{k|k-1}\right)_i \tag{4-8}$$

$$\mathbf{S}_k = \sum_{i=0}^{2L+1} W_i^{(c)} \left[\left(\mathcal{Y}_{k|k-1}\right)_i - \hat{\mathbf{x}}_k^-\right]\left[\left(\mathcal{Y}_{k|k-1}\right)_i - \hat{\mathbf{x}}_k^-\right]^T + \mathbf{R} \tag{4-9}$$

3. Calculate the Kalman Gain and the *a posteriori* optimal state prediction

[3] The square root matrix in (4-3) is usually calculated using the Cholesky factorization. If the calculation is not accurate though, divergence might occur due to the covariance matrix becoming non symmetric. For increased efficiency and accuracy, the reader is referred to the Square Root Unscented Kalman Filter [34].

$$\mathcal{K}_k = \mathbf{P}_k^- \mathbf{C}^T \left(\mathbf{S}_k\right)^{-1} \tag{4-10}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathcal{K}_k \mathbf{y}_k \tag{4-11}$$

4. Calculate the covariance for the *a posteriori* state prediction

$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathcal{K}_k C\right)\mathbf{P}_k^- \tag{4-12}$$

4.3 The Unscented Kalman Filter for concurrent state and parameter estimation

Two separate methods exist to attack the problem of concurrent state and parameter estimation using the UKF, the *joint* and the *dual* approach. In the joint approach, one appends the parameters that require identification to the state vector and treats them as state variables with zero dynamics. In the dual approach two separate filters are set, one estimating the state using the best currently available value of the parameters and another estimating the parameters based on the current best estimation of the state. Both approaches have been widely used by the research community with common joint cases being [13][42] and dual approaches [31][41].

According to the literature, it is hard to make general conclusions about which performs better in the general case. The Dual approach provides more flexibility because the state and parameters estimation processes are decoupled. Furthermore, the parameter estimation can be switched off after convergence, increasing the accuracy of the state estimation [31]. While sources exist that suggest that the dual configuration yields better results, according to [28] the joint configuration can outperform the dual if tuned properly.

4.3.1 Joint approach

For the general case, consider the discrete system of (4-1) where the dependence of the model with a vector of parameters π is explicitly stated

$$\begin{aligned} \mathbf{x}_{k+1} &= F(\mathbf{x}_k, \boldsymbol{\pi}, \mathbf{u}_k) + \mathbf{w}_k \\ \mathbf{y}_k &= H(\mathbf{x}_k, \mathbf{v}_k) \end{aligned} \tag{4-13}$$

The concept of the joint approach is to augment the parameter vector in the state vector and assume zero dynamics in the appended state variables. This translates to the system

$$\begin{aligned} \begin{bmatrix} \mathbf{x}_{k+1} \\ \boldsymbol{\pi}_{k+1} \end{bmatrix} &= \begin{bmatrix} F(\mathbf{x}_k, \boldsymbol{\pi}_k, \mathbf{u}_k) + \mathbf{w}_k \\ \boldsymbol{\pi}_k \end{bmatrix} + \mathbf{w}_k \\ \mathbf{y}_k &= H(\mathbf{x}_k, \mathbf{v}_k) \end{aligned} \tag{4-14}$$

Assuming that the augmented system has full state observability the classic UKF equations presented in Chapter 4.2 can be directly used to reconstruct the system's state. To present the strength of this approach, one can refer to [13] where an adaptive control system based on the joint-UKF is proposed for stabilizing an inverted double pendulum with unknown model parameters.

Seeing Figure 4-3, the results are indeed very impressive given the complex and highly non-linear dynamics governing the double pendulum model, however it is noted that this benchmark is a case where the dynamics of the system are indeed very close to the analytic model running in the Kalman Filter equations, unlike the fuel sloshing models that are of interest here, where considerable model

mismatch might be present. Furthermore, it should be noted that all states of the inverted pendulum system are measured, and the UKF is used for denoising, rather that estimating the pendulum angles states.

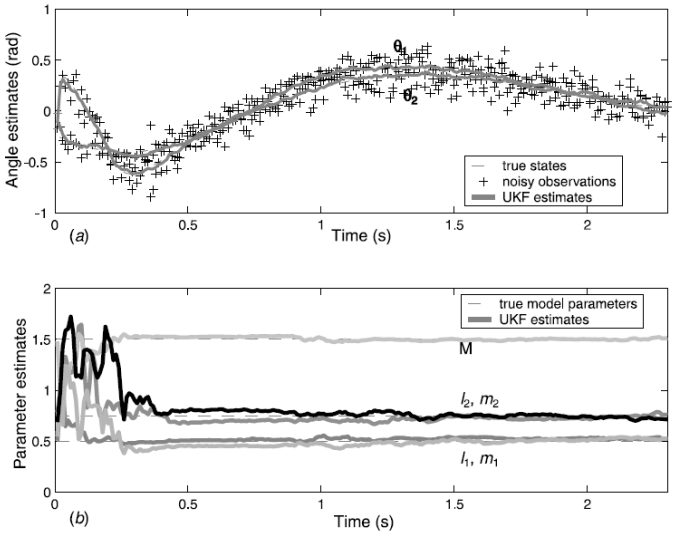


Figure 4-3. Joint state-parameter estimation and adaptive control of a double inverted pendulum using a joint UKF [12].

4.3.2 Dual approach

In the dual approach, two separate, interconnected filters are created. Before presenting the full algorithm, some comments regarding the parameter estimation part are interesting. The algorithm uses the Kalman filter for estimating the parameters of a nonlinear mapping $G(\mathbf{x}_k, \boldsymbol{\pi})$ from clean data x_k. Provided a training set of input output data $\{\mathbf{x}_k, \mathbf{d}_k\}$, one wishes to minimize the error $\mathbf{e}_k = \mathbf{d}_k - G(\mathbf{x}_k, \boldsymbol{\pi})$. Note that $G(\cdot)$ might be any nonlinear function, ranging from a neural network to a parametrized dynamic system. If it is a dynamic model then $\mathbf{x}_k$ and $\mathbf{d}_k$ are essentially the state at an instant *k* and the prediction of the state according to $G(\cdot)$ at instant *k+1* respectively. Reframing the problem in the Kalman Filter framework the equations are

$$\begin{aligned} \boldsymbol{\pi}_{k+1} &= \boldsymbol{\pi}_k + \mathbf{r}_k \\ \mathbf{d}_k &= G(\mathbf{x}_k, \boldsymbol{\pi}_k) + \mathbf{e}_k \end{aligned} \tag{4-15}$$

where a stationary process model along with a nonlinear measurement model is used. The $\mathbf{r}_k$ and $\mathbf{e}_k$ are the process and measurement noise respectively, with covariances $\mathbf{R}_k^r$ and $\mathbf{R}_k^e$. If the Kalman filter is employed in this scenario it can be shown that it minimizes the mean square error

$$J(\boldsymbol{\pi}) = \sum_{k=1}^{\infty} \left[\mathbf{d}_k - G\left(\mathbf{x}_k, \mathbf{w}\right)\right]^T \left(\mathbf{R}_e\right)^{-1} \left[\mathbf{d}_k - G\left(\mathbf{x}_k, \mathbf{w}\right)\right] \tag{4-16}$$

According to [12], if the "noise" covariance $\mathbf{R}^e$ is a constant diagonal matrix it cancels out of the algorithm and hence can be set arbitrarily. The innovations covariance $\mathbf{R}_k^r$ does not represent noise but instead works as the driving component for the parameter estimation, affecting its convergence rate and tracking performance. The larger the value of the $\mathbf{R}_k^r$, the faster old data get discarder. Multiple approaches exist for setting $\mathbf{R}_k^r$. Some of the most common are to set it to an arbitrary value, reducing in to zero as training continues or setting $\mathbf{R}_k^r = (\lambda^{-1} - 1)\mathbf{P}_{\pi_k}$ where $\lambda \in (0,1]$ is referred to as the

forgetting factor. More advanced methods exist as well but won't be examined here. It is interesting to comment that the EKF in this case is equivalent to a modified Newton method, reducing the cost function (4-16) in every iteration.

Although using the Kalman Filter in this fashion might seem unorthodox at first, it has been used successfully for Neural Network training and model parameter estimation among others. In order to formulate the concurrent state-parameter estimation problem according to (4-15) all that remains is to somehow obtain the training set $\{\mathbf{x}_k, \mathbf{d}_k\}$ which would require full state measurements. To this end, a second filter is implemented solely for computing a noise-free estimation of the full state vector from the noisy measurements available and the current best guess of the model parameters. The hole process can be summarized as follows:

Parameter UKF equations		**State UKF equations**	
$\boldsymbol{\pi}_{k+1} = \boldsymbol{\pi}_k + \mathbf{r}_k$ $\mathbf{d}_k = F(\mathbf{x}_k, \boldsymbol{\pi}_k) + \mathbf{e}_k$	(4-17)	$\mathbf{x}_{k+1} = F(\mathbf{x}_k, \boldsymbol{\pi}_k, \mathbf{u}_k) + \mathbf{w}_k$ $\mathbf{y}_k = H(\mathbf{x}_k, \mathbf{v}_k)$	(4-18)

1. Initialize both filters with

State estimation	**Parameter estimation**
$\mathbf{x}_0 = E[\hat{\mathbf{x}}], \mathbf{P}_0 = E\left[\left(\hat{\mathbf{x}}_0 - \mathbf{x}_0\right)^T \left(\hat{\mathbf{x}}_0 - \mathbf{x}_0\right)\right]$	$\hat{\boldsymbol{\pi}}_0 = E\left[\boldsymbol{\pi}_0\right], P_0^{\pi} = E\left[\left(\hat{\boldsymbol{\pi}}_0 - \boldsymbol{\pi}_0\right)^T \left(\hat{\boldsymbol{\pi}}_0 - \boldsymbol{\pi}_0\right)\right]$

Iterate for all data available for $k = 1, 2..$

2. Use the UKF equations on the state model (4-18) to calculate the *a priori* state estimation and covariance $\hat{\mathbf{x}}_{k+1}^{-}$ $\mathbf{P}_{k+1}^{-}$ based on the current value of the system parameters $\hat{\boldsymbol{\pi}}_{k-1}^{+} = \hat{\boldsymbol{\pi}}_k$
3. Calculate the Kalman Gain and correct the *a priori* prediction based on the current measurement $\mathbf{y}_k$ yielding the *a posteriori* estimation $\hat{\mathbf{x}}_k^{+}$.
4. Substituting $\hat{\mathbf{x}}_k^{+} \to \mathbf{d}_k$ and $\hat{\mathbf{x}}_{k-1}^{+} \to \mathbf{x}_k$, calculate the optimal Kalman gain and correct the *a priori* $\hat{\boldsymbol{\pi}}_k^{-}$ to the *a posteriori* $\hat{\boldsymbol{\pi}}_k^{+}$

4.4 Other methods for parameter estimation in the presence of unmeasurable states

Two other methods can be found in the literature that are applicable in the case of parameter identification in the presence of unmeasurable state variables. The first is based on the ubiquitous particle filters and can be considered as an extension of the methods already discussed while the second is nonlinear numerical optimization methods.

4.4.1 Particle filters

Particle filters are a more general approach to dealing with the propagation of a random variable through a nonlinear system. Having their roots in the Monte Carlo technique, instead of using a small number of points to approximate the random variable they use exhaustive sampling. This highly increases the computational burden, however the Gaussian distribution hypothesis for the random variables is eliminated, giving the method much higher performance. Multiple papers exist in the literature where the method has been used for concurrent state and parameter estimation, with famous examples being

from chemistry, where state variables, usually being concentration of substances in solutions, might be hard to measure while the computational load in laboratory conditions does not pose a problem [6]. A benefit of this approach is that the result of the parameter estimation is a probability distribution function representing the probability distribution of the actual parameter [32].

4.4.2 Nonlinear numerical optimization methods

Another popular alternative used in the literature is none other than numerical optimization techniques [15]. These methods are almost exclusively used offline both due to their high computational requirements but also because they process the data as a hole, after the identification experiment data gathering has been completed. This is in contrast with the recursive methods presented so far where the estimation happens in real-time while the data is still being gathered. The formulation of the problem in this case is simple and can be summarized as follows. Consider a continuous time dynamic model of known structure in the form

$$\begin{aligned} \dot{\mathbf{x}} &= f(\mathbf{x}, \boldsymbol{\pi}, \mathbf{u}) \\ \mathbf{y} &= g(\mathbf{x}, \boldsymbol{\pi}) \end{aligned} \tag{4-19}$$

where the parameter vector $\boldsymbol{\pi}$ will be used as the variables for optimization. From some identification experiment we have gathered data $\mathbf{y}_{\exp}(t)$ where $t \in [0, t_{end}]$ and we wish to find the parameters of the model in (4-19) in order for the model predicted response to match the experimental one. This can be expressed in the following cost function

$$J = \int_0^T \left(\mathbf{y}(t) - \mathbf{y}(t)_{\exp}\right)^T \left(\mathbf{y}(t) - \mathbf{y}(t)_{\exp}\right) dt \tag{4-20}$$

Assuming some gradient based numerical optimization method is selected like the Gradient Descent method, or a quasi-Newton method like the BFGS, the problem reduces to the calculation of the sensitivity derivatives $\partial J / \partial \boldsymbol{\pi}$. Note that if the initial conditions in (4-19) are unknown they can be appended to the optimization variables as well. The procedure, described in Figure 4-4, needs to be initialized from some initial guess of the optimization variables $\boldsymbol{\pi}_0$ and using an iterative algorithm, converge to a local minimum of the cost function. A disadvantage of the method is that convergence cannot be guaranteed nor can one know whether the algorithm has reached the global or a local minimum of the cost function.

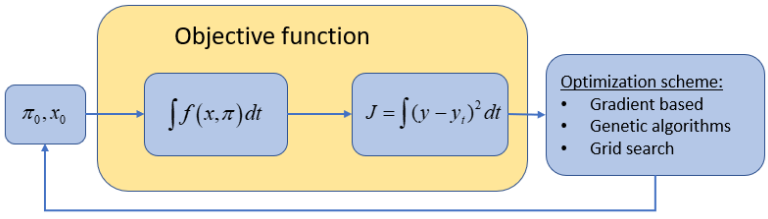


Figure 4-4. Numerical optimization scheme for solving parameter estimation problem with unmeasurable state variables.

The most computationally expensive part of the procedure is the calculation of the sensitivity derivatives $\partial J / \partial \boldsymbol{\pi}$. In order to calculate in using finite differences for a parameter vector of size *n*, the ODE in (4-19) needs to be integrated *n+1* times and given the iterative nature of the algorithm, the process might be slow. Current advances in optimization methods [25] have replaced this burdensome calculation with the adjoint method, that requires only a single integration of the dynamic system

equations and one integration of the adjoint equations that are of similar complexity. This might allow for future potential uses of these methods in online parameter estimation scenarios.

In this thesis the numerical optimization method is used for comparison against the Kalman Filter based methods when the true model parameters are unknown. These are for example scenarios where the measurements fed to the estimator come from CFD simulations of the sloshing plant. The optimization method used will be the Matlab's integrated *fminunc()* and *fmincon()* or even genetic algorithms like *ga()* which are less prone to converging to local minima.

4.5 Tuning methods

As previously mentioned, the performance of the UKF based identification is very highly dependent on the tuning of the filter. Although guidelines exist for this process, it still remains a arduous task that requires a lot of trial and error. More sophisticated automatic tuning methods based on grid search or other numerical optimization methods exist as well, however they lack the intuition that is gained on the effect of each tuning variable while hand tuning the filter, so it is advised that they are used for final fine-tuning rather that from the beginning of the procedure.

In the joint configuration tuning the filter is similar to tuning a classic UKF for state estimation. The most important tuning parameters are the process noise and measurement noise covariances. These essentially control how new uncertainty enters the system with each measurement and every time the system state equations are used. Starting with the simpler case of measurement noise a first approach is to turn to the sensor's noise characteristics. However, at least in this study, the measurements fed to the UKF are not directly measured since an integration process is taking place to compute velocities from accelerations. This is therefore not an option so the covariance will be set empirically. Figure 2-6 helps visualizing the magnitude of the measurement error and set this variable accordingly. It is advised one can start with a diagonal matrix with values being set in a $\pm 3\sigma$ fashion in the sense that the error between the measured and true value of the input signal always lies within the assumed error distribution.

Setting the process noise is a more challenging task. For one thing, the uncertainty that is linked with this variable is much more complex. In general, process noise covers errors from the following cases:

1. Errors due to the inexact first order discretization of the continuous time state equations. This is one of the biggest sources of uncertainty contained in process noise. Quantifying it is not easy in the general case, although one could get an estimate of its magnitude by looking at the differences of the first order scheme with the Runge Kutta based numerical integration of the equations of motion.
2. Errors due to intentional system model simplifications. This could be the linearization of the plant equations or the deliberated negligence of some terms, like in the higher order terms in the flexible panel kinetic energy, to make the equations simpler.
3. Errors due to unmodelled dynamics.

Kalman filter-based algorithms are in general not tolerant to modelling inaccuracies because they assume the process noise is gaussian and uncorrelated with the system's state. It is clear that trying to analytically estimate the values of the individual sources of uncertainty is not a practical solution for setting the process noise covariance matrix. A helpful alternative is using a simple visual guideline to access the filter's performance and tune it accordingly. One of the Kalman filter's assumptions is that the state vector is a Gaussian Random Variable (GRV) with a mean equal to its true value and a

corresponding covariance. In each iteration of the filter, a realization of the GRV is used, which in order for the above assumption to hold, should lie within the assumed distribution. Exploiting access to the state's covariance matrix in each iteration, one could export each state variable's variance σ_i^2 from the diagonal elements of the covariance matrix for plotting. The plotted variables are the state's posterior prediction $\hat{x}_i^+$ along with the true value of the state plus minus three σ_i.

In the case of a system parameter, and when tuning the filter in laboratory conditions or using artificially generated data, its true value is constant and known. As seen in Figure 4-5 the 3-sigma bound is constantly decreasing but the parameter estimation always falls inside it. This should be the case for a well-tuned filter. Checking the errors in the state variable predictions in a real scenario is important as well, but harder to be done since the true value of the system's state is neither known nor constant so one can only retrieve it from simulations.

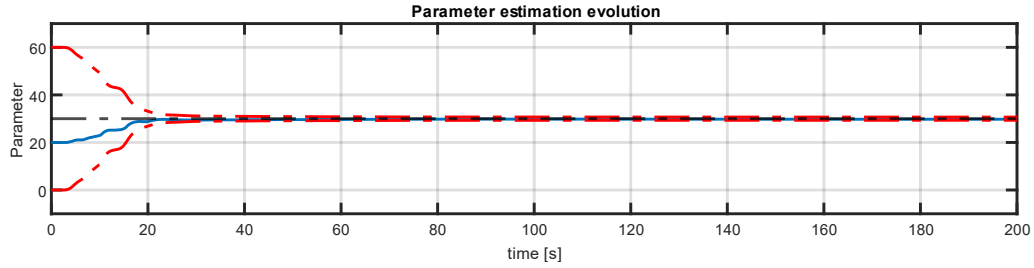


Figure 4-5. Typical convergence plot for a parameter when the joint - UKF algorithm is used.

One more important consideration is looking at the steady state estimation error distribution after convergence. Apart from being within the limits of the 3-sigma rule, it should have a zero mean in order for the estimator to be unbiased and resemble a gaussian distribution. These two conditions are hardly ever met even in ideal scenarios where the estimator is fed with noisy simulation data, much more in real experimental ones. Figure 4-6 contains the absolute error corresponding to Figure 4-5 with an appropriate zoom. One can see the error never converges to zero possibly due to the non-Gaussian errors contained in the process noise described in Chapter 2-5. This could be a potential disadvantage of the joint UKF approach, however not much can be done about it. It is worth noting that for this particular case the true value of the parameter seen in Figure 4-5 is 30 and the "steady state" absolute error is about 0.2, so the relative accuracy of the prediction is about 0.7%, which is considered rather good given the presence of noise.

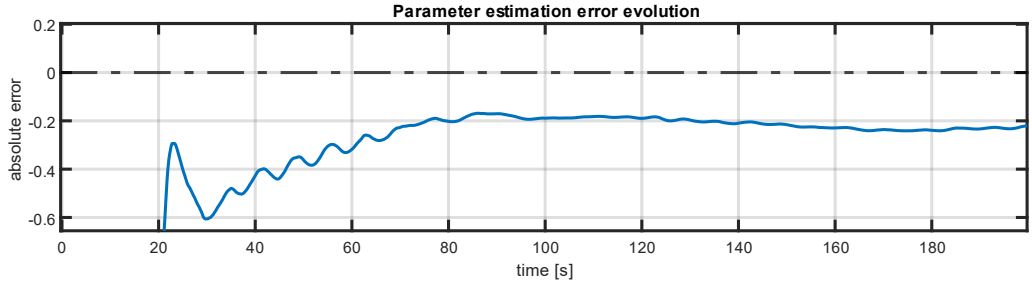
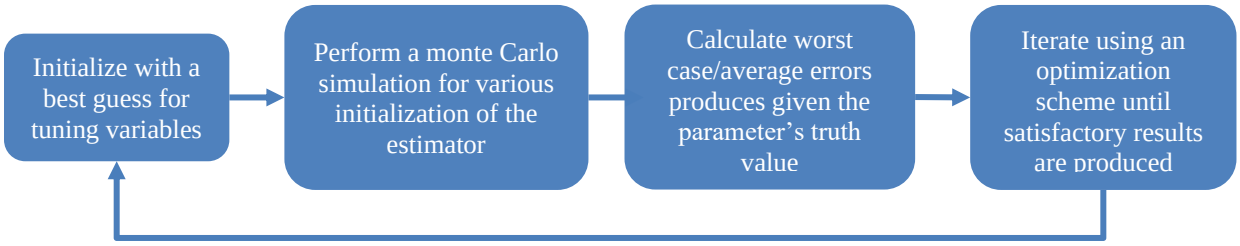


Figure 4-6. Absolute estimation error of Figure 4-5.

4.5.1 Automated Methods

As already mentioned, automated methods exist for Kalman Filter tuning. These could be distinguished into online and offline methods. Online methods run in parallel with the filter and adapt its parameters

to the possibly time varying statistical parameters of the real plant. Offline approaches are used to find the filter parameters in order for the filter to perform best under a given set of conditions. In this thesis, only offline methods will be considered for tuning. These usually work by defining a cost function that represents the filter's consistency and then maximizing it using classic optimization methods. One difficulty in this approach is that the performance of the UKF is stochastic since it is dependent on the RNG seed that represents noise and other parameters like the initial error in the state estimates. In order to accurately calculate its performance for a given set of tunings averaging multiple simulations is required rather than a single experiment. This highly increases the computational cost of this procedure making the use of surrogate model optimization techniques more appropriate. The optimization procedure can be summarized as follows:



The algorithm used for solving the optimization problem is the Matlab's surrogate optimization algorithm as implemented in *surrogateopt()*. Surrogate optimization is particularly efficient for optimization problems where the objective function calculation is time consuming and classic gradient-based methods or genetic algorithms would be too time consuming to implement. The algorithm works by initially searching the optimization variables domain at random. When the objective function has been calculated in enough random points it uses radial basis functions (RBFs) to interpolate the objective function and find promising search areas for continuing the search. The more points evaluated the better the accuracy of the predictions of the surrogate model become, finally converging to the local minima or maxima of the actual function.

4.6 Input Design

Finally, one last parameter affecting the performance of the estimation is the input design. In order for the estimation process to be accurate, especially in the presence of noise, the input applied should be strong enough to excite all system dynamics in a way that their effect in the measurable output is significantly larger than noise. Although systematic methods exist for designing such inputs, while trying to also minimize the total energy spent, they are usually hard to implement for UKF based algorithms. One way of doing this is computing the local observability matrix for the augmented system and design trajectories that maximize its rank condition, as proposed in [35]. In high dimensional nonlinear systems this can be particularly challenging to do in real-time and computing them offline might be unavoidable, but the results are superior mainly in the time needed for the parameters to converge and in the accuracy of the estimation.

In all models considered in this thesis the input, the input variables are the torques and forces excreted at the satellite base from the AOCS of the satellite. The profile of the input is chosen empirically, by trying a series of different signals and selecting the one that performs best. In most identification experiments a sequence of successive rectangular pulses in each input variable, like in Figure 4-7, seems to provide satisfactory results. Care is also taken to assure that the required input

would be within the limits of the AOCS of a typical satellite. For forces, this means that they should not exceed 20 N while for torques they should not exceed 10 Nm, if they are coming from thrusters, or 1 Nm if they are from reaction wheels. Among other input profiles tested, exciting all inputs simultaneously or using sinusoidal profiles resulted in worse estimates with the identified parameters converging to their true values with worse accuracy or not converging at all.

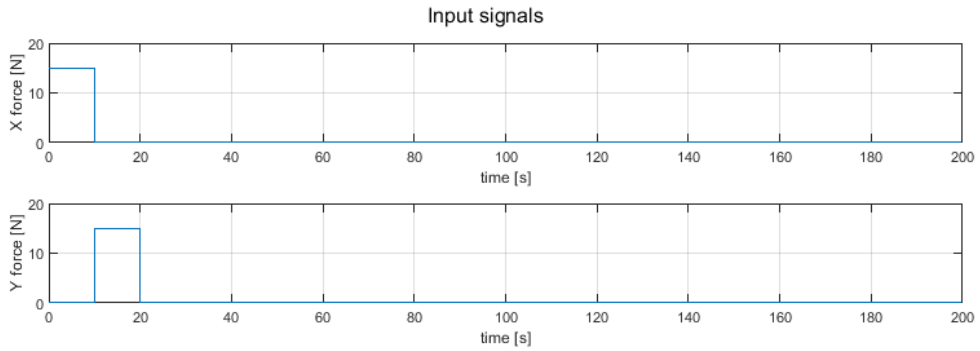


Figure 4-7. Typical input shape

An exception to the inputs described above is the case of the models containing fuel sloshing, where as previously mentioned can't be selected explicitly due to the limitations of CFD simulations. In such scenarios, the inputs calculated from the procedure described in Section 3.1.2 are used.

4.7 Parameter estimation case 1 – Sloshing tank

The first parameter estimation problem studied in this thesis is that of the 1D sloshing tank, as presented in Chapter 2.2, using data from CFD created in Section 3.1.2. The parameters contained in the dynamic model can be found in Table 4-1. Before proceeding to the presentation of the results based on the joint-UKF algorithm, the identification problem will be tackled with the numerical optimization methods presented in chapter 4.4.2. This serves two purposes. First, given the iterative nature of algorithm according to the distinction made earlier, along with an appropriate selection of an optimizer, an informal guarantee can be given about finding global best-fit model to the CFD data. This gives a sense of how well can the model can capture the actual effect, rather than how well the identification algorithm works. Furthermore, the global best-fit model parameters, can be used as a ground truth to evaluate the performance of the joint-UKF algorithm. Two different algorithms will be used for solving the numerical optimization problem, Matlab's *ga()* and *fminunc()*.

Table 4-1. Model parameters for 1D spacecraft with fuel sloshing.

Parameter Description	Symbol	Assumption	Units	Value
Fixed mass (spacecraft +non-sloshing fuel)	m_f	Unknown	Kg	-
Sloshing mass	m_s	Unknown	Kg	-
Sloshing spring constant	k	Unknown	N/m	-
Sloshing damping constant	b	Neglected	Ns/m	0

Proceeding with the results, the values presented in Table 4-2, are the results of two totally different optimization schemes that have converged to the same values, that being a strong indicator that this a

global minimum of the cost function. Also, focusing on Figure 4-8, because the identification results can't be evaluated on sight by looking at the velocity response of the container, which is the actual optimized response, its acceleration is provided as well, as predicted by the CFD model and the best-fit MSD model.

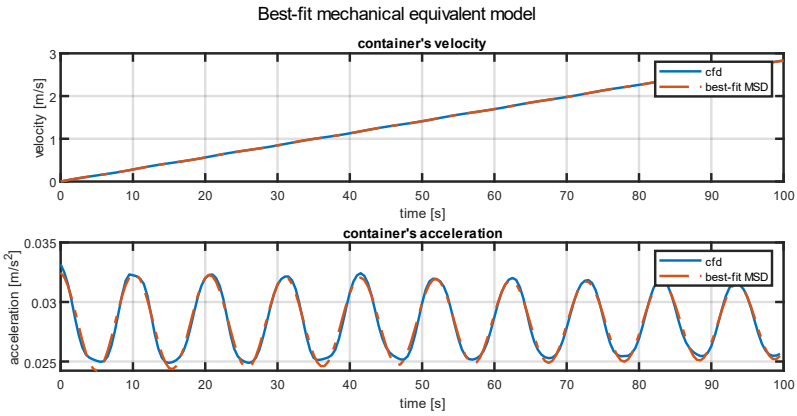


Figure 4-8. Results for 1D sloshing parameter estimation based on numerical methods

Table 4-2. Results for the parameter estimation of 1D sloshing tank using optimization methods used

Parameter	Units	Best-fit value (*fminunc*)	Best-fit value (*ga*)
m_f	Kg	922.3956	922.3768
m_s	Kg	136.2134	136.2326
k	N/m	43.1204	43.1257
b	Ns/m	0.6576	0.6583

As for the equations used in the joint-UKF algorithm they are the discretized version of the model (2-6) appended with a vector of parameters

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\pi}_{k+1}\end{bmatrix}=\begin{bmatrix}\left(\mathbf{x}_k+T_s f(\mathbf{x}_k,\mathbf{u}_k\right)\\ \boldsymbol{\pi}_k\end{bmatrix}$$
$$\mathbf{y}_k=\begin{bmatrix}1 & \mathbf{0}_{1\times 5}\end{bmatrix}\mathbf{x}_k \tag{4-21}$$

where $\boldsymbol{\pi}=\begin{bmatrix}m_f & k & m_s\end{bmatrix}^T$. The sampling time given the sloshing frequency is not very high is set to $T_s=0.01\,s$ in order to keep the first order approximate discretization errors small, while attenuating the high frequency noise.

Given that in this scenario the algorithm only receives one measurement and tries to estimate 3 state variables and 3 parameters, the UKF-based algorithm struggles. Also, looking at the first plot of Figure 4-8 one gest an idea of how small effect the sloshing phenomenon has on the velocity of the spacecraft, which resembles a rigid body motion. The results will be presented in detail though, along with possible solutions and suggestions for future work for increasing its performance. Finally, it is noted that the sloshing damping is neglected since it has not been appended to the sate vector as a parameter towards estimation. This is because the model is observed to have very small sensitivity to this parameter and the damping can be safely excluded from the analytic model contained in the UKF estimator without seriously affecting its performance. Appending it to the state would increase the

number of unknowns in the estimation and possibly make the results worse. Regarding the measurements fed to the UKF estimator, the integrated acceleration response of the container as calculated from the CFD with added white noise corresponding to the IMU characteristics were used. The *Unscented Kalman Filter* block will be used from the *Control System Toolbox*. Its exact parameters can be viewed in Table 4-3.

Table 4-3. UKF parameters for 1D sloshing parameter estimation.

Parameter	Symbol	Nominal noise level
Measurement Noise	R	$3.13 \cdot 10^{-4}$
Process Noise	$\mathbf{Q}$	$diag(31.25, 4.06, 9.06, 9.06, 2.19, 7.81) \cdot 10^{-6}$
Initial Covariance	$\mathbf{P}_0$	$diag(0.01, 0.01, 0.01, 30, 40, 200)$

The values of the for the measurement noise and process noise have been optimized to yield the best estimation results using the scheme presented in Section 4.5.1. To give a sense of the time required for this optimization problem to be solved, each call of the objective function contains 50 joint-UKF estimations and takes around 50 seconds to execute. In order for the surrogate optimization to yield accurate results, about 500 evaluations are needed, resulting in a total run time of about 7 hours in a desktop computer. The search domains for the diagonal elements of the parameters $\mathbf{Q}$ and R were $Q_{ii} \in \left[10^{-5}, 10^{-12}\right], R \in \left[10^{-8}, 10^{-2}\right]$. Given that the search domain ranges in multiple orders of magnitude a logarithmic transformation was used for faster searching, using $\log(Q_{ii}), \log(R)$ as optimization variables.

Given than the above simulation includes stochastic variables like the artificially generated white noise, one simulation is not enough for evaluating the performance of the algorithm. It is observed that changing even the random number generator seed that produces the white noise affects the final results. To this end, a 250 Monte Carlo simulation is conducted, initializing the filter randomly in the range of ±50% relative error with respect to the estimation variables, while changing the RNG seed as well.

Even using the optimized tunings described by the process above, the filter still doesn't perform particularly well. As depicted in Figure 4-9 showing the relative error histogram of the final estimation, the worst-case estimate reaches up to 40% while being in the vicinity of 10% for most simulations. This is partially expected though, given that the joint-UKF algorithm employed is typically used in scenarios where there are multiple measured quantities and few parameters for estimation. Here, we are trying to estimate all the model parameters and unmeasurable state variables just from a single measured quantity.

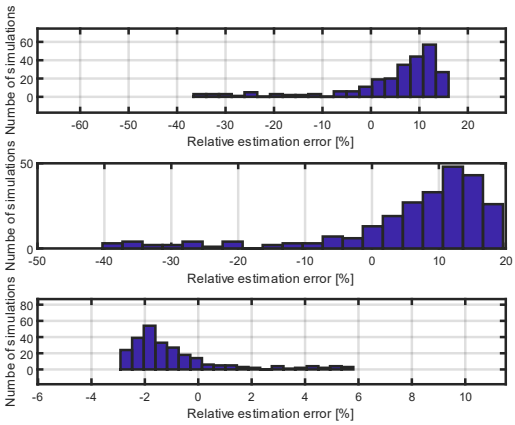


Figure 4-9. Monte Carlo simulation results for 1D fuel sloshing parameter estimation using joint-UKF.

4.8 Identification case 2 – Satellite with sloshing

4.8.1 Pendulum model with CFD data

Although the UKF based algorithm in Chapter 4.7 didn't yield satisfactory results for the 1D sloshing plant, this was mainly due to the small number of measurements compared to the large number of estimated quantities. In a 3D scenario, the measurement vector is expanded since 3 linear and 3 angular velocities can be measured instead of a single velocity measurement in the 1D case, while the sloshing plant retains the same number of parameters. Instead of using an MSD equivalent model though, the the pendulum model of (2-23) is tested here since it is found to be able to predict the CFD results better. The parameters in the analytic model are the pendulum length as well as the fixed and sloshing fuel masses. The inertial parameters of the spacecraft and the position of the tank are considered known. The complete parameters entering the system model equations can be found in Table 4-4.

Table 4-4. Model parameters for spacecraft with sloshing using a pendulum model.

Parameter Description	Symbol	Assumption	Units	Value
Satellite rigid mass	m	Known	Kg	800
Satellite inertia around y axis	$\mathbf{I}_b$	Known	Kg m^2	$600\mathbf{I}_3$
Sloshing tank center position	$\mathbf{p}_0$	Known	m	[0 0 0.2]
Sloshing mass	m_s	Unknown	Kg	-
Body fixed mass	m_f	Unknown	Kg	-
Pendulum length	l_p	Unknown	m	-
Sloshing damping	b	Neglected	Ns/m	0

As for the equations used in the joint-UKF algorithm they are the discretized state-space version of model in (2-23) appended with a vector of parameters $\boldsymbol{\pi} = \begin{bmatrix} m_s & l_p & m_s \end{bmatrix}^T$. The sampling is set to $T_s = 0.01$ seconds for the same reasons as in the previous case.

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\pi}_{k+1}\end{bmatrix}=\begin{bmatrix}\left(\mathbf{x}_k+T_s f\left(\mathbf{x}_k,\mathbf{u}_k\right)\right)\\ \boldsymbol{\pi}_k\end{bmatrix}$$
$$\mathbf{y}_k=\begin{bmatrix}\mathbf{I}_{6\times 6} & \mathbf{0}_{7\times 6}\end{bmatrix}\mathbf{x}_k \tag{4-22}$$

In order to evaluate the performance of the joint-UKF algorithm as well as tuning it correctly, the best-fit sloshing parameters need to be calculated first in the same sense as in Chapter 4.6. These are calculated using the numerical optimization methods presented in Section 4.4.2. Because these process the data as a hole rather than sequentially they are more accurate and reliable than the UKF algorithm in the expense of computational load as discussed earlier. The best estimates are used as ground truth for plotting the 3 sigma bounds used for tuning as well as for calculating the final estimation error. Using Matlab's *fmincon()* and minimizing the squared difference of the CFD and analytical response for this system one gets.

Table 4-5. Best-fit parameters for the pendulum sloshing subsystem calculated from a CFD response.

Parameter	Best-fit value
m_s	151.1724
m_f	108.7225
l_p	0.3122
b	2.5910

It is worth noting that the best fit pendulum model captures the CFD data reasonably well, as can be seen in Figure 4-10. The success of the UKF algorithm highly depends on that since as discussed earlier Kalman Filters are generally not tolerant to unmodelled dynamics. The fact that the lumped parameter system has the ability to capture the phenomenon for some value of its parameters is therefore crucial. Also note that as in the 1D case the damping constant is neglected.

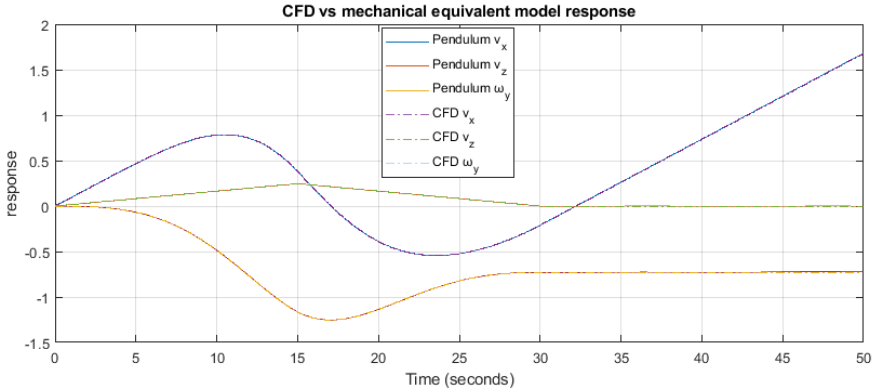


Figure 4-10. CFD vs mechanical equivalent model response.

Proceeding with presenting the results for the joint-UKF algorithm, the input-output data from the CFD are first corrupted by noise according to Chapter 2.5. The filter is initialized with a relative error within a ±70% range for the best fit estimates parameters. The results are based on a 250 simulation Monte Carlo analysis in order to make sure the filter performs well under all possible initializations. The results will be presented for actual CFD data, as well as for the case where the estimator is fed with noisy data coming the analytic equations of the plant. This way, we can see how the performance of the algorithm is affected from the model mismatch between the CFD and the model response. The results

based on data from the analytic equations serve the purpose of a providing an upper limit of how the filter could perform in the ideal case where the analytic model could describe the phenomenon perfectly. In both cases the estimator is tuned with the same parameters. Those are

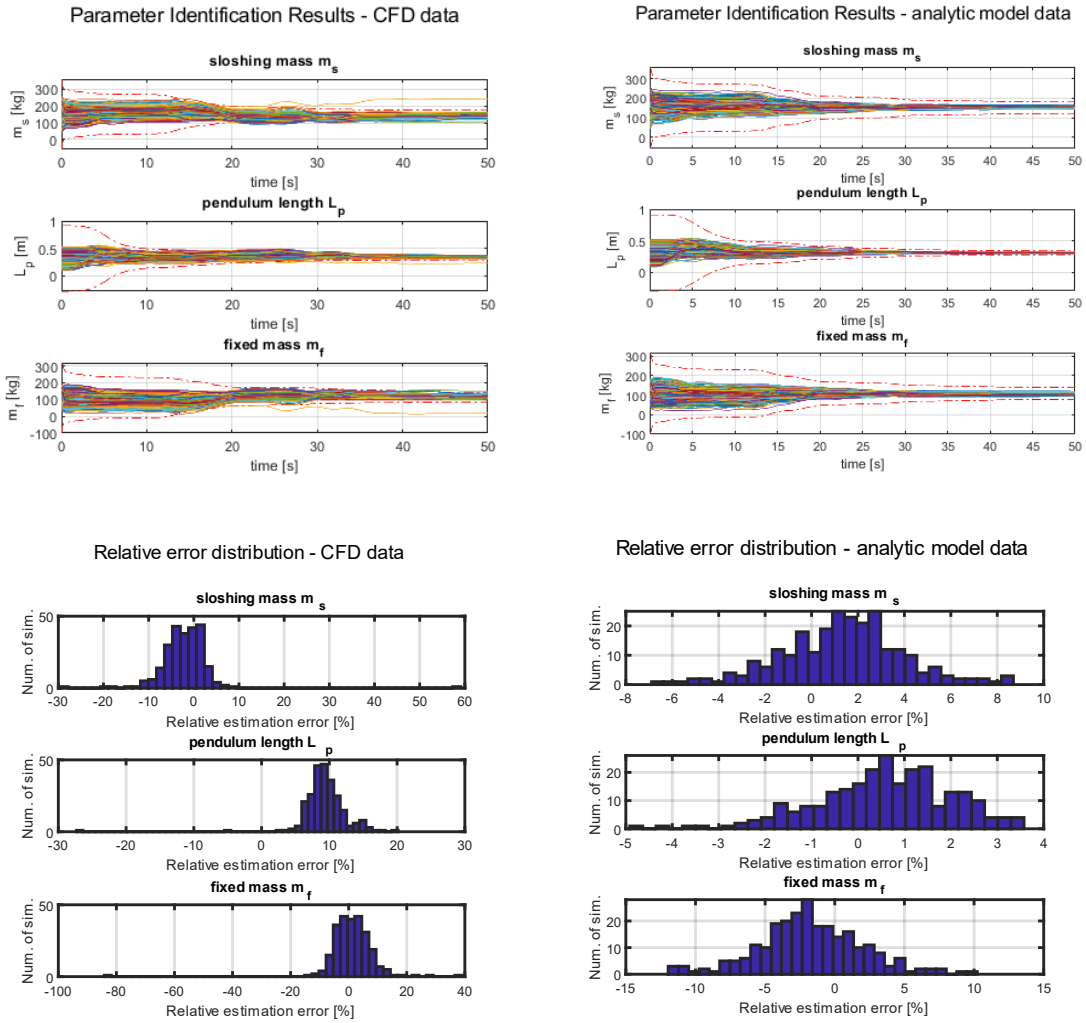
Table 4-6. Joint-UKF Parameters for pendulum sloshing model.

Parameter	Symbol	Case 1
Measurement Noise	R	$\begin{bmatrix} 10^{-5} I_{3x3} & 0 \\ 0 & 10^{-4} I_{3x3} \end{bmatrix}$
Process Noise	Q	$10^{-9} I_{8x8}$
Initial Covariance	P_0	$\begin{bmatrix} 0.01 I_{10x10} & 0 & 0 & 0 \\ 0 & 70 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 70 \end{bmatrix}$

The results are presented into two forms. On Figure 4-11 one can see the time domain diagrams of the evolution of the parameter estimation as well as the histograms of the final relative error. The 3-sigma bounds of the estimations are plotted as well (for only the first of the 250 runs) in order to verify that all estimations fall inside it, as described in the tuning section. The histograms on the other hand are useful for evaluating the worst-case estimation error the filter might produce. To this end, Table 4-7 is constructed. Looking at the results, one can see that in estimation based on the CFD data the errors are quite large, while in the estimation from the analytic model data they are in acceptable limits. This illustrates the main disadvantage of the method: it's very high dependance in the knowledge of model's structure. Furthermore, looking closely at the estimation from CFD data one can even see that in a couple of the 250 runs the simulation has not even converged to the correct region of values of the parameter. This can be depicted in the extremes of the error histograms as well as the time evolution of the prediction.

Table 4-7. Worst-case relative estimation error of pendulum sloshing subsystem from CFD data using joint-UKF.

Parameter	Relative error CFD data	Relative error analytic model data
m_s	12%	8%
m_f	15%	5%
l_p	15%	10%



4.8.2 MSD model - 3D motion

Apart from the pendulum model the MSD model of (2-19) will be tested as well. The joint UKF approach will be considered for the estimation. The overall scenario is similar with the previous case. The complete set of parameters in the dynamic equations in can be viewed in Table.

Table 4-8. Model parameters for a spacecraft with an MSD sloshing model.

Parameter Description	Symbol	Assumption	Units	Value
Satellite total mass	m	Known	Kg	800
Satellite Inertia Tensor	$\mathbf{I}_b$	Known	Kg m^2	$600I_{3\times3}$
Sloshing tank center position with respect to satellite CM	$\mathbf{p}_0$	Known	m	[0;0;0.2]
Sloshing mass	m_s	Unknown	Kg	40
Sloshing mass	m_f	Unknown	Kg	40
Sloshing stiffness	k_x, k_y	Unknown	N/m	30
Sloshing damping (common in all axes)	b	Neglected	Ns/m	0.5

The main difference with the pendulum model presented earlier is that the MSD mechanical equivalent is not found to be able to predict the CFD data as well. As with the pendulum model, the MSD model response was optimized to match the data from the CFD. As seen in Figure 4-12 the main difference lies in the failure to predict the rotational velocity. The problem arises from the fact that the pendulum model, due to the rotational joint transfers only a very small torque to the spacecraft though the attachment point, only due to the damping coefficient. This is in accordance with the true plant, which in general doesn't transmit torques either[4]. On the MSD model however because the mass is constrained to move on the plane, torques are generated when the sloshing mass is off the equilibrium point.

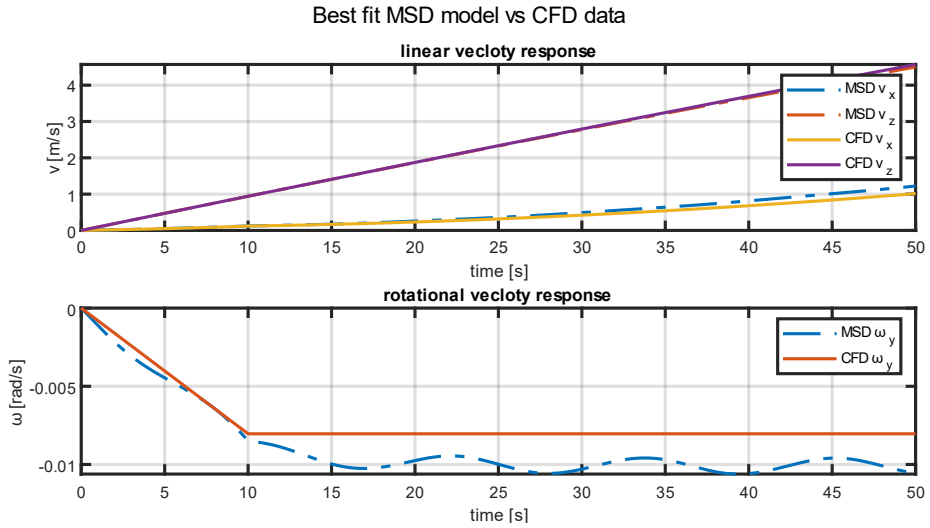


Figure 4-12. Model mismatch between the MSD model and the CFD data

This structural difference with the actual plant causes the MSD best fit model to "significantly" disagree with the CFD data, at least in the sense of what the UKF based algorithm can tolerate. To this end, this case won't be studied using data from CFD but rather from a simscape model, using practically the same equations as the analytic model running in the Kalman Filter. This only serves the purpose to check if the structure of the model allows for identification based on this technique, without addressing the problem of model mismatch between the actual and the mechanical equivalent plants.

[4] In an actual sloshing container, forces are transferred to from the liquid to the walls though pressure and shear stresses. The shear stress force is in generally much smaller that pressure related force. For a given point in a spherical tank, the direction of the pressure related forces, being vertical to the local surface, pass from the tanks center resulting in zero moments with respect to it. The moments from shear stresses are related to damping.

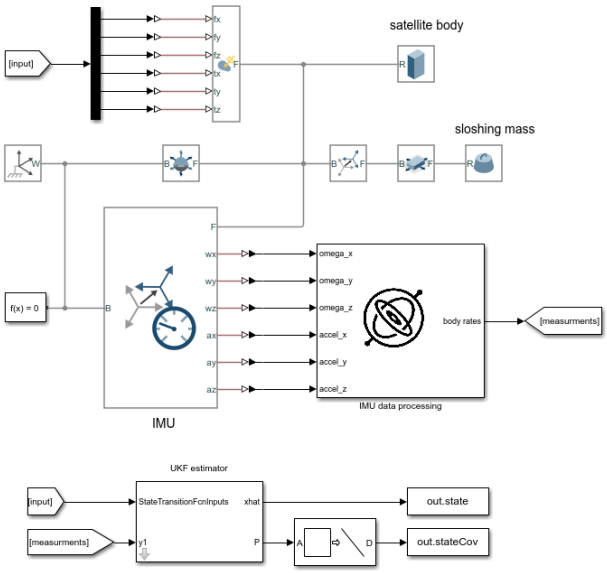


Figure 4-13. Simscape model for simulating the Spacecraft's under in the presence of sloshing.

To this end, the equations presented in (2-19) are brought into a discrete-time state space form. They are also augmented by a vector of parameters. The final system is

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\pi}_{k+1}\end{bmatrix}=\begin{bmatrix}\left(\mathbf{x}_k+T_s f\left(\mathbf{x}_k,\mathbf{u}_k\right)\right)\\ \boldsymbol{\pi}_k\end{bmatrix} \qquad (4\text{-}23)$$
$$\mathbf{y}_k=\left[\mathbf{I}_{6\times 6}\ \ \mathbf{0}_{8\times 6}\right]\mathbf{x}_k$$

Where $\boldsymbol{\pi}=\left[m_s\ \ k_x\ \ k_y\ \ m\right]$. The sampling frequency is the same as with the other experiments. The results will be presented for two levels of noise as described in chapter 2.5, the nominal and a higher one. Regarding the inputs, the axial thrust force is considered to be 100N while the lateral force is two consecutive pulses as shown in Figure 4-7 in the X and Y direction only. No moment input is used. The UKF filter settings used were:

Parameter	Symbol	Case 1	Case 2
Measurement Noise	R	$\begin{bmatrix}10^{-4}\mathbf{I}_{3x3} & 0\\ 0 & 10^{-2}\mathbf{I}_{3x3}\end{bmatrix}$	$\begin{bmatrix}25\cdot10^{-4}\mathbf{I}_{3x3} & 0\\ 0 & 25\cdot10^{-2}\mathbf{I}_{3x3}\end{bmatrix}$
Process Noise	Q	$3\cdot10^{-10}\mathbf{I}_{14x14}$	$75\cdot10^{-10}\mathbf{I}_{14x14}$
Initial Covariance	P_0	$\begin{bmatrix}0.1\mathbf{I}_{10x10} & 0 & 0 & 0 & 0\\ 0 & 50^2 & 0 & 0 & 0\\ 0 & 0 & 10^2 & 0 & 0\\ 0 & 0 & 0 & 10^2 & 0\\ 0 & 0 & 0 & 0 & 200^2\end{bmatrix}$	$\begin{bmatrix}0.1\mathbf{I}_{10x10} & 0 & 0 & 0 & 0\\ 0 & 50^2 & 0 & 0 & 0\\ 0 & 0 & 10^2 & 0 & 0\\ 0 & 0 & 0 & 10^2 & 0\\ 0 & 0 & 0 & 0 & 200^2\end{bmatrix}$

The above parameters conclude all preliminaries required for the analysis and one can proceed with presenting the results. As seen in Figure 4-14 indeed the parameters converge to their true values very well as expected, since no modelling inaccuracies are present. As with the previous cases, a Monte Carlo simulation is performed for both noise levels, consisting of 250 runs in which the initial values

of the estimated variables are picked from a constant distribution in a ± 70% relative error with respect to their ground truth value. The RNG seed is also changed randomly in each run.

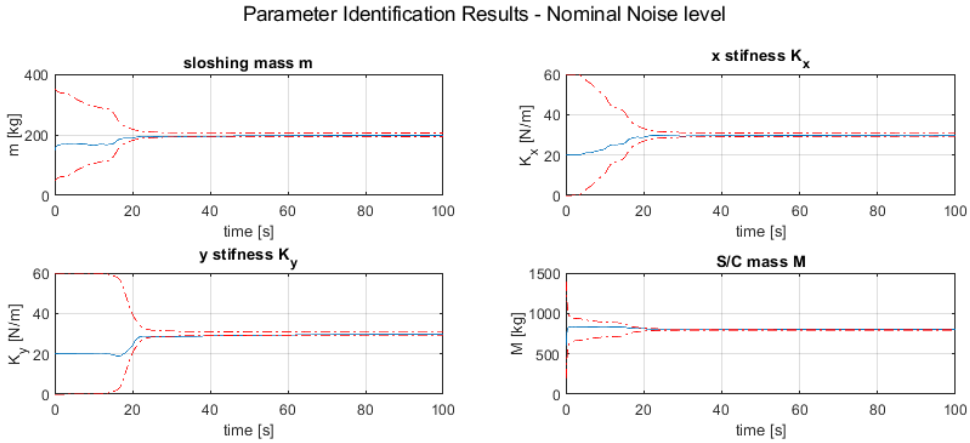


Figure 4-14. Convergence of parameters for spacecraft with MSD sloshing model.

Looking at the histograms of the relative error in each variable in Figures 4-9 and 4-10, one can deduce several conclusions. The most obvious one is that none of the distributions presented are centered at zero. As mentioned previously, it is rarely the case that a joint configuration UKF algorithm will have zero bias even under the assumption of no modelling inaccuracies. Certainly, the biased velocity measurements resulting from the integrated IMU data worsen this effect. Finally, one can estimate the worst-case relative accuracy of the estimator in both noise levels by looking at the histogram graphs.

Parameter	Relative error Nominal noise	Relative error High noise
m_s	7%	15%
k_x	5%	10%
k_x	5%	10%
m	1.5%	4%

Table 4-9. 3D sloshing parameter estimation results depicted as worst-case errors

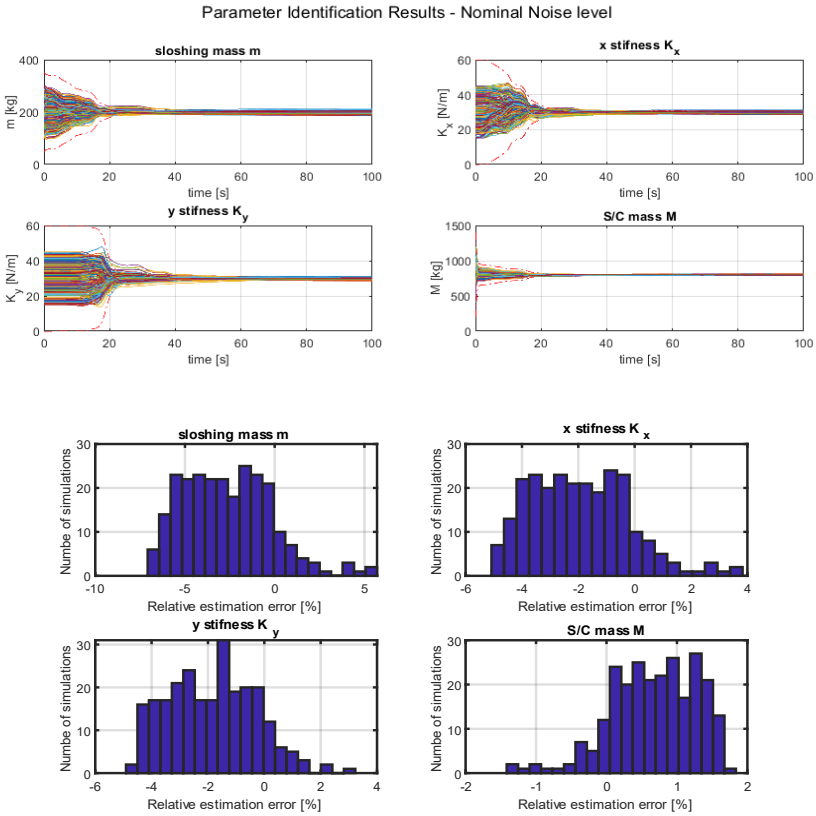


Table 4-10. 3D sloshing parameter estimation results based on MSD model – Nominal Noise levels

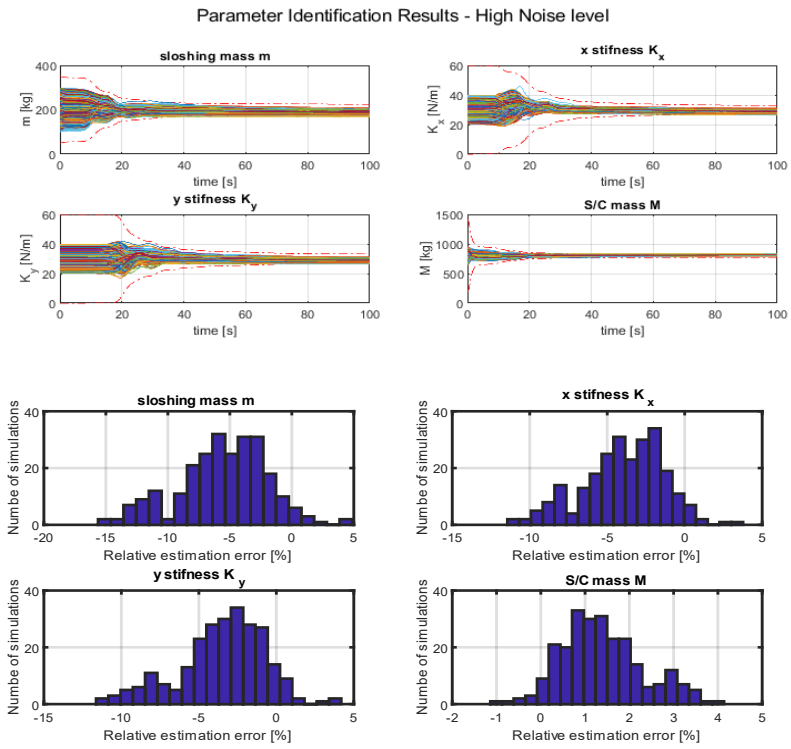


Table 4-11. 3D sloshing parameter estimation results based on MSD model – High Noise levels

4.9 Identification case 3 – Satellite with flexible appendage

Apart from the case of satellite with sloshing, the scenario of a spacecraft with a flexible appendage will be studied as well, given its wide used in modern satellites. The joint approach will be used here too. The specific model used consists of a rigid spacecraft with two adjacent solar arrays modelled as Euler Bernoulli beams. Given the material properties and the size of each array, all their properties can be calculated analytically. The motive of using this time domain approach in the case flexible panels, is that even if the two panels are identical, their contribution in the dynamic equations differs because they are attached to different points and with different orientations allowing for the UKF based algorithm to estimate the flexible parameters of one separately. This would not be possible in a frequency domain-based method, since it would be hard, if not impossible, to match an identified natural frequency with its corresponding appendage. To stress the value of the approach the two panels are considered identical in all properties. The complete set of parameters contained in the dynamic model equations can be found in Table 4-12.

Table 4-12. Model parameters for spacecraft with 2 flexible solar arrays.

Parameter Description	Symbol	Assumption	Value	Units
Spacecraft mass	m	Known	800	kg
Spacecraft Inertia Tensor	$\mathbf{I}_b$	Known	$600\mathbf{I}_{3\times3}$	kg m^2
Panel 1 attachment point	p_{01}	Known	[0;0;0.5]	m
Panel 2 attachment point	p_{01}	Known	[0;0;-0.5]	m
Material's density	ρ	Known/Unknown	3000	kg/m^3
Panel 1,2 width	w_1, w_2	Known	1	m
Panel 1,2 thickness	h_1, h_2	Known	0.02	m
Panel 1,2 length	L_1, L_2	Known	4	m
Panel 1,2 flexural stiffness	Σ_1, Σ_2	Unknown	1333	Pa m^4
Panel's modal damping	-	Neglected	0	s

Compared to the fuel sloshing cases, one needs to carefully consider which variables need to be estimated and which are derived. The desired tradeoff is to give the algorithm enough "degrees of freedom" to be able to adjust the analytic model to fit the data but not include variables that are enter the equations in the same manner and therefore destroy the observability of the joint system. Considering Table 4-12, the parameters can be split into two categories. The dimensional characteristics of the panel, and the dynamic ones. The dimensional characteristics can be easily measured on earth. The dynamic properties of the array on the other hand, consisting of its flexural rigidity and material density are harder to estimate since simulating the zero g environment in terrestrial laboratories is an extremely challenging task. To this end, two experiments will be conducted only estimating the panel's dynamic properties. In the first, only each panel's flexural rigidity will be estimated, considering the material's density known. In the second, the material's density will be estimated as well. Note that according to the Euler Bernoulli hypothesis, estimating all the natural frequencies of a beam separately is not possible. Once the parameters Σ, ρ and the cross-sectional area of the beam S, or equivalently the first natural frequency, are selected the rest natural frequencies are derived according to

$$\omega_i = \beta_i \sqrt{\frac{\Sigma}{\rho S}} \tag{4-24}$$

In order to ensure the model's potential to capture the response of the actual plant, one has to include enough flexible modes in the equations of motion in all directions prone to oscillations. In this study, only two modes are added in y-bending, given that with the above dimensions and parameters of the system, all other modes had only a minor influence on the system dynamics and in the case of a control system model they would most likely be treated as white noise, rather than modelled analytically. Finally, note that in contrast with the sloshing scenarios presented earlier, the estimation algorithm is fed with data from simscape rather that a more sophisticated simulation environment. As briefly mentioned in Chapter 2-5, Simscape also uses the Euler-Bernoulli simplification for modelling flexible members. The testing of the algorithm against actual experimental data, or data from a completely coupled FEA system is suggested as future work. For small deflections of the panel however, the Euler-Bernoulli assumption is expected to work reasonably well, especially for the y-wise bending that is the main focus here.

4.9.1 Identification of flexural stiffness only

First, the case of known panel densities will be presented. The parameters being augmented in the state vector in this case are only $\boldsymbol{\pi} = \begin{bmatrix} \Sigma_1 & \Sigma_2 \end{bmatrix}^T$. The sampling time is reduced to $T_s = 0.001$ seconds considering the higher natural frequency of the vibrating panels. As with the sloshing case, the results will be presented for the two levels of noise of the end of 2.5.

The UKF filter settings used are shown in Table 4-13.

Table 4-13. joint-UKF parameters for spacecraft with flexible solar arrays.

Parameter	Symbol	Nominal noise level	High noise level
Measurement Noise	**R**	$\begin{bmatrix} 10^{-5}\mathbf{I}_{3x3} & 0 \\ 0 & 10^{-4}\mathbf{I}_{3x3} \end{bmatrix}$	$\begin{bmatrix} 25\cdot10^{-5}\mathbf{I}_{3x3} & 0 \\ 0 & 25\cdot10^{-4}\mathbf{I}_{3x3} \end{bmatrix}$
Process Noise	**Q**	$10^{-12} I_{16x16}$	$10^{-12} I_{16x16}$
Initial Covariance	$\mathbf{P}_0$	$\begin{bmatrix} 10^{-2}\mathbf{I}_{10x10} & 0 & 0 \\ 0 & 500 & 0 \\ 0 & 0 & 500 \end{bmatrix}$	$\begin{bmatrix} 0.1\mathbf{I}_{10x10} & 0 & 0 \\ 0 & 500 & 0 \\ 0 & 0 & 500 \end{bmatrix}$

The estimation results, shown in Figures 4-15 and 1-16 for the two noise levels are presented in a similar fashion to the previous cases. The worst-case estimation errors can be found in Table 4-14.

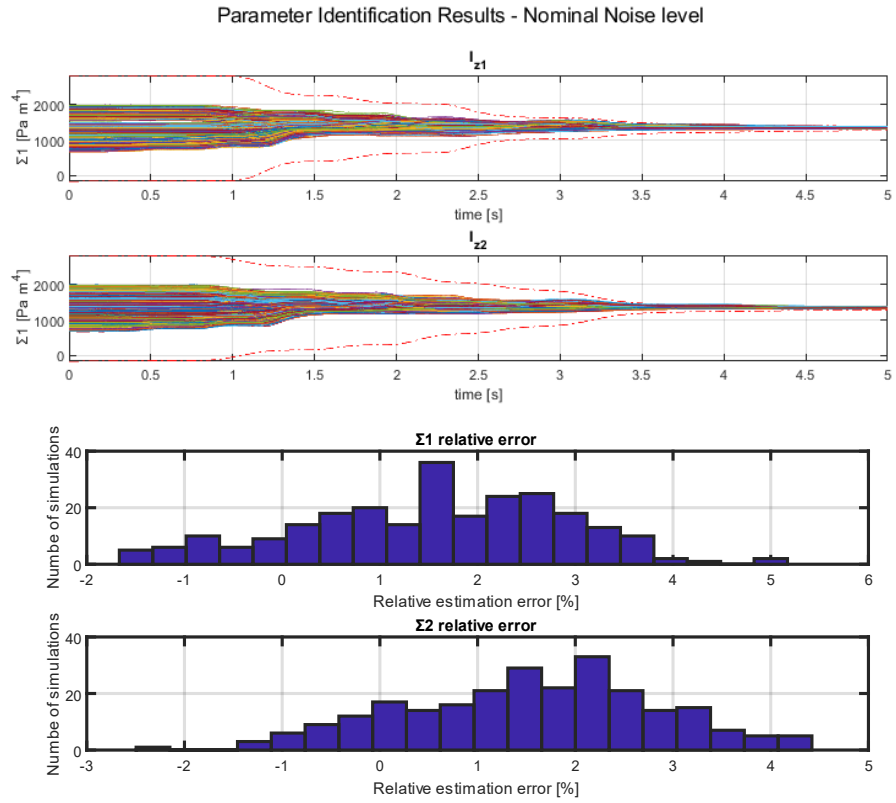


Figure 4-15. Flexural rigidity estimation of solar panels - Nominal Noise

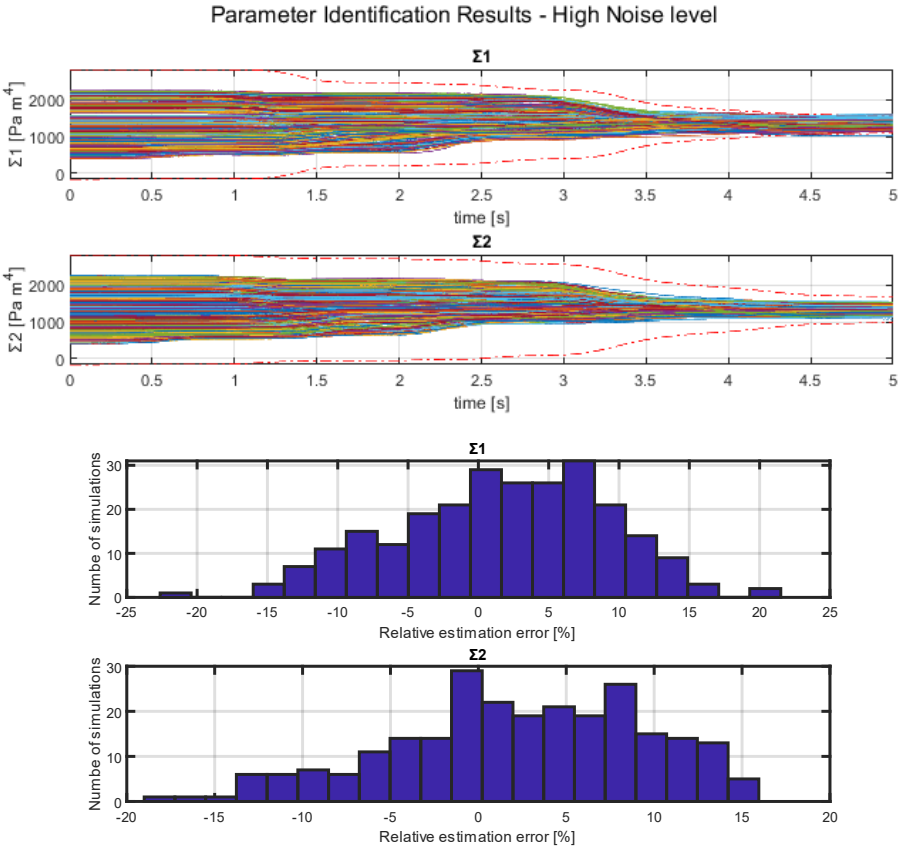


Figure 4-16. Flexural rigidity estimation of solar panels - High Noise

Table 4-14. Worst-case relative estimation error of Flexural rigidity of the solar panels.

Parameter	Relative error Nominal noise	Relative error High noise
Σ_1	4%	15%
Σ_2	4%	15%

4.9.2 Identification of flexural stiffness and density

This scenario concerns the case of identifying both the material's density and the flexural stiffness of each solar array. Based on the empirical rule discussed earlier regarding the joint-UKF algorithm, the more variables are appended to the state the worse the identification accuracy gets, the results are expected to be somewhat worse here. The case of both unknown material density and flexural stiffness might be of interest though because it allows more degrees of freedom for the identification algorithm to match the experimental data and might be useful for fault detection or estimating cracks. The estimation process behaves reasonably well under nominal noise conditions, but produces large errors in the increased noise scenario. To this end, only the nominal noise case is presented.

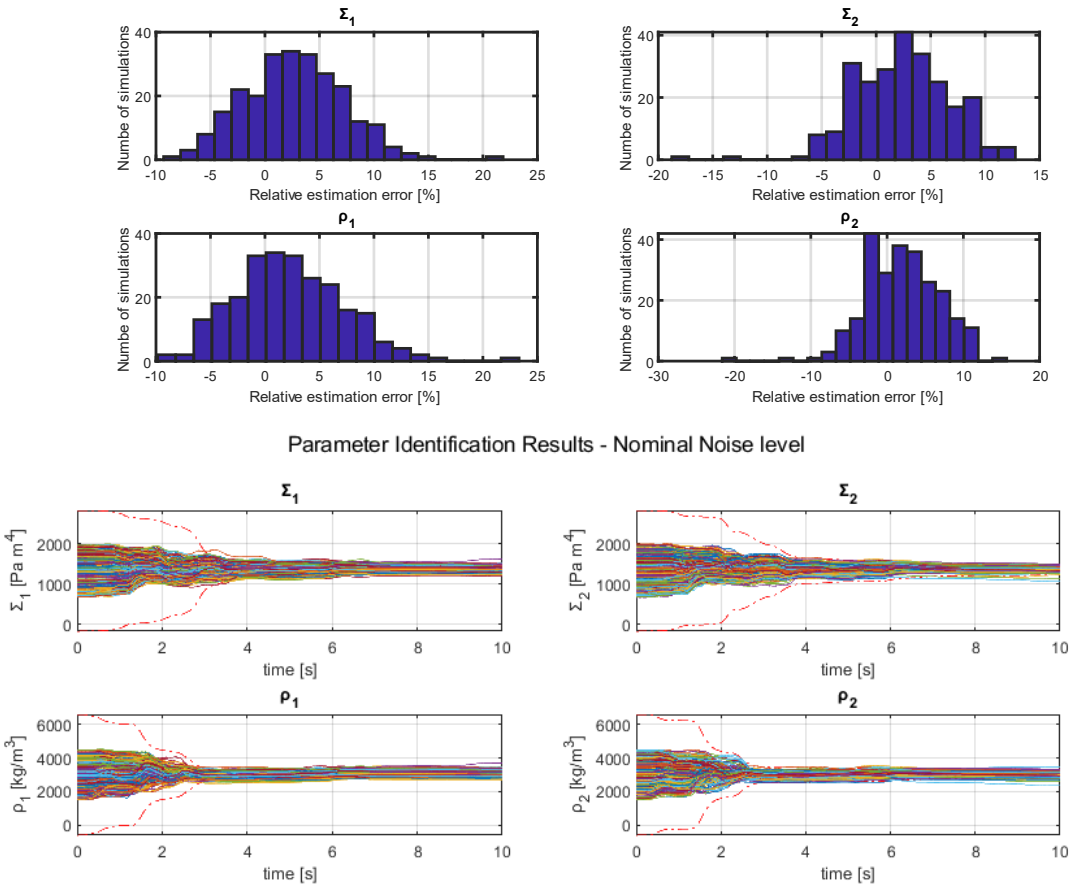


Table 4-15. Worst-case estimation errors of flexural rigidity and density of flexible solar arrays.

Parameter	Relative error Nominal noise
Σ_1	15%
Σ_2	15%
ρ_1	15%
ρ_2	15%

Possible room for improvement might exist if batch tuning methods are employed in order to optimize the process noise and initial covariance matrix to have possibly non diagonal matrices, or at least diagonal matrices with different elements for each state variable. The computational cost of this procedure though renders it infeasibly to be implemented in a local computer in the context of this thesis, given that each Monte Carlo simulation needed to estimate the performance of a give set of tunings takes at least 30 minutes to run. Considering that an optimization scheme would require hundreds of iterations, this option could be further explored only by utilizing cluster computing techniques.

5 Outcomes and future work

5.1 Conclusions

To conclude this work, the outcomes are summarized. The main focus of this thesis was the concurrent state and parameter estimation of non-linear dynamic systems using mainly the Unscented Kalman Filter in a joint state-parameter configuration. This method was tested in the cases of a satellite with fuel sloshing represented using both a pendulum and a mass spring damper equivalent model, and a satellite with two flexible solar arrays modelled as Euler-Bernoulli beams using the assumed mode method.

In the case of sloshing, the UKF-based algorithm is shown to be capable of estimating the dynamics of the system, in 3D motion scenarios but seems to be fundamentally lacking in the 1D motion scenario, giving poor estimates under noise, because of the very small ratio of measured to estimated number of parameters and states. To this end, in the 1D motion a more powerful offline method is used. Apart from the question of whether the UKF estimator is applicable it this particular system as far as the structure of the dynamic equations, i.e. observability of the system, the more pertinent issue of the model mismatch between the mechanical equivalent models and the actual plant is studied using CFD. It is observed that in 1D motion scenarios the mass spring damper can faithfully represent sloshing. In more general 3D motions however there are small differences in the dynamic response of this model with the CFD data. The pendulum model is found to be more potent to represent sloshing in a satellite systems in a high-g scenario, a result that can be confirmed in the literature as well. Using the CFD simulation, a pair of input-output data was generated to test the UKF algorithm with regards to its ability to handle modelling errors existing because of this model mismatch. In the case of the pendulum model, the algorithm is shown to have a reasonable performance being sensitive however to the initialization of the sloshing model parameters. In the mass spring damper model, the error produced by the UKF algorithms were larger given that the model showed small inconsistencies with respect to the CFD data even using the best-fit parameters calculated by the more powerful numerical methods presented.

In the case of a satellite with flexible solar panels, the UKF estimator worked accurately. An advantage of the time domain methods used in this work to evaluate the flexible parameters and by extension the natural frequencies of each array from only data gathered form the IMU of the satellite have the advantage of being able correctly identify the parameters of multiple arrays, even when these are similar or even identical. This is in contradiction with a frequency domain method where it would be hard to match the eigenfrequency captured in the spectrum of the output signal with its corresponding array. The parameter estimation scenario that performed best was that where the panel's mass, and therefore density is known, and only the flexural stiffness must be computed. A scenario where the array's material density is also estimated is tested and found to have reduced performance because of the greater ratio of estimated to measured quantities. Finally, it is noted that the modelling assumption of the Euler-Bernoulli equations in each bending direction is not tested here since the model developed in Matlab's Simscape that is used for data generation is also based on Euler-Bernoulli beam finite elements. The only modelling inconsistencies that are tested are those of the contribution of higher frequency modes and higher order terms in the dynamic equations, while also the assumption of the clamped end and the analytic mode shapes that were used in the dynamic equation derivation.

5.2 Recommendations for future work

As usually in research, there is always room for improvement, and this thesis is no exception. To this end, three main areas are recommended for further investigation. The first is with regards to the joint-UKF estimator algorithm, which was the main tool used throughout this thesis for online parameter estimation in non-linear systems with unmeasurable state variables. Although this tool has an elegant formulation, as with all Kalman Filter methods it is very sensitive to modelling errors such as unmodelled dynamics or model mismatch as in the case of sloshing. This is indeed a very important drawback since in most realistic scenarios these issues exist to different degrees, smaller like in the case of flexible solar arrays, or more serious like in the case of sloshing. An effective parameter estimation algorithm should be tolerant to these issues, being still able to provide estimates close to the global best-fit, with the ultimate goal to maximize controller performance. To this end, especially for non-linear systems with unmeasurable state variables, two methods could be investigated. The first could be to replace the Unscented Kalman Filter with Particle Filters, eliminating the Gaussian assumption in the statistics of the random variables representing the state and parameters. This will certainly increase performance, of course with in the expense of computational power required, and possibly robustness in the sense of the model mismatch errors discussed above. A second solution might be to further develop the numerical optimization method presented in [25] to decrease their computational cost while feed them with data from a small time window, rather that the hole experiment at once.

One more very promising area of research would be unlocked by the ability to perform coupled CFD simulations with the dynamics of a general dynamic system, ranging from the closed loop equations of a satellite with controllers and stochastic feedback to open-loop models including the dynamics of various appendages like solar arrays or manipulators. While it is true that under free floating conditions the sloshing phenomenon is extremely hard to reproduce with simple mechanical equivalent models, it might be the case that it can be safely ignored if proper, robust control algorithms are employed. To this end, a completely coupled CFD simulator, something that to the authors knowledge is almost non-existent among published papers, would allow to study this issue. Of course, this assumes that CFD is indeed an accurate tool to describe sloshing in zero-g, something that few CFD solvers are actually validated to be capable of doing so.

6 References

[1] Barber, J. 'Kalman filtering techniques for parameter estimation', p. 49.

[2] Bayle, O. *et al.* (2002) 'Influence of the ATV Propellant Sloshing on the GNC Performance', in *AIAA Guidance, Navigation, and Control Conference and Exhibit*. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Monterey, California: American Institute of Aeronautics and Astronautics. doi:10.2514/6.2002-4845.

[3] Bourdelle, A. *et al.* 'Modeling and Control of Propellant Slosh Dynamics in Observation Spacecraft with Actuators Saturations', p. 16.

[4] Bourdelle, A. *et al.* (2019) 'Towards New Controller Design Oriented Models of Propellant Sloshing in Observation Spacecraft', in *AIAA Scitech 2019 Forum*. *AIAA Scitech 2019 Forum*, San Diego, California: American Institute of Aeronautics and Astronautics. doi:10.2514/6.2019-0115.

[5] Buseck, R. and Benaroya, H. (1993) 'MECHANICAL MODELS FOR SLOSH OF LIQUID FUEL', in *Aerospace Design Conference*. *Aerospace Design Conference*, Irvine,CA,U.S.A.: American Institute of Aeronautics and Astronautics. doi:10.2514/6.1993-1093.

[6] Chen, T., Morris, J. and Martin, E. (2005) 'Particle filters for state and parameter estimation in batch processes', *Journal of Process Control*, 15(6), pp. 665–673. doi:10.1016/j.jprocont.2005.01.001.

[7] De Luca, A. and Siciliano, B. (1991) 'Closed-form dynamic model of planar multilink lightweight robots', *IEEE Transactions on Systems, Man, and Cybernetics*, 21(4), pp. 826–839. doi:10.1109/21.108300.

[8] de Souza, L.C.G. and de Souza, A.G. (2014) 'Satellite Attitude Control System Design considering the Fuel Slosh Dynamics', *Shock and Vibration*, 2014, pp. 1–8. doi:10.1155/2014/260206.

[9] Dodge, F.T. and Antonio, S. 'THE NEW "DYNAMIC BEHAVIOR OF LIQUIDS IN MOVING CONTAINERS"', p. 202.

[10] Dubanchet, V. (2016) *Modeling and Control of a Flexible Space Robot to Capture a Tumbling Debris*.

[11] Enright, P. and Wong, E. (1994) 'Propellant slosh models for the Cassini spacecraft', in *Astrodynamics Conference*. *Astrodynamics Conference*, Scottsdale,AZ,U.S.A.: American Institute of Aeronautics and Astronautics. doi:10.2514/6.1994-3730.

[12] Goldstein, H., Poole, C. and Safko, J. (2001) *Classical Mechanics (3rd Edition)*.

[13] Haykin, S.S. (2001) *Kalman filtering and neural networks*. New York: Wiley.

[14] Hughes, P.C. (1986), "Spacecraft Attitude Dynamics", John Wiley, New York

[15] Isermann, R. and Münchhof, M. (2011) *Identification of Dynamic Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-78879-9.

[16] Ivanov, D. *et al.* (2017) 'Methods for the vibration determination and parameter identification of spacecraft with flexible structures', *Journal of Computer and Systems Sciences International*, 56, pp. 311–327. doi:10.1134/S1064230717010087.

[17] Julier, S.J. and Uhlmann, J.K. 'A New Extension of the Kalman Filter to Nonlinear Systems', p. 12.

[18] Khosla, P.K. and Kanade, T. (1985) 'Parameter identification of robot dynamics', in *1985 24th IEEE Conference on Decision and Control*. *1985 24th IEEE Conference on Decision and Control*, pp. 1754–1760. doi:10.1109/CDC.1985.268838.

[19] Kopp, R.E. and Orford, R.J. (1963) 'LINEAR REGRESSION APPLIED TO SYSTEM IDENTIFICATION FOR ADAPTIVE CONTROL SYSTEMS', *AIAA Journal*, 1(10), pp. 2300–2306. doi:10.2514/3.2056.

[20] Kreppel, S. 'Scaling and Modeling of Propellant Sloshing and Zero Gravity Equilibrium for the Orion Service Module Propellant Tanks', p. 17.

[21] Ljung, L. (1998) 'System Identification', in Procházka, A. et al. (eds) Signal Analysis and Prediction. Boston, MA: Birkhäuser Boston, pp. 163–173. doi:10.1007/978-1-4612-1768-8_11.

[22] McGee, L.A. and Schmidt, S.F. 'Discovery of the Kalman Filter as a Practical Tool for Aerospace and Industry', p. 25.

[23] Meirovitch, L. (1991) 'Hybrid state equations of motion for flexible bodies in terms of quasi-coordinates', *Journal of Guidance, Control, and Dynamics*, 14(5), pp. 1008–1013. doi:10.2514/3.20743.

[24] Meirovitch, L. (1970), Methods of Analytical Dynamics, McGraw-Hill, New York,.

[25] Nachbagauer, K. *et al.* (2015) 'The Use of the Adjoint Method for Solving Typical Optimization Problems in Multibody Dynamics', *Journal of Computational and Nonlinear Dynamics*, 10(6), p. 061011. doi:10.1115/1.4028417.

[26] Nanos, K. and Papadopoulos, E. (2019) 'On Parameter Estimation of Space Manipulator Systems with Flexible Joints Using the Energy Balance', in *2019 International Conference on Robotics and Automation (ICRA)*. *2019 International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada: IEEE, pp. 3570–3576. doi:10.1109/ICRA.2019.8793960.

[27] Navabi, M. and Davoodi, A. (2017) '3D modeling and control of fuel sloshing in a spacecraft', in *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*. *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran: IEEE, pp. 0913–0918. doi:10.1109/KBEI.2017.8324928.

[28] Nelson, A.T. "Nonlinear estimation and modeling of noisy time-series by dual Kalman filtering methods"

[29] Rackl, W., Gerstmann, J. and Lampariello, R. (2018) 'Analysis of liquid fuel sloshing on free-floating robot dynamics under low-gravity condition', in *2018 IEEE Aerospace Conference*. *2018 IEEE Aerospace Conference*, Big Sky, MT: IEEE, pp. 1–12. doi:10.1109/AERO.2018.8396687.

[30] Reyhanoglu, M. (2003a) 'Maneuvering control problems for a spacecraft with unactuated fuel slosh dynamics', in *Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003. Conference on Control Applications*, Istanbul, Turkey: IEEE, pp. 695–699. doi:10.1109/CCA.2003.1223522.

[31] Sekhavat, P., Gong, Q. and Ross, I.M. (2007) 'NPSAT1 Parameter Estimation Using Unscented Kalman Filtering', in *2007 American Control Conference*. *2007 American Control Conference*, New York, NY, USA: IEEE, pp. 4445–4451. doi:10.1109/ACC.2007.4283031.

[32] Sen, S. and Bhattacharya, B. (2018) 'Non-Gaussian parameter estimation using generalized polynomial chaos expansion with extended Kalman filtering', *Structural Safety*, 70, pp. 104–114. doi:10.1016/j.strusafe.2017.10.009.

[33] *The NEAR Rendezvous Burn Anomaly of December 1998*

[34] Theureau, D., Mignot, J. and Tanguy, S. (2013) 'Integration of low g sloshing models with spacecraft attitude control simulators', in *AIAA Guidance, Navigation, and Control (GNC) Conference*. *AIAA Guidance, Navigation, and Control (GNC) Conference*, Boston, MA: American Institute of Aeronautics and Astronautics. doi:10.2514/6.2013-4961.

[35] Travers, M. and Choset, H. (2015) 'Use of the nonlinear observability rank condition for improved parametric estimation', in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA: IEEE, pp. 1029–1035. doi:10.1109/ICRA.2015.7139303.

[36] Van der Merwe, R. and Wan, E.A. (2001) 'The square-root unscented Kalman filter for state and parameter-estimation', in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings*, Salt Lake City, UT, USA: IEEE, pp. 3461–3464. doi:10.1109/ICASSP.2001.940586.

[37] Vreeburg (2005) 'Spacecraft maneuvers and slosh control', *IEEE Control Systems*, 25(3), pp. 12–16. doi:10.1109/MCS.2005.1432593.

[38] Vreeburg, J.P.B. (2005) 'Measured states of Sloshsat FLEVO', in *56th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*. *56th International Astronautical Congress of the International Astronautical Federation, the International*

Academy of Astronautics, and the International Institute of Space Law, Fukuoka, Japan: American Institute of Aeronautics and Astronautics. doi:10.2514/6.IAC-05-C1.2.09.

[39] Wan, E.A. 'Dual Estimation and the Unscented Transformation', p. 7.

[40] Wan, E.A. and Van Der Merwe, R. (2000) 'The unscented Kalman filter for nonlinear estimation', in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373). Symposium on Adaptive Systems for Signal Processing Communications and Control*, Lake Louise, Alta., Canada: IEEE, pp. 153–158. doi:10.1109/ASSPCC.2000.882463.

[41] Wenzel, T.A. *et al.* (2006) 'Dual extended Kalman filter for vehicle state and parameter estimation', *Vehicle System Dynamics*, 44(2), pp. 153–171. doi:10.1080/00423110500385949.

[42] Wielitzka, M., Dagen, M. and Ortmaier, T. (2015) 'Joint unscented Kalman filter for state and parameter estimation in vehicle dynamics', in *2015 IEEE Conference on Control Applications (CCA). 2015 IEEE Conference on Control Applications (CCA)*, Sydney, Australia: IEEE, pp. 1945–1950. doi:10.1109/CCA.2015.7320894.

Appendix A

In this appendix two openFOAM case files will be presented, for the coupled and prescribed motion simulations. Only the dynamic mesh dictionaries will be provided. OpenFOAM-v2012 was used. The rest files can be retrieved from *...\tutorials\multiphase\interFoam\laminar\sloshingCylinder*.

Coupled rigid body motion – interFOAM simulation using a Newmark scheme

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2012                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      dynamicMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dynamicFvMesh       dynamicMotionSolverFvMesh;

motionSolverLibs   (rigidBodyMeshMotion);

motionSolver       rigidBodyMotion;

rigidBodyMotionCoeffs
{
    report          on;

    solver
    {
        type Newmark;
    }

    accelerationRelaxation 0.7;

    bodies
    {
        sloshingCylinder
        {
            type            cuboid;
            parent          root;

            // Cuboid dimensions
            Lx              0.3;
            Ly              0.2;
            Lz              0.5;

            // Density of the cuboid
            rho             500;

            // Cuboid mass
            mass             400;
            L               ($Lx $Ly $Lz);
            centreOfMass    (0 0 0);
            transform       (1 0 0 0 1 0 0 0 1) (0.5 0.45 0.1);

            joint
            {
                type            composite;
```

```
                joints
                (
                    {
                        type Px;
                    }
                );
            }

            patches         (sloshingCylinder);
            innerDistance   999;
            outerDistance   1000;
        }
    }

       restraints
       {

              force
              {
                     body   sloshingCylinder;
                     type        externalForce;
                     location    (0 0 0);
                     force       (1 0 0);
              }
       }
}
```

Prescribed 3D motion

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  v2012                                 |
|   \\  /    A nd           | Website:  www.openfoam.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      dynamicMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dynamicFvMesh   dynamicMotionSolverFvMesh;

motionSolver    solidBody;

solidBodyMotionFunction tabulated6DoFMotion;

tabulated6DoFMotionCoeffs
{
    CofG            ( 0 0 0 );
    timeDataFileName    "$FOAM_CASE/constant/motionProfile.dat";
}




// ************************************************************************* //
```

Structure of “motionProfile.dat” is also provided below. Rotations are according to the XYZ euler angle sequence.

```
#number_of_points
(
(time_1  ((x_1 y_1 z_1) (roll_1 pitch_1 yaw_1)))
(time_2  ((x_2 y_2 z_2) (roll_2 pitch_2 yaw_2)))
.
.
.
)
```

Appendix B

The matlab code for deriving the symbolic equations using the mixed quasi-coordinates formulation used in this thesis will be presented.

```
%*
Code for symbolic computation of equations of motion for a mixed generanlized-
quasicoordinate formulation.
Each degree of freedom is declared as a variable i.e. q1,q1_d,q1_dd named holder
variable, and as a function i.e. x(t), called actual variable, to facilitate symbolic
differentiation with respect to time, i.e. diff(x,t) or with respect to the variable
itself i.e. diff(L_holder, q1).
The output is the generalized inertia matrix "H" with respect to second_order_terms
and a vector of nonlinear terms contained in "c".

Author: George Rapakoulias - 1/10/2021

*%

clear;
clc;
% usefull functions
crossMat = @(x) [0 -x(3) x(2); x(3) 0 -x(1); -x(2) x(1) 0];

syms t x_s(t) y_s(t) z_s(t) theta_s(t) vx(t) vy(t) vz(t) omega_x(t) omega_y(t)
omega_z(t)
syms r v rd vd [3,1]
syms q qd qdd    [3,1]
syms M m x0 y0 z0 Ibxx Ibyy Ibzz Kx Ky Kz bz bx by real

Ib = diag([Ibxx, Ibyy, Ibzz]);

v_b = [vx; vy; vz];
omega_b = [omega_x; omega_y; omega_z];

p_s = [x0; y0; z0] + [x_s; y_s; z_s];

v_s = v_b + diff(p_s) + crossMat(formula(omega_b))*p_s;

K = 1/2*M*(v_b.'*v_b) + 1/2*m*(v_s.'*v_s)+ 1/2*omega_b.'*Ib*omega_b; %system kinetic
energy

T = 1/2*Kx*x_s^2 + 1/2*Ky*y_s^2 + 1/2*Kz*z_s^2;

L = K - T; %system Lagrangian in terms of function symbols

Rayleigh  = 1/2*bx*diff(x_s)^2 + 1/2*by*diff(y_s)^2 + 1/2*bz*diff(z_s)^2;

actual_gc_list = formula([x_s y_s z_s]); % theta_s]);
holder_gc_list = q.' ;

actual_gc_list_d = diff(actual_gc_list, t);
holder_gc_list_d = qd.';

actual_gc_list_dd = diff(actual_gc_list_d, t);
holder_gc_list_dd = qdd.';

actual_symbols = [actual_gc_list actual_gc_list_d actual_gc_list_dd ...
                  vx, vy, vz, diff(vx), diff(vy), diff(vz), ...
                  omega_x,    omega_y,omega_z,    diff(omega_x),    diff(omega_y),
diff(omega_z)];

holder_symbols = [holder_gc_list holder_gc_list_d holder_gc_list_dd v.' vd.' r.' rd.'
];

L_q = subs(L, formula(actual_symbols), holder_symbols);
```

```
Rayleigh_q = subs(Rayleigh, formula(actual_symbols), holder_symbols);

n_gen_dofs = length(actual_gc_list);
n_dofs = 6 + n_gen_dofs;

dLdV_act     = sym(zeros(3,1));
dLdV_act_dt  = sym(zeros(3,1));

for kk = 1:3
    dLdV = diff(L_q, v(kk));
    dLdV_act(kk) = subs(dLdV, holder_symbols, actual_symbols);
    dLdV_act_dt(kk) = diff(dLdV_act(kk), t);
end

LHS_V = simplify(dLdV_act_dt + crossMat(formula(omega_b))*dLdV_act);

dLdomega_act     = sym(zeros(3,1));
dLdomega_act_dt  = sym(zeros(3,1));

for kk = 1:3
    dLdomega = diff(L_q, r(kk));
    dLdomega_act(kk) = subs(dLdomega, holder_symbols, actual_symbols);
    dLdomega_act_dt(kk) = diff(dLdomega_act(kk), t);
end

LHS_omega = simplify(dLdomega_act_dt + crossMat(formula(omega_b))*dLdomega_act +
crossMat(formula(v_b))*dLdV_act);


dLdq_act    = sym(zeros(n_gen_dofs,1));
dRdq_act    = sym(zeros(n_gen_dofs,1));
dLdqd_act_dt = sym(zeros(n_gen_dofs,1));

for kk = 1:n_gen_dofs
    dLdqd = diff(L_q, holder_gc_list_d(kk));
    dLdqd_act = subs(dLdqd, holder_symbols, actual_symbols);
    dLdqd_act_dt(kk) = diff(dLdqd_act, t);
    dLdq = diff(L_q, holder_gc_list(kk));
    dRdq = diff(Rayleigh_q, holder_gc_list_d(kk));
    dLdq_act(kk) = subs(dLdq, holder_symbols, actual_symbols);
    dRdq_act(kk) = subs(dRdq, holder_symbols, actual_symbols);
end

LHS_g = dLdqd_act_dt - dLdq_act + dRdq_act;

LHS = [LHS_V; LHS_omega; LHS_g];

LHS_holder = subs(LHS, actual_symbols, holder_symbols);

second_order_terms = [vd.' rd.' holder_gc_list_dd];

H_calc = sym(zeros(n_dofs,n_dofs));
c = sym(zeros(n_dofs,1));
c_flag = zeros(n_dofs,1);

for i = 1:n_dofs
    currentLine = LHS_holder(i);
    terms = children(expand(currentLine));
    n_terms = length(terms);
    disp(n_terms);
    for kk = 1:n_terms
        c_flag = zeros(n_dofs,1);
        for j = 1:n_dofs
            [C,~] = coeffs(terms(kk), second_order_terms(j), 'All');
            if(length(C) == 2)
                H_calc(i,j) = H_calc(i,j) + C(1);
            else
                c_flag(j) = 1;
```

```
            end
        end
        if(min(c_flag) == 1)
            c(i) = c(i)+ terms(kk);
        end
    end
end

H = simplify(H_calc);
c = simplify(c);
if(simplify(LHS_holder-(H_calc*second_order_terms.'+c)) == zeros(n_dofs,1))
    disp("It looks like we made it!");
end
```