KALMAN FILTERING FOR DYNAMIC MOTION AND MODEL ESTIMATION

by

Randal Schumacher, B.Eng
Ryerson University, 2003

A thesis
presented to Ryerson University

in partial fulfillment of the
requirements for the degree of
Master of Applied Science
in the Program of
Mechanical Engineering

Toronto, Ontario, Canada, 2006

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis.
I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

ABSTRACT

KALMAN FILTERING FOR DYNAMIC MOTION AND MODEL ESTIMATION

Master of Applied Science, 2006, Randal Schumacher

School of Graduate Studies, Ryerson University

The fundamental task of a space vision system for rendezvous, capture, and servicing of satellites on-orbit is the real-time determination of the motion of the *target* vehicle as observed on-board a *chaser* vehicle. Augmenting the architecture to incorporate the highly regarded Kalman filtering technique can synthesize a system that is more capable, more efficient and more robust. A filter was designed and testing was conducted in an inertial environment and then in a more realistic relative motion orbital rendezvous scenario. The results indicate that a *Dynamic Motion Filter* based on *extended Kalman filtering* can provide the vision system routines with excellent initialization leading to faster convergence, reliable pose estimation at slower sampling rates, and the ability to estimate target position, velocity, orientation, angular velocity, and mass center location.

ACKNOWLEDGEMENTS

The author would like to thank:

Dr. D.J. McTavish for providing guidance and advice on the topic of Kalman filtering for dynamic motion estimation. Special thanks for formulating much of the theory and for providing direction during implementation and testing.

Dr. G. Okouneva for providing knowledge in the field of vision-based pose estimation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

BHT	Bounded Hough Transform
CoM	Center of Mass
DARPA	Defense Advanced Research Projects Agency
DCM	Direction Cosine Matrix
EKF	Extended Kalman Filter
ICP	Iterative Closest Point
KF	Kalman Filter
LHS	Left Hand Side
LTI	Linear Time-Invariant
MEKF	Multiplicative Extended Kalman Filter
MEX	Matlab EXecutable
QUEST	QUaternion ESTimator
RHS	Right Hand Side
RPY	Roll Pitch Yaw
SRKF	Square-Root Kalman Filter
UKF	Unscented Kalman Filter

NOMENCLATURE

Standard Symbols

a_{Transfer}	Semi-major axis of the Hohmann transfer ellipse
$\mathbf{C}_{ba}$	Direction Cosine Matrix
$\underline{\mathbf{c}}$	First moment of inertia vector
$c(\cdot)$	Cosine of an angle
$E[\cdot]$	Expectation of a random variate or multi-variate
$\mathbf{F}$	Dynamic coefficient matrix / Jacobian (size *n x n*)
$\mathcal{F}$	Reference frame (origin & three non-coplanar basis vectors)
$\hat{\mathcal{F}}$	A Vectrix
$\underline{\mathbf{f}}$	Total external force vector
$\underline{\bar{\mathbf{f}}}$	Total external *specific* force vector
$\mathbf{G}$	Input coupling matrix / Jacobian (size *n x r*)
$\underline{\mathbf{g}}$	Total external torque vector
$\mathbf{H}_k$	Measurement sensitivity matrix / Jacobian (size m *x n*)
$\underline{\mathbf{h}}_o$	Absolute angular momentum vector about an arbitrary point in rigid body $\mathcal{R}$
h_{p_v}	Perigee altitude of the Hohmann transfer ellipse for the chaser vehicle
h_{a_v}	Apogee altitude of the Hohmann transfer ellipse for the chaser vehicle
$\underline{\mathbf{I}}$	Second moment of inertia dyadic (assuming CoM reference frame)
$\underline{\mathbf{J}}$	Second moment of inertia dyadic
$\mathbf{K}_k$	Kalman Gain (size *n x m*)
K	Normalizing constant for quaternion reset in the *Attitude MEKF*
$\mathbf{L}$	Process noise coupling matrix / Jacobian (size *n x s*)
m	Mass of a rigid body
$\mathbf{n}_{\text{noise}}$	Measurement noise in the context of the translational measurement
$\mathbf{P}$	State covariance matrix (size *n x n*)
$\underline{\mathbf{p}}$	Absolute momentum vector of a point $\mathcal{P}$
$\mathbf{Q}$	Process noise covariance matrix (size *n x n)*
$\mathbf{Q}_c$	Process noise spectral density matrix (size *s x s)*
$\mathbf{q}$	Quaternion (scalar 4th component) or Euler Parameters Vector (scalar 1st component)
$\mathbf{R}_m$	Measurement noise covariance, expressed in meas. frame (size *m x m*)
$\mathbf{R}_k$	Measurement noise covariance (expressed in filter "workable" frame) (size *m x m*)
R	Inertial position vector magnitude (used in 2-body orbital motion equation)

$\mathbf{r}_c$	The inertial position of the *target* center of mass, expressed in an inertial frame $\mathcal{F}_i$
$\mathbf{r}_{cg,g}$	The mass center offset of the Center of Mass Frame relative to the Geometric Frame, expressed in the Geometric Frame
$\underline{\mathbf{r}}_n$	Position of the n^{th} material point from an arbitrary point of interest in rigid body $\mathcal{R}$
$\underline{\hat{\mathbf{r}}}$	A general unit vector
r_{p_v}	Perigee radius of the Hohmann transfer ellipse for the chaser vehicle
r_{a_v}	Apogee radius of the Hohmann transfer ellipse for the chaser vehicle
r_{c_c}	Circular orbit radius of the target vehicle
$\mathcal{R}$	A rigid body
$s(\cdot)$	Sine of an angle
T_{Transfer}	Period of the Hohmann transfer ellipse
$T_{\frac{1}{2}\text{Transfer}}$	Half period of the Hohmann transfer ellipse
$\mathbf{u}$	Controlled input (size r)
$\mathbf{v}_k$	Measurement error / noise (size *m*)
$\mathbf{v}_c$	The inertial velocity of the *target* center of mass, expressed in an inertial frame $\mathcal{F}_i$
$\underline{\mathbf{v}}_b$	Inertial velocity vector of a rigid body $\mathcal{R}$
$\mathbf{w}$	Disturbance input / Process noise (size *s*)
$\mathbf{x}$	System state (size *n*)
x_u, x_l	Upper and lower bounds for bounded uniformly distributed random number function
$\mathbf{z}_k$	The k^{th} measurement to the filter

Greek Symbols

$\Delta\alpha$	Angular distance traveled of target during $T_{\frac{1}{2}\text{Transfer}}$
γ_k	The k^{th} sample value of the norm of position errors
ζ	Norm of the attitude errors
η_*	True Anomaly (general case) – of the reference orbital motion
$\dot{\eta}_*$	Sweep Rate (general case) – of the reference orbital motion
$\boldsymbol{\Theta}_{ab}$	Generic notation to represent relative 3-D orientation between two frames $\mathcal{F}_a$ and $\mathcal{F}_b$
$\boldsymbol{\Theta}_{\text{ref}}$	Generic notation for a reference orientation
$\delta\boldsymbol{\Theta}$	Generic notation for a small deviation orientation
$\boldsymbol{\theta}$	The *small* rotation vector
θ_*	True Anomaly (circular case) – of the reference orbital motion
$\boldsymbol{\kappa}$	A generic error representation
$\mu_\oplus$	Earth's Gravitational Parameter (= 3.986012 x 10^5 km^3/s^2)
$\boldsymbol{\Xi}(\boldsymbol{\varpi})$	Quaternion forward multiplication construct (size *4 x 4*)
$\bar{\boldsymbol{\Xi}}(\mathbf{q})$	Quaternion reverse multiplication construct (size *4 x 4*)
ξ_k	The k^{th} sample value of the norm of attitude errors

σ	Standard deviation
σ^2	Variance
$\mathbf{\Phi}$	The rotation vector
φ	Initial phase between target and chaser at start of Hohmann transfer
χ	Bounded uniformly distributed random number function
$\mathbf{\Omega(q)}$	Quaternion multiplication construct (size *4 x 3*)
Ω_*	Sweep Rate (circular case) – of the reference orbital motion
$\underset{\rightharpoonup}{\boldsymbol{\omega}}_b$	Body inertial angular velocity vector
$\boldsymbol{\varpi}$	Body-rate quaternion (first element is zero)
ω_c	Mean orbital motion of the target
ω_*	Mean orbital motion of the Reference Orbital-Nadir Frame
$\{\hat{\mathbf{a}}, \mathbf{\Phi}\}$	Euler's Axis and Angle
$\{\eta, \boldsymbol{\varepsilon}\}$	Euler's Parameters
$\{\phi, \theta, \psi\}$	Roll-Pitch-Yaw sequence of Euler Angles

Subscripts, Sidescripts, Superscripts, Miscellaneous

$\underset{\rightharpoonup}{\mathbf{a}}$	A general vector
$\underset{\rightharpoonup}{\hat{\mathbf{a}}}$	A general unit vector
$\underset{\rightharpoonup}{\mathbf{a}}_{bc}$	A relative vector, "*b*" relative to "*c*"
$\mathbf{a}$	A column-matrix (a vector $\underset{\rightharpoonup}{\mathbf{a}}$ expressed in some reference frame $\mathscr{F}$)
$\hat{\mathbf{a}}_k^+$	An *a posteriori* estimate
$\hat{\mathbf{a}}_k^-$	An *a priori* estimate
$\tilde{\mathbf{a}}_k$	A measurement
$\mathbf{a}^\times$	Cross product expression for matrix algebra
$\mathbf{a}_{,b}$	The components of a vector expressed in Frame b
$\mathbf{a}_{bc,c}$	The components of a relative vector, "*b*" relative to "*c*", expressed in Frame c
$\oplus$	Quaternion rotation multiplier
$\mathbf{\Theta}$	Rotational partition
$\mathbf{r}$	Translational partition
k	Discrete-time
$*$	Denoting *reference orbital motion*

1 Introduction

1.1 Progress in Autonomous Satellite Capture

Currently a number of enabling technologies are being used to design and test systems for rendezvous, capture, and service of satellites on-orbit. Designed to be operational in a semi-autonomous or fully autonomous mode, these systems often rely heavily on computer vision. With the launch of the Air Force Research Lab's XSS-11 Flight Mission in early 2005 and the planned launch of the DARPA Orbital Express Mission, depicted in Figure 1-1, it is apparent that the space community is moving to develop highly capable unmanned space robotic systems.

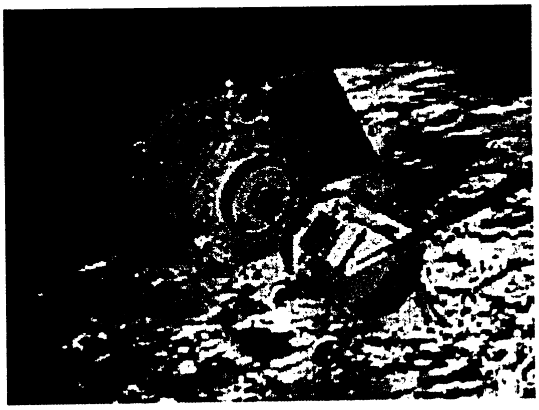


Figure 1-1: DARPA Orbital Express Concept [1]

At the heart of these on-orbit rendezvous, capture, and servicing systems is computer vision as described in [2-5]. In this thesis, the term *vision system* will be used to mean any sensor-algorithm system combination that produces a static 'snapshot' of the pose that is *not dynamic model based.* Examples include the Iterative Closest Point (ICP) – Stereo Camera combination in [3], the Bounded Hough Transform (BHT)/ICP – Lidar Scanner combination in [2] and [5], and other such combinations.

The fundamental task of these systems is the real-time determination of the motion of the *target* vehicle as observed on-board a *chaser* vehicle. Although typically these systems have been designed to use vision-based estimation as a stand-alone process, recent studies have looked to exploit the highly regarded Kalman filtering technique to add a propagation/prediction capability into the system architecture to synthesize a system that is more capable, more efficient and more robust. The potential benefits of such a *Hybrid Pose/Motion Estimation System* include:

- Better initialization of vision system routines, faster convergence, less computation.
- Reliable vision system pose estimation at slower sampling rates due to the propagation inherent in the filter.
- Simultaneous estimation of target mass properties: mass center location, principal inertia ratios, principal axis directions.
- Incorporation of "measurement slew" effects into the filter to compensate for time delay in measurement processing.

One such hybrid architecture as described in [6] involves essentially three distinct subsystems for the target motion estimation system: i) kinematic data fusion, to fuse sensory data into a coarse estimate of target pose; ii), Kalman filtering, to filter these coarse estimates and obtain the full target dynamic state and inertial parameters; and iii) shape estimation, used to build a probabilistic map of the target shape based on the filtered output. This hybrid of computer vision techniques and optimal estimation techniques is likely to produce a system that is more accurate and robust than a typical stand-alone vision system architecture.

In this thesis, the focus is on the design and performance of a reliable Kalman filter for such a hybrid pose/motion estimation system. As such, the details of the computer vision and sensor components are outside of the scope of this document. From the stand-point of the filter, the measurement may come from any vision system combination imaginable as long as it provides the filter a measurement of the target pose and/or motion. This Hybrid Pose/Motion Estimation System is depicted in Figure 1-2.

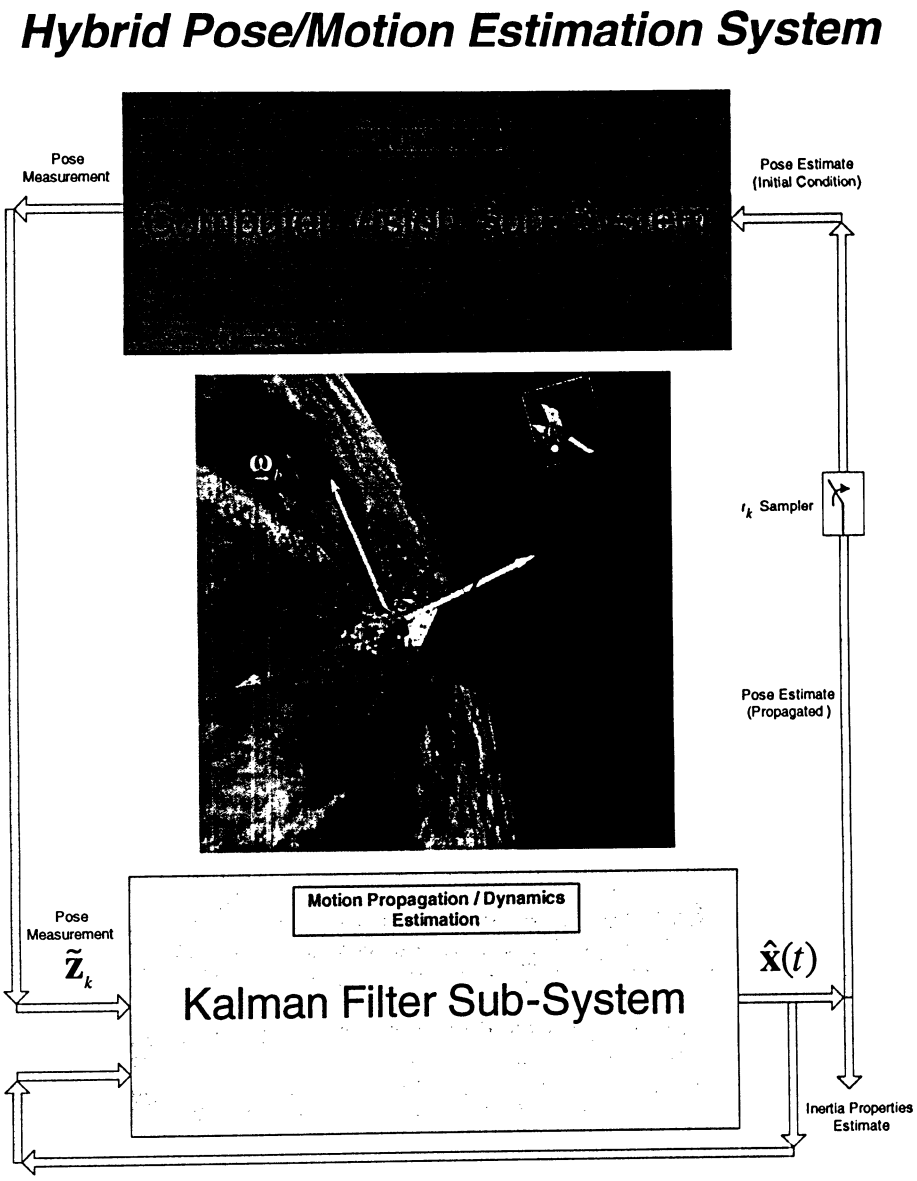

Hybrid Pose/Motion Estimation System
Pose Measurement
Computer Vision Sub-System
Pose Estimate (Initial Condition)
ω
t_k Sampler
Pose Estimate (Propagated)
Pose Measurement
$\tilde{\mathbf{z}}_k$
Motion Propagation / Dynamics Estimation
Kalman Filter Sub-System
$\hat{\mathbf{x}}(t)$
Inertia Properties Estimate

Figure 1-2: Hybrid Pose/Motion Estimation System Interaction

As the Kalman filter sub-system design is central to the hybrid architecture a concise literature review is presented of the various Kalman filtering implementations in the following section.

1.2 Literature Review

A brief review of general high-level Kalman filtering techniques is presented, followed by the important and problematic Kalman filtering techniques as applied to attitude determination.

1.2.1 Kalman Filtering – General Techniques

The Kalman filter is an optimal recursive data processing algorithm [7]. It was originally developed as a tool in linear estimation theory, but its extension to nonlinear problems was first introduced for application in the Apollo missions by Schmidt et al [8]. This *extended Kalman filter (EKF)* is basically a linearized version of the original formulation where a re-linearization occurs about each current best state estimate before covariance propagation and measurement update. Therefore, as soon as the new state estimate has been made, a new and better reference state trajectory is incorporated into the estimation process. A general procedure for the Kalman filtering algorithm is shown in Appendix A.

In theory, the standard Kalman filter recursion is the optimal solution to linear filtering problems perturbed by Gaussian white noise and the standard EKF recursion is a near-optimal solution to nonlinear filtering problems perturbed by white Gaussian noise. However, in application, the standard recursions can be prone to numerical difficulties. In fact, a modified implementation of the filter known as a *Square-Root Kalman filter* was used for the Apollo spacecraft navigation filters [7]. The covariance in the state estimate error, $\mathbf{P}$, has generated the most numerical difficulty over the past 40 years. For instance, as described by Maybeck [7], although it is theoretically impossible for the covariance matrix to have negative eigenvalues, such a condition can result due to numerical computation using finite word-length. These *ill-conditioned* problems are often noted when: i) the measurements are very accurate; or ii) a linear combination of state vector components is known with great precision while others are nearly unobservable [7]. To help alleviate the numerical problems inherent in the standard Kalman filter formulations, a number of alternative reformulations have been developed over the years. Figure 1-3 shows a sample of the different implementations available.

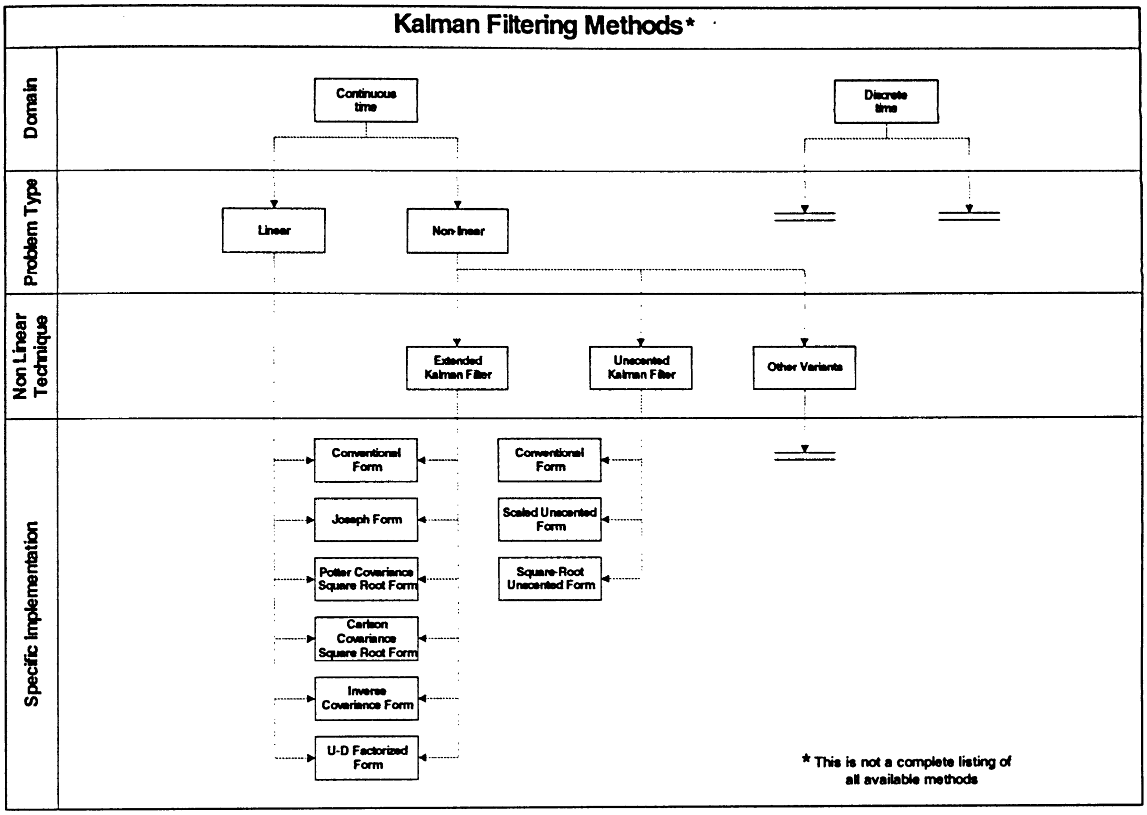


Figure 1-3: Listing of current Kalman Filtering Methods

In [7] a review of the relative computational efficiency of the alternative implementations is given (KF/ EKF types only) and it is suggested that the *Bierman-Thornton U-D Factorized Form* is possibly the most advantageous of the KF/EKF alternatives. It is also suggested that the approach to filter design should be to design and tune the KF/EKF types using the *Conventional Form*, but then implementing one of the more numerically stable forms for real-time operation.

A new breed of Kalman filtering for nonlinear problems was introduced in the mid-1990s. This is the so-called *Unscented Kalman Filter (UKF)* developed by Julier and Uhlmann [9, 10]. The motivation behind the UKF was that in some nonlinear cases it is easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function. The UKF uses a deterministic sampling approach to capture the mean and covariance estimates with a minimal set of sample points [11]. Furthermore, as outlined by Julier and Uhlmann [10], it is argued that the EKF has two main drawbacks in practice:

i) Linearization can produce highly unstable filters if the assumptions of local linearity is violated, and,

ii) The derivation of the Jacobian matrices is nontrivial in most applications and can often lead to significant implementation difficulties.

Depending on the application, the UKF has often been shown to out-perform the EKF as seen in [10-13], but at a greater computational cost.

1.2.2 Kalman Filtering for Spacecraft Attitude Estimation

An application sub-category of Kalman filtering with almost an equally large variety of techniques in the literature has been Kalman filtering for spacecraft attitude estimation. Over 35 years ago the earliest known publication for this application was presented by Farrel [14] as mentioned in a recent paper by Markley [15]. A fundamental issue in the literature has been how best to parameterize the attitude of the rigid body for estimation. Although a three-parameter set is attractive since it has as many parameters as there are degrees of freedom, Stuelpnagel [16] proves mathematically that *"no three-parameter set can be both global and nonsingular"*. As mentioned by Hughes [17], at least four parameters are required to treat all orientations uniformly. Thus, a four or more parameter set is required for treating attitude in an absolute sense. However, if it is anticipated that a certain rotation region will not be encountered, then a three-parameter set is fine as long as its inevitable singularity region is not entered [17]. Thus, a three-parameter set can be appropriate when dealing with small rotations as in the case of attitude deviations from a particular reference attitude. There is general agreement in the literature to adopt the

Rotation Matrix (also called the *Direction Cosine Matrix (DCM)* or *Attitude Matrix*) as the universal parameterization that specifies the orientation of a rigid body [17, 18].

The key paper produced by Lefferts, Markley, and Shuster [8] provides an excellent historical survey of the various parameterizations and methodologies used up to 1981. The formulation in [8] uses the quaternion parameterization of attitude explicitly throughout the derivations. As noted, advantages of this parameterization are that the quaternion has: i) prediction equations that are treated linearly; ii) a representation that is free from singularities (i.e. avoiding gimbal-lock such as that found in a *Roll-Pitch-Yaw (RPY)* sequence); and iii) the rotation matrix is algebraic in the quaternion components (no transcendental functions). However, the lack of independence in the four quaternion components leads to the constraint that the quaternion have unit norm. According to [8], this constraint results in the singularity of the covariance matrix of the quaternion state which led to the "The Body-Fixed Covariance Representation" to circumvent this difficulty. The approach used in [8] to avoid the singularity of the covariance matrix is to represent $\mathbf{P}$ by a matrix of smaller dimension. This approach later became known as the *Multiplicative Extended Kalman Filter (MEKF)* as it treats the reset operation that is implicit in the standard EKF as an explicit step so that the *a posteriori* estimate of the attitude is handled correctly (i.e. maintains orthonormality as an equivalent DCM). As described by Markley [15], the basic idea of the MEKF is to compute an unconstrained estimate of some three-component attitude *deviation* parameterization such as the small rotation vector, $\boldsymbol{\theta}$, while using the correctly normalized four-component quaternion, $\mathbf{q}_{\text{ref}}$, to provide a globally nonsingular attitude representation.

The EKF has performed admirably in the majority of attitude determination applications [19], but poor performance or even divergence due to the linearization implicit in the EKF has lead to other developments in spacecraft attitude estimation. A listing of the various nonlinear filtering implementations for attitude estimation is shown in Figure 1-4. Of all the methods it remains the case that the EKF in the form known as the MEKF has been the method of choice for the vast majority of applications [19]. Its performance will generally not be sufficient only in cases that have highly nonlinear dynamics / measurements models or those that lack decent initial estimates of the state.

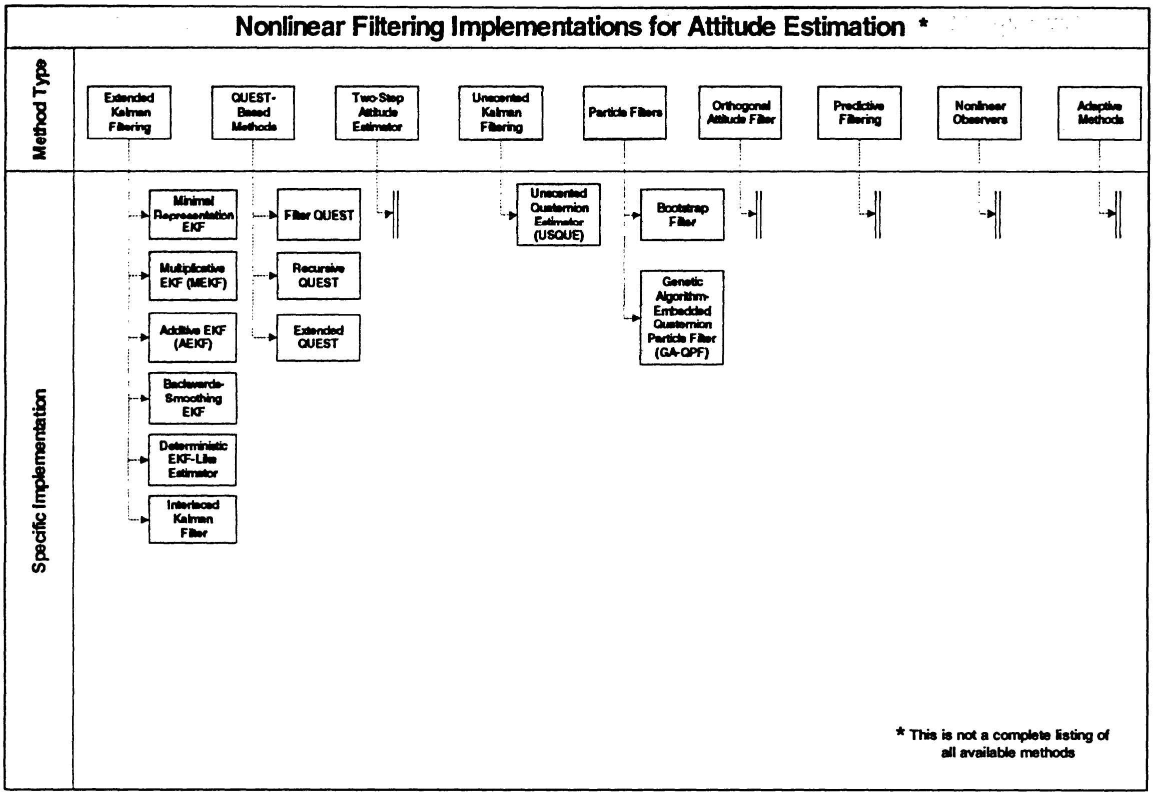

Nonlinear Filtering Implementations for Attitude Estimation *
Method Type
Specific Implementation
Extended Kalman Filtering
QUEST-Based Methods
Two-Step Attitude Estimator
Unscented Kalman Filtering
Particle Filters
Orthogonal Attitude Filter
Predictive Filtering
Nonlinear Observers
Adaptive Methods
Minimal Representation EKF
Multiplicative EKF (MEKF)
Additive EKF (AEKF)
Backwards-Smoothing EKF
Deterministic EKF-Like Estimator
Interlaced Kalman Filter
Filter QUEST
Recursive QUEST
Extended QUEST
Unscented Quaternion Estimator (USQUE)
Bootstrap Filter
Genetic Algorithm-Embedded Quaternion Particle Filter (GA-QPF)
* This is not a complete listing of all available methods

Figure 1-4: Listing of Nonlinear Filtering Implementations for Attitude Estimation [19]

In cases where the performance of the EKF is not sufficient, the UKF methods or “sigma-point filters” [19] serve as an attractive alternative. As mentioned in the literature, they are especially attractive when it is difficult or impossible to compute analytic partial derivatives of the dynamics or measurement models. The backwards-smoothing EKF has also been shown to work in cases where the EKF has failed but generally has not been competitive with a well-designed sigma-point filter due to its extra computational burden [19]. Particle filters can be used when the density function of the random variables to be estimated are non-Gaussian, but this method is computationally expensive. The QUEST (QUaternion-ESTimator) methods are all suboptimal compared to the standard EKF implementations but are often used for contingency designs or as simple analysis tools [19]. The Two-Step Attitude Estimator, Orthogonal Attitude Filter, Predictive Filter, and

Nonlinear Observers have future potential applications but have been limited to academic studies thus far. The Adaptive methods are useful when system parameters (such as vehicle inertia properties) are not well known or during cases where spacecraft anomalies may occur [19].

1.2.3 Objective

In this thesis, the emphasis is on key feature inclusion and general performance validation of a filter for use in a hybrid pose/motion estimation system as described in Section 1.1. The work, as presented, shall be easily adapted to any modern KF implementation. Since the vision system component in the hybrid architecture will already be computational demanding, achieving a filter design that is balanced in terms of performance and computational efficiency is desirable. In this study, the focus is on the design of an extended Kalman filter for this purpose and the assessment of its performance via simulation.

2 Theoretical Background

2.1 Assumptions

In studying the performance characteristics of a dynamic motion filter for use in a hybrid pose/motion estimation system several prominent assumptions have been made [20]. These include:

i) The target body is assumed to be passive with external actions arising from the environment only (often negligible depending on orbital characteristics).

ii) A reliable estimate of the chaser vehicle position and orientation is always available.

iii) Position and orientation measurements of the target body relative to the chaser are available as a discrete-time signal, presumed to have been produced by a vision system.

iv) A body inertia model of the target body is available that is assumed to be accurate, although the precise center of mass location is unknown.

The goal of this study is to demonstrate the performance of a dynamic motion filter based on an extended Kalman filter implementation. Design of the dynamic motion filter will be progress in three distinct parts. The translational motion estimation of the target body is coupled to the rotational motion estimation. First, a multiplicative extended Kalman filter will be designed to provide a near-optimal estimated of the rotational state of the target body. Second, the state will be augmented to additionally provide a near-optimal estimate of the translational state of the target body and this dynamic motion filter will be tested in a simple inertial simulation environment. Finally, the dynamic motion filter will be tested in an orbital environment where the states will be augmented to emphasize relative motion for a rendezvous and docking scenario.

In this study, effects not considered are as follows:

i) Target body flexible motion, or motion due to internal variations of inertia - the target body is assumed rigid.

ii) Partial or "smeared / delayed" measurements of target body motion by the chaser vehicle.

iii) Self-induced actions by the target body (thrust, magnetorquers, reaction wheels, etc).

iv) Uncertainty in the second moment of inertia.

2.2 Rotational and Translational Body Kinematics

2.2.1 Reference Frames In General

As outlined by Hughes [17], a *reference frame* consists of an origin location and a set of three non-coplanar basis vectors. The most predominantly used set of reference vectors is the *dextral orthonormal triad* (i.e. right-handed, mutually perpendicular, and of unit length) called a Cartesian reference frame. The orientation of any vector $\underset{\rightharpoonup}{\mathbf{v}}$ can be specified with respect to any reference frame of interest $\mathcal{F}$ by the three direction cosines between $\underset{\rightharpoonup}{\mathbf{v}}$ and the reference vectors of $\mathcal{F}$. Similarly, one can define the orientation of one frame $\mathcal{F}_a$ with respect to a second frame $\mathcal{F}_b$ by the three basis vectors of $\mathcal{F}_a$ projected onto $\mathcal{F}_b$. A notational device that is handy when dealing with multiple reference frames is the *vectrix*. From Hughes [17], it is stated that "*a vectrix has a split personality, possessing simultaneously the characteristics of both a vector and a matrix*".

Therefore, the elements of the vectrix $\hat{\mathcal{F}}_a$ corresponding to reference frame $\mathcal{F}_a$ are the basis vectors in a column matrix,

$$\hat{\mathcal{F}}_a = \begin{bmatrix} \underset{\rightarrow}{\hat{\mathbf{x}}}_a \\ \underset{\rightarrow}{\hat{\mathbf{y}}}_a \\ \underset{\rightarrow}{\hat{\mathbf{z}}}_a \end{bmatrix} \tag{2.1}$$

The orientation of one frame $\mathcal{F}_a$ can be defined with respect to a second frame $\mathcal{F}_b$ using vectrices,

$$\hat{\mathcal{F}}_b \cdot \hat{\mathcal{F}}_a^T = \begin{bmatrix} \underset{\rightarrow}{\hat{\mathbf{x}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{x}}}_a & \underset{\rightarrow}{\hat{\mathbf{x}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{y}}}_a & \underset{\rightarrow}{\hat{\mathbf{x}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{z}}}_a \\ \underset{\rightarrow}{\hat{\mathbf{y}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{x}}}_a & \underset{\rightarrow}{\hat{\mathbf{y}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{y}}}_a & \underset{\rightarrow}{\hat{\mathbf{y}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{z}}}_a \\ \underset{\rightarrow}{\hat{\mathbf{z}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{x}}}_a & \underset{\rightarrow}{\hat{\mathbf{z}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{y}}}_a & \underset{\rightarrow}{\hat{\mathbf{z}}}_b \cdot \underset{\rightarrow}{\hat{\mathbf{z}}}_a \end{bmatrix} \triangleq \mathbf{C}_{ba} \tag{2.2}$$

This gives the well known Rotation Matrix parameterization (or DCM) between the two frames. The rotation matrix $\mathbf{C}_{ba}$ transforms the components of a vector expressed in frame $\mathcal{F}_a$, to the components of that same vector in frame $\mathcal{F}_b$. This can be written as,

$$\mathbf{r}_{,b} = \mathbf{C}_{ba}\mathbf{r}_{,a} \tag{2.3}$$

The opposite transformation can be done using the transpose of $\mathbf{C}_{ba}$ since it is an orthonormal matrix,

$$\mathbf{r}_{,a} = \mathbf{C}_{ba}^T\mathbf{r}_{,b} = \mathbf{C}_{ab}\mathbf{r}_{,b} \tag{2.4}$$

From (2.2), it is easily seen that $\mathbf{C}_{ba}$ can be written instead as,

$$\mathbf{C}_{ba} = \begin{bmatrix} \hat{\mathbf{x}}_{a,b} & \hat{\mathbf{y}}_{a,b} & \hat{\mathbf{z}}_{a,b} \end{bmatrix} \tag{2.5}$$

where the columns represent the basis vectors of the frame $\mathcal{F}_a$ expressed in frame $\mathcal{F}_b$.

2.2.2 Problem-Specific Reference Frames

In this study, the dynamic motion filter is formulated and tested separately for both a simple inertial environment and an accelerated orbital environment. Reference frames relevant to both contexts are shown in Table 2-1. A graphical representation of these frames in the context of motion about some abstract inertial reference is shown in Figure 2-1. A graphical representation of these frames in the context of orbital motion about the Earth is shown in Figure 2-2.

	Inertial and Related Frames
$\mathcal{F}_i$	The Generic Inertial Frame.
$\mathcal{F}_\oplus$	Geocentric Equatorial Frame. ▪ A specific non-rotating Earth centered inertial frame
$\mathcal{F}_{p_*}$	Reference Orbital-Periapsis Frame. ▪ Shares origin with $\mathcal{F}_\oplus$ ▪ z-axis aligned with reference orbit normal: $\underline{\hat{\mathbf{z}}}_{p_*} = \underline{\mathbf{n}}_*$ ▪ x-axis aligned with reference orbital periapsis direction: $\underline{\hat{\mathbf{x}}}_{p_*} = \underline{\mathbf{p}}_*$
$\mathcal{F}_{p_v}$	Chaser Orbital-Periapsis Frame (if orbital plane different from $\mathcal{F}_{p_*}$).
$\mathcal{F}_{p_c}$	Target Orbital-Periapsis Frame (if orbital plane different from $\mathcal{F}_{p_*}$).
$\mathcal{F}_o$	Reference Orbital-Nadir Frame. ▪ Origin located at position $\underline{\mathbf{r}}_*$ ▪ z-axis local downward (nadir) ▪ y-axis in negative orbit normal direction
	The Chaser Vehicle Frames
$\mathcal{F}_v$	Chaser Vehicle Body Frame.
$\mathcal{F}_m$	Chaser Pose Measurement Frame. ▪ The frame in which the pose of the target body is directly estimated. *Measurement errors are directly specified in this reference frame.*
	The Target Body Frames
$\mathcal{F}_g$	Target Geometry Frame. ▪ The frame that is observed and measured by a vision system in $\mathcal{F}_m$
$\mathcal{F}_c$	Target Mass Center Frame. ▪ A frame parallel to $\mathcal{F}_g$ but placed at the target mass center
$\mathcal{F}_P$	Target Principal-Axis Frame. ▪ A frame in which the first moment and cross second moments of inertia are zero

Table 2-1: Specific Reference Frames for the Dynamic Motion Filter Study

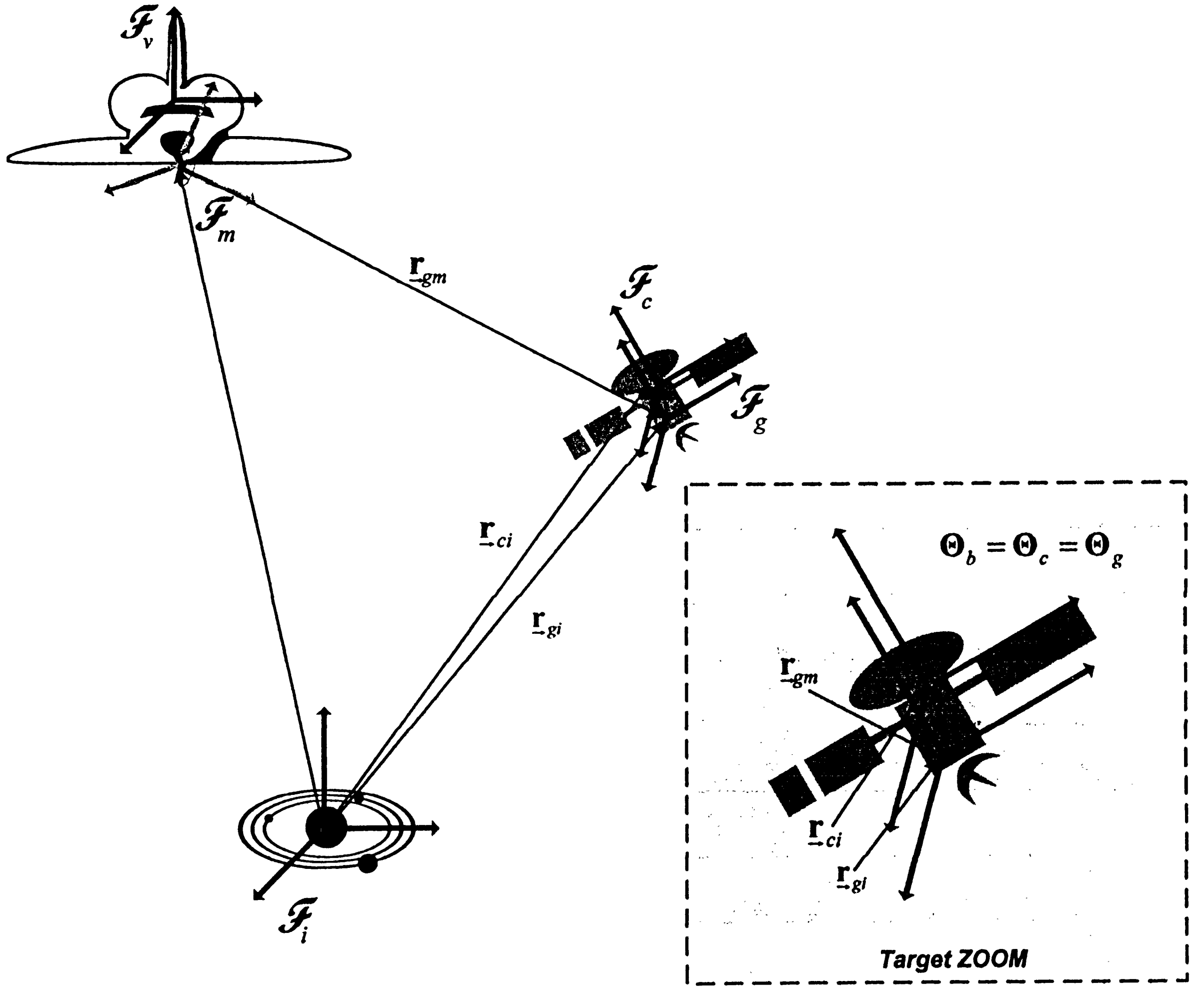

$\mathcal{F}_v$
$\mathcal{F}_m$
$\underset{\rightarrow}{\mathbf{r}}_{gm}$
$\mathcal{F}_c$
$\mathcal{F}_g$
$\underset{\rightarrow}{\mathbf{r}}_{ci}$
$\underset{\rightarrow}{\mathbf{r}}_{gi}$
$\mathcal{F}_i$
$\Theta_b = \Theta_c = \Theta_g$
$\underset{\rightarrow}{\mathbf{r}}_{gm}$
$\underset{\rightarrow}{\mathbf{r}}_{ci}$
$\underset{\rightarrow}{\mathbf{r}}_{gi}$
Target ZOOM

Figure 2-1: Inertial Environment Reference Frames

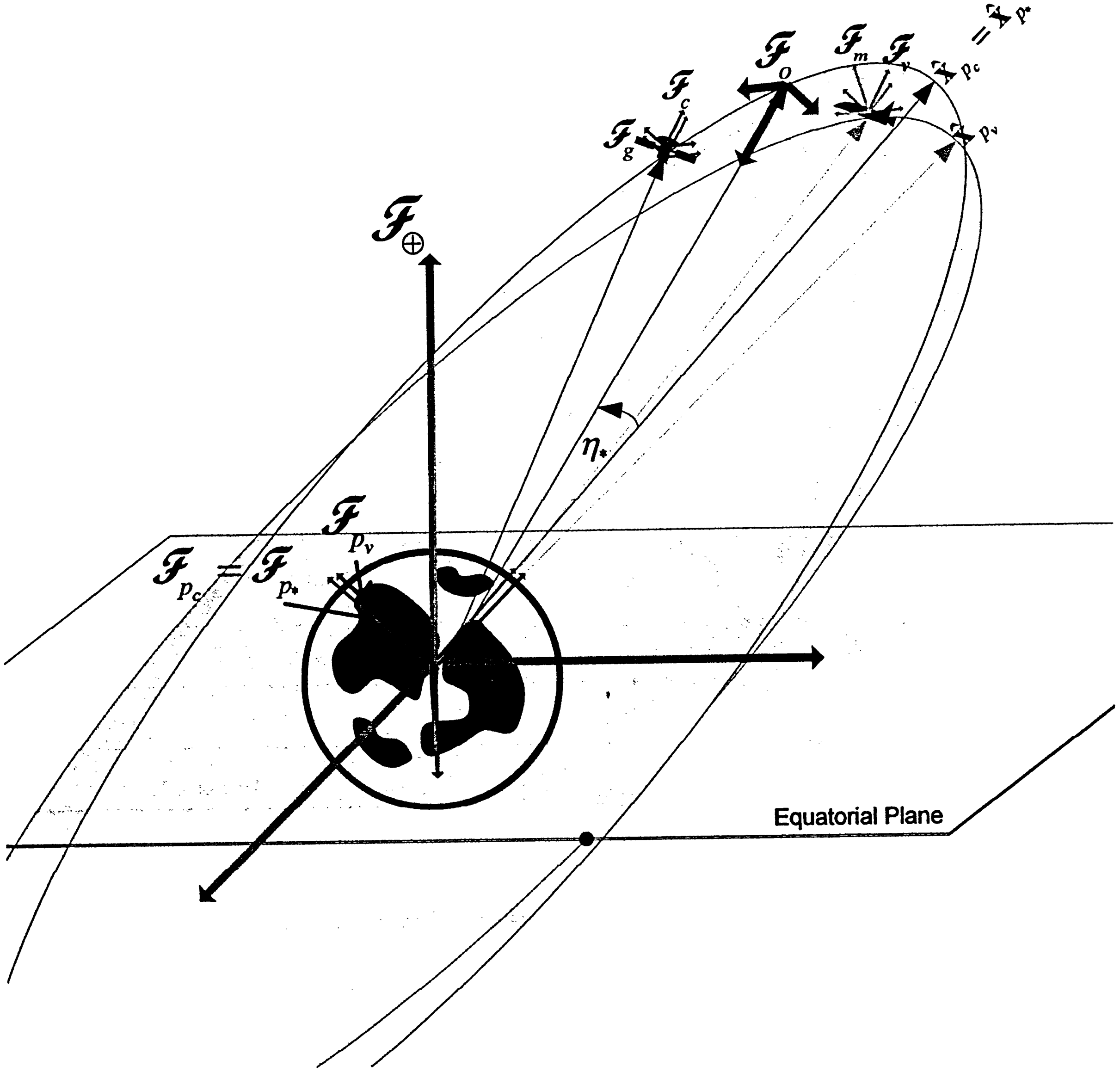


Figure 2-2: Orbital Environment Reference Frames

2.2.3 General Orientation and Parameterizations

By way of [21], a generic notation $\mathbf{\Theta}_{ab}$ is used to describe the orientation of some reference frame $\mathcal{F}_b$ relative to another reference frame $\mathcal{F}_a$. If a third frame $\mathcal{F}_c$ is involved the orientation of $\mathcal{F}_c$ relative to $\mathcal{F}_a$ can be found using $\mathbf{\Theta}_{ab}$ such that,

$$\mathbf{\Theta}_{ac} = \mathbf{\Theta}_{ab} \oplus \mathbf{\Theta}_{bc} \tag{2.6}$$

This generic form can be used and manipulated without resorting to any specific parameterization. As an example of a specific numerical representation, (2.6) can be represented in terms of the DCM parameterization as,

$$\mathbf{C}_{ca} = \mathbf{C}_{cb}\mathbf{C}_{ba} \tag{2.7}$$

When only two reference frames such as $\mathcal{F}_a$ and $\mathcal{F}_b$ are involved, $\mathbf{C}_{ba}$ can be simply referred to as $\mathbf{C}$. As is well known, there are other specific parameterizations that can represent a rotation $\mathbf{\Theta}$. These include:

i) Euler's Axis and Angle, $\{\hat{\mathbf{a}}, \Phi\}$

ii) The Rotation Vector, $\mathbf{\Phi} = \Phi\hat{\mathbf{a}}$

iii) Euler Angle Sets such as the standard Roll-Pitch-Yaw, $\{\phi, \theta, \psi\}$

iv) Euler Parameters, $\left\{\eta = \cos\left(\frac{\Phi}{2}\right) \quad , \quad \boldsymbol{\varepsilon} = \sin\left(\frac{\Phi}{2}\right)\hat{\mathbf{a}}\right\}$

These can all be related back to the "base" parameterization (i.e., the DCM) by the following relations,

$$\mathbf{C} = \mathbf{C}(\hat{\mathbf{a}}, \Phi) = \cos(\Phi)\mathbf{1} + (1 - \cos(\Phi))\hat{\mathbf{a}}\hat{\mathbf{a}}^T - \sin(\Phi)\hat{\mathbf{a}}^{\times} \tag{2.8}$$

$$\mathbf{C} = \mathbf{C}(\mathbf{\Phi}) = \cos(\Phi)\mathbf{1} + (1-\cos(\Phi))\frac{\mathbf{\Phi}\mathbf{\Phi}^T}{\Phi^2} - \sin(\Phi)\frac{\mathbf{\Phi}^\times}{\Phi} \tag{2.9}$$

$$\mathbf{C} = \mathbf{C}(\phi,\theta,\psi) = \mathbf{C}_1(\phi)\mathbf{C}_2(\theta)\mathbf{C}_3(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{bmatrix} \begin{bmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} c\psi & s\psi & 0 \\ -s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

$$\mathbf{C} = \mathbf{C}(\eta,\boldsymbol{\varepsilon}) = (\eta^2 - \boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon})\mathbf{1} + 2\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T - 2\eta\boldsymbol{\varepsilon}^\times \tag{2.11}$$

In this study, the DCM or the quaternion is used to describe the rate of change of orientation.

In terms of the DCM, the differential equation for the time evolution of the attitude is,

$$\dot{\mathbf{C}}_{bi} = -\boldsymbol{\omega}_b^\times \mathbf{C}_{bi} \tag{2.12}$$

where $\boldsymbol{\omega}_b = \boldsymbol{\omega}_{bi,b}$ is the angular velocity of a body frame $\mathcal{F}_b$ relative to an inertial frame $\mathcal{F}_i$, expressed in the body frame $\mathcal{F}_b$.

2.2.4 Euler Parameter Vector (Quaternion) Description of Orientation

As per [21], a 4x1 column matrix is defined to represent the Euler parameter vector,

$$\mathbf{q} = \begin{bmatrix} \eta \\ \boldsymbol{\varepsilon} \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{2.13}$$

and refer to it as the "quaternion" as a shorthand name. The normalization condition is then,

$$\mathbf{q}^T\mathbf{q} = 1 \tag{2.14}$$

and the identity quaternion is indicated by

$$\mathbf{q}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.15}$$

Symbolically, it is possible to introduce consecutive rotations with the quaternion as,

$$\mathbf{q}_3 = \mathbf{q}_1 \oplus \mathbf{q}_2 \tag{2.16}$$

with the rotation of $\mathbf{q}_1$ followed by $\mathbf{q}_2$. Numerically, this is implemented by

$$\mathbf{q}_3 = \Xi(\mathbf{q}_2)\mathbf{q}_1 \tag{2.17}$$

or (preserving the "forward order"):

$$\mathbf{q}_3 = \bar{\Xi}(\mathbf{q}_1)\mathbf{q}_2 \tag{2.18}$$

with the coefficient matrix constructs Ξ and $\bar{\Xi}$ defined as:

$$\Xi(\mathbf{q}) = \begin{bmatrix} \eta & -\boldsymbol{\varepsilon}^T \\ \boldsymbol{\varepsilon} & \eta\mathbf{1} - \boldsymbol{\varepsilon}^\times \end{bmatrix} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \tag{2.19}$$

$$\bar{\mathbf{\Xi}}(\mathbf{q}) = \begin{bmatrix} \eta & -\boldsymbol{\varepsilon}^T \\ \boldsymbol{\varepsilon} & \eta\mathbf{1} + \boldsymbol{\varepsilon}^\times \end{bmatrix} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \tag{2.20}$$

See [21] for more information regarding the Euler parameter vector.

To describe the rate of change of orientation in terms of the quaternion the differential equation is,

$$\dot{\mathbf{q}} = \mathbf{q} \oplus \boldsymbol{\varpi} \tag{2.21}$$

or, expressed in a format suitable for computation,

$$\dot{\mathbf{q}} = \mathbf{\Xi}(\boldsymbol{\varpi})\mathbf{q} = \bar{\mathbf{\Xi}}(\mathbf{q})\boldsymbol{\varpi} = \mathbf{\Omega}(\mathbf{q})\boldsymbol{\omega} \tag{2.22}$$

with the additional constructs defined as,

$$\boldsymbol{\varpi} = \begin{bmatrix} 0 \\ \frac{1}{2}\boldsymbol{\omega} \end{bmatrix}, \tag{2.23}$$

$$\mathbf{\Omega}(\mathbf{q}) = \frac{1}{2}\begin{bmatrix} -\boldsymbol{\varepsilon}^T \\ \eta\mathbf{1} + \boldsymbol{\varepsilon}^\times \end{bmatrix} = \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \tag{2.24}$$

For a more detailed discussion of the various attitude parameterizations that can be used to represent $\mathbf{\Theta}$, see [21, 22].

2.3 Motion Equations for a Rigid Body

The dynamic equations for a rigid body $\mathscr{R}$ is developed from Newton-Euler laws as derived in Hughes [17]. It is assumed below that the vectors and vector expressions have their components expressed in a general body-fixed frame, $\mathscr{F}_b$. In the following derivation the frame $\mathscr{F}_b$ is not assumed at the center of mass nor is it assumed that it is a principal-axis (mass centered) frame.

The translational and absolute angular momentum rates of change are given as (where o is defined as some point of interest fixed in $\mathscr{F}_b$),

$$\begin{aligned} \dot{\mathbf{p}} &= -\boldsymbol{\omega}_b^\times \mathbf{p} + \mathbf{f} \\ \dot{\mathbf{h}}_o &= -\boldsymbol{\omega}_b^\times \mathbf{h} - \mathbf{v}_o^\times \mathbf{p} + \mathbf{g}_o \end{aligned} \tag{2.25}$$

where the translational and absolute angular momentum are given as,

$$\begin{aligned} \mathbf{p} &= m\mathbf{v}_o - \mathbf{c}^\times \boldsymbol{\omega}_b \\ \mathbf{h}_o &= \mathbf{c}^\times \mathbf{v}_o + \mathbf{J}\boldsymbol{\omega}_b \end{aligned} \tag{2.26}$$

After some manipulation with (2.25) and (2.26), the standard results are found:

$$\begin{aligned} m\dot{\mathbf{v}}_o - \mathbf{c}^\times \dot{\boldsymbol{\omega}}_b + m\boldsymbol{\omega}_b^\times \mathbf{v}_o - \boldsymbol{\omega}_b^\times \mathbf{c}^\times \boldsymbol{\omega}_b &= \mathbf{f} \\ \mathbf{c}^\times \dot{\mathbf{v}}_o + \mathbf{J}\dot{\boldsymbol{\omega}}_b + \mathbf{c}^\times \boldsymbol{\omega}_b^\times \mathbf{v}_o + \boldsymbol{\omega}_b^\times \mathbf{J}\boldsymbol{\omega}_b &= \mathbf{g}_o \end{aligned} \tag{2.27}$$

or in matrix form as,

$$\begin{bmatrix} m\mathbf{1} & -\mathbf{c}^\times \\ \mathbf{c}^\times & \mathbf{J} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}}_o \\ \dot{\boldsymbol{\omega}}_b \end{bmatrix} + \begin{bmatrix} m\boldsymbol{\omega}_b^\times & -\boldsymbol{\omega}_b^\times \mathbf{c}^\times \\ \mathbf{c}^\times \boldsymbol{\omega}_b^\times & \boldsymbol{\omega}_b^\times \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{v}_o \\ \boldsymbol{\omega}_b \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{g}_o \end{bmatrix} \tag{2.28}$$

with the associated variables and parameters described in Table 2-2.

	Target Body Dynamics Notation
m	The total mass of the rigid body $\mathscr{R}$.
$\mathbf{r}$	Location of a point in the rigid body $\mathscr{R}$.
$\mathbf{c} = \int \mathbf{r}\,dm$	The first moment of inertia of $\mathscr{R}$.
$\mathbf{J} = -\int \mathbf{r}^\times \mathbf{r}^\times dm$	The second moment of inertia of $\mathscr{R}$.
$\mathbf{f} = \mathbf{f}_{,b}$	The total of all external forces acting on $\mathscr{R}$, expressed in $\mathscr{F}_b$.
$\mathbf{g} = \mathbf{g}_{,b}$	The total of all external forces torques on $\mathscr{R}$, expressed in $\mathscr{F}_b$.
$\mathbf{v}_o = \mathbf{v}_b = \mathbf{v}_{bi,b}$	The body inertial velocity of $\mathscr{R}$, expressed in $\mathscr{F}_b$.
$\boldsymbol{\omega}_b = \boldsymbol{\omega}_{bi,b}$	The body inertial angular velocity of $\mathscr{R}$, expressed in $\mathscr{F}_b$.

Table 2-2: Target Body Dynamics Notation

For this study, it is sufficient to simulate the motion of the target body in a mass center frame, $\mathscr{F}_c$. In this case $\mathscr{F}_b$ is the same as $\mathscr{F}_c$, and the dynamic motion equations of (2.27) then simplify to,

$$\begin{aligned} m\dot{\mathbf{v}}_b + m\boldsymbol{\omega}_b^\times \mathbf{v}_b &= \mathbf{f} \\ \mathbf{I}\dot{\boldsymbol{\omega}}_b + \boldsymbol{\omega}_b^\times \mathbf{I}\boldsymbol{\omega}_b &= \mathbf{g} \end{aligned} \tag{2.29}$$

since at the mass center $\mathbf{c} = \mathbf{0}$ and denoting the body inertia $\mathbf{J} = \mathbf{I}$ which is conventional in the literature when explicitly using a mass-center frame, $\mathscr{F}_c$. As noted in [20], if the body velocity is expressed directly in an inertial frame, $\mathscr{F}_i$, rather than a body frame, $\mathscr{F}_b$, then Newton's law "*F=ma*" is realized in the translational motion equation,

$$\begin{aligned} m\dot{\mathbf{v}}_{bi,i} &= \mathbf{f}_{,i} \\ \mathbf{I}\dot{\boldsymbol{\omega}}_b + \boldsymbol{\omega}_b^\times \mathbf{I}\boldsymbol{\omega}_b &= \mathbf{g} \end{aligned} \tag{2.30}$$

2.3.1 Motion/Relative Motion in the Orbital Context

In an orbital environment about the Earth, the motion of the body mass center is governed by the equation of perturbed orbital motion for a rigid object,

$$\ddot{\underline{\mathbf{r}}} = -\frac{\mu_{\oplus}}{R^2}\hat{\underline{\mathbf{r}}} + \delta\underline{\mathbf{a}} \tag{2.31}$$

where the first term on the RHS of (2.31) represents the standard $1/R^2$ gravity law with the applied force acting along the radial direction $\hat{\underline{\mathbf{r}}}$, and the second term on the RHS of (2.31) accounting for any non-Keplerian acceleration.

As shown in Figure 2-2, in this study the orbital motion of three distinct positions are specified corresponding to the target satellite mass center, chaser satellite mass center, and a reference orbital position. The target satellite is considered to be in free orbit, assumed mostly Keplerian but not perfectly. Its equation of motion is given by,

$$\ddot{\underline{\mathbf{r}}}_c = -\frac{\mu_{\oplus}}{R_c^{\,2}}\hat{\underline{\mathbf{r}}}_c + \delta\underline{\mathbf{a}}_c \tag{2.32}$$

with the small acceleration $\delta\underline{\mathbf{a}}_c = \delta\bar{\underline{\mathbf{f}}}_c$ accounting for any non-Keplerian motion. The chaser spacecraft is assumed roughly co-orbiting with the target with possibly gentle thrust/torque applied during the proximity operations. Its equation of motion is given by,

$$\ddot{\underline{\mathbf{r}}}_v = -\frac{\mu_{\oplus}}{R_v^{\,2}}\hat{\underline{\mathbf{r}}}_v + \delta\underline{\mathbf{a}}_v \tag{2.33}$$

with the small acceleration $\delta\underline{\mathbf{a}}_v = \bar{\underline{\mathbf{F}}}_v + \delta\bar{\underline{\mathbf{f}}}_v$ accounting for any maneuvering action and non-Keplerian motion, respectively. A reference orbital position, denoted by $\underline{\mathbf{r}}_*$, is introduced and is governed by perfect Keplerian motion of the form,

$$\ddot{\underline{\mathbf{r}}}_* = -\frac{\mu_\oplus}{R_*^{\,2}}\hat{\underline{\mathbf{r}}}_* \tag{2.34}$$

At this position, $\underline{\mathbf{r}}_*$ is the origin of a reference orbital-nadir frame, $\mathcal{F}_o$, and maintains a fixed attitude with respect to the local nadir and orbit normal directions associated with $\mathcal{F}_{p*}$. The relative position of the chaser or target from the reference orbit is given as,

$$\underline{\mathbf{r}}'_i = \underline{\mathbf{r}}_i - \underline{\mathbf{r}}_* \tag{2.35}$$

where *i* is used generically to associate any particular object being considered.

2.4 The Modified EKF Procedure for Attitude Filtering

The most widespread implementation of the EKF for spacecraft attitude estimation, the Multiplicative Extended Kalman Filter (MEKF), parameterizes the global attitude $\mathbf{\Theta}$ with a unit quaternion $\mathbf{q}$, while employing a three-component representation for the attitude deviation $\delta\mathbf{\Theta}$ [15] . In this section, the three-component parameterization of the attitude deviation is discussed explicitly using the small rotation vector, $\boldsymbol{\theta}$.

Originally formulated using the quaternion parameterization explicitly [8], the basic idea of the MEKF is to describe the true orientation $\mathbf{\Theta}$ as the rotational sum of a reference orientation $\mathbf{\Theta}_{\text{ref}}$ and a small rotational deviation $\delta\mathbf{\Theta}$, as given by,

$$\mathbf{\Theta} = \mathbf{\Theta}_{\text{ref}} \oplus \delta\mathbf{\Theta} \tag{2.36}$$

The filter then performs an unconstrained estimate of a three-component attitude deviation parameterization $\boldsymbol{\theta}$ corresponding to $\delta\mathbf{\Theta}$, with a correctly normalized quaternion $\mathbf{q}_{\text{ref}}$ being used as the parameterization of $\mathbf{\Theta}_{\text{ref}}$ about which the attitude deviation is defined [15]. Although an EKF is traditionally thought to have only two steps: i) time propagation; and ii) measurement update, one must be more careful for the MEKF during the measurement update and explicitly define an additional sub-step called the *reset* step.

In the following, the use of the superscripts "-" and "+" indicate the estimate just before and just after the measurement update at time t_k, respectively. For example, the shorthand notation $\hat{\mathbf{a}}_k^-$ has the same meaning as $\hat{\mathbf{a}}(t_k^-)$ (i.e. the *a priori* estimate at time t_k).

In the MEKF, the continuous-time propagation is arranged such that $\hat{\boldsymbol{\theta}}_k^- \equiv \mathbf{0}$, meaning that the reference orientation is the best estimate of the true quaternion prior to incorporation of the measurement at t_k. During the discrete measurement update the filter estimates a finite value of the three-component attitude deviation $\hat{\boldsymbol{\theta}}_k^+$. As this occurs the reference orientation still retains its pre-update value $\boldsymbol{\Theta}_{\text{ref},k}^-$ such that it is no longer the optimal estimate of the body orientation. The reset step then corrects this by passing the updated information in $\hat{\boldsymbol{\theta}}_k^+$ to a post-update reference orientation $\boldsymbol{\Theta}_{\text{ref},k}^+$ and resetting the estimate of $\hat{\boldsymbol{\theta}}$ to zero [15].

The general orientation description of (2.36) can be written symbolically in terms of the DCM or quaternion as,

$$\begin{aligned} \mathbf{C} &= \mathbf{C}_{\text{ref}} \oplus \delta\mathbf{C}(\boldsymbol{\theta}) \\ \mathbf{q} &= \mathbf{q}_{\text{ref}} \oplus \delta\mathbf{q}(\boldsymbol{\theta}) \end{aligned} \tag{2.37}$$

or written in their "workable" matrix and quaternion algebraic format,

$$\begin{aligned} \mathbf{C} &= \delta\mathbf{C}(\boldsymbol{\theta})\mathbf{C}_{\text{ref}} \\ \mathbf{q} &= \Xi(\delta\mathbf{q}(\boldsymbol{\theta}))\mathbf{q}_{\text{ref}} \end{aligned} \tag{2.38}$$

The best estimate of the true orientation as processed in the reset step can be seen symbolically in terms of the DCM or quaternion as,

$$\begin{aligned}\mathbf{C}^{+}_{\mathrm{ref},k} &= \mathbf{C}^{-}_{\mathrm{ref},k} \oplus \delta\mathbf{C}(\hat{\boldsymbol{\theta}}^{+}_{k}) \\ \mathbf{q}^{+}_{\mathrm{ref},k} &= \mathbf{q}^{-}_{\mathrm{ref},k} \oplus \delta\mathbf{q}(\hat{\boldsymbol{\theta}}^{+}_{k})\end{aligned} \tag{2.39}$$

As discussed by Markley [15], the reset step is implicit in the standard EKF where the true state $\mathbf{x}$ is the sum of a reference state $\mathbf{x}_{\mathrm{ref}}$ and a small error $\boldsymbol{\kappa}$,

$$\mathbf{x}_k = \mathbf{x}_{\mathrm{ref},k} + \boldsymbol{\kappa}_k \tag{2.40}$$

During the measurement update in the standard EKF, an updated estimate of the error is formed,

$$\hat{\boldsymbol{\kappa}}^{+}_{k} = \hat{\boldsymbol{\kappa}}^{-}_{k} + \Delta\hat{\boldsymbol{\kappa}}_k \tag{2.41}$$

where $\Delta\hat{\varepsilon}_k$ is the correction resulting from the measurement update. The reset operation that implicitly (shown explicitly here) moves the ûpdate information from the error state to the full state estimate is seen as,

$$\mathbf{x}^{+}_{\mathrm{ref},k} = \mathbf{x}^{-}_{\mathrm{ref},k} + \hat{\boldsymbol{\kappa}}^{+}_{k} - \hat{\boldsymbol{\kappa}}^{-}_{k} = \mathbf{x}^{-}_{\mathrm{ref},k} + \Delta\hat{\boldsymbol{\kappa}}_k \tag{2.42}$$

Since orientation is not strictly *additive* in the sense of (2.42), the reset operation must be treated explicitly using the symbolic rotational sum operator, $\oplus$.

2.4.1 Attitude Deviation Representation

As mentioned, in this thesis, a unique three-component representation is defined in terms of the small rotation vector $\boldsymbol{\theta}$ description in [23]. The attitude deviation in (2.37) can be defined in terms of the quaternion as,

$$\delta\mathbf{q}(\boldsymbol{\theta}) = \frac{1}{K}\begin{bmatrix} 1-\frac{1}{8}\boldsymbol{\theta}^T\boldsymbol{\theta} \\ \frac{1}{2}\boldsymbol{\theta} \end{bmatrix} \tag{2.43}$$

where K is,

$$K = \sqrt{\left(1+\frac{1}{64}\theta^4\right)} \tag{2.44}$$

3 Procedure

In this thesis, the complete solution was designed and tested in three successive stages, as depicted in Figure 3-1. First, the filter was designed and tested to estimate angular motion only in an inertial environment. Second, the filter state was augmented to include translational states and tested in an inertial environment. Finally, the filter state was augmented to include relative orbital motion and tested in an orbital environment. For the complete implementation details of each stage please refer to Appendix B, Appendix F, and Appendix H, respectively.

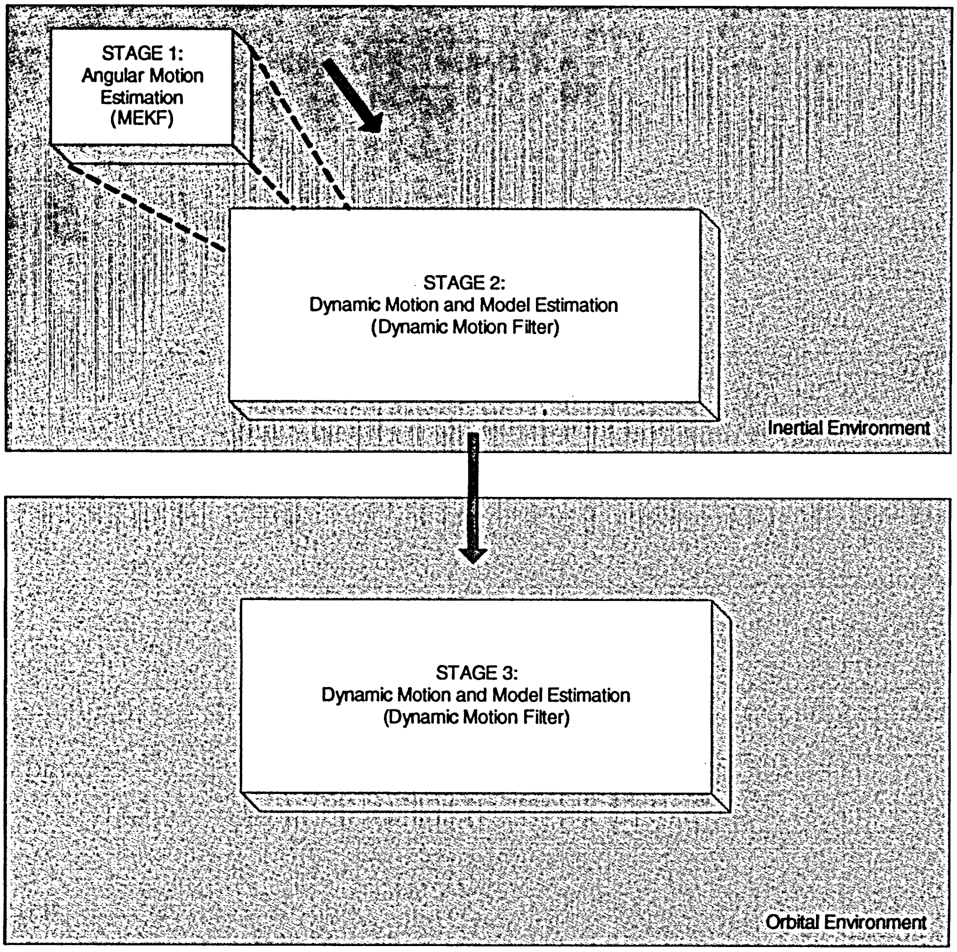


Figure 3-1: Design and Testing Stages

3.1 Modeling/Simulation Tools

Testing of the EKF algorithms was conducted by building the truth models in Simulink. Because the rotational and translational dynamics of the target body are only coupled when the translational motion is considered, one can build the environment for the Attitude MEKF initially and then augment the state to the dynamic motion filter which incorporates the translational part for the full motion and model estimation. All EKF algorithms and related functions were written in C and integrated into the testing environment using C MEX S-functions. See the Matlab documentation for specifics related to the C MEX S-function interface [24].

3.2 Attitude Estimation in an Inertial Context

3.2.1 Inertial Attitude Dynamics Truth Model

For the attitude dynamics truth model, Euler's rotational motion equation (expressed in a CoM Body-Frame) is implemented,

$$\mathbf{I}\dot{\boldsymbol{\omega}}_b + \boldsymbol{\omega}_b^{\times}\mathbf{I}\boldsymbol{\omega}_b = \mathbf{g}_{\text{ext}} \tag{3.1}$$

The acting torques can be partitioned as $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}} + \mathbf{g}_{\text{disturb}}$. For the truth model, the target object is assumed to be rigid and basically in free motion with only small external disturbance torques acting on the body such that $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{disturb}}$. The model was verified using standard tests such as the constant energy check in the case of $\mathbf{g}_{\text{ext}} = \mathbf{0}$, the orthonormality check of the DCM ($\mathbf{C}_{bi}\mathbf{C}_{bi}^T = \mathbf{1}$) or constraint check of the quaternion ($\mathbf{q}^T\mathbf{q} = 1$), etc. All tests verified that the rotational dynamics truth model was behaving as expected.

3.2.2 Inertial Attitude Dynamics Filter Model

Again, for the attitude dynamics filter model, Euler's rotational motion equation (expressed in a CoM Body-Frame) is implemented,

$$\mathbf{I}\dot{\boldsymbol{\omega}}_b + \boldsymbol{\omega}_b^{\times}\mathbf{I}\boldsymbol{\omega}_b = \mathbf{g}_{\mathrm{ext}} \tag{3.2}$$

The acting torques can be partitioned as $\mathbf{g}_{\mathrm{ext}} = \mathbf{g}_{\mathrm{known}} + \mathbf{g}_{\mathrm{disturb}}$. For the filter model the target object is assumed to be rigid and basically in free motion with only small external disturbance torques acting on it. Since only small non-deterministic external disturbance torques are assumed to be acting on the body the filter model propagates the torque free case with $\mathbf{g}_{\mathrm{ext}} = \mathbf{g}_{\mathrm{disturb}} = \mathbf{0}$ (since the expectation of the disturbances is zero mean).

3.2.3 Inertial Attitude Measurement Model

Part of the direct output of the vision system will be a measurement of the target orientation $\mathcal{F}_g$ relative to the measurement frame $\mathcal{F}_m$, whose orientation is assumed to be known relative to the chaser vehicle frame $\mathcal{F}_v$.,The chaser vehicle frame is assumed absolutely know relative to the inertial frame $\mathcal{F}_i$. Working in the inertial environment it is assumed that the measurement frame errors $\delta\mathbf{C}_{gm}$ are applied as follows:

$$\tilde{\boldsymbol{\Theta}}_g = \boldsymbol{\Theta}_g \oplus \delta\boldsymbol{\Theta}_{mg} = \boldsymbol{\Theta}_{im} \oplus \tilde{\boldsymbol{\Theta}}_{mg} = \boldsymbol{\Theta}_{im} \oplus \left(\boldsymbol{\Theta}_{mg} \oplus \delta\boldsymbol{\Theta}_{mg}\right) \tag{3.3}$$

where the notation is described in Table 3-1.

	Generic Attitude Measurement Notation
$\boldsymbol{\Theta}_g$	The orientation of the target geometry frame relative to the inertial frame.
$\tilde{\boldsymbol{\Theta}}_g$	A measurement of the target geometry frame orientation relative to the inertial frame.
$\boldsymbol{\Theta}_{im}$	The known orientation of the pose measurement frame relative to the inertial frame.
$\tilde{\boldsymbol{\Theta}}_{mg}$	The measurement of the target orientation in the measurement frame.
$\boldsymbol{\Theta}_{mg}$	The true target orientation in the measurement frame.
$\delta\boldsymbol{\Theta}_{mg}$	The small rotational deviation between $\tilde{\boldsymbol{\Theta}}_{mg}$ and $\boldsymbol{\Theta}_{mg}$.

Table 3-1: Generic Attitude Measurement Notation (Inertial Context)

As specified in [20], it is assumed that the x-axis basis direction of $\mathcal{F}_m$ forms the boresight of the imaging system and as such local RPY errors of the attitude measurement can be specified as follows:

$$\delta\boldsymbol{\Theta}_{mg} \to \mathbf{C}_{RPY}(\delta\phi,\delta\theta,\delta\psi) \tag{3.4}$$

To match the performance of a typical pose instrument for the application, the roll error expectation is expected to be less than that of the pitch and yaw errors. The expectation of the pitch and yaw errors are assumed equivalent in this study. For the pose instrument model the measurement accuracy is arbitrarily assumed to be similar to that presented in [6]. The measurements entering the filter (four co-operative sensors case) have a mean attitude error of 0.852 deg. It is assumed in this study that a vision system upstream of the filter can produce similar accuracy or better. No detailed description is found as to the exact meaning of this attitude accuracy in [6] (on each RPY axis, norm of errors, etc.) so it is assumed in this study that it is associated with the size of the attitude error, $\delta\boldsymbol{\Theta}_{mg}$. If

$\delta\Theta_{mg}$ was parameterized in terms of the associated rotation vector the norm of the attitude error would be given as,

$$\text{If} \quad \|\Phi(\delta\Theta_{mg})\| \equiv \zeta \tag{3.5}$$

and the mean or expected value is,

$$E[\zeta] \approx 0.852 \text{ deg} \tag{3.6}$$

For the purpose of generating a simple attitude measurement noise model that is comparable to the measurement accuracy used in [6], the following is done:

i) Assume,

$$\|\Phi(\delta\Theta_{mg})\| \equiv \zeta \tag{3.7}$$

and

$$\Phi(\delta\Theta_{mg}) \cong \begin{bmatrix} \delta\phi \\ \delta\theta \\ \delta\psi \end{bmatrix} \equiv \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{3.8}$$

ii) As noted above, the roll error is expected to be less than the yaw and pitch errors. Arbitrarily specify $\delta\psi = \delta\theta = 3\delta\phi$.

iii) Also assume,

$$\begin{aligned} E\left[(\delta\phi - \delta\bar{\phi})^2\right] &= \sigma_\phi^2 \\ E\left[(\delta\theta - \delta\bar{\theta})^2\right] &= \sigma_\theta^2 \\ E\left[(\delta\psi - \delta\bar{\psi})^2\right] &= \sigma_\psi^2 \end{aligned} \tag{3.9}$$

and the RPY errors are assumed zero mean.

iv) Then,

$$\begin{aligned} \|\Phi(\delta\Theta_{mg})\| &\equiv \sqrt{x_1^2+x_2^2+x_3^2}=\sqrt{\delta\phi^2+\delta\theta^2+\delta\psi^2} \\ &= \sqrt{\delta\phi^2+(3\delta\phi)^2+(3\delta\phi)^2} \\ &=\sqrt{19\delta\phi^2} \approx 1 \\ &\text{So, } \delta\phi=0.2294 \end{aligned} \tag{3.10}$$

and assume this particular $\delta\phi$ represents the standard deviation in roll error.

v) The noise $\delta\Theta_{mg}$ is simulated as follows:

a. Generate $\{\delta\phi,\delta\theta,\delta\psi\}$ noise that is zero mean Gaussian with standard deviations σ_ϕ, σ_θ, and σ_ψ.

b. Convert $\{\delta\phi,\delta\theta,\delta\psi\}$ to $\delta\mathbf{C}_{gm}$ to output the actual measurement in the DCM parameterization,

$$\tilde{\mathbf{C}}_k=\tilde{\mathbf{C}}_g(t_k)=\tilde{\mathbf{C}}_{gm}\mathbf{C}_{mi}=\delta\mathbf{C}_{gm}\mathbf{C}_{gm}\mathbf{C}_{mi}=\delta\mathbf{C}_{gm}\mathbf{C}_g(t_k) \tag{3.11}$$

3.2.4 Inertial Attitude Measurement Covariance

Recall equation (3.4) above. As noted in [20], these small angles can be collected together as a small rotation vector,

$$\delta\boldsymbol{\theta}_m=\begin{bmatrix}\delta\phi\\ \delta\theta\\ \delta\psi\end{bmatrix} \tag{3.12}$$

It is assumed that the covariance of this small rotation vector is specified and defined as,

$$\operatorname{cov}\delta\boldsymbol{\theta}_m = E\left\{\delta\boldsymbol{\theta}_m \delta\boldsymbol{\theta}_m^T\right\} = \mathbf{R}_{\Theta,m} \tag{3.13}$$

However, to be used by the filter, this small rotation vector is manipulated to get the transformed error vector specified in the "reference" body frame (the moving reference frame at each filter sampling time),

$$\delta\boldsymbol{\theta}_{\text{ref}} = \mathbf{C}_{\text{ref},m}\delta\boldsymbol{\theta}_m = \left(\mathbf{C}_{\text{ref}}\mathbf{C}_{mi}^T\right)\delta\boldsymbol{\theta}_m \tag{3.14}$$

Note the relationship between the measurement and the small rotation vector,

$$\mathbf{z}_{\Theta,k} = \delta\boldsymbol{\varepsilon} \approx \frac{1}{2}\boldsymbol{\theta} \tag{3.15}$$

From this the measurement covariance for the filter is the projection of the measurement frame error covariance into the current estimate of the reference body frame,

$$\mathbf{R}_{\Theta,k} = E\left\{\delta\boldsymbol{\varepsilon}\delta\boldsymbol{\varepsilon}^T\right\} = \frac{1}{4}E\left\{\delta\boldsymbol{\theta}_{\text{ref}}\delta\boldsymbol{\theta}_{\text{ref}}^T\right\} = \frac{1}{4}\mathbf{C}_{\text{ref},m}\mathbf{R}_{\Theta,m}\mathbf{C}_{\text{ref},m}^T \tag{3.16}$$

where

$$\mathbf{C}_{\text{ref}\,m} = \mathbf{C}_{\text{ref},k}^{-}\mathbf{C}_{mi}^T \tag{3.17}$$

The values selected to populate $\mathbf{R}_{\Theta,m}$ are detailed in Section 4.1.2.

3.3 Motion Estimation in an Inertial Context

3.3.1 Inertial Translational Dynamics Truth Model

In addition to the attitude dynamics truth model described in Section 3.2.1, for complete motion estimation, a simple translational dynamics truth model is added by simulating Newton's motion equations (expressed in inertial space),

$$m\dot{\mathbf{v}}_{bi,i} = \mathbf{f}_{,i} \tag{3.18}$$

The acting forces can be partitioned as $\mathbf{f}_{,i} = \mathbf{f}_{\text{known}} + \mathbf{f}_{\text{disturb}}$. For the truth model the target object is assumed to be rigid and basically in free motion with only small external disturbance forces acting on the body such that $\mathbf{f}_{,i} = \mathbf{f}_{\text{disturb}}$. The mass center offset from the target Geometric Frame is constant and therefore $\dot{\mathbf{r}}_{cg,g} = \mathbf{0}$. The model was verified using standard tests such as the constant energy check in the case of $\mathbf{f}_{,i} = \mathbf{0}$, and other simple tests to verify that the translational dynamics truth model was behaving as expected.

3.3.2 Inertial Translational Dynamics Filter Model

Similarly, in addition to the attitude dynamics filter model described in Section 3.2.2, for complete motion estimation a simple translational dynamics filter model is added by simulating Newton's motion equations (expressed in inertial space),

$$m\dot{\mathbf{v}}_{bi,i} = \mathbf{f}_{,i} \tag{3.19}$$

The acting forces can be partitioned as $\mathbf{f}_{,i} = \mathbf{f}_{\text{known}} + \mathbf{f}_{\text{disturb}}$. For the filter model the target is assumed to be basically in free motion with only small external disturbance forces acting on it. Since only small non-deterministic external disturbance forces are assumed to be acting on the body, the filter model propagates the force free case with $\mathbf{f}_{,i} = \mathbf{f}_{\text{disturb}} = \mathbf{0}$ (since the expectation of the disturbances is zero mean).

3.3.3 Translational Measurement Model

As noted in [20], at discrete times t_k, the imaging system of the chaser vehicle returns a measurement of the target's position and orientation. In the inertial testing environment, this corresponds to the measurement set $\underline{\mathbf{r}}_{gi}$ and $\mathbf{\Theta}_b$. The pose measurement frame $\mathcal{F}_m$ is the frame in which the pose of the *target* is directly estimated (by possibly a camera or lidar instrument). In this frame, the uncertainty associated with the pose measurement sensor is directly defined.

The pose measurement sensor provides to the filter a measurement of the target pose, $\tilde{\mathbf{r}}_g$ and $\tilde{\mathbf{\Theta}}_b$. Referring to Figure 2-1, the following relations can be specified,

$$\tilde{\mathbf{r}}_{gm,m} = \mathbf{r}_{gm,m} + \mathbf{n}_{\text{noise}} \tag{3.20}$$

In words, the position measurement (produce by the vision system) is an estimate of the Geometry frame $\mathcal{F}_g$ origin relative to the pose measurement frame $\mathcal{F}_m$ origin, expressed in the pose measurement frame $\mathcal{F}_m$. Again, the relationship between $\mathcal{F}_m$ and $\mathcal{F}_v$ is exactly known. Referencing this measurement back into the *inertial reference frame* $\mathcal{F}_i$,

$$\tilde{\mathbf{r}}_g = \mathbf{r}_v + \mathbf{C}_{mi}^T \tilde{\mathbf{r}}_{gm,m} \tag{3.21}$$

where $\mathbf{C}_{mi}^T = \mathbf{C}_{im} = \mathbf{C}_{vi}^T \mathbf{C}_{mv}^T$.

For the actual simulated noise model (3.20) is emulated. The noise in the sensor, $\mathbf{n}_{\text{noise}}$, can be captured by generating normally distributed random numbers multiplied by the *sigma* values expected for that particular instrument. Then during every measurement,

$$\tilde{\mathbf{r}}_{gm,m} = \mathbf{r}_{gm,m} + \Delta\mathbf{r}_{gm,m} \tag{3.22}$$

where $\Delta\mathbf{r}_{gm,m}$ is zero mean and Gaussian. In this study a simple approach will be used to simulate the noise and select sigma values for $\Delta\mathbf{r}_{gm,m}$ based on the assumed distance from the target, $\|\mathbf{r}_{gm,m}\|$, applicable over a specified interval. A representative sensor arrangement is given in [6], and sensor accuracy results are presented as a mean error (in terms of % target size). The author in [6] assumes $\|\mathbf{r}_{gm,m}\| \approx 10\text{ m}$. For an equivalent distance from the target it is assumed that the vision system produces a relative Cartesian position measurement, $\tilde{\mathbf{r}}_{gm,m}$, with errors that are zero mean and standard deviations on all axes of 5 cm.

Since any true sensors (such as cameras or LIDAR equipment) incorporated into a vision system would have an accuracy that is a function of the distance away from the target, a more sophisticated approach would be to generate these noise values using the approach in [25] where the $\Delta\mathbf{r}$ values are driven by the range and bearing of the target from the sensor instrument.

3.3.4 Translational Measurement Covariance

The measurement noise covariance for the translational motion filter is,

$$\operatorname{cov} \delta\mathbf{r}_{gm,m} = E\left\{\delta\mathbf{r}_{gm,m}\delta\mathbf{r}_{gm,m}^{T}\right\} = \mathbf{R}_{r,m} \tag{3.23}$$

Projecting into the reference inertial frame,

$$
\begin{aligned}
\delta\mathbf{r}_{gm,i} &= \mathbf{C}_{mi}^{T}\delta\mathbf{r}_{gm,m} \\
\mathbf{R}_{\mathbf{r},k} &= E\left\{\delta\mathbf{r}_{gm,i}\delta\mathbf{r}_{gm,i}^{T}\right\} = E\left\{\mathbf{C}_{mi}^{T}\delta\mathbf{r}_{gm,m}\left[\mathbf{C}_{mi}^{T}\delta\mathbf{r}_{gm,m}\right]^{T}\right\} = \mathbf{C}_{mi}^{T}\mathbf{R}_{\mathbf{r},m}\mathbf{C}_{mi}
\end{aligned}
\tag{3.24}
$$

The values selected to populate $\mathbf{R}_{\mathbf{r},m}$ are detailed in Section 4.2.2.

3.4 Motion Estimation in an Orbital Context

3.4.1 Orbital Dynamics Truth Model

Again, in addition to the attitude dynamics model described in Section 3.2.1, for motion estimation it is possible to add the orbital dynamics model by propagating the center of mass of each body (chaser and target) using the perturbed two-body orbital motion form for each,

$$\ddot{\underline{\mathbf{r}}} = -\frac{\mu_{\oplus}}{R^2}\hat{\underline{\mathbf{r}}} + \delta\underline{\mathbf{a}} \tag{3.25}$$

The numerical solution is typically resolved in the inertial Geocentric Equatorial Frame, $\mathcal{F}_{\oplus}$. However, in the present study the scope is restricted to the simplified case where both the chaser and target remain unperturbed such that the orbital motion evolves as the analytical solution to Keplerian motion. Under this assumption it is possible to incorporate $\delta\underline{\mathbf{a}} \neq \mathbf{0}$ if required using an osculating orbit formulation or restricting such acceleration to being impulsive [20]. Since the filter will deal with relative motion, a reference orbit is defined whose position is located by $\underline{\mathbf{r}}_{*}$, at which a Reference Orbital-Nadir Frame resides, $\mathcal{F}_{o}$. This reference orbit motion is assumed circular and governed by

$$\ddot{\underline{\mathbf{r}}}_* = -\frac{\mu_\oplus}{R_*^2}\hat{\underline{\mathbf{r}}}_* \qquad (3.26)$$

The frame of interest to express these quantities in is the Reference Orbital-Nadir Frame, $\mathcal{F}_o$. Each object relative to the reference orbit is described by $\underline{\mathbf{r}}_i' = \underline{\mathbf{r}}_i - \underline{\mathbf{r}}_*$. The situation is shown in Figure 3-2. As described in [20], a cylindrical coordinate description for the position of each object (chaser and target) is used, with respect to the Reference Orbital Periapsis Frame, $\mathcal{F}_{p_*}$. In general the chaser, target, and any reference orbital-nadir frame could all be out of plane with one another but these cylindrical coordinates are reflected/projected in the common reference orbital periapsis frame, $\mathcal{F}_{p_*}$. The object cylindrical variables and relative cylindrical variables are described in Table 3-2.

	Cylindrical Coordinates/Relative Coordinates Notation
R_i	Orbital radius of the object reflected in the Reference Orbital-Periapsis Frame, $\mathcal{F}_{p_*}$.
ϑ_i	Angular position of the object in $\mathcal{F}_{p_*}$ from the reference periapsis.
y_i	Out-of-plane position of the object in $\mathcal{F}_{p_*}$, defined in the negative orbit-normal direction.
$r_i = R_i - R_*$	Differential orbital radius reflected in $\mathcal{F}_{p_*}$.
$\theta_i = \vartheta_i - \eta_*$	Differential angular position in the reference orbital-plane.

Table 3-2: Generic Cylindrical Coordinates Notation for an Object

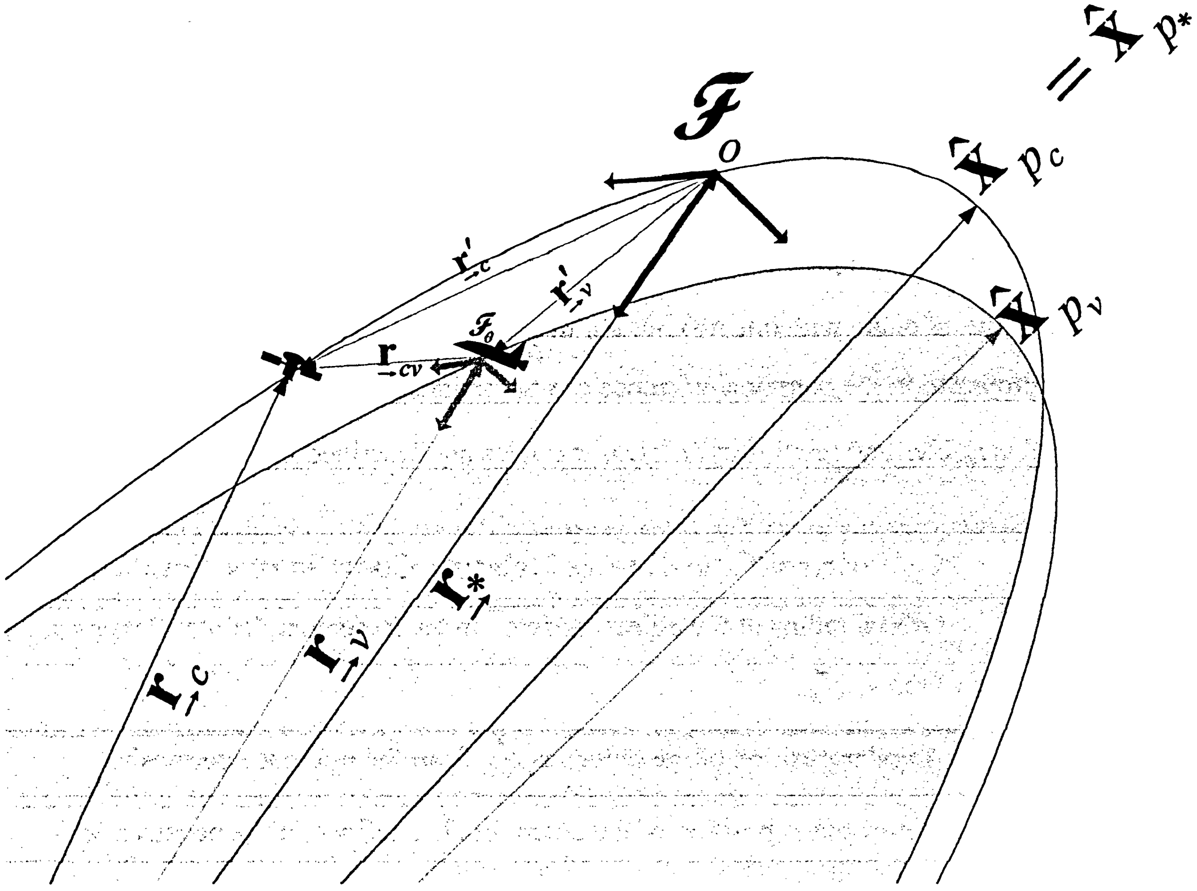


Figure 3-2: Position vector relations in Orbital Context

The relationships between the cylindrical variables are shown in Figure 3-3 with a view looking in the negative orbit-normal direction, $-\hat{\mathbf{n}}$, from above the plane of the Reference Orbital-Periapsis Frame, $\mathcal{F}_{p_*}$.

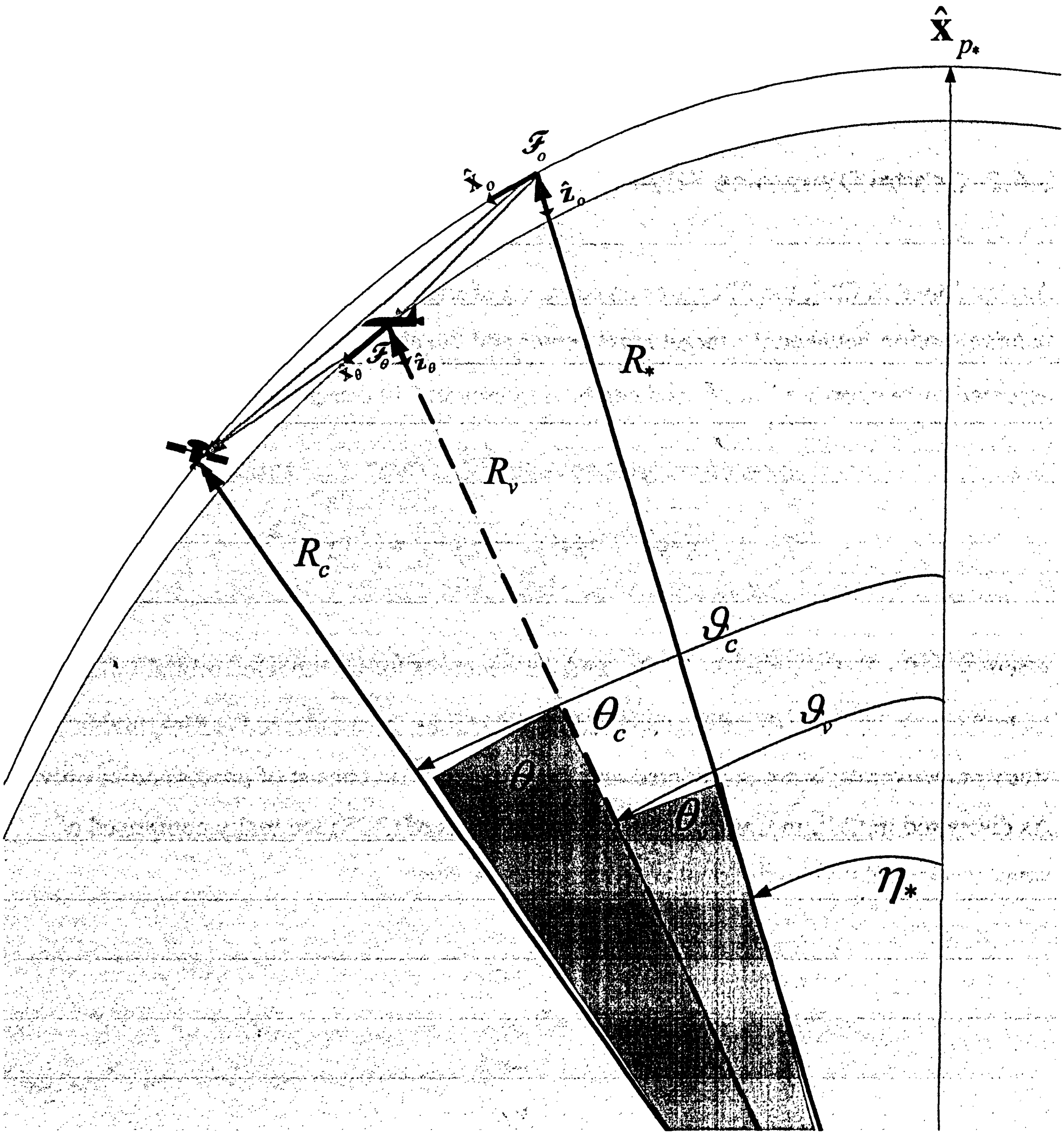


Figure 3-3: Cylindrical Coordinates relations in Reference Orbital-Periapsis Frame

From Figure 3-3, it is easily seen that the relative position coordinates (cylindrical) between the chaser and target are specified as

$$\begin{aligned} r &= r_c - r_v \\ \theta &= \theta_c - \theta_v \\ y &= y_c - y_v \end{aligned} \tag{3.27}$$

3.4.2 Orbital Dynamics Filter Model

As developed in [20], the filter uses an equation of small relative motion to relate the relative motion between the target mass center and the chaser mass center. The system is assumed to be expressed in $\mathscr{F}_o$ and can be implemented in direct Cartesian-like form as

$$\begin{bmatrix} \dot{\bar{\mathbf{r}}} \\ \dot{\bar{\mathbf{v}}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ -\Omega_*^2(\hat{\mathbf{y}}\hat{\mathbf{y}}^T - 3\hat{\mathbf{z}}\hat{\mathbf{z}}^T) & 2\Omega_*\hat{\mathbf{y}}^\times \end{bmatrix} \begin{bmatrix} \bar{\mathbf{r}} \\ \bar{\mathbf{v}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} [\delta\mathbf{a}] \tag{3.28}$$

where $\hat{\mathbf{y}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$, $\hat{\mathbf{z}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ and $\delta\mathbf{a}$ accounts for un-modeled orbital influences including any maneuvering thrust applied by the chaser. As mentioned earlier, in this study maneuvering is not considered and $\delta\mathbf{a}$ is assumed to consist of process noise only. As discussed in [20], in Cartesian-like form the states of (3.28) are really comprised of small motion relative cylindrical coordinates of the form

$$\begin{aligned} \bar{\mathbf{r}} &= \begin{bmatrix} R_*\theta \\ y \\ -r \end{bmatrix} \\ \bar{\mathbf{v}} &= \begin{bmatrix} R_*\dot{\theta} \\ \dot{y} \\ -\dot{r} \end{bmatrix} \end{aligned} \tag{3.29}$$

The orbital truth model and approximate relative form for the filter were verified to be correctly implemented by using standard tests from analytically predictable Keplerian motion.

3.4.3 Relative Orientation Measurement Model

For the relative orientation measurement we have the following generic relation,

$$\boldsymbol{\Theta}_{mg} = \boldsymbol{\Theta}_{mv} \oplus \boldsymbol{\Theta}_{v\oplus} \oplus \boldsymbol{\Theta}_{\oplus b} \tag{3.30}$$

where the notation is described in Table 3-3.

	Generic Attitude Measurement Notation
$\boldsymbol{\Theta}_{mg}$	The true target orientation in the measurement frame.
$\boldsymbol{\Theta}_{mv}$	The known measurement frame orientation in the chaser vehicle.
$\boldsymbol{\Theta}_{v\oplus}$	The dynamically specified chaser vehicle orientation in the geocentric equatorial frame.
$\boldsymbol{\Theta}_{v\oplus}$	The dynamically simulated target vehicle orientation in the geocentric equatorial frame.

Table 3-3: Generic Attitude Measurement Notation (Orbital Context)

Again, as specified in [20], it is assumed that the x-axis basis direction of $\mathcal{F}_m$ forms the bore-sight of the imaging system and as such local RPY errors of the attitude measurement can be specified as follows:

$$\delta\boldsymbol{\Theta}_{mg} \rightarrow \mathbf{C}_{RPY}(\delta\phi, \delta\theta, \delta\psi) \tag{3.31}$$

To match the performance of a typical pose instrument for the application, the roll error expectation is expected to be less than that of the pitch and yaw errors. The expectation of the pitch and yaw errors are assumed equivalent in this study.

Note that all parameters for the injected noise are identical to those as specified in Section 3.2.3 and will not be repeated for brevity. The measurement is generated as follows:

$$\tilde{\mathbf{C}}_{gm} = \delta\mathbf{C}_{gm}\mathbf{C}_{gm} = \delta\mathbf{C}_{gm}\mathbf{C}_{g\oplus}\mathbf{C}_{m\oplus}^{T} \tag{3.32}$$

3.4.4 Relative Orientation Measurement Covariance

As seen in Section 3.2.4, we have the following relation:

$$\delta\boldsymbol{\theta}_m = \begin{bmatrix} \delta\phi \\ \delta\theta \\ \delta\psi \end{bmatrix} \tag{3.33}$$

It is assumed that the covariance of this small rotation vector is specified and defined as

$$\operatorname{cov}\delta\boldsymbol{\theta}_m = E\left\{\delta\boldsymbol{\theta}_m\delta\boldsymbol{\theta}_m^T\right\} = \mathbf{R}_{\boldsymbol{\Theta},m} \tag{3.34}$$

However, since the applied measurement is the axis-part of the Euler parameters quaterion,

$$\mathbf{z}_{\boldsymbol{\Theta},k} = \delta\tilde{\boldsymbol{\varepsilon}} \tag{3.35}$$

and noting the following relationship between the measurement and the small rotation vector,

$$\mathbf{z}_{\boldsymbol{\theta},k} = \delta\boldsymbol{\varepsilon} \approx \frac{1}{2}\boldsymbol{\theta} \tag{3.36}$$

The applied measurement covariance is then defined as:

$$\mathbf{R}_{\boldsymbol{\theta},k} = E\left\{\delta\boldsymbol{\varepsilon}\delta\boldsymbol{\varepsilon}^T\right\} = \frac{1}{4}E\left\{\delta\boldsymbol{\theta}\delta\boldsymbol{\theta}^T\right\} = \frac{1}{4}\mathbf{C}_{gm}\mathbf{R}_{\boldsymbol{\theta},m}\mathbf{C}_{gm}^T \tag{3.37}$$

where

$$\mathbf{C}_{gm} = \mathbf{C}_{g\oplus}\mathbf{C}_{m\oplus}^T \tag{3.38}$$

The values selected to populate $\mathbf{R}_{\boldsymbol{\theta},m}$ are detailed in Section 4.3.2.

3.4.5 Relative Position Measurement Model

The pose measurement sensor provides to the filter a measurement of the relative position of the Target Geometry Frame origin relative to the Chaser Vehicle Frame origin, expressed in the Reference Orbital-Nadir Frame. In symbols, this is $\mathbf{r}_{gv,o}$. However, the physical measurement is of the form,

$$\tilde{\mathbf{r}}_{gm,m} = \mathbf{r}_{gm,m} + \mathbf{n}_{\text{noise}} \tag{3.39}$$

In words, the position measurement (produce by the vision system) is an estimate of the Geometry frame $\mathcal{F}_g$ origin relative to the pose measurement frame $\mathcal{F}_m$ origin, expressed in the pose measurement frame $\mathcal{F}_m$.

The physical relative position measurement is simulated by first extracting $\mathbf{r}_{gm,\oplus}$ from the truth model. It is related to the chaser and target vehicles by,

$$\mathbf{r}_{gm,\oplus} = \mathbf{r}_{g,\oplus} - \mathbf{r}_{m,\oplus} \tag{3.40}$$

with the following relations for the RHS of (3.40),

$$\begin{aligned} \mathbf{r}_{g,\oplus} &= \mathbf{r}_{c,\oplus} - \mathbf{C}_b^T \mathbf{r}_{cg,g} \\ \mathbf{r}_{m,\oplus} &= \mathbf{r}_{v,\oplus} - \mathbf{C}_v^T \mathbf{r}_{mv,v} \end{aligned} \tag{3.41}$$

Note, all quantities on the RHS of (3.41) are either generated by the truth model or specified during initialization. The quantity $\mathbf{r}_{gm,\oplus}$ is then converted to the measurement frame using the following relation,

$$\mathbf{r}_{gm,m} = \mathbf{C}_{mv} \mathbf{C}_{v\oplus} \mathbf{r}_{gm,\oplus} \tag{3.42}$$

To emulate (3.39) the noise in the sensor, $\mathbf{n}_{\text{noise}}$, can be captured by generating normally distributed random numbers multiplied by the *sigma* values expected for that particular instrument. Then during every measurement,

$$\tilde{\mathbf{r}}_{gm,m} = \mathbf{r}_{gm,m} + \Delta \mathbf{r}_{gm,m} \tag{3.43}$$

where $\Delta \mathbf{r}_{gm,m}$ is zero mean and Gaussian. In this study, a simple approach will be used to simulate the noise and select sigma values for $\Delta \mathbf{r}_{gm,m}$ based on the assumed distance from the target, $\| \mathbf{r}_{gm,m} \|$, applicable over a specified interval. A representative sensor arrangement is given in [6], and sensor accuracy results are presented as a mean error (in terms of % target size). To maintain consistency with Section 3.3.3, it is assumed that the vision system produces a relative Cartesian position measurement, $\tilde{\mathbf{r}}_{gm,m}$, with errors that are zero mean and standard deviations on all axes of 5 cm.

Finally, for conversion to the applied measurement used by the filter (namely, $\tilde{\mathbf{r}}_{gv,o}$), the following relationship is noted,

$$\tilde{\mathbf{r}}_{gv,o} = \mathbf{C}_{o\oplus}\mathbf{C}_{v}^{T}\left(\mathbf{r}_{mv,v} + \mathbf{C}_{mv}^{T}\tilde{\mathbf{r}}_{gm,m}\right) \tag{3.44}$$

3.4.6 Relative Position Measurement Covariance

The measurement noise covariance associated with the relative position measurement is:

$$\operatorname{cov}\, \delta\mathbf{r}_{gm,m} = E\left\{\delta\mathbf{r}_{gm,m}\delta\mathbf{r}_{gm,m}^{T}\right\} = \mathbf{R}_{\mathbf{r},m} \tag{3.45}$$

Projecting into the Reference Orbital-Nadir Frame $\mathcal{F}_o$,

$$\begin{aligned} \delta\mathbf{r}_{gv,o} &= \mathbf{C}_{o\oplus}\mathbf{C}_{v}^{T}\mathbf{C}_{mv}^{T}\delta\mathbf{r}_{gm,m} = \mathbf{C}_{om}\delta\mathbf{r}_{gm,m} \\ \mathbf{R}_{\mathbf{r},k} &= E\left\{\delta\mathbf{r}_{gv,o}\delta\mathbf{r}_{gv,o}^{T}\right\} = E\left\{\mathbf{C}_{om}\delta\mathbf{r}_{gm,m}\left[\mathbf{C}_{om}\delta\mathbf{r}_{gm,m}\right]^{T}\right\} = \mathbf{C}_{om}\mathbf{R}_{\mathbf{r},m}\mathbf{C}_{om}^{T} \end{aligned} \tag{3.46}$$

The values selected to populate $\mathbf{R}_{\mathbf{r},m}$ are detailed in Section 4.2.2.

3.4.7 Additional Tracking Frame

As described in [20] and shown in the equations of Appendix H, an additional orbital-nadir frame is defined called the “tracking frame”, $\mathcal{F}_\theta$, associated with the angular position of the chaser vehicle in the plane of the reference orbit. This frame can be seen in both Figure 3-2 and Figure 3-3. The relationship between $\mathcal{F}_\theta$ and $\mathcal{F}_o$ is,

$$\mathbf{C}_{\theta o} = \begin{bmatrix} \cos\theta_v & 0 & \sin\theta_v \\ 0 & 1 & 0 \\ -\sin\theta_v & 0 & \cos\theta_v \end{bmatrix} \tag{3.47}$$

This frame was introduced since the target-from-chaser relative cylindrical coordinates $\bar{\mathbf{r}}$ are mostly related to $\underline{\mathbf{r}}_{cv}$ expressed in this tracking frame $\mathcal{F}_\theta$. The general output equation used in the filter measurement update,

$$\mathbf{h}_{\mathbf{r}}(\mathbf{x},t) = \mathbf{r}_{gv,o} = \mathbf{r}_{cv,o} - \mathbf{C}_{o\oplus}\mathbf{C}_b^T\mathbf{r}_{cg,g} \tag{3.48}$$

is rewritten as,

$$\mathbf{r}_{gv,o} = \mathbf{C}_{\theta o}^T\mathbf{r}_{cv,\theta} - \mathbf{C}_{o\oplus}\mathbf{C}_b^T\mathbf{r}_{cg,g} \tag{3.49}$$

where $\mathbf{r}_{cv,\theta}$ is computed from $\bar{\mathbf{r}}$ and R_v as follows:

$$\mathbf{r}_{cv,\theta} = \begin{bmatrix} (R_v + r)\sin\theta \\ y_v + y \\ -(R_v + r)\cos\theta \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -R_v \end{bmatrix} = \begin{bmatrix} (R_v + r)\sin\theta \\ y_v + y \\ R_v - (R_v + r)\cos\theta \end{bmatrix} \tag{3.50}$$

4 Preliminary Design and Testing

4.1 Attitude Estimation (Inertial Context)

4.1.1 Simulation Setup

The simulation setup to design and test the Attitude MEKF is made up of the inertial truth model as outlined in Section 3.2.1, the attitude measurement model as outlined in Section 3.2.3, and the Attitude MEKF written in C code and integrated into the MATLAB/Simulink environment through a C MEX S-function. The Attitude MEKF implementation is described in detail in Appendix B, where the small rotation vector is used as the attitude deviation representation in the reset step. For the reference orientation, the quaternion parameterization is preferred over the DCM parameterization in this study for the following reasons:

i) The quaternion, $\mathbf{q}_{\text{ref}}$, is carried in the S-function state vector rather than the DCM, $\mathbf{C}_{\text{ref}}$. This reduces the size of the overall S-function state vector.

ii) Since the vector part of the measured error quaternion, $\tilde{\boldsymbol{\varepsilon}}_k$, is used during the update, using the quaternion $\mathbf{q}_{\text{ref}}$ reduces the number of parameterization conversions ($\mathbf{q} \rightarrow \mathbf{C}, \mathbf{C} \rightarrow \mathbf{q}$, etc.) within the filter algorithm.

The top-level of the simulation testbed used to design and test the Attitude MEKF is shown in Figure 4-1.

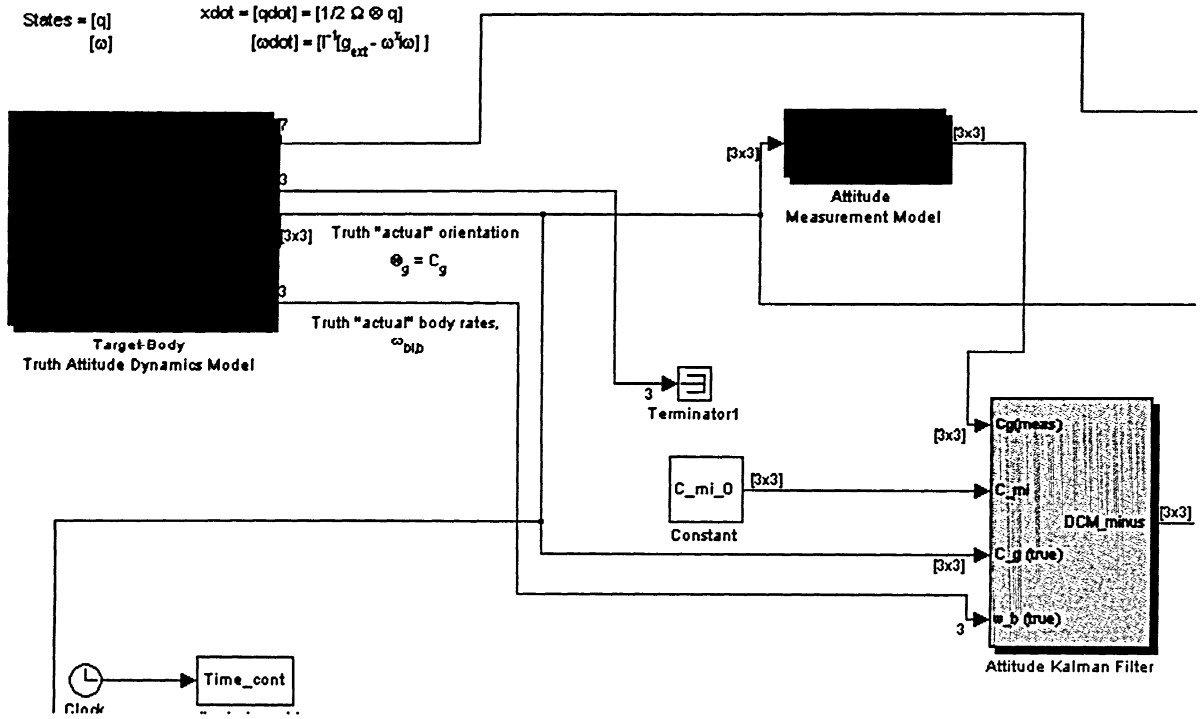


Figure 4-1: Top-Level inertial testbed for Attitude MEKF design and testing

There are many variables and parameters associated with the Attitude MEKF and the simulation testbed. To assess the performance of the Attitude MEKF, a *baseline setup* is established. Selection of the initialization variables and parameters should be chosen such that they represent a realistic scenario for the application being studied. All relevant variables and parameters for the baseline setup can be seen in Appendix C.

4.1.2 Covariance Specification and Filter Tuning

The noise covariance matrices, namely $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$, must be properly specified and *tuned* to achieve an acceptable numerical performance for a specified structure (i.e., totally specified except for $\mathbf{P}_{\Theta,0}$ and the time histories of $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$). Although the $\mathbf{P}_{\Theta,0}$ matrix is key in determining the initial transient performance of the filter, the primary concern is on finding an optimal balance between the time histories of $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$, as they dictate the *steady-state* performance of the filter [7].

The author initially became aware of the importance of covariance specification and filter tuning while conducting an initial test of the Attitude MEKF. For this initial test, both $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ were selected such that they corresponded exactly to the simulated process and measurement noise $\mathbf{w}_{\boldsymbol{\theta}}$ and $\mathbf{v}_{\boldsymbol{\theta},k}$. If no disturbance excitation is considered and $\mathbf{Q}_{\boldsymbol{\theta},c} = \mathbf{0}$, the result is shown in Figure 4-2 with the slow divergence of the filter as the roll error becomes arbitrarily large as time progresses, albeit unknown to the filter. This sort of performance is clearly unacceptable.

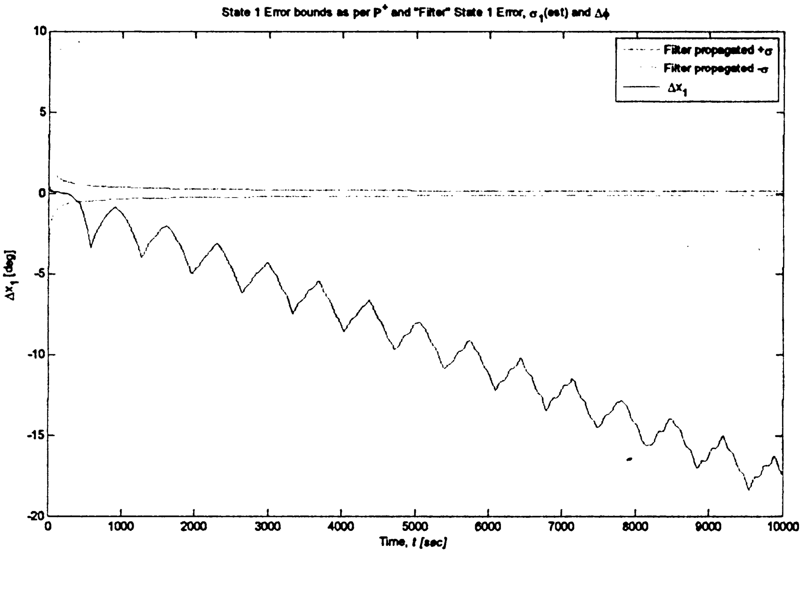


Figure 4-2: Roll error of $\hat{\mathbf{C}}^{+}_{\text{ref}}$ in an Attitude MEKF with $\mathbf{Q}_{\boldsymbol{\theta},c} = \mathbf{0}$

The problem with this divergent case was improper noise covariance specification and filter tuning. As expected, if $\mathbf{Q}_{\boldsymbol{\theta},c} = \mathbf{0}$, in essence it is assumed that "we have complete confidence that our dynamic system model within the filter is true", which is obviously false, and the filter essentially operates *open-loop* ignoring any measurements once $\mathbf{Q}_{\boldsymbol{\theta},c} = \mathbf{0}$ pulls the Kalman gain $\mathbf{K}_{\boldsymbol{\theta},k}$ down to negligible levels.

For more information on why proper specification of $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ is required for acceptable Attitude MEKF performance, see Appendix D. According to the latter reference, the balancing of $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ is often determined experimentally using automatic search methods or by manual calculation. In this study, the tuning process starts by choosing process noise covariance matrices based on the simulated noises injected into the system, then varying $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ via a *parametric sweep* through a range of values.

For the Attitude MEKF, the tuning procedure is as follows:

i) The baseline setup is used for all variables and parameters to *freeze* the truth/estimator structure (all specified except $\mathbf{P}_{\boldsymbol{\theta},0}$, $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$).

ii) $\mathbf{P}_{\boldsymbol{\theta},0}$ is set diagonal with all off-diagonal elements set to zero. The diagonal elements used are those in the baseline setup indicating that moderate uncertainty exists about the initial filter state estimate, $\hat{\mathbf{x}}_{\boldsymbol{\theta},0}$.

iii) $\mathbf{R}_{\boldsymbol{\theta},m}$ is set diagonal with all off-diagonal elements set to zero. The diagonal elements were initially set to match the noise $\mathbf{v}_{\boldsymbol{\theta},k}$ in the measurement process.

iv) $\mathbf{Q}_{\boldsymbol{\theta},c}$ is set diagonal with all off-diagonal elements set to zero. An arbitrarily small value of $(2\times10^{-4}\,\mathrm{N.m})^2$ was initially used for all diagonal elements to force the filter to pay attention to measurements over time.

v) The diagonal values of $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ are modulated via a *parametric-sweep* until the *local* best performance is found in the range of the sweep.

vi) The local best performance is assessed by running each $\mathbf{Q}_{\boldsymbol{\theta},c}$ and $\mathbf{R}_{\boldsymbol{\theta},m}$ combination for a fixed but lengthy simulation time until either steady-state performance is achieved or divergence is apparent.

vii) The $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$ combination that gives the local best performance is that which has the lowest value of the sample mean of the norm of the attitude errors.

Since $\mathbf{\Theta}_{\text{True}} = \mathbf{\Theta}_{\text{ref}} \oplus \delta\mathbf{\Theta}$, or in terms of the DCM, $\mathbf{C}_g = \delta\mathbf{C}^+\hat{\mathbf{C}}^+_{\text{ref}}$, then the actual *a posteriori* attitude error is,

$$\delta\mathbf{C}^+ = \mathbf{C}_g\hat{\mathbf{C}}^{+\ T}_{\text{ref}} \tag{4.1}$$

This can be converted to a three-parameter set such as the rotation vector

$$\mathbf{\Phi}(\delta\mathbf{C}^+) \tag{4.2}$$

and the norm $\|\mathbf{\Phi}(\delta\mathbf{C}^+)\|$ can be found during each filter sampling interval, T_s. The sample mean of this norm $\|\mathbf{\Phi}(\delta\mathbf{C}^+)\|$ can be found over the entire simulation interval. Let $\xi_k = \|\mathbf{\Phi}(\delta\mathbf{C}^+)\|_k$ and n is the total number of filter samples k. The sample mean of the norm of the attitude errors is then,

$$\bar{\xi}_k = \frac{\sum_{k=1}^{n}\xi_k}{n} \tag{4.3}$$

and the $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$ combination in the range of all $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$ combinations considered that gives,

$$\left[\bar{\xi}_k = \frac{\sum_{k=1}^{n}\xi_k}{n}\right]_{\min} \tag{4.4}$$

is selected to give the local best performance.

A more robust method such as a *Genetic Algorithm* that searches for a *global* best combination of $\mathbf{Q}_{\boldsymbol{\Theta},c}$ and $\mathbf{R}_{\boldsymbol{\Theta},m}$ for a specified truth/estimator structure could be used. This was the methodology in [26], but the *local* search technique described above is acceptable for the present study. The output of this search is the local best $\mathbf{Q}_{\boldsymbol{\Theta},c}$ and $\mathbf{R}_{\boldsymbol{\Theta},m}$ combination as shown in Appendix E.

4.1.3 Filter Performance Evaluation

In the inertial environment the MEKF state scope is:

$$\mathbf{x} = \mathbf{x}_{\boldsymbol{\Theta}} = \begin{bmatrix} \boldsymbol{\theta}_b \\ \boldsymbol{\omega}_{bi,b} \end{bmatrix} \tag{4.5}$$

with the states defined in Table 4-1. The measurement as viewed by the filter is related as follows:

$$\mathbf{z}_{\boldsymbol{\Theta},k} = \tilde{\boldsymbol{\varepsilon}}_k \tag{4.6}$$

The output vector used in the state update equation is as follows:

$$\mathbf{y}_{\boldsymbol{\Theta}} = \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x},t) = \boldsymbol{\varepsilon} = \frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \tag{4.7}$$

where

$$K = \sqrt{1 + \frac{1}{64}\theta^4} \tag{4.8}$$

	MEKF State Notation
$\mathbf{q}_b$	The "body-frame inertial orientation" parameterized as the quaternion as an alias for the common orientations of the Geometry Frame $\mathcal{F}_g$ and Mass-Center Frame $\mathcal{F}_c$. (Recall that $\mathcal{F}_c$ is defined, for convenience, to have the same orientation as $\mathcal{F}_g$, hence $\mathbf{C}_{cg} = \mathbf{1}$).
$\boldsymbol{\theta}$	Small rotation vector parameterization to describe the attitude deviation.
$\boldsymbol{\omega}_{bi,b}$	The "body-frame inertial angular velocity" expressed in the Body Frame, which is again an alias for the common inertial angular velocity of the Geometry Frame $\mathcal{F}_g$ and the Mass-Center Frame $\mathcal{F}_c$. (i.e. the vectors $\underset{\rightarrow}{\boldsymbol{\omega}}_{bi} \triangleq \underset{\rightarrow}{\boldsymbol{\omega}}_{ci} \equiv \underset{\rightarrow}{\boldsymbol{\omega}}_{gi}$ and column matrices $\boldsymbol{\omega}_{bi,b} = \boldsymbol{\omega}_{gi,g} = \boldsymbol{\omega}_{ci,c}$ have equivalent numerical values. $\boldsymbol{\omega}_{bi,b}$ is often abbreviated as $\boldsymbol{\omega}_b$).
	MEKF Measurement Notation
$\tilde{\boldsymbol{\varepsilon}}_k$	Vector part of the quaternion from measured orientation deviation.

Table 4-1: MEKF Notation (Inertial Context)

The measurement sensitivity Jacobian is as follows:

$$\mathbf{H}_{\boldsymbol{\theta},k} = \left[\frac{1}{2}\mathbf{1}_{3x3} \;\vdots\; \mathbf{0}_{3x3}\right]_{3x6} \tag{4.9}$$

Using the baseline setup case, the following results indicate that the Attitude MEKF performs relatively well. Quantifying the error similar to [12], the norm of the attitude errors are shown in Figure 4-3. After some initial transient behavior, the filter settles to a norm of the attitude errors within 1 deg on average.

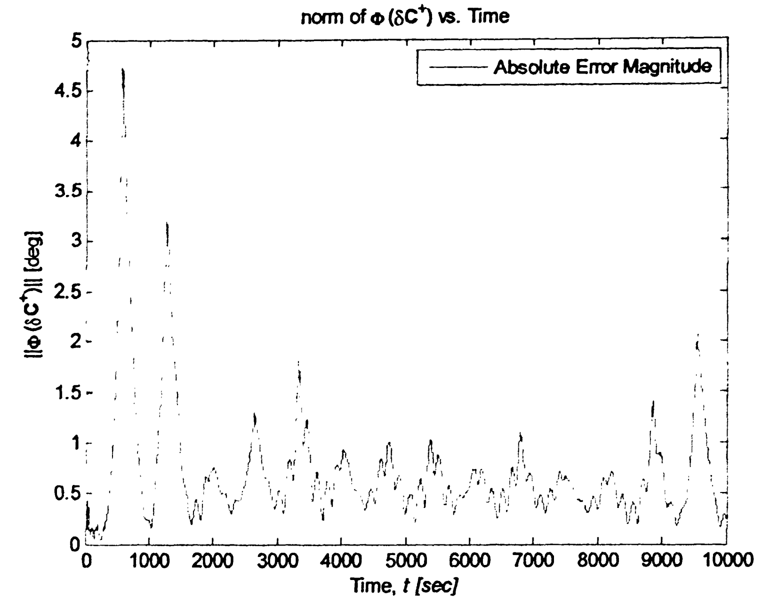


Figure 4-3: Norm of Attitude Errors (Baseline Setup)

The *a posteriori* attitude errors in terms of a RPY sequence, $RPY(\delta C^{+})$, vs. time are shown in Figure 4-4. Similar to the performance assessments used in [27], the 1-sigma covariance bounds are plotted along with the true error to assess whether the filter is performing as well as it "believes" it is performing. Integration of the Gaussian probability density function, to find the distribution function, indicates that there is a 68% probability that the Gaussian random variable is with one standard deviation ($\pm\sigma$) of the mean, 95% probability it is within two standard deviations of the mean, and 99% probability that it is within three standard deviations of the mean [27]. Figure 4-3 indicates the filter's representation of its own errors match reality as indicated by the fact that the true errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time. In Figure 4-5, it is seen that the angular rate errors are within approx. 0.01-0.02 deg/s and the filter's covariance in the state estimate error represents reality well.

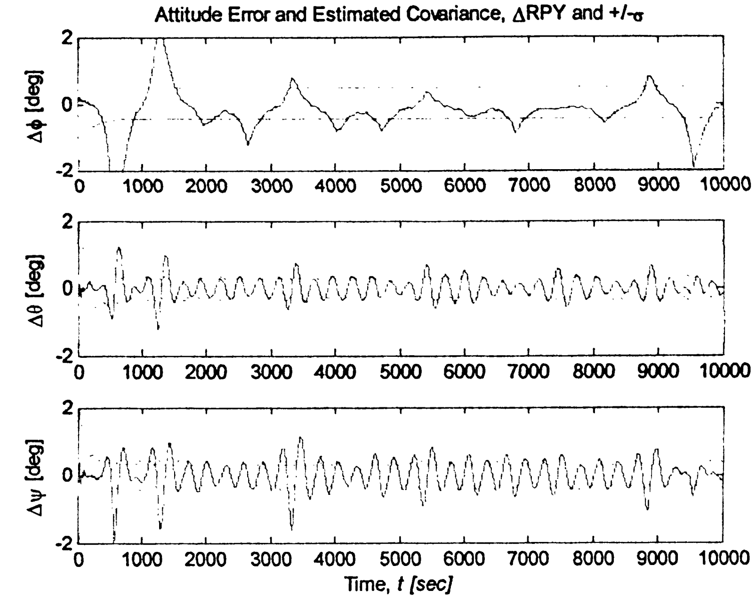


Figure 4-4: Attitude Errors parameterized as RPY (Baseline Setup)

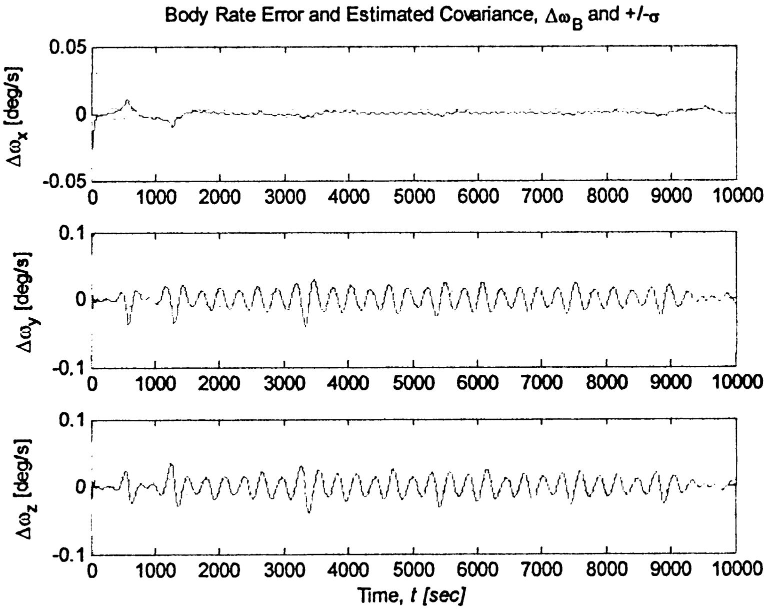


Figure 4-5: Body-Rate Errors (Baseline Setup)

Additional testing to check the integrity of the filter is as follows:

1. First, using the same basic baseline setup it is expected that if the measurement noise covariance matrix, $\mathbf{R}_{\boldsymbol{\theta},m}$, is relatively small then the filter assumes the measurements are "near-perfect" and should more or less discard any information coming from its embedded dynamic model.
2. Second, if the filter's dynamic model is slightly off in its representation of the target's second moment of inertia matrix, $\mathbf{I}$, it should still do a decent tracking job from the inclusion of measurement feedback (assuming the filter noise covariance has been specified such that $\mathbf{Q}_{\boldsymbol{\theta},c} \neq \mathbf{0}$).

<u>*INTEGRITY TEST 1*</u>

With the same basic setup as per Appendix C, the measurement noise covariance matrix is modified to put nearly total confidence in the measurements,

$$\mathbf{R}_{\boldsymbol{\theta},m} = \begin{bmatrix} \hat{\sigma}^2_{v_{xx}} & 0 & 0 \\ 0 & \hat{\sigma}^2_{v_{yy}} & 0 \\ 0 & 0 & \hat{\sigma}^2_{v_{zz}} \end{bmatrix} \text{ where } \hat{\sigma}^2_{v_{xx}} = \hat{\sigma}^2_{v_{yy}} = \hat{\sigma}^2_{v_{zz}} = 1\text{e-6 deg} \qquad (4.10)$$

With this modification, the filter behaves as expected with large values in $\mathbf{K}_{\boldsymbol{\theta},k}$, meaning the weighting is heavily biased towards the measurement. This can be seen in Figure 4-6 with the *a posteriori* estimates being very noisy and mostly overlapping the measured values.

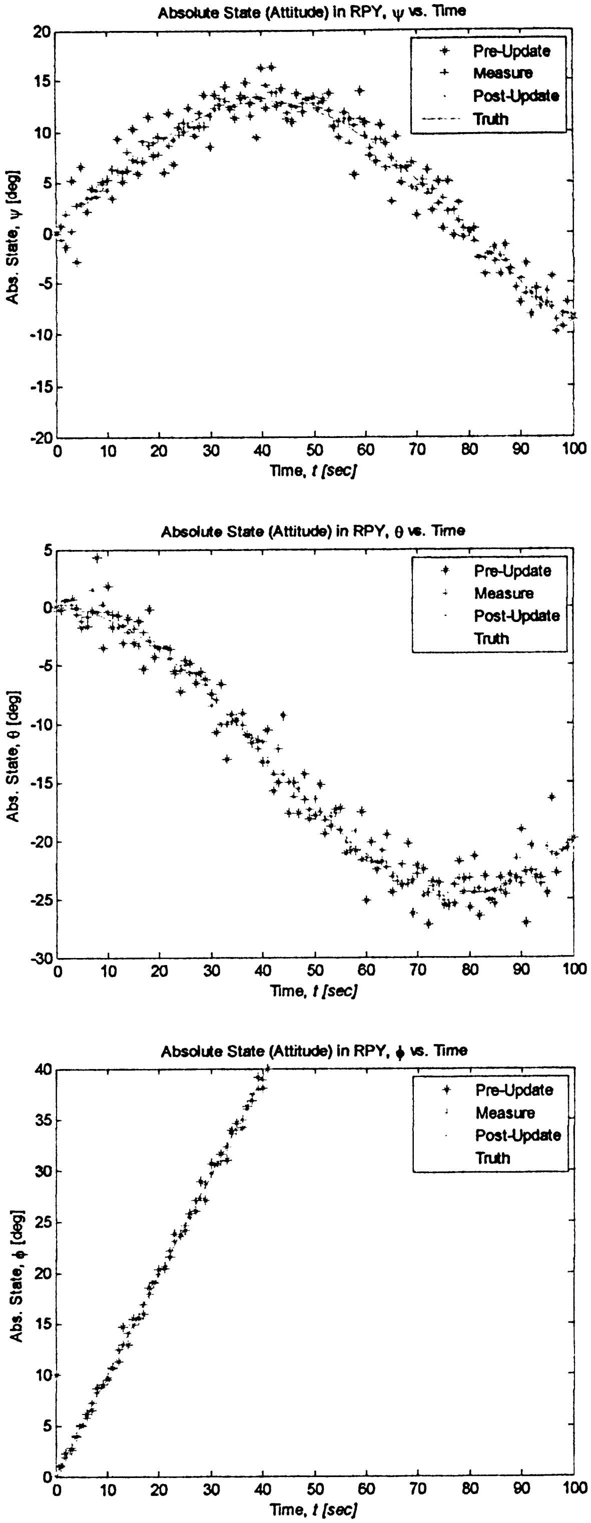


Figure 4-6: Attitude MEKF Integrity Test 1 – Near Perfect Measurement Behavior

INTEGRITY TEST 2

With the same basic setup as per Appendix C, only the filter's representation of the target inertia matrix is modified by adding/subtracting 1% of the minor axis inertia to all principal inertias,

$$\mathbf{I}=\begin{bmatrix}1462 & 0.0 & 0.0\\ 0.0 & 790 & 0.0\\ 0.0 & 0.0 & 511\end{bmatrix} \text{ and } \hat{\mathbf{I}}=\begin{bmatrix}1467 & 0.0 & 0.0\\ 0.0 & 785 & 0.0\\ 0.0 & 0.0 & 516\end{bmatrix} \text{ kg.m}^2 \qquad (4.11)$$

As seen in Figure 4-7, the filter does a decent job of estimating the attitude given that it is working with an incorrect inertia model of the target and the inertias are not being estimated online. Future work beyond the scope of this study is to add the inertia to the state so that it can be estimated online since in practical applications the inertia would be initialized in the filter with some uncertainty.

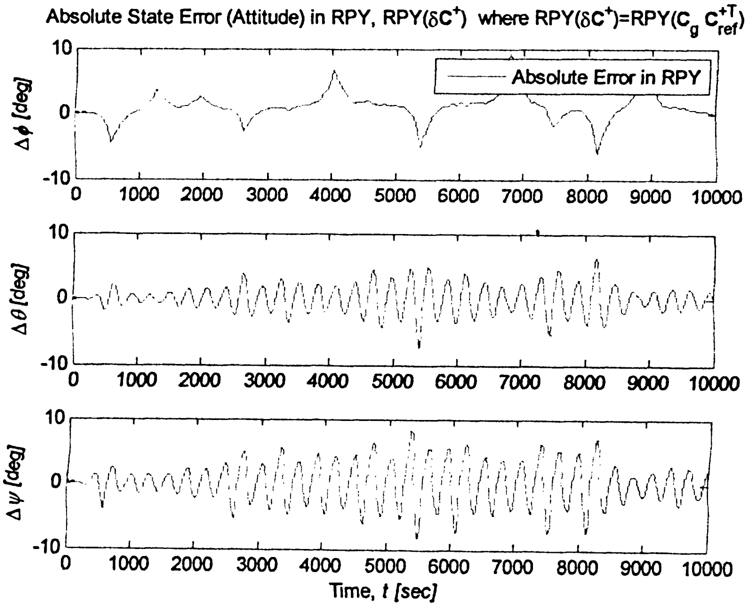


Figure 4-7: Attitude MEKF Integrity Test 2 – Uncertain Filter Inertia Model

Next, to partially characterize the robustness of the Attitude MEKF, a small sample *Monte-Carlo Analysis* is performed involving approx. 300 simulations where the initial filter estimates of attitude and body-rates are varied from the true initial attitude and body-rates. This was performed as follows:

i) The truth model variables and parameters were those specified in Appendix C.

ii) The filter initial estimates are determined in the following manner:

First, uniformly distributed random numbers are generated that are bounded between -1 and 1. Specifying these upper and lower bounds, $x_l = -1$ and $x_u = 1$, the random number is given as,

$$\chi = x_l + (x_u - x_l) \times rand() \tag{4.12}$$

where *rand()* is the standard Matlab uniformly distributed pseudo-random number function. The bounded random number is then multiplied by an amplitude factor that acts as the max allowable size of the error from the truth. For this study, the max allowable filter initialization error size factor was incremented for attitude and body-rates as follows,

$$\begin{array}{ll} \Delta\phi_{\max} = \Delta\theta_{\max} = \Delta\psi_{\max} & \text{in the interval [0,50] deg} \\ \Delta\omega_{b_x(\max)} = \Delta\omega_{b_y(\max)} = \Delta\omega_{b_z(\max)} & \text{in the interval [0,1] deg/s} \end{array} \tag{4.13}$$

The initial attitude estimate conditions were specified using

$$\begin{bmatrix} \hat{\phi} \\ \hat{\theta} \\ \hat{\psi} \end{bmatrix}_{t=0} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\phi_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \\ \Delta\theta_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \\ \Delta\psi_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \end{bmatrix} \tag{4.14}$$

The initial body-rate estimate conditions were specified using

$$\begin{bmatrix} \hat{\omega}_{bx} \\ \hat{\omega}_{by} \\ \hat{\omega}_{bz} \end{bmatrix}_{t=0} = \begin{bmatrix} \omega_{bx} \\ \omega_{by} \\ \omega_{bz} \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\omega_{b_x(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_y(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_z(\max)} \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \tag{4.15}$$

The results of the small sample Monte-Carlo Analysis are shown in the scatter plot in Figure 4-8. Each case was simulated for 10,000 seconds (170 minutes, or about 1.5 orbits at 500 km altitude). To condense the amount of information displayed, only the size of the initial attitude and body-rate errors are displayed on the abscissa and ordinate. The vertical axis normal to the abscissa, and ordinate indicates the average size (mean) of the attitude errors over the entire interval simulated. Performance was arbitrarily categorized as convergent if the mean of the norm of attitude errors was 2 deg or less. These cases are indicated by a "red" dot, • . Cases with a mean of the norm of attitude errors beyond this threshold are indicated by a "black" dot, •. Although this only displays insights into how well the filter can handle errors in its initial estimates for the dynamically evolving baseline case, it gives a rough order of magnitude of filter robustness to initialization errors. Beyond initialization errors of $\| \delta RPY \| \approx 35$ deg and $\| \Delta\omega_b \| \approx 1$ deg/s it is shown that convergence below 2 deg mean norm attitude error is sparse.

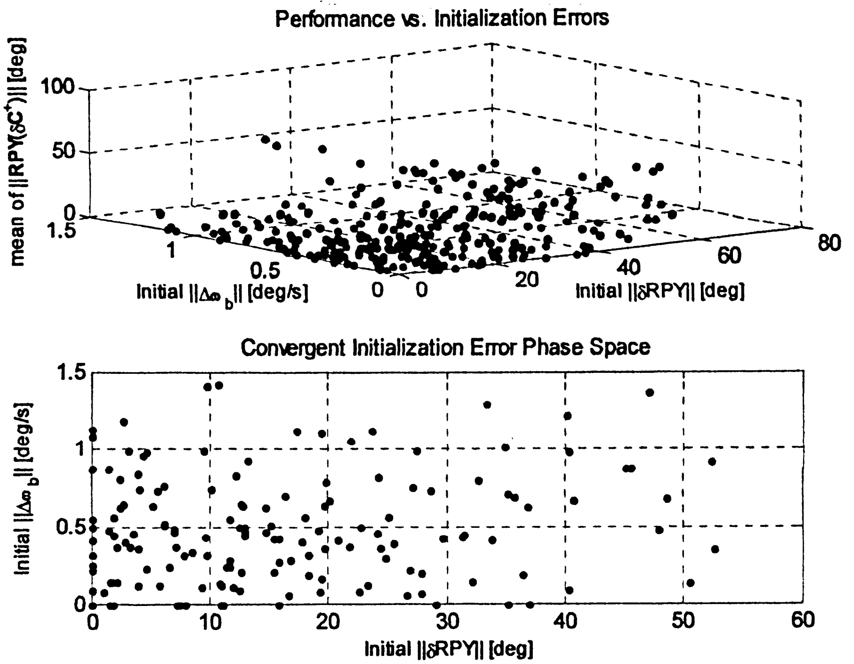


Figure 4-8: Attitude MEKF Robustness to Initial Estimate Errors

Similar to the results found in [28], in the top 3-D plot in Figure 4-8, certain cases of initialization result in unacceptable performance with large steady state error or divergent values as seen by the size of the mean of $\| RPY(\delta \mathbf{C}^{+}) \|$. In light of this, it may be possible to improve performance if other state variable measurements are available (target body-rates inferred by the vision system), further tuning of the filter for a better combination of $\mathbf{Q}_{\theta,c}$ and $\mathbf{R}_{\theta,m}$, or other *ad hoc* improvements. Since the performance is only acceptable within the bounds of the linear approximation implicit in the EKF, the tedious procedure of tuning $\mathbf{Q}_{\theta,c}$ and $\mathbf{R}_{\theta,m}$ may give some improvement in performance, although a high-order implementation such as the Unscented Kalman Filter (UKF) for spacecraft attitude estimation outlined in [12] should also be considered.

4.2 Motion Estimation (Inertial Context)

4.2.1 Simulation Setup

The simulation setup to design and test the dynamic motion filter is made up of the inertial truth models as outlined in Section 3.2.1 and Section 3.3.1, the measurement models as outlined in Section 3.2.3 and Section 3.3.3, and the dynamic motion filter written in C code and integrated into the MATLAB/Simulink environment through a C MEX S-function. The dynamic motion filter implementation used in the inertial context is described in detail in Appendix F.

The top-level of the simulation testbed used to design and test the dynamic motion filter is shown in Figure 4-9.

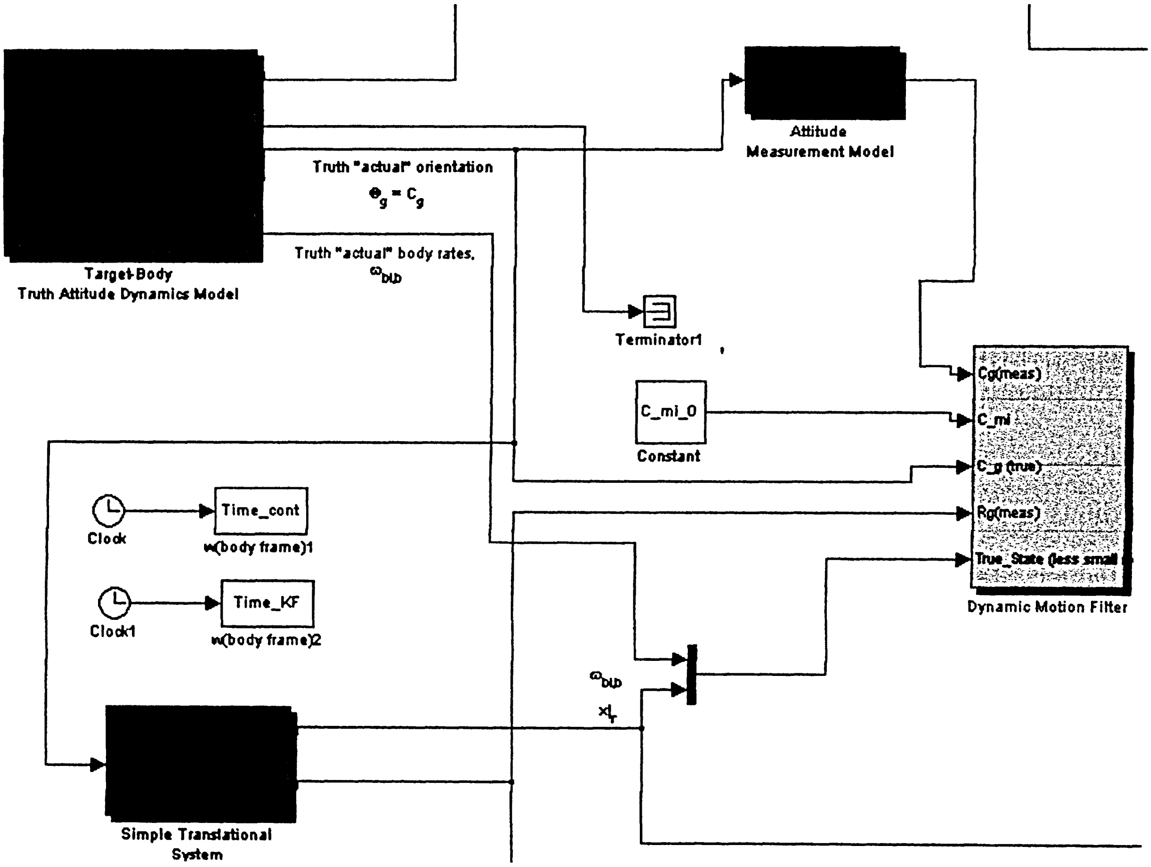


Figure 4-9: Top-Level Inertial Simulation testbed for Dynamic Motion Filter testing

There are many variables and parameters associated with the dynamic motion filter and the simulation testbed. To assess the performance of the dynamic motion filter, a *baseline setup* is established. All relevant variables and parameters for the baseline setup can be seen in Appendix C and Appendix G.

4.2.2 Covariance Specification and Filter Tuning

A discussion regarding covariance specification and filter tuning was presented in Section 4.1.2 and Appendix D. To avoid redundancy, the results are simply presented of a manual tuning effort for the dynamic motion filter in finding the *local* best combination of $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$. Additionally, the same local combination of $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$ is used that was associated with the orientation states as discussed in Section 4.1.2.

Again, the tuning process starts by choosing process noise covariance matrices based on the simulated noises injected into the system, then varying $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ via a *parametric sweep* through a range of values.

A much simplified version of the tuning procedure used in Section 4.2.2 is used here:

i) The baseline setup is used for all variables and parameters to *freeze* the truth/estimator structure (all specified except $\mathbf{P}_{\mathrm{r},0}$, $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$).

ii) $\mathbf{P}_{\mathrm{r},0}$ is set diagonal with all off-diagonal elements set to zero. The diagonal elements used are those in the baseline setup indicating that there is moderate uncertainty about the initial filter state estimate corresponding to the position $\mathbf{r}_c$ and quite confident about the other initial state estimates.

iii) $\mathbf{R}_{\mathrm{r},m}$ is set diagonal with all off-diagonal elements set to zero. The diagonal elements are initially set to match the noise characteristics $\mathbf{v}_{\mathrm{r},k}$ in the measurement process.

iv) $\mathbf{Q}_{\mathrm{r},c}$ is set diagonal with all off-diagonal elements set to zero. An arbitrarily small value of $(1\times10^{-4}\ \mathrm{N.m})^2$ is used for all diagonal elements to force the filter to pay attention to measurements over time.

v) The diagonal values of $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ are modulated via a *parametric-sweep* until the *local* best performance is found in the range of the sweep.

vi) The local best performance is assessed by running each $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ combination for a fixed but lengthy simulation time until either steady-state performance is achieved or divergence is apparent.

vii) The $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ combination that gives the local best performance is that which has the lowest value of the sample mean of the norm of the position errors.

The actual *a posteriori* position error is,

$$\Delta\mathbf{r}_c^+ = \mathbf{r}_c - \hat{\mathbf{r}}_c^+ \tag{4.16}$$

and the norm $\|\Delta\mathbf{r}_c^+\|_k$ can be found during each filter sampling interval, T_s. The sample mean of this norm $\|\Delta\mathbf{r}_c^+\|_k$ can be found over the entire simulation interval. Let $\gamma_k = \|\Delta\mathbf{r}_c^+\|_k$ and n is the total number of filter samples k. The sample mean of the norm of the position errors is then,

$$\bar{\gamma}_k = \frac{\sum_{k=1}^{n}\gamma_k}{n} \tag{4.17}$$

and the $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ combination in the range of all $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ combinations considered that gives

$$\left[\bar{\gamma}_k = \frac{\sum_{k=1}^{n} \gamma_k}{n}\right]_{\min} \tag{4.18}$$

is selected to give the local best performance.

Since setting the diagonal elements of $\mathbf{R}_{\mathrm{r},m}$ to match the noise $\mathbf{v}_{\mathrm{r},k}$ in the measurement process gave excellent results, this parameter was not modulated. Only a few iterations were performed by modulating $\mathbf{Q}_{\mathrm{r},c}$ and the local best $\mathbf{Q}_{\mathrm{r},c}$ and $\mathbf{R}_{\mathrm{r},m}$ combination are the values shown in Appendix G.

4.2.3 Filter Performance Evaluation

In the inertial environment, the dynamic motion filter state scope is,

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{\Theta} \\ \mathbf{x}_{\mathrm{r}} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_b \\ \boldsymbol{\omega}_{bi,b} \\ \hline \mathbf{r}_{ci,i} \\ \mathbf{v}_{ci,i} \\ \mathbf{r}_{cg,g} \end{bmatrix} \tag{4.19}$$

with the states defined in Table 4-2. The measurement as viewed by the filter is related as follows,

$$\mathbf{z}_k = \begin{bmatrix} \tilde{\boldsymbol{\varepsilon}}_k(\delta\hat{\mathbf{q}}_k) \\ \tilde{\mathbf{r}}_g \end{bmatrix} \tag{4.20}$$

The output vector used in the state update equation is the following,

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{\boldsymbol{\theta}} \\ \mathbf{y}_{\mathbf{r}} \end{bmatrix} = \begin{bmatrix} \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x},t) \\ \mathbf{h}_{\mathbf{r}}(\mathbf{x},t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\varepsilon} \\ \mathbf{r}_{g,i} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{K}\left(\dfrac{1}{2}\boldsymbol{\theta}\right) \\ \mathbf{r}_{c,i} - \mathbf{C}_b^T \mathbf{r}_{cg,g} \end{bmatrix} \tag{4.21}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\theta^4} \tag{4.22}$$

	Dynamic Motion Filter State Notation
$\mathbf{x}_{\boldsymbol{\theta}}$	Rotational partition of the dynamic motion filter state.
$\mathbf{x}_{\mathbf{r}}$	Translational partition of the dynamic motion filter state.
$\mathbf{q}_b$	The "body-frame inertial orientation" parameterized as the quaternion as an alias for the common orientations of the Geometry Frame $\mathcal{F}_g$ and Mass-Center Frame $\mathcal{F}_c$. (Recall that $\mathcal{F}_c$ is defined, for convenience, to have the same orientation as $\mathcal{F}_g$, hence $\mathbf{C}_{cg} = \mathbf{1}$).
$\boldsymbol{\theta}$	Small rotation vector parameterization to describe the attitude deviation.
$\boldsymbol{\omega}_{bi,b}$	The "body-frame inertial angular velocity" expressed in the Body Frame, which is again an alias for the common inertial angular velocity of the Geometry Frame $\mathcal{F}_g$ and the Mass-Center Frame $\mathcal{F}_c$. (i.e. the vectors $\underset{\rightarrow}{\boldsymbol{\omega}}_{bi} \triangleq \underset{\rightarrow}{\boldsymbol{\omega}}_{ci} \equiv \underset{\rightarrow}{\boldsymbol{\omega}}_{gi}$ and column matrices $\boldsymbol{\omega}_{bi,b} = \boldsymbol{\omega}_{gi,g} = \boldsymbol{\omega}_{ci,c}$ have equivalent numerical values. $\boldsymbol{\omega}_{bi,b}$ is often abbreviated as $\boldsymbol{\omega}_b$).
$\mathbf{r}_{ci,i}$	The inertial position of the Mass-Center Frame $\mathcal{F}_c$ origin, expressed in an inertial frame $\mathcal{F}_i$.
$\mathbf{v}_{ci,i}$	The inertial velocity of the Mass-Center Frame $\mathcal{F}_c$ origin, expressed in an inertial frame $\mathcal{F}_i$.
$\mathbf{r}_{cg,g}$	The mass-center offset or target satellite mass center location relative to the Geometry-Frame $\mathcal{F}_g$ origin, expressed in the Geometry-Frame $\mathcal{F}_g$.

	Dynamic Motion Filter Measurement Notation
$\tilde{\boldsymbol{\varepsilon}}_k$	Vector part of the quaternion from measured orientation deviation.
$\tilde{\mathbf{r}}_g$	Measured inertial position of the Geometry-Frame $\mathcal{F}_g$ origin, expressed in an inertial frame $\mathcal{F}_i$.
	Dynamic Motion Filter Output Notation
$\mathbf{y}_{\boldsymbol{\Theta}}$	Rotational partition of the output vector.
$\mathbf{y}_{\mathbf{r}}$	Translational partition of the output vector.
$\mathbf{C}_b$	The rotation matrix between $\mathcal{F}_i$ and $\mathcal{F}_b$, where $\mathcal{F}_b$ defines the same orientation of the target Geometry-Frame $\mathcal{F}_g$ and Center-of-Mass Frame $\mathcal{F}_c$.

Table 4-2: Dynamic Motion Filter Notation (Inertial Context)

The translational state is coupled to the rotational state through its dependence on the target reference attitude estimate. This can be seen in the measurement sensitivity Jacobian as follows,

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} = \begin{bmatrix} \mathbf{H}_{\boldsymbol{\Theta},k} \\ \mathbf{H}_{\mathbf{r},k} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_{\text{ref}}^T \mathbf{r}_{cg,g}^{\times} & \mathbf{0} & \mathbf{1} & \mathbf{0} & -\mathbf{C}_b^T \end{bmatrix} \tag{4.23}$$

where $\mathbf{C}_{\text{ref}}^T = \hat{\mathbf{C}}_{\text{ref}}^{T\,-}$ is coupled to the translational states.

Using the baseline setup cases of Appendix C and Appendix G, the following results indicate that the dynamic motion filter performs well. In this preliminary section, only the results of the translational states are shown. The norm of the position errors are shown in Figure 4-10. After some initial transient behavior, the filter settles to a norm of the position errors within 2 cm.

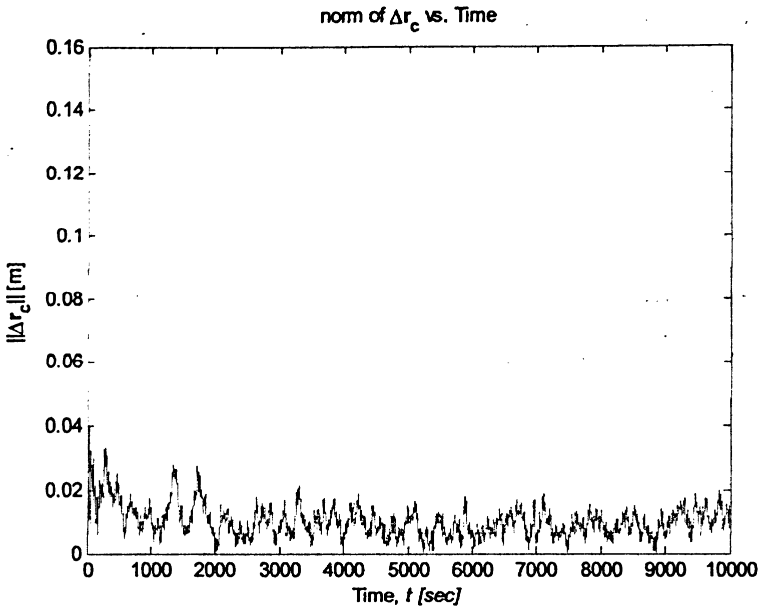


Figure 4-10: Norm of Inertial Position Errors (Baseline Setup)

The *a posteriori* position errors vs. time are shown in Figure 4-11. The figure indicates that the filter's own representation of its errors match reality as indicated by the fact that the true errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time. In Figure 4-12, it is seen that the velocity errors are within approx. 0.01 cm/s and the filter's covariance in the state estimate error represents reality well.

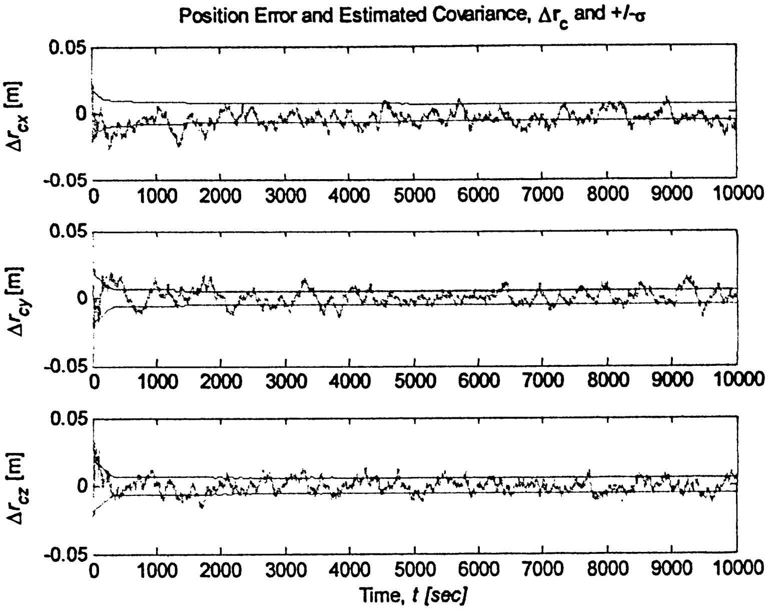


Figure 4-11: Position Errors vs. Time (Baseline Setup)

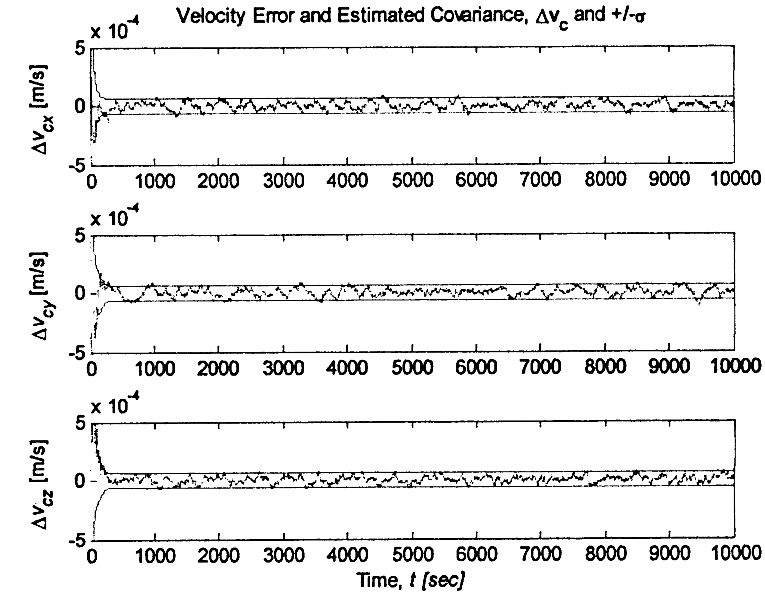


Figure 4-12: Velocity Errors vs. Time (Baseline Setup)

The CoM offset errors vs. time can be seen in Figure 4-13 and the norm of the CoM offset errors in Figure 4-14.

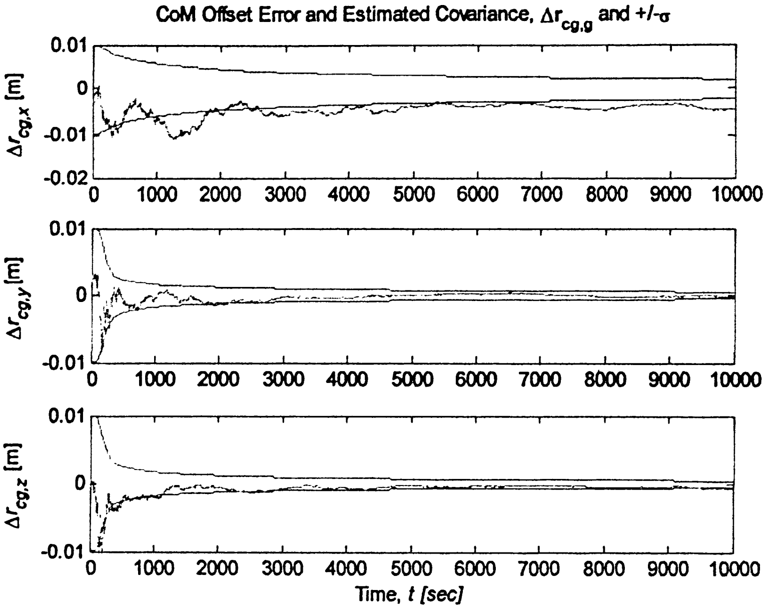


Figure 4-13: CoM Offset Errors vs. Time (Baseline Setup)

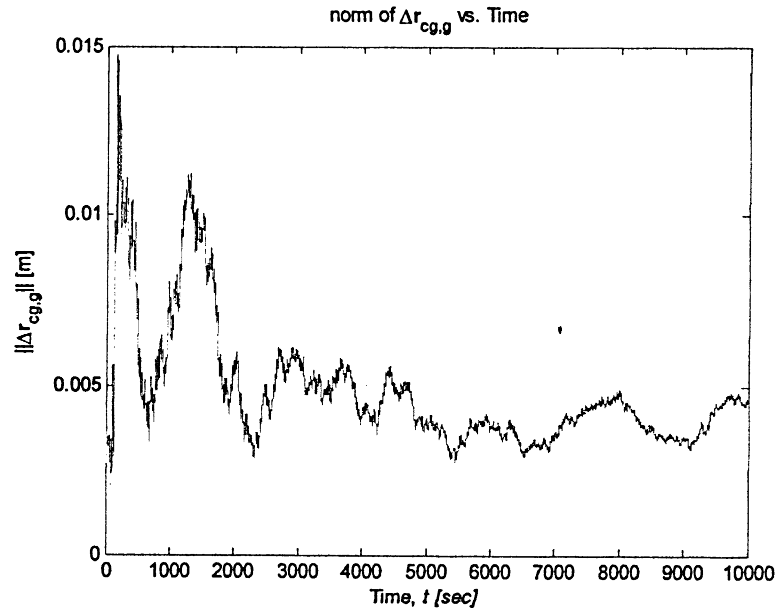


Figure 4-14: Norm of CoM Offset Errors (Baseline Setup)

Note that the *y* and *z* components of the estimated $\hat{\mathbf{r}}_{cg,g}$ in Figure 4-13 are good but the x component is hovering around the lower edge of the covariance bounds. This may indicate that additional filter tuning is required or that a possible observability issue exists. However, the covariance in the state estimate error, **P**, is well behaved in that it does not grow arbitrarily large. The behavior of the translation state and attitude state will be further investigated in Section 5 when the dynamic motion filter is fully assessed in an inertial simulation test environment and then in a more realistic orbital simulation test environment.

4.3 Motion Estimation (Orbital Context)

To emulate the relative motion of the chaser vehicle and target vehicle during a rendezvous proximity operation, a representative scenario of the relative motion is required. The hybrid pose/motion estimation system is assumed to be utilized when the distance from chaser-to-target is approximately 100m or less. In all cases, it is expected that the chaser is approaching the target from behind in roughly the same orbital plane. Therefore, a *Hohmann transfer* scenario is used to generate the representative relative motion for simulation testing. Figure 4-15 shows the basic Hohmann scenario,

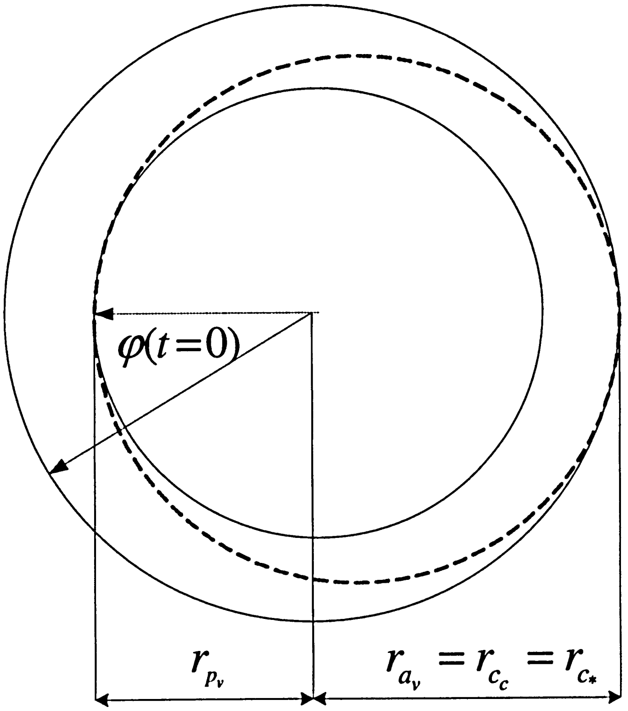


Figure 4-15: Hohmann Transfer for Relative Motion Scenario

Assume the chaser transfer ellipse has the following perigee/apogee altitudes,

$$\begin{aligned} h_{p_v} &= 500 \text{ km} \\ h_{a_v} &= 500.04 \text{ km} \end{aligned} \tag{4.24}$$

with the following properties,

$$\begin{aligned} a_{\text{Transfer}} &= \frac{r_{p_v} + r_{a_v}}{2} = 6878.02 \text{ km} \\ T_{\text{Transfer}} &= 2\pi\sqrt{\frac{a_{\text{Transfer}}{}^3}{\mu_\oplus}} = 5676.836 \text{ sec} \\ T_{\frac{1}{2}\text{Transfer}} &= 2838.418 \text{ sec} \end{aligned} \tag{4.25}$$

The mean motion of the target is,

$$\omega_* = \omega_c = \sqrt{\frac{\mu_\oplus}{r_{c_c}{}^3}} = 0.0011068062 \text{ rad/s} = 3.8049 \text{ deg/min} \tag{4.26}$$

During $T_{\frac{1}{2}\text{Transfer}}$ (180 deg swoop) the target moves through

$$\Delta\alpha = \omega_c . T_{\frac{1}{2}\text{Transfer}} = 3.141578950 = 179.99921489 \text{ deg} \tag{4.27}$$

To "near perfectly" dock with the target, the chaser should have an initial starting phase of (just behind of the target)

$$\varphi = 180 - \Delta\alpha \approx 0.000785106 \text{deg} \tag{4.28}$$

The orbital elements related to the relative motion Hohmann transfer scenario are seen in Table 4-3.

ORBITAL SCENARIO CHARACTERISTICS		
REFERENCE O-N FRAME	TARGET	CHASER
$h_p =$ 500.04 km $h_a =$ 500.04 km $\Omega =$ 153.4 deg $i =$ 36.7 deg $\omega =$ 0.0 deg $l_0 =$ 0.000785106 deg	$h_p =$ 500.04 km $h_a =$ 500.04 km $\Omega =$ 153.4 deg $i =$ 36.7 deg $\omega =$ 0.0 deg $l_0 =$ 0.000785106 deg	$h_p =$ 500 km $h_a =$ 500.04 km $\Omega =$ 153.4 deg $i =$ 36.7 deg $\omega =$ 0.0 deg $l_0 =$ 0.0 deg

Table 4-3: Relative Motion Hohmann Transfer Scenario Initialization

4.3.1 Simulation Setup

The simulation setup to design and test the dynamic motion filter is made up of the truth models as outlined in Section 3.2.1 and Section 3.4.1, the measurement models as outlined in Section 3.2.3 and Section 3.4.5, and the dynamic motion filter written in C code and integrated into the MATLAB/Simulink environment through a C MEX S-function. The dynamic motion filter implementation used in the orbital context is described in detail in Appendix H.

The top-level of the simulation testbed used to design and test the dynamic motion filter is shown in Figure 4-16.

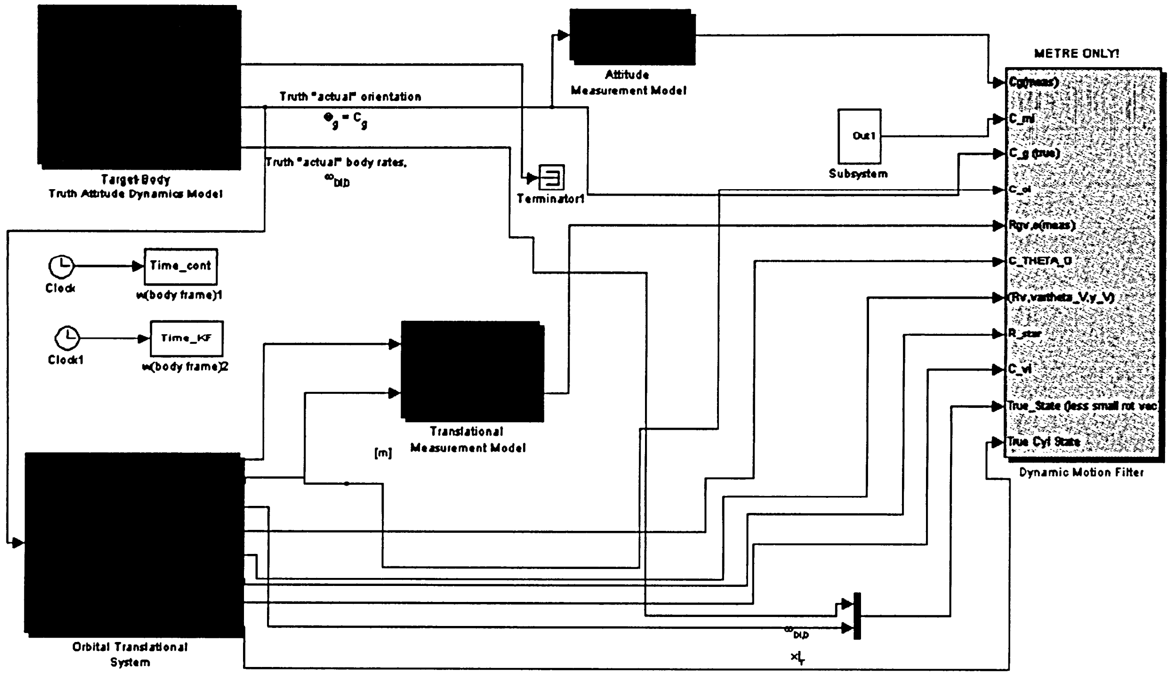


Figure 4-16: Top-Level Orbital Simulation testbed for Dynamic Motion Filter testing

There are many variables and parameters associated with the dynamic motion filter and the simulation testbed. To assess the performance of the dynamic motion filter a *baseline setup* is established. All relevant variables and parameters for the baseline setup can be seen in Appendix C and Appendix I.

4.3.2 Covariance Specification and Filter Tuning

For the dynamic motion filter in the orbital context, the covariance specification and filter tuning procedure remains identical to that outlined in Section 4.2.2. Only a few iterations were performed by modulating $\mathbf{Q}_{r,c}$, and the *local* best $\mathbf{Q}_{r,c}$ and $\mathbf{R}_{r,m}$ combination are the values shown in Appendix I.

4.3.3 Filter Performance Evaluation

In the orbital environment, the dynamic motion filter state scope is,

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{\theta} \\ \mathbf{x}_{\mathbf{r}} \end{bmatrix} = \left[\begin{array}{c} \boldsymbol{\theta}_b \\ \boldsymbol{\omega}_{bi,b} \\ \hline \bar{\mathbf{r}} \\ \bar{\mathbf{v}} \\ \mathbf{r}_{cg,g} \end{array}\right] \tag{4.29}$$

with the states defined in Table 4-4. The measurement as viewed by the filter is related as follows:

$$\mathbf{z}_k = \begin{bmatrix} \tilde{\boldsymbol{\varepsilon}}_k(\delta\hat{\mathbf{q}}_k) \\ \tilde{\mathbf{r}}_{gv,o} \end{bmatrix} \tag{4.30}$$

The output vector used in the state update equation is as follows:

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{\theta} \\ \mathbf{y}_{\mathbf{r}} \end{bmatrix} = \begin{bmatrix} \mathbf{h}_{\theta}(\mathbf{x},t) \\ \mathbf{h}_{\mathbf{r}}(\mathbf{x},t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\varepsilon} \\ \mathbf{r}_{gv,o} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{K}\left(\dfrac{1}{2}\boldsymbol{\theta}\right) \\ \mathbf{C}_{\theta o}^{T}\mathbf{r}_{cv,\theta} - \mathbf{C}_{o\oplus}\mathbf{C}_{b}^{T}\mathbf{r}_{cg,g} \end{bmatrix} \tag{4.31}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\theta^4} \tag{4.32}$$

	Dynamic Motion Filter Notation
$\mathbf{x}_{\theta}$	Rotational partition of the dynamic motion filter state.
$\mathbf{x}_{\mathbf{r}}$	Translational partition of the dynamic motion filter state.
$\mathbf{q}_b$	The "body-frame inertial orientation" parameterized as the quaternion as

	an alias for the common orientations of the Geometry Frame $\mathscr{F}_g$ and Mass-Center Frame $\mathscr{F}_c$. (Recall that $\mathscr{F}_c$ is defined, for convenience, to have the same orientation as $\mathscr{F}_g$, hence $\mathbf{C}_{cg} = \mathbf{1}$).
$\boldsymbol{\theta}$	Small rotation vector parameterization to describe the attitude deviation.
$\boldsymbol{\omega}_{bi,b}$	The "body-frame inertial angular velocity" expressed in the body frame, which is again an alias for the common inertial angular velocity of Geometry Frame $\mathscr{F}_g$ and Mass-Center Frame $\mathscr{F}_c$. (i.e. the vectors $\underrightarrow{\boldsymbol{\omega}}_{bi}$ □ $\underrightarrow{\boldsymbol{\omega}}_{ci} \equiv \underrightarrow{\boldsymbol{\omega}}_{gi}$ and column matrices $\boldsymbol{\omega}_{bi,b} = \boldsymbol{\omega}_{gi,g} = \boldsymbol{\omega}_{ci,c}$ have equivalent numerical values. $\boldsymbol{\omega}_{bi,b}$ is often abbreviated as $\boldsymbol{\omega}_b$).
$\bar{\mathbf{r}} = \begin{bmatrix} R_* \theta \\ y \\ -r \end{bmatrix}$	The small relative cylindrical coordinates Cartesian-like vector-matrix of the target relative to the chaser, expressed in the Reference Orbital-Nadir Frame $\mathscr{F}_o$.
$\bar{\mathbf{v}} = \begin{bmatrix} R_* \dot{\theta} \\ \dot{y} \\ -\dot{r} \end{bmatrix}$	The small relative cylindrical coordinates Cartesian-like vector-matrix *rates* of the target relative to the chaser, expressed in the Reference Orbital-Nadir Frame $\mathscr{F}_o$.
$\mathbf{r}_{cg,g}$	The mass-center offset or target satellite mass center location relative to the Geometry-Frame $\mathscr{F}_g$ origin, expressed in the Geometry-Frame $\mathscr{F}_g$.
	Dynamic Motion Filter Measurement Notation
$\tilde{\boldsymbol{\varepsilon}}_k$	Vector part of the quaternion from measured orientation deviation.
$\tilde{\mathbf{r}}_{gv,o}$	Measured position of the Geometry-Frame $\mathscr{F}_g$ origin relative to the Chaser Vehicle Frame $\mathscr{F}_v$ origin, expressed in the Reference Orbital-Nadir Frame $\mathscr{F}_o$.
	Dynamic Motion Filter Output Notation
$\mathbf{y}_{\boldsymbol{\Theta}}$	Rotational partition of the output vector.
$\mathbf{y}_{\mathbf{r}}$	Translational partition of the output vector.
$\mathbf{C}_b$	The rotation matrix between $\mathscr{F}_{\oplus}$ and $\mathscr{F}_b$, where $\mathscr{F}_b$ defines the same

	orientation of the target Geometry-Frame $\mathcal{F}_g$ and Center-of-Mass Frame $\mathcal{F}_c$.
$\mathbf{r}_{cv,\theta}$	Position of the target Mass-Center Frame $\mathcal{F}_c$ origin relative to the Chaser Vehicle Frame $\mathcal{F}_v$ origin, expressed in the Tracking Frame $\mathcal{F}_\theta$.
$\mathbf{C}_{o\oplus}$	The rotation matrix between $\mathcal{F}_\oplus$ and $\mathcal{F}_o$.
$\mathbf{C}_{\theta o}$	The rotation matrix between $\mathcal{F}_o$ and $\mathcal{F}_\theta$.

Table 4-4: Dynamic Motion Filter Notation (Orbital Context)

The translational state is coupled to the rotational state through its dependence on the target reference attitude estimate. This can be seen in the measurement sensitivity Jacobian as follows:

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} = \begin{bmatrix} \mathbf{H}_{\Theta,k} \\ \mathbf{H}_{\mathbf{r},k} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{2}\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_{o\oplus}\mathbf{C}_{\text{ref}}^T\mathbf{r}_{cg,g}^{\times} & \mathbf{0} & \mathbf{C}_{\theta o}^T \begin{bmatrix} \frac{(R_v + r)}{R_*}\cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \frac{(R_v + r)}{R_*}\sin\theta & 0 & \cos\theta \end{bmatrix} & \mathbf{0} & -\mathbf{C}_{o\oplus}\mathbf{C}_{\text{ref}}^T \end{bmatrix} \tag{4.33}$$

where $\mathbf{C}_{\text{ref}}^T = \hat{\mathbf{C}}_{\text{ref}}^{T\,-}$ is the coupling term.

Using the baseline setup cases of Appendix C and Appendix I, the following results indicate that the dynamic motion filter performs well. In this preliminary section, only the results of the translational states are shown. The simulation is run for 3000 seconds, just beyond the point of intersection between the target and chaser at $T_{\frac{1}{2}\text{Transfer}}$. In Figure 4-17, the true and *a posteriori* estimate of the relative cylindrical coordinates is shown, along with the error in the relative cylindrical coordinates estimate in Figure 4-18. The radial and out-of-plane errors are within 1-2 cm. The angular error is almost negligible. The norm of the position errors in Cartesian-like form are shown in Figure 4-19. After some initial transient behavior, the filter settles to a norm of the Cartesian-like position errors within 3-4 cm.

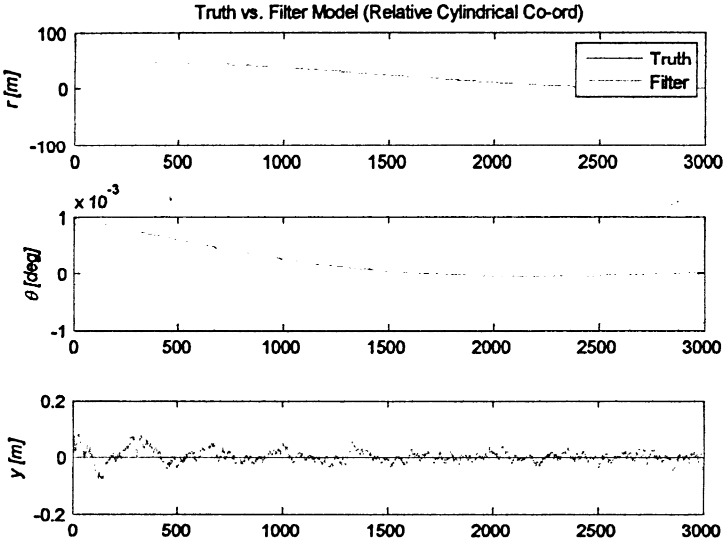


Figure 4-17: Relative Cylindrical Coordinates Baseline Case

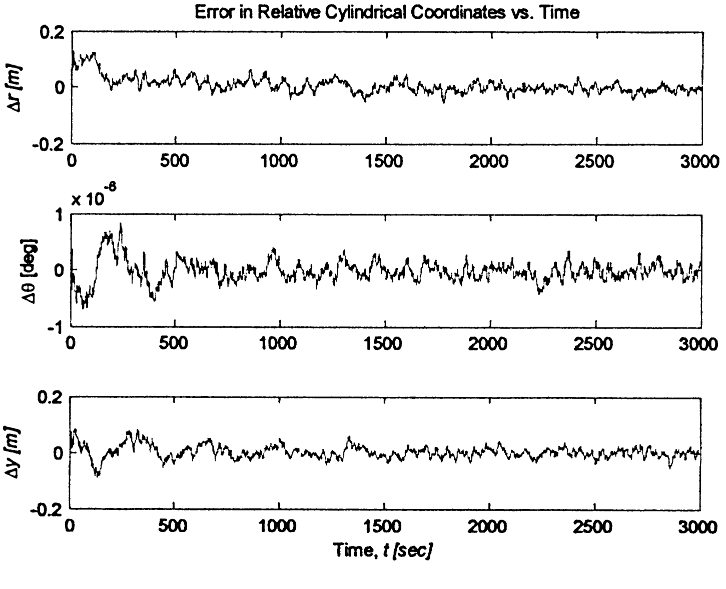


Figure 4-18: Relative Cylindrical Coordinate Errors vs. Time (Baseline Setup)

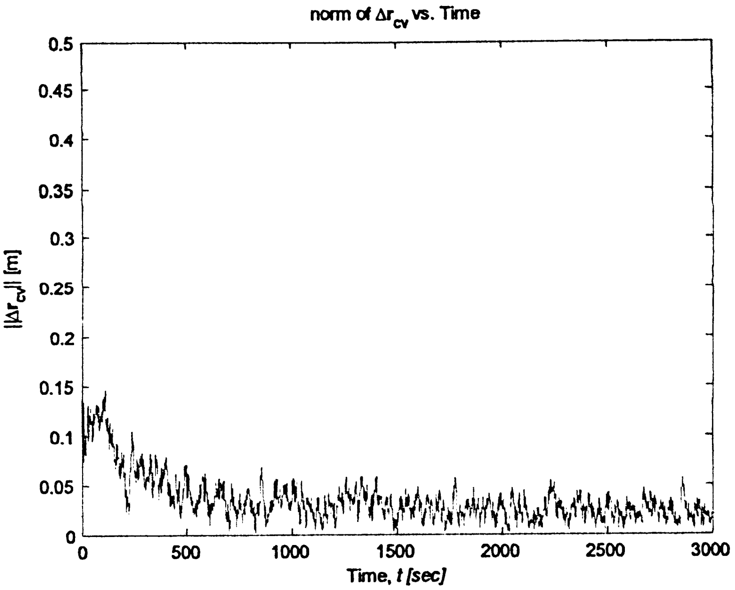


Figure 4-19: Norm of Cartesian-like Position Errors (Baseline Setup)

The *a posteriori* relative position errors vs. time in Cartesian-like form are shown in Figure 4-20. The figure indicates the filter's own representation of its errors match reality as indicated by the fact that the true errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time.

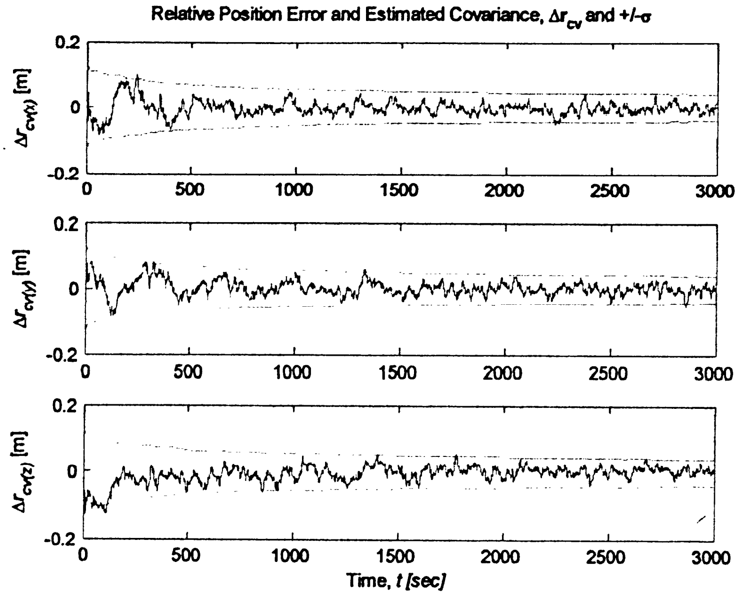


Figure 4-20: Relative Cartesian-like Position Errors vs. Time (Baseline Setup)

The CoM offset errors vs. time can be seen in Figure 4-21 and the norm of the CoM offset errors in Figure 4-22. Again, the true CoM offset errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time.

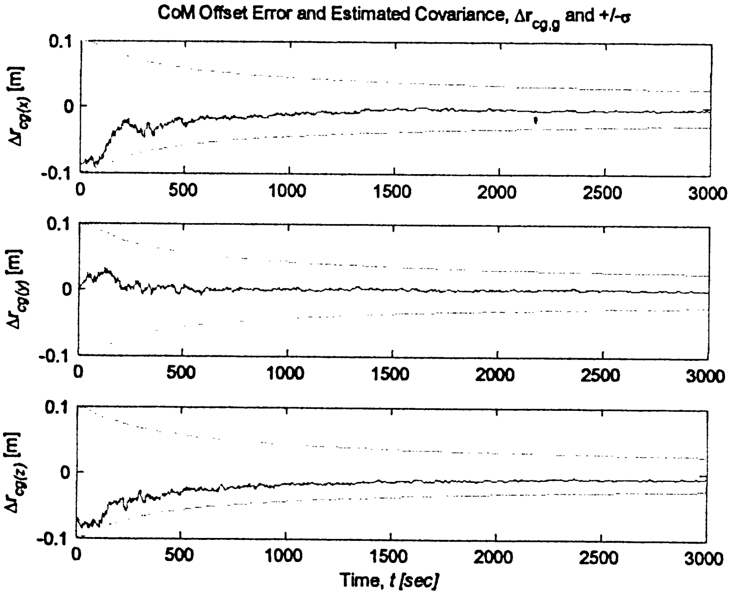


Figure 4-21: CoM Offset Errors vs. Time (Baseline Setup)

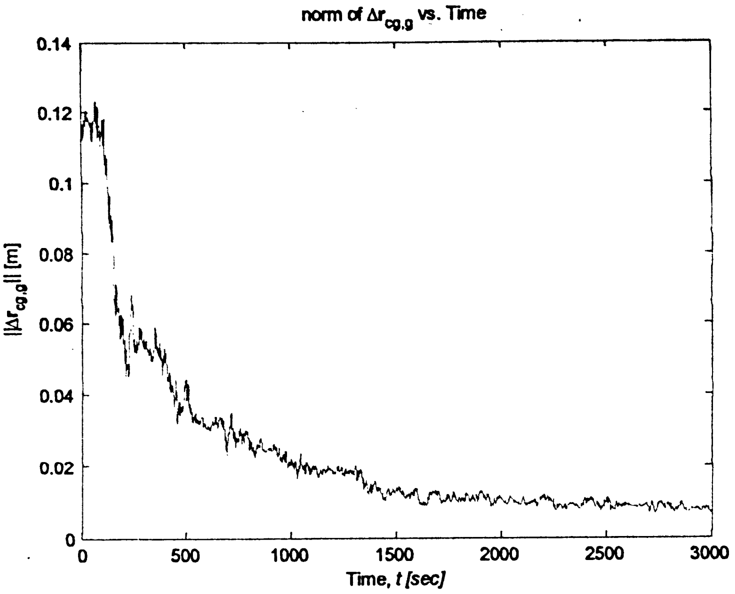


Figure 4-22: Norm of CoM Offset Errors (Baseline Setup)

The behavior of the complete dynamic state will be further investigated in Section 5 when the dynamic motion filter is fully assessed in the orbital simulation test environment.

5 Dynamic Motion Filter Results

In this section, the performance of the dynamic motion filter is assessed in an inertial environment and in an orbital environment.

5.1 *Dynamic Motion Filter Performance - Inertial Environment*

In the inertial environment, the dynamic motion filter state scope is,

$$\mathbf{x}=\begin{bmatrix}\mathbf{x}_{\boldsymbol{\theta}}\\ \mathbf{x}_{\mathbf{r}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{\theta}_{b}\\ \underline{\boldsymbol{\omega}_{bi,b}}\\ \mathbf{r}_{cl,l}\\ \mathbf{v}_{ci,i}\\ \mathbf{r}_{cg,g}\end{bmatrix} \tag{5.1}$$

The measurement as viewed by the filter is related as follows:

$$\mathbf{z}_{k}=\begin{bmatrix}\tilde{\boldsymbol{\varepsilon}}_{k}(\delta\hat{\mathbf{q}}_{k})\\ \tilde{\mathbf{r}}_{g}\end{bmatrix} \tag{5.2}$$

The output vector used in the state update equation is as follows:

$$\mathbf{y}=\begin{bmatrix}\mathbf{y}_{\boldsymbol{\theta}}\\ \mathbf{y}_{\mathbf{r}}\end{bmatrix}=\begin{bmatrix}\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x},t)\\ \mathbf{h}_{\mathbf{r}}(\mathbf{x},t)\end{bmatrix}=\begin{bmatrix}\boldsymbol{\varepsilon}\\ \mathbf{r}_{g,i}\end{bmatrix}=\begin{bmatrix}\dfrac{1}{K}\left(\dfrac{1}{2}\boldsymbol{\theta}\right)\\ \mathbf{r}_{c,i}-\mathbf{C}_{b}^{T}\mathbf{r}_{cg,g}\end{bmatrix} \tag{5.3}$$

where,

$$K=\sqrt{1+\frac{1}{64}\theta^{4}} \tag{5.4}$$

All variables of the above equations were previously defined in Table 4-2 of Section 4.2.3 and will not be re-defined for brevity. A graphic of the inertial operating environment for dynamic motion filter testing is shown in Figure 5-1.

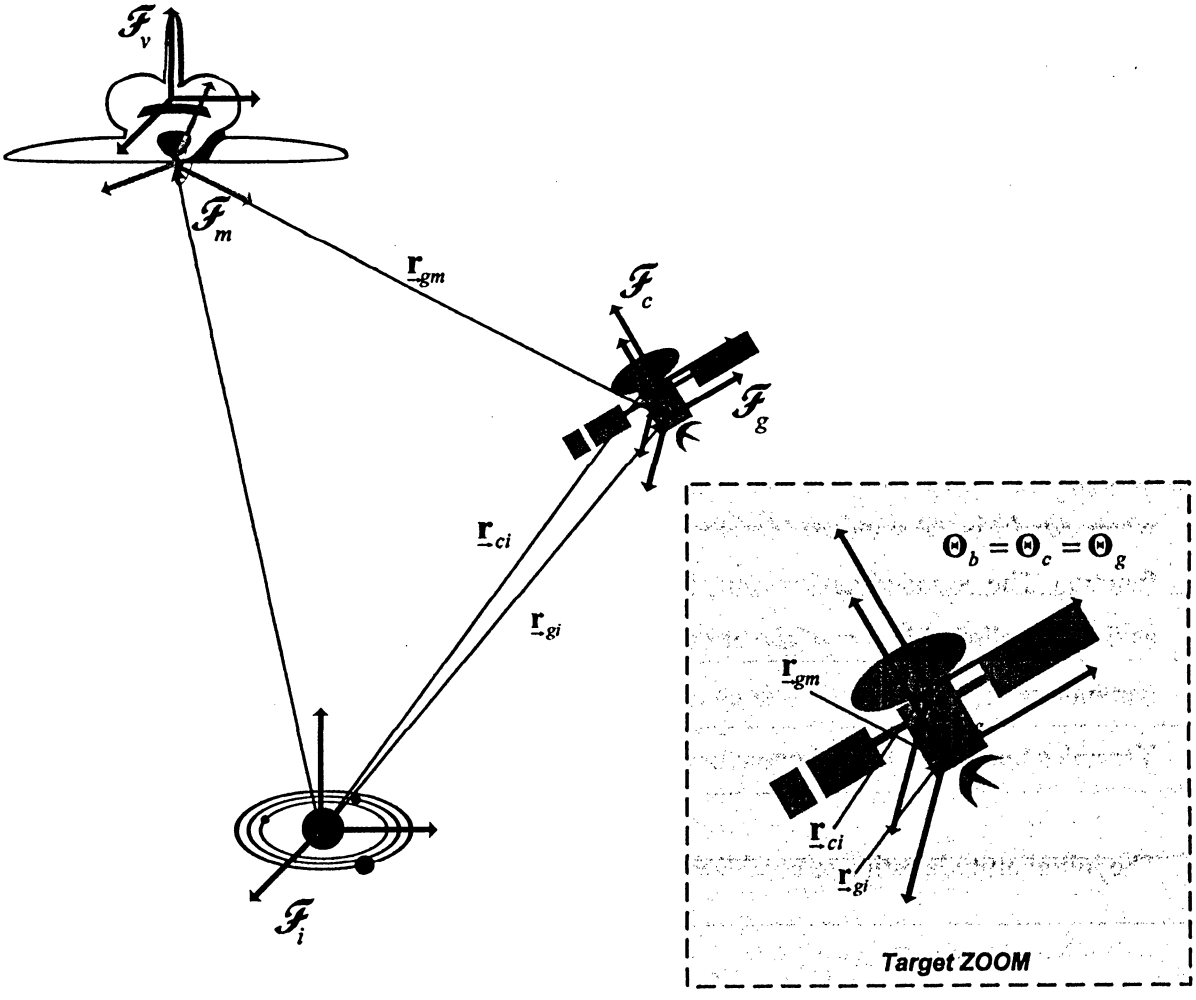

$\mathcal{F}_v$
$\mathcal{F}_m$
$\underset{\rightarrow}{\mathbf{r}}_{gm}$
$\mathcal{F}_c$
$\mathcal{F}_g$
$\underset{\rightarrow}{\mathbf{r}}_{ci}$
$\underset{\rightarrow}{\mathbf{r}}_{gi}$
$\mathcal{F}_i$
$\Theta_b = \Theta_c = \Theta_g$
$\underset{\rightarrow}{\mathbf{r}}_{gm}$
$\underset{\rightarrow}{\mathbf{r}}_{ci}$
$\underset{\rightarrow}{\mathbf{r}}_{gi}$
Target ZOOM

Figure 5-1: Inertial Environment Frames and Vectors

To assess the performance of the dynamic motion filter in the inertial environment, a number of simulations are run with random but bounded truth initial conditions, $\{\mathbf{x}_0\}$, and random but bounded error in the filter initial conditions, $\{\Delta\mathbf{x}_0 = \hat{\mathbf{x}}_0 - \mathbf{x}_0\}$. Each sample simulation will be run for a sufficiently long period of time to assess the occurrence of good performance (convergence within a specified level), bad performance (convergence above a specified level), and divergence.

Similar to Section 4.1.3, the robustness of the dynamic motion filter is assessed in an inertial environment by running approx. 100 simulations where the initial filter estimates of attitude and body-rates are varied from the true initial attitude and body-rates.

Again, uniformly distributed random numbers are generated that are bounded between -1 and 1. Specifying these upper and lower bounds $x_l = -1$ and $x_u = 1$, the random number is given as,

$$\chi = x_l + (x_u - x_l) \times rand() \tag{5.5}$$

where *rand()* is the standard Matlab uniformly distributed pseudo-random number function. The bounded random number is then multiplied by an amplitude factor that acts as the max allowable size of the error from the truth. For this study, all truth model parameters (**I**, m, etc) and filter covariance parameters ($\mathbf{P}_0, \mathbf{Q}_c, \mathbf{R}_m$) are held constant. Variables that differ from Appendix C and Appendix G are specified as follows:

The initial attitude estimate conditions are specified using,

$$\begin{bmatrix} \hat{\phi} \\ \hat{\theta} \\ \hat{\psi} \end{bmatrix}_{t=0} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\phi_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \\ \Delta\theta_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \\ \Delta\psi_{\max} \times \left(x_l + (x_u - x_l) \times rand()\right) \end{bmatrix} \text{ deg} \tag{5.6}$$

where the deviations from true attitude are bounded by $\Delta\phi_{\max} = \Delta\theta_{\max} = \Delta\psi_{\max} = 15$ deg. The initial body-rate estimate conditions are specified using,

$$\begin{bmatrix} \hat{\omega}_{b_x} \\ \hat{\omega}_{b_y} \\ \hat{\omega}_{b_z} \end{bmatrix}_{t=0} = \begin{bmatrix} \omega_{b_x} \\ \omega_{b_y} \\ \omega_{b_z} \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\omega_{b_x(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_y(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_z(\max)} \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \text{ deg/s} \tag{5.7}$$

where $\Delta\omega_{b_x(\max)} = \Delta\omega_{b_y(\max)} = \Delta\omega_{b_z(\max)} = 0.1$ deg/s. Furthermore, it is assumed that the target will be in a relatively docile motion state such that it is possible to dock with. The true target initial motion is specified as,

$$\begin{bmatrix} \omega_{b_x} \\ \omega_{b_y} \\ \omega_{b_z} \end{bmatrix}_{t=0} = \begin{bmatrix} 1.0 \\ 0.1 \times (x_l + (x_u - x_l) \times rand()) \\ 0.5 \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \text{ deg/s} \tag{5.8}$$

The initial inertial position estimate conditions are specified using,

$$\begin{bmatrix} \hat{r}_{c_x} \\ \hat{r}_{c_y} \\ \hat{r}_{c_z} \end{bmatrix}_{t=0} = \begin{bmatrix} r_{c_x} \\ r_{c_y} \\ r_{c_z} \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta r_{c_x(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta r_{c_y(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta r_{c_z(\max)} \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \text{ m} \tag{5.9}$$

where $\Delta r_{c_x(\max)} = \Delta r_{c_y(\max)} = \Delta r_{c_z(\max)} = 5$ m. Similarly, the initial inertial velocity estimate conditions are specified with $\Delta v_{c_x(\max)} = \Delta v_{c_y(\max)} = \Delta v_{c_z(\max)} = 0.1$ m/s, and the initial CoM offset estimate conditions are specified with $\Delta r_{cg,g_x(\max)} = \Delta r_{cg,g_y(\max)} = \Delta r_{cg,g_z(\max)} = 0.005$ m. To collapse the performance information into a compact format, the norm or size of the errors for each simulation case is shown. Performance related to the orientation state estimate, $\hat{\mathbf{x}}_{\Theta}$, is presented in Figure 5-2 displaying the mean of the norm of the attitude errors and the mean of the norm of the body-rate errors for each simulation case. Performance related to the translational state estimate, $\hat{\mathbf{x}}_{\mathbf{r}}$, is presented in Figure 5-3 displaying the mean of the norm of the position errors and the mean of the norm of the CoM offset errors for each simulation case.

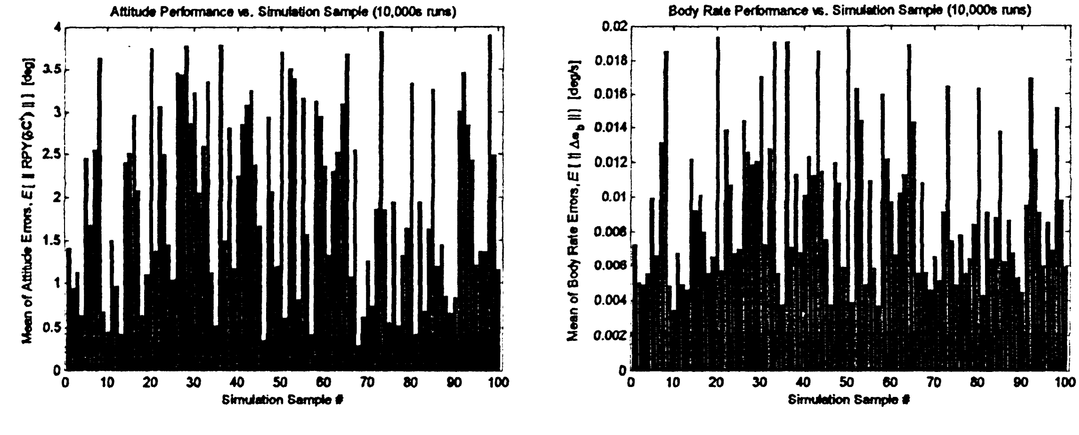


Figure 5-2: Average Error performance for the Rotational State Estimate

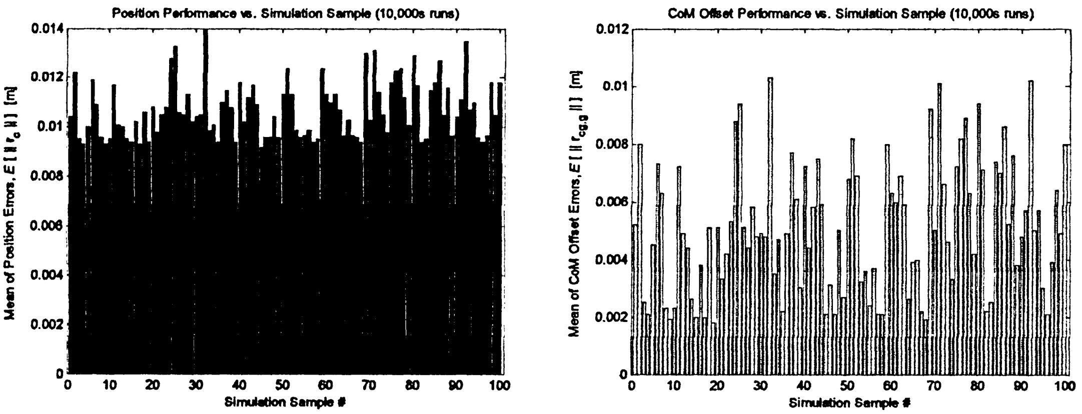


Figure 5-3: Average Error performance for the Translational State Estimate

During the simulations, it was noticed that the attitude estimation performance, $\hat{\Theta}_b$, was very dependent on how well the filter was estimating the body-rates, $\hat{\omega}_b$. For example, Figure 5-4 shows the attitude errors in terms of the RPY parameterization for Case 50 and Case 10. Since the body-rate estimates of case 50 ("black" line –) are the worst of the entire batch, its attitude estimates have much larger "peaks and troughs" when compared to Case 10 ("blue" line –) which is doing a good job of inferring body-rates. This behavior is expected since the body-rates drive the attitude motion. If the vision system could also provide rough estimates (measurements) of the target body-rates to the filter the entire performance of the system would likely be enhanced.

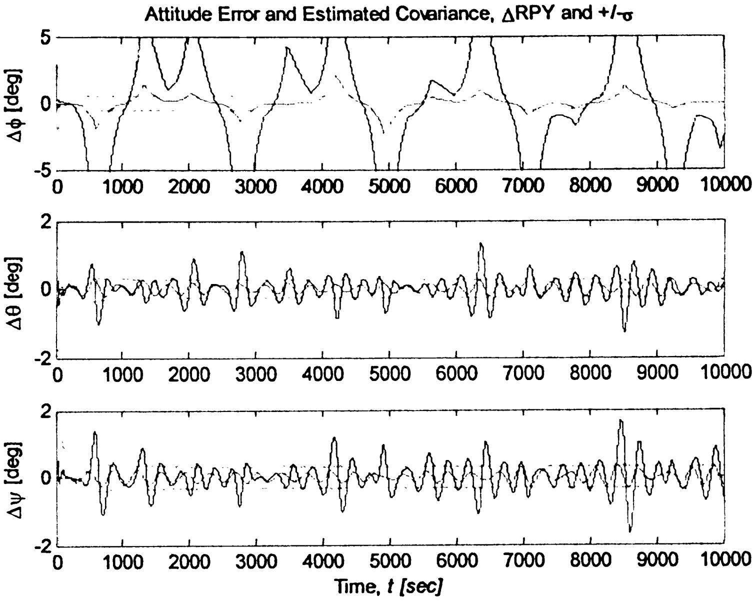


Figure 5-4: Effect of Body-Rate Estimation Performance on Attitude Estimation

The estimation performance of a typical case (Case 25) is shown in Figure 5-5, Figure 5-6, Figure 5-7, Figure 5-8, and Figure 5-9. For Case 25 the random error generation produced the following initial filter estimates ,

$$\begin{Bmatrix} \phi \\ \theta \\ \psi \end{Bmatrix}_{t=0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \text{deg} \quad \text{and} \quad \begin{Bmatrix} \hat{\phi} \\ \hat{\theta} \\ \hat{\psi} \end{Bmatrix}_{t=0} = \begin{bmatrix} 10.98 \\ 11.67 \\ -14.61 \end{bmatrix} \text{deg} \tag{5.10}$$

$$\omega_{b(t=0)} = \begin{bmatrix} 1.0 \\ -0.09 \\ 0.0706 \end{bmatrix} \text{deg/s} \quad \text{and} \quad \hat{\omega}_{b(t=0)} = \begin{bmatrix} 0.9361 \\ -0.1797 \\ 0.1174 \end{bmatrix} \text{deg/s} \tag{5.11}$$

$$\mathbf{r}_{c(t=0)} = \begin{bmatrix} -49.15 \\ 13.9 \\ -68.147 \end{bmatrix} \text{ m} \quad \text{and} \quad \hat{\mathbf{r}}_{c(t=0)} = \begin{bmatrix} -45.514 \\ 14.572 \\ -63.342 \end{bmatrix} \text{ m} \tag{5.12}$$

$$\mathbf{v}_{c(t=0)} = \begin{bmatrix} 0.19 \\ -0.34 \\ 0.3172 \end{bmatrix} \text{ m/s} \quad \text{and} \quad \hat{\mathbf{v}}_{c(t=0)} = \begin{bmatrix} 0.2471 \\ -0.4073 \\ 0.3838 \end{bmatrix} \text{ m/s} \tag{5.13}$$

$$\mathbf{r}_{cg,g(t=0)} = \begin{bmatrix} 0.05 \\ 0.05 \\ 0.0 \end{bmatrix} \text{ m} \quad \text{and} \quad \hat{\mathbf{r}}_{cg,g(t=0)} = \begin{bmatrix} 0.0469 \\ 0.0514 \\ 0.0017 \end{bmatrix} \text{ m} \tag{5.14}$$

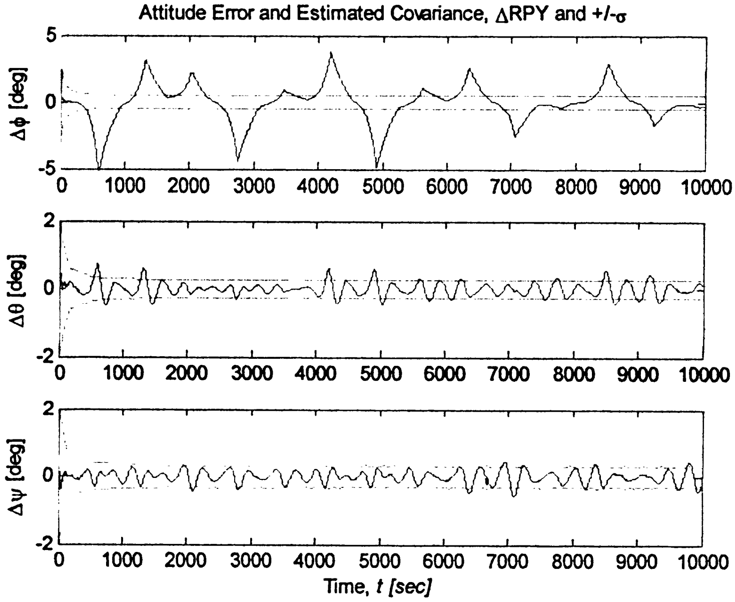


Figure 5-5: Typical Attitude Error Performance (Case 25)

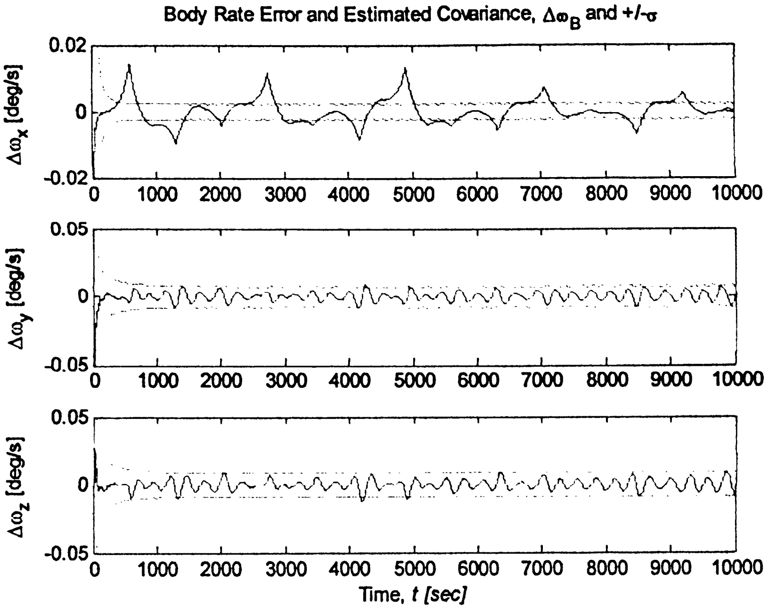


Figure 5-6: Typical Body-Rate Error Performance (Case 25)

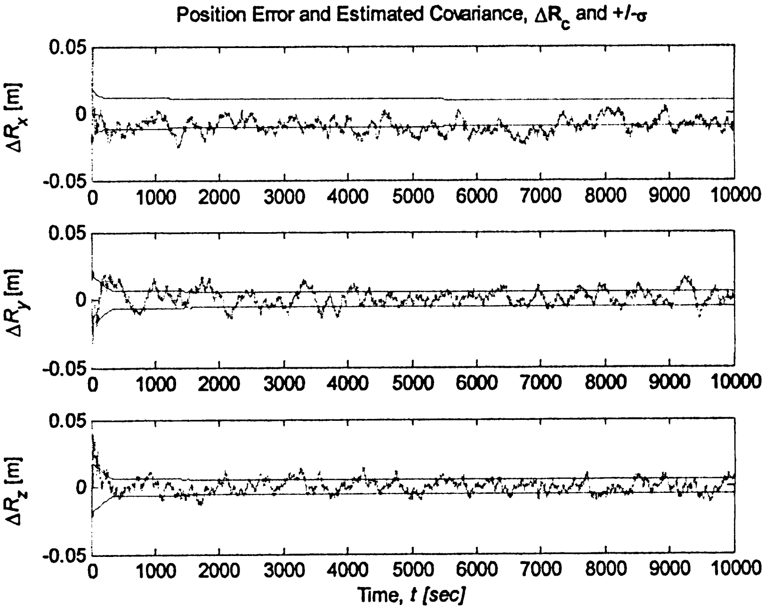


Figure 5-7: Typical Inertial Position Error Performance (Case 25)

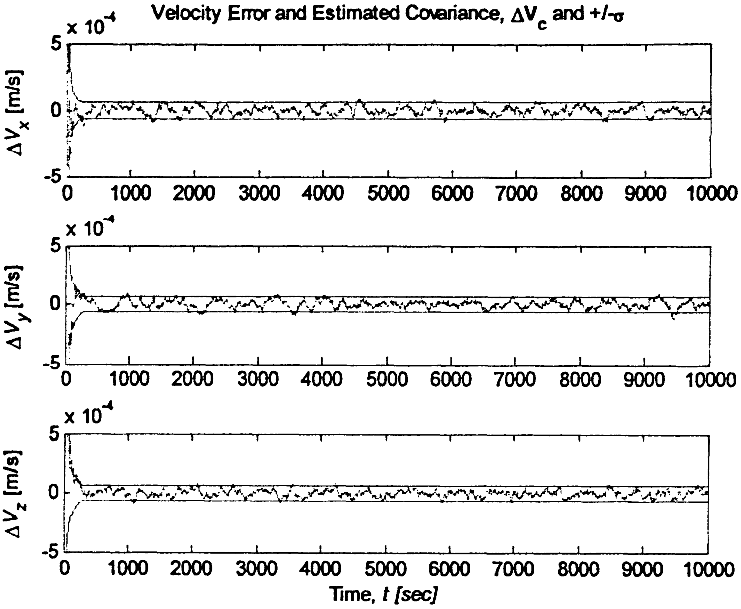


Figure 5-8: Typical Inertial Velocity Error Performance (Case 25)

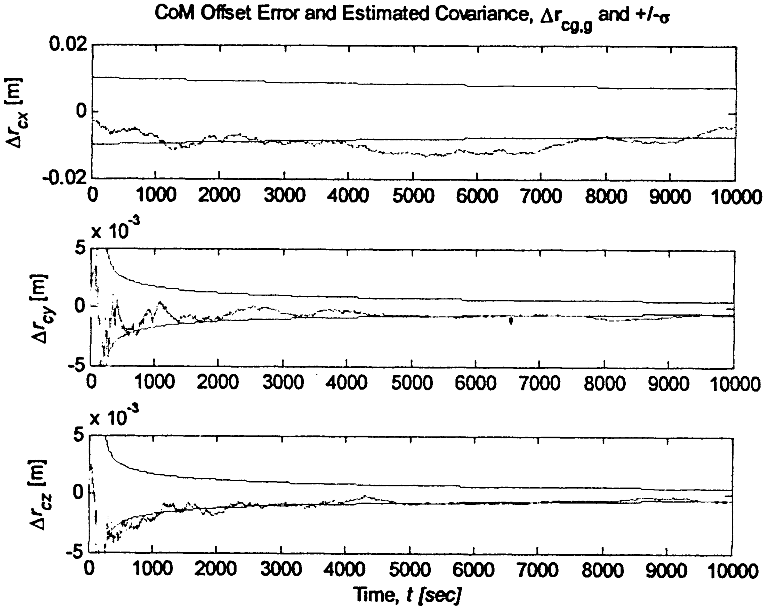


Figure 5-9: Typical CoM Offset Error Performance (Case 25)

As can be seen in Figure 5-5, the roll error, $\Delta\phi$, is bounded but not ideal in matching with the filter's covariance representation of the roll error. This indicates that additional effort should be spent tuning the filter to find a better combination of the noise covariance matrices, $\mathbf{Q}_{\Theta,c}$ and $\mathbf{R}_{\Theta,m}$, or upgrade to a more robust implementation such as the *noise-adaptive filter* to estimate the best combination on-line [29].

5.2 Dynamic Motion Filter Performance - Orbital Environment

In the orbital environment, the dynamic motion filter state scope is,

$$\mathbf{x}=\begin{bmatrix}\mathbf{x}_{\Theta}\\ \mathbf{x}_{\mathrm{r}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{\theta}_b\\ \boldsymbol{\omega}_{bi,b}\\ \hline \bar{\mathbf{r}}\\ \bar{\mathbf{v}}\\ \mathbf{r}_{cg,g}\end{bmatrix} \tag{5.15}$$

The measurement as viewed by the filter is related as follows:

$$\mathbf{z}_k=\begin{bmatrix}\tilde{\boldsymbol{\varepsilon}}_k(\delta\hat{\mathbf{q}}_k)\\ \tilde{\mathbf{r}}_{gv,o}\end{bmatrix} \tag{5.16}$$

The output vector used in the state update equation is as follows:

$$\mathbf{y}=\begin{bmatrix}\mathbf{y}_{\Theta}\\ \mathbf{y}_{\mathrm{r}}\end{bmatrix}=\begin{bmatrix}\mathbf{h}_{\Theta}(\mathbf{x},t)\\ \mathbf{h}_{\mathrm{r}}(\mathbf{x},t)\end{bmatrix}=\begin{bmatrix}\boldsymbol{\varepsilon}\\ \mathbf{r}_{gv,o}\end{bmatrix}=\begin{bmatrix}\frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right)\\ \mathbf{C}_{\theta o}^{T}\mathbf{r}_{cv,\theta}-\mathbf{C}_{o\oplus}\mathbf{C}_{b}^{T}\mathbf{r}_{cg,g}\end{bmatrix} \tag{5.17}$$

where,

$$K=\sqrt{1+\frac{1}{64}\theta^{4}} \tag{5.18}$$

All variables of the above equations were previously defined in Table 4-4 of Section 4.3.3 and will not be re-defined for brevity.

A close-up graphic of the relative orbital operating environment for dynamic motion filter testing is shown in Figure 5-10.

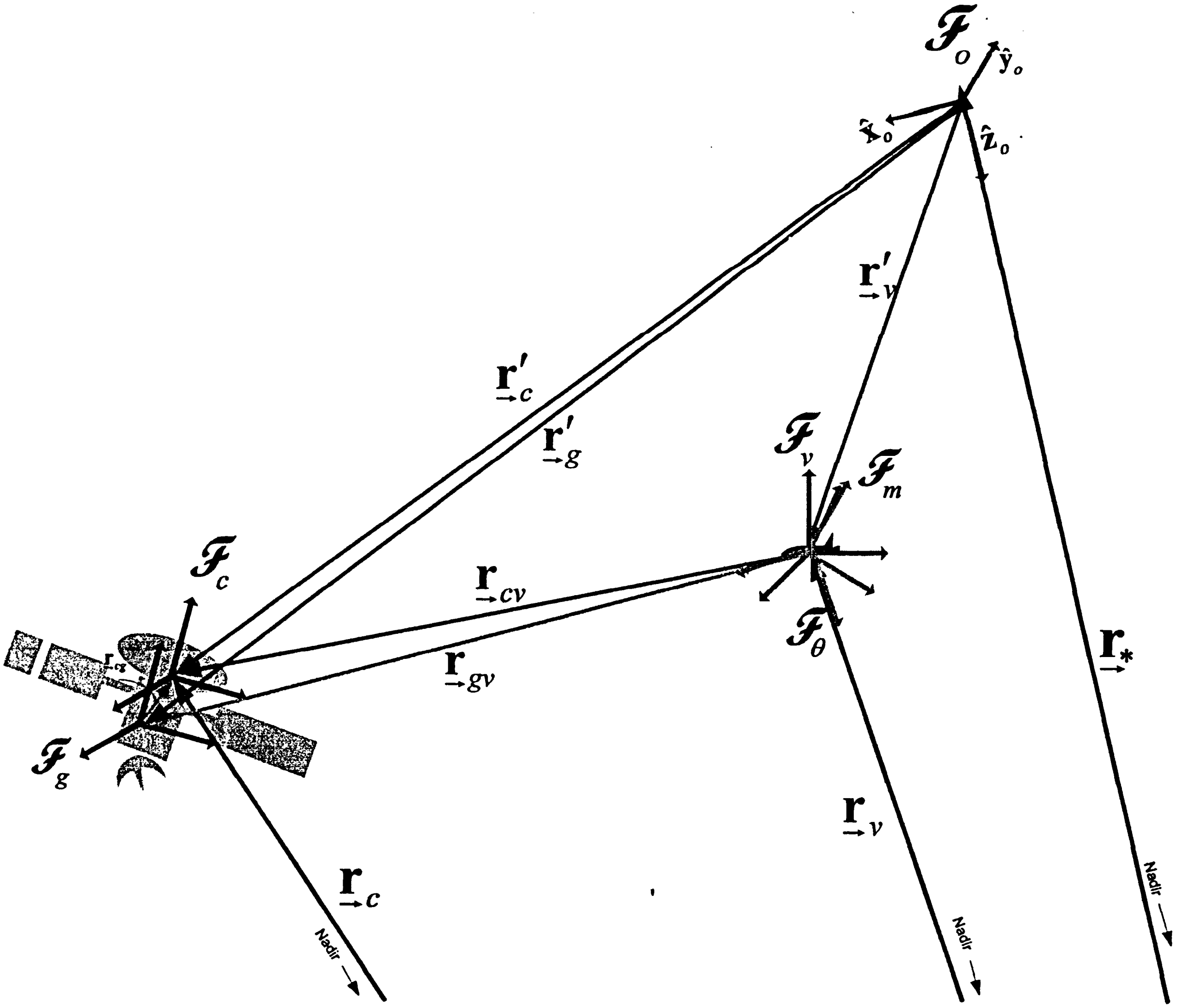

$\mathcal{F}_o$
$\hat{y}_o$
$\hat{x}_o$
$\hat{z}_o$
$\underset{\rightarrow}{r}'_v$
$\underset{\rightarrow}{r}'_c$
$\underset{\rightarrow}{r}'_g$
$\mathcal{F}_v$
$\mathcal{F}_m$
$\mathcal{F}_c$
$\underset{\rightarrow}{r}_{cv}$
$\underset{\rightarrow}{r}_{gv}$
$\mathcal{F}_\theta$
$\underset{\rightarrow}{r}_*$
$\underset{\rightarrow}{r}_\alpha$
$\mathcal{F}_g$
$\underset{\rightarrow}{r}_v$
$\underset{\rightarrow}{r}_c$
Nadir
Nadir
Nadir

Figure 5-10: Orbital Environment Frames and Vectors

To assess the performance of the dynamic motion filter in the orbital environment, a small set of simulations are run with random but bounded truth initial conditions, $\{\mathbf{x}_0\}$, and random but bounded error in the filter initial conditions, $\{\Delta\mathbf{x}_0 = \hat{\mathbf{x}}_0 - \mathbf{x}_0\}$. Each sample simulation will be run for 3000 seconds which is a small time beyond the chaser-to-target intersection in the representative Hohmann transfer scenario.

Similar to Section 4.1.3 and Section 5.1, the robustness of the dynamic motion filter is assessed in an orbital environment by running a number of simulations where the filter initial state estimates are varied from the true initial states. The same randomized initialization procedures outlined in Section 4.1.3 and Section 5.1 are used here and will not to recited for brevity.

For this study, all truth model parameters ($\mathbf{I}$, m, etc) and filter covariance parameters ($\mathbf{P}_0, \mathbf{Q}_c, \mathbf{R}_m$) are held constant. Variables that differ from Appendix C and Appendix I are specified as follows:

The initial attitude estimate conditions are specified using,

$$\begin{bmatrix} \hat{\phi} \\ \hat{\theta} \\ \hat{\psi} \end{bmatrix}_{t=0} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\phi_{\max} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\theta_{\max} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\psi_{\max} \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \text{deg} \qquad (5.19)$$

where the deviations from true attitude are bounded by $\Delta\phi_{\max} = \Delta\theta_{\max} = \Delta\psi_{\max} = 10$ deg. The initial body-rate estimate conditions are specified using,

$$\begin{bmatrix} \hat{\omega}_{b_x} \\ \hat{\omega}_{b_y} \\ \hat{\omega}_{b_z} \end{bmatrix}_{t=0} = \begin{bmatrix} \omega_{b_x} \\ \omega_{b_y} \\ \omega_{b_z} \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta\omega_{b_x(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_y(\max)} \times (x_l + (x_u - x_l) \times rand()) \\ \Delta\omega_{b_z(\max)} \times (x_l + (x_u - x_l) \times rand()) \end{bmatrix} \text{deg/s} \qquad (5.20)$$

where $\Delta\omega_{b_x(\max)} = \Delta\omega_{b_y(\max)} = \Delta\omega_{b_z(\max)} = 0.1$ deg/s. Furthermore, it is assumed that the target will be in a relatively docile motion state such that it is possible to dock with. The true target initial motion is specified as,

$$\begin{bmatrix} \omega_{b_x} \\ \omega_{b_y} \\ \omega_{b_z} \end{bmatrix}_{t=0} = \begin{bmatrix} 1.0 \\ 0.1\times\left(x_l+(x_u-x_l)\times rand()\right) \\ 0.5\times\left(x_l+(x_u-x_l)\times rand()\right) \end{bmatrix} \text{ deg/s} \tag{5.21}$$

The initial relative position estimate conditions are specified using,

$$\begin{bmatrix} \hat{r}_{cv_x} \\ \hat{r}_{cv_y} \\ \hat{r}_{cv_z} \end{bmatrix}_{t=0} = \begin{bmatrix} r_{cv_x} \\ r_{cv_y} \\ r_{cv_z} \end{bmatrix}_{t=0} + \begin{bmatrix} \Delta r_{cv_x(\max)}\times\left(x_l+(x_u-x_l)\times rand()\right) \\ \Delta r_{cv_y(\max)}\times\left(x_l+(x_u-x_l)\times rand()\right) \\ \Delta r_{cv_z(\max)}\times\left(x_l+(x_u-x_l)\times rand()\right) \end{bmatrix} \text{ m} \tag{5.22}$$

where $\Delta r_{cv_x(\max)} = \Delta r_{cv_y(\max)} = \Delta r_{cv_z(\max)} = 5$ m. Similarly the initial relative velocity estimate conditions are specified with $\Delta v_{c_x(\max)} = \Delta v_{c_y(\max)} = \Delta v_{c_z(\max)} = 0.1$ m/s, and the initial CoM offset estimate conditions are specified with $\Delta r_{cg,g_x(\max)} = \Delta r_{cg,g_y(\max)} = \Delta r_{cg,g_z(\max)} = 0.005$ m. The performance of the dynamic motion filter is displayed similar to [27] where 5 sample runs are displayed on the plots including the baseline case. Note the baseline case is indicated by a solid "blue" line, while the additional sample runs are indicated by dashed/dashed-dot lines. For Figure 5-11 to Figure 5-18, the line styles are defined in Table 5-1.

———	Baseline Case
-------	Random Case 1
— —	Random Case 2
— - — ·	Random Case 3
— —	Random Case 4

Table 5-1: 5 Sample Run Line Style Definitions

From the sample simulations shown in Figure 5-11, it can be seen that the attitude portion of the dynamic motion filter is sensitive to the accuracy of the initial estimates.

The random sample case 4 in "cyan" was initialized with the most accurate body-rate estimate, $\|\Delta\hat{\omega}_b(t=0)\| \approx 0.024$ deg/s, along with an accurate attitude estimate.

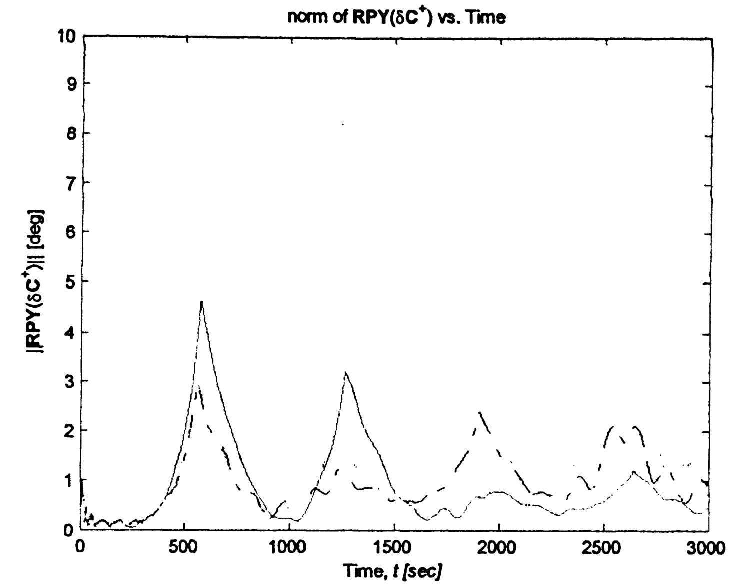


Figure 5-11: Norm of Attitude Errors (5 Sample Runs)

The *a posteriori* attitude errors in terms of a RPY sequence, $RPY(\delta C^+)$, vs. time are shown in Figure 5-12. For these sample runs the figure indicates that the filter's representation of its own errors match reality as indicated by the fact that the true errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time. In Figure 5-13, it is seen that the angular rate errors corresponding to these cases are within approx. 0.01-0.02 deg/s and the filter's covariance in the state estimate error represents reality well.

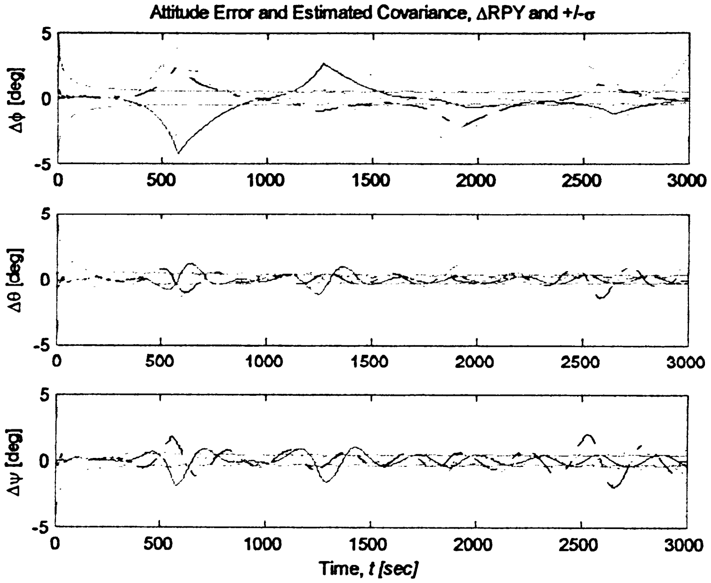


Figure 5-12: Attitude Errors parameterized as RPY (5 Sample Runs)

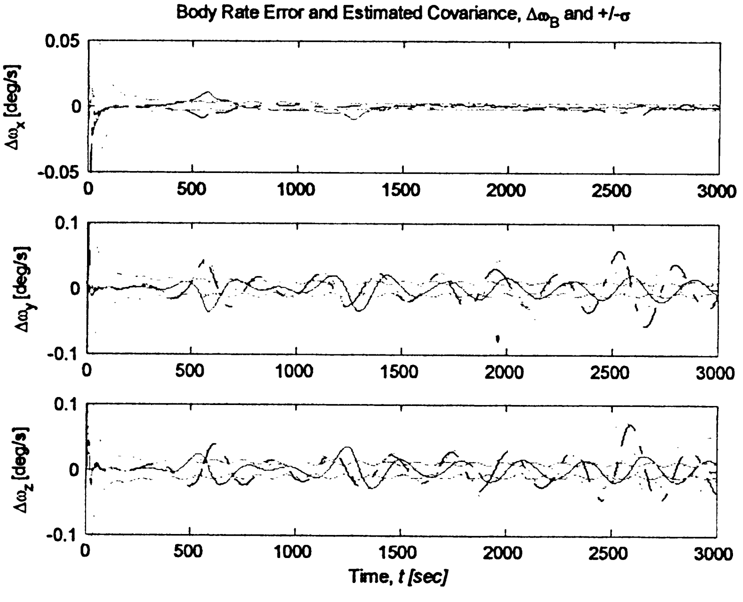


Figure 5-13: Body-Rate Errors (5 Sample Runs)

The relative position errors represented in cylindrical coordinates are shown in Figure 5-14. The filter does an excellent job of tracking the relative position of the target as seen by the radial and out-of-plane errors bounded within 2 cm and angular errors within a fraction of a degree. Expressing the relative position errors in terms of there Cartesian-like form in the reference orbital-nadir frame, the error magnitude is bounded within 3 cm.

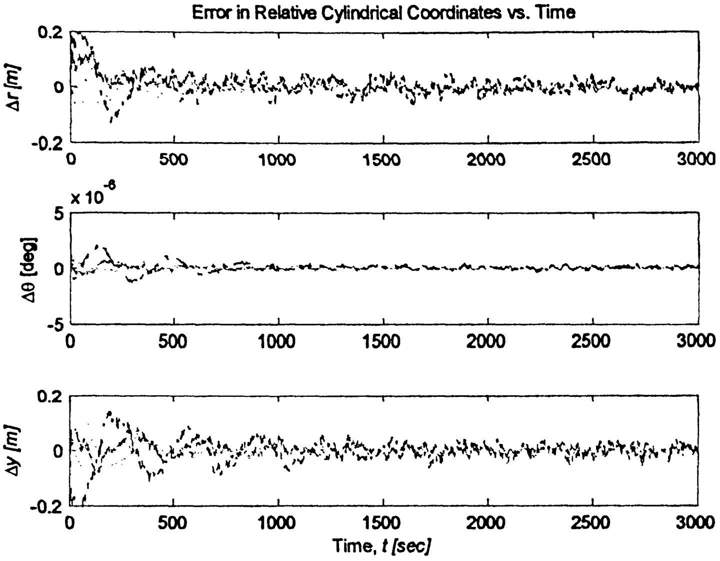


Figure 5-14: Relative Cylindrical Coordinate Errors vs. Time (5 Sample Runs)

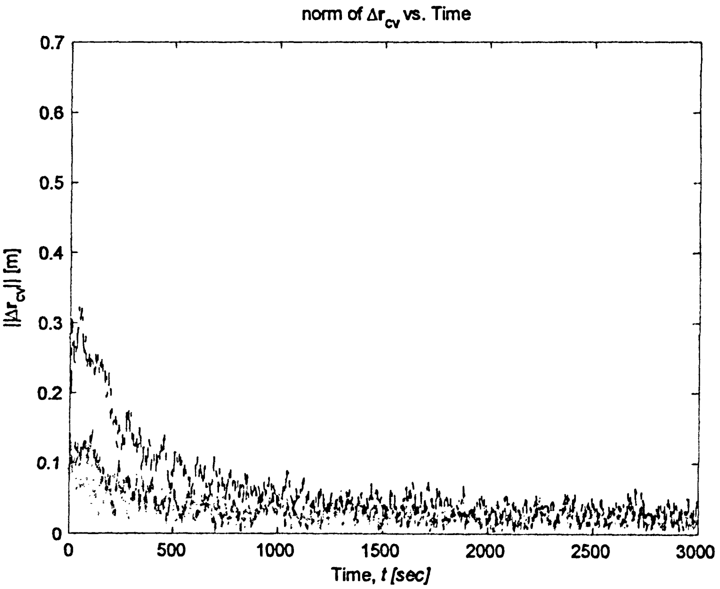


Figure 5-15: Norm of Cartesian-like Position Errors (5 Sample Runs)

The *a posteriori* position errors vs. time expressed in Cartesian-like form are shown in Figure 5-16. The figure indicates that the filter's own representation of its errors match reality as indicated by the fact that the true errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time.

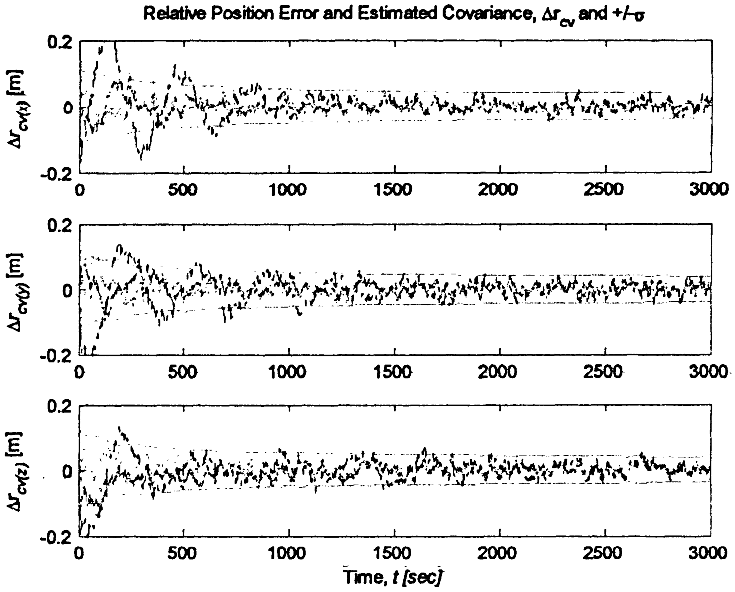


Figure 5-16: Relative Cartesian Position Errors vs. Time (5 Sample Runs)

The CoM offset errors vs. time can be seen in Figure 5-17 and the norm of the CoM offset errors in Figure 5-18. Again, the true CoM offset errors are within the covariance bounds ($\pm\sigma$) approx. 68% of the time. For the sample runs shown, the magnitude of the CoM offset error is bounded within 0.6 cm as seen in Figure 5-18.

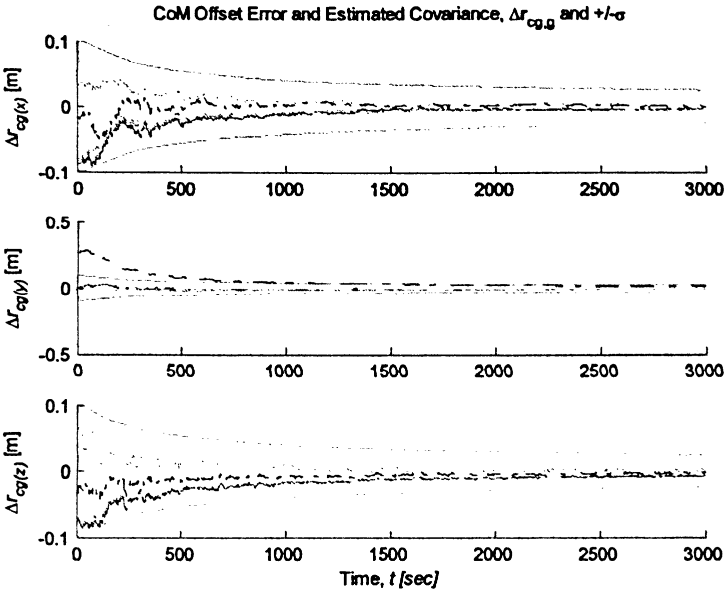


Figure 5-17: CoM Offset Errors vs. Time (5 Sample Runs)

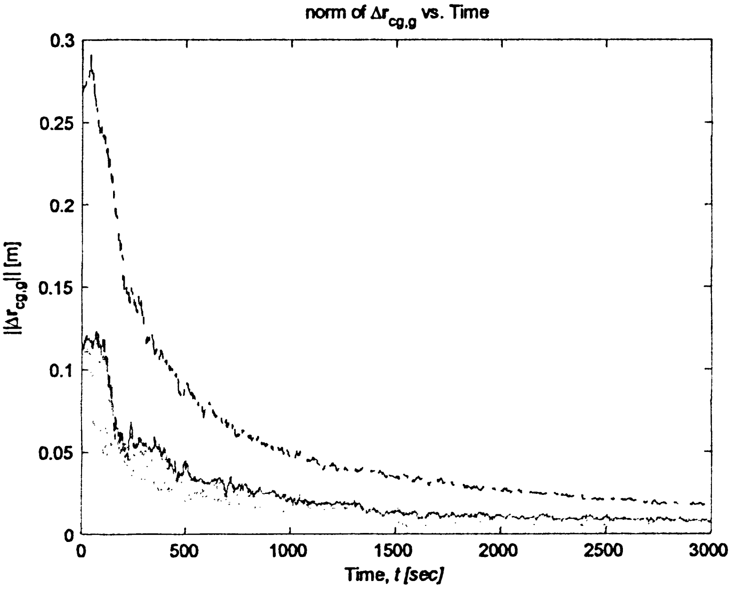


Figure 5-18: Norm of CoM Offset Errors (5 Sample Runs)

As mentioned in Section 5.1, during the orbital scenario simulations, it was noticed that the attitude partition of the filter is less robust to initialization errors than the relative translation partition but the entire state will be affected since the two partitions are coupled. Even larger values of $\Delta\mathbf{r}_{cv(\max)}$ in (5.22) were tested (≈ 10 m error) and the filter did an excellent job of estimating the relative motion. However, large initialization errors in attitude and body-rates beyond the thresholds mentioned in Section 4.1.3 caused delayed and even no convergence of the filter states. If the vision system could also provide rough estimates (measurements) of the target body-rates to the filter, the entire performance of the system would likely be enhanced and the dynamic motion filter would be more robust against initialization errors in the orientation states.

6 Conclusions and Recommendations

Using the computationally efficient extended Kalman filtering technique, a dynamic motion filter was designed and tested for use in a hybrid pose/motion estimation system to assist an orbital rendezvous vision system in its motion estimation duties. Specifically, the results shown in Section 5 indicate that a dynamic motion filter can provide the vision system routines with excellent initialization leading to faster convergenc, reliable pose estimation at slower sampling rates, and the ability to estimate target position, velocity, orientation, angular velocity, and mass center location.

Although the filter performs well with moderately accurate initialization, it was particularly sensitive to initialization errors in the target body-rates, $\hat{\omega}_b$, a similar finding in [12] and [28]. This lack of robustness to initialization errors is likely due to the first-order approximation implicit in the EKF formulation as described in the literature. Also, particular attention should be made to the subtlety in covariance specification and tuning of the nonlinear attitude partition of the filter. Issues and techniques are covered in Appendix D.

Future work should focus on: i) augmenting the state for the addition of inertia parameters (inertia components, principal directions) and recognition of dynamic effects to differentiate them; ii) implementation of "advanced" formulations (SRKF, UKF, etc.) as required for viability and/or computational stability and efficiency; and iii) testing with real data. The filter can then be introduced into the hybrid pose/motion estimation system to compliment the vision system in its motion estimation duties.

Appendix A Kalman Filtering – Implementation Overview

For Kalman filtering, the physical situation of most common interest in aerospace applications is a hybrid one:

- System dynamics are continuous in time
- Measurements are processed at discrete points in time

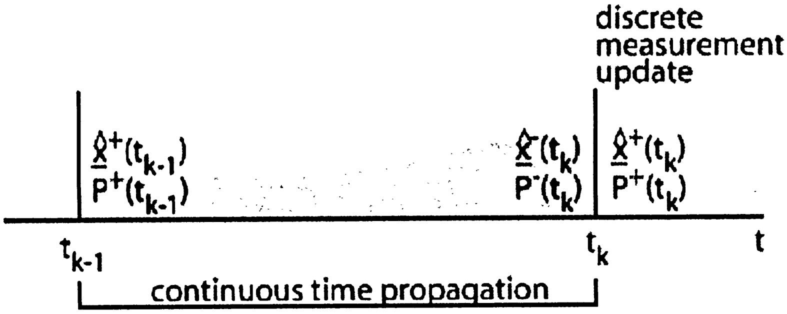

discrete measurement update
$\hat{\underline{x}}^{+}(t_{k-1})$
$P^{+}(t_{k-1})$
$\hat{\underline{x}}^{-}(t_k)$
$P^{-}(t_k)$
$\hat{\underline{x}}^{+}(t_k)$
$P^{+}(t_k)$
t_{k-1}
t_k
t
continuous time propagation

Figure A-1: Implementation Process Step [30]

Where a discrete time measurement is processed, there is a discontinuity in the estimate, the estimate error, and the error covariance matrix.

In the diagram above, the superscript "-" is used to indicate values before incorporating the measurement at a measurement time (*a priori)* and the superscript "+" to indicate values after incorporating the measurement at a measurement time (*a posteriori).*

A graphical overview of the typical interaction between a real system and the Kalman filter is shown in Figure A-2.

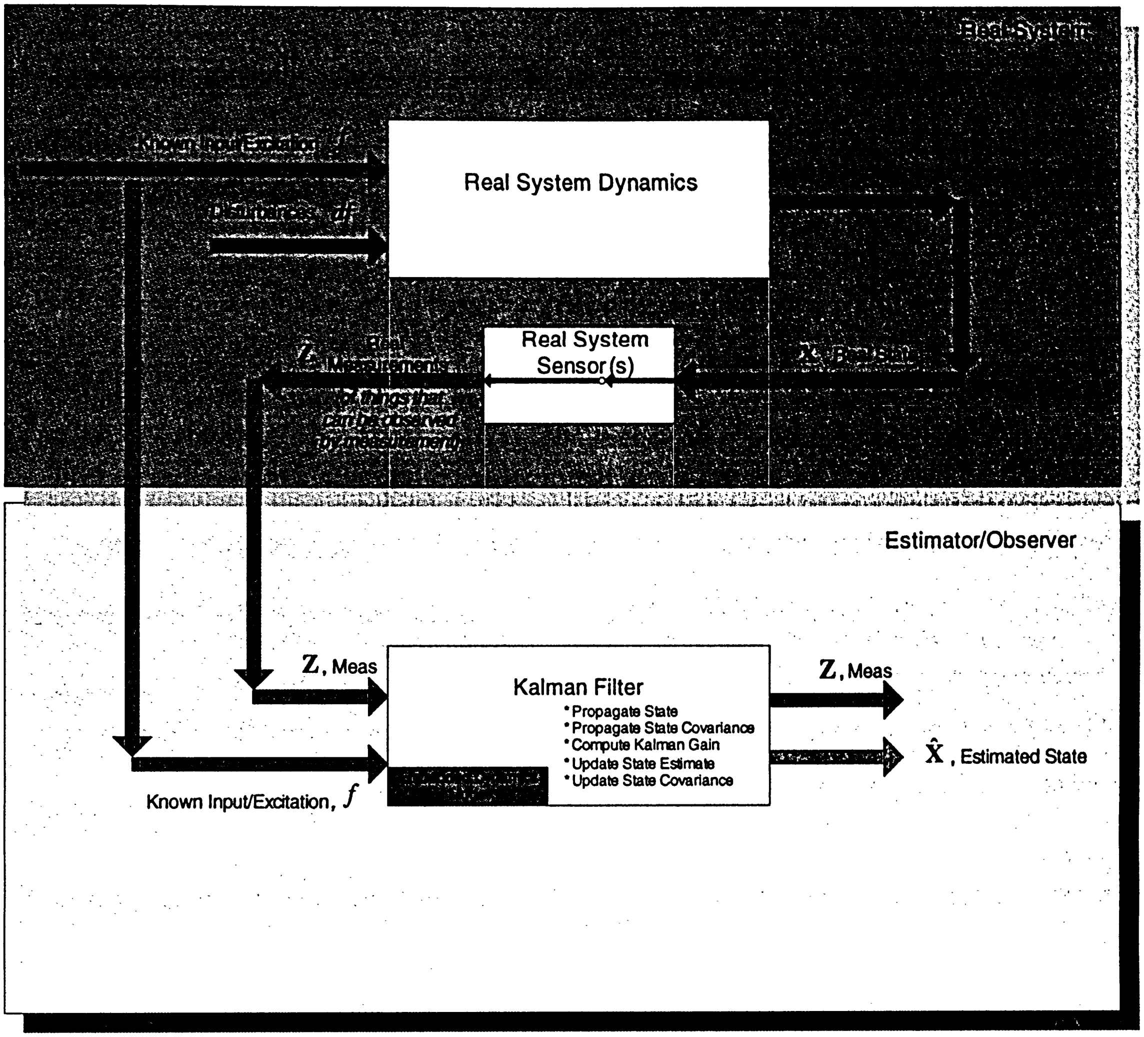


Figure A-2: Overview of Kalman filter with Real System Interaction [31]

Below, the basic extended Kalman filtering equations are shown that can be found in any standard text including [7, 27, 29, 32]. Although not shown below, there are a variety of equivalent formulations in both discrete-time and continuous-time that have numerical advantages over the standard formulations.

A.1 Procedure: Time Propagation

After processing the measurement at t_{k-1}, one acquires the estimate $\hat{\mathbf{x}}_{k-1}^{+}$ and the estimation error covariance matrix $\mathbf{P}_{k-1}^{+}$.

The filter operates open loop until the next measurement point at t_k. In this interval the system dynamics are described by a differential equation. In general the true and filter interpretations of the system motion will be given by,

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t),\mathbf{w}(t),\mathbf{p}(t),t) \\ \dot{\hat{\mathbf{x}}}(t) &= \mathbf{f}(\hat{\mathbf{x}}(t),\mathbf{u}(t),\mathbf{0},t)\end{aligned} \tag{A.1}$$

and it is noted that since we do not know the process noise input $\mathbf{w}(t)$, we express the filter dynamics equation with the expectation of $\mathbf{w}(t)$ (namely, $E[\mathbf{w}(t)] \equiv \mathbf{0}$). The estimate, $\hat{\mathbf{x}}_{k-1}^{+}$, is the mean of the distribution of $\mathbf{x}$ conditioned on all the measurements processed up to that point. The estimated error is zero mean. To preserve the error unbiased until the next measurement, the estimate can be driven with the mean or expectation of the system dynamics.

The propagation step can be taken by integrating the following differential equations

$$\begin{aligned}\dot{\hat{\mathbf{x}}} &= \mathbf{f}(\hat{\mathbf{x}}(t),\mathbf{u}(t),t) \rightarrow \text{where} \rightarrow \hat{\mathbf{x}}(t_{k-1}) = \hat{\mathbf{x}}_{k-1}^{+} \\ \dot{\mathbf{P}} &= \mathbf{FP} + \mathbf{PF}^{T} + \mathbf{LQ}_c\mathbf{L}^{T} \rightarrow \text{where} \rightarrow \mathbf{P}(t_{k-1}) = \mathbf{P}_{k-1}^{+}\end{aligned} \tag{A.2}$$

More explicitly, the general propagation of the system takes on the form

State Estimate Propagation

$$\hat{\mathbf{x}}_k^{-} = \hat{\mathbf{x}}_{k-1}^{+} + \int_{t_{k-1}}^{t_k} \mathbf{f}\left(\hat{\mathbf{x}}(\tau),\mathbf{u}(\tau),\tau\right) d\tau \tag{A.3}$$

Covariance Estimate Propagation

$$\mathbf{P}_k^{-} = \mathbf{P}_{k-1}^{+} + \int_{t_{k-1}}^{t_k} \left[\mathbf{F}(\tau)\mathbf{P}(\tau) + \mathbf{P}(\tau)\mathbf{F}^{T}(\tau) + \mathbf{L}(\tau)\mathbf{Q}_c(\tau)\mathbf{L}^{T}(\tau)\right] d\tau \tag{A.4}$$

The matrices that comprise the error covariance derivative $\dot{\mathbf{P}}(t)$ are generally partial derivative Jacobian matrices evaluated at the current best estimate of the variables $\hat{\mathbf{x}}$ and $\mathbf{u}$ for a non-linear system of the form in (A.1). These include,

$$\mathbf{F}(\tau)=\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(\tau),\mathbf{u}(\tau),\tau)\bigg|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{A.5}$$

$$\mathbf{L}(\tau)=\frac{\partial}{\partial \mathbf{w}}\mathbf{f}(\mathbf{x}(\tau),\mathbf{u}(\tau),\mathbf{w}(\tau),\tau)\bigg|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{A.6}$$

This will make a non-linear system appear linear over short time intervals. The state propagation can be accomplished by directly applying a numerical routine. An established algorithm such as the *classical fourth-order Runge-Kutta method* can be used for the numerical propagation.

A.2 Procedure: Measurement Update

At the next measurement point, t_k, one has the *a priori* estimate, $\hat{\mathbf{x}}_k^-$, it's error covariance matrix, $\mathbf{P}_k^-$, and a new measurement of the general form,

$$\mathbf{z}_k=\mathbf{h}(\mathbf{x}_k,\mathbf{u}_k,t_k)+\mathbf{v}_k \tag{A.7}$$

where the noise, $\mathbf{v}_k$, is an unbiased, independent sequence of random variables associated with measurement noise.

The update and gain equations are given in discrete-time as follows,

Filter Gain Computation

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left[\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \right]^{-1} \tag{A.8}$$

where

$$\mathbf{H}_k = \frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{x}(t), t) \bigg|_{\mathbf{x}=\hat{\mathbf{x}}_k} \tag{A.9}$$

State Estimate Update

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k, \mathbf{u}_k, t_k) \right] \tag{A.10}$$

Covariance Estimate Update

$$\mathbf{P}_k^+ = \left[\mathbf{1} - \mathbf{K}_k \mathbf{H}_k \right] \mathbf{P}_k^- \tag{A.11}$$

Appendix B Attitude Estimation Filter Equations (MEKF)

The implementation of the MEKF is summarized as follows:

B.1 Initial Conditions

At time t_0 ...

$$\begin{aligned} \hat{\mathbf{q}}_{\mathrm{ref},0} &= \hat{\mathbf{q}}_{\mathrm{ref}}(t_0) \\ \hat{\boldsymbol{\omega}}_{b,0} &= \hat{\boldsymbol{\omega}}_b(t_0) \\ \mathbf{P}_0 &= \mathbf{P}(t_0) \\ \hat{\boldsymbol{\theta}}_k^- &\equiv \mathbf{0} \ \text{ by definition} \end{aligned} \tag{B.1}$$

NOTE: the '~' (tilde) under the variable is used to indicate "The Error Attitude State" to avoid confusion with the '~' (tilde) atop a variable that indicates a measurement.

The many different states are specified as,

The Reference Attitude State

$$\mathbf{x}_{\boldsymbol{\Theta}}(t_k) = \begin{bmatrix} \mathbf{q}_{\mathrm{ref},k} \\ \boldsymbol{\omega}_{b,k} \end{bmatrix} \tag{B.2}$$

The Error Attitude State

$$\underset{\sim}{\mathbf{x}}_{\boldsymbol{\Theta}}(t_k) = \begin{bmatrix} \boldsymbol{\theta}_k \\ \boldsymbol{\omega}_{b,k} \end{bmatrix} \tag{B.3}$$

The S-function Attitude State

$$\mathbf{x}_{\boldsymbol{\theta},\text{S-func}}(t_k) = \begin{bmatrix} \hat{\underset{\sim}{\mathbf{x}}}^{+}_{\boldsymbol{\theta},k} \\ \hat{\underset{\sim}{\mathbf{x}}}^{-}_{\boldsymbol{\theta},k} \\ \mathbf{P}^{+}_{\boldsymbol{\theta},k} \\ \mathbf{P}^{-}_{\boldsymbol{\theta},k} \\ \hat{\mathbf{q}}^{-}_{\text{ref},k} \end{bmatrix} \tag{B.4}$$

The following steps implement the filter...

B.2 Propagating the State Estimate

Integrate starting from $\hat{\mathbf{q}}^{+}_{\text{ref},k-1}$ and $\hat{\boldsymbol{\omega}}^{+}_{b,k-1}$ to get $\hat{\mathbf{q}}^{-}_{\text{ref},k}$ and $\hat{\boldsymbol{\omega}}^{-}_{b,k}$.

The propagation is carried out using the following equations of motion,

$$\dot{\mathbf{q}} = \mathbf{q} \oplus \boldsymbol{\varpi} = \boldsymbol{\Xi}(\boldsymbol{\varpi})\mathbf{q} = \bar{\boldsymbol{\Xi}}(\mathbf{q})\boldsymbol{\varpi} \tag{B.5}$$

$$\dot{\boldsymbol{\omega}}_b = \mathbf{I}^{-1}\left[\mathbf{g}_{\text{ext}} - \boldsymbol{\omega}_b^{\times}\mathbf{I}\boldsymbol{\omega}_b\right] \tag{B.6}$$

where, as defined earlier,

$$\boldsymbol{\varpi} = \begin{bmatrix} 0 \\ \frac{1}{2}\boldsymbol{\omega}_b \end{bmatrix}, \tag{B.7}$$

and

$$\boldsymbol{\Xi}(\mathbf{q}) = \begin{bmatrix} \eta & -\boldsymbol{\varepsilon}^T \\ \boldsymbol{\varepsilon} & \eta\mathbf{1} - \boldsymbol{\varepsilon}^{\times} \end{bmatrix} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \tag{B.8}$$

The acting torques in (B.3) can be partitioned as $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}} + \mathbf{g}_{\text{disturb}}$. However, for the filter propagation the dynamics can be driven with only known inputs such that $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}}$.

In terms of the reference state estimate, $\hat{\mathbf{x}}(t_k)$,

$$\hat{\mathbf{x}}_k^- = \hat{\mathbf{x}}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \mathbf{f}\{\hat{\mathbf{x}}[\tau(-)], \mathbf{u}(\tau), \tau\} d\tau \tag{B.9}$$

And you get,

$$\begin{matrix} \hat{\mathbf{q}}_{\text{ref},k}^- \\ \hat{\boldsymbol{\omega}}_{\text{b},k}^- \end{matrix} \tag{B.10}$$

Regarding the *a priori* estimate of the attitude error state, $\hat{\boldsymbol{\theta}}_k^- \equiv \mathbf{0}$ by definition.

B.3 Extrapolating the State-Error Covariance

Propagate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{P}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \left[\mathbf{F}(\tau)\mathbf{P}(\tau) + \mathbf{P}(\tau)\mathbf{F}^T(\tau) + \mathbf{L}(\tau)\mathbf{Q}_c(\tau)\mathbf{L}^T(\tau)\right] d\tau \tag{B.11}$$

where,

$$\mathbf{F}(\tau) = \frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau)\Big|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{B.12}$$

$$\mathbf{L}(\tau) = \frac{\partial}{\partial \mathbf{w}} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \mathbf{w}(\tau), \tau)\Big|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{B.13}$$

or alternatively extrapolate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^+\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \tag{B.14}$$

where the state transition matrix, $\mathbf{\Phi}_k$, is the Taylor-series expansion of $e^{\mathbf{F}T_s}$,

$$\mathbf{\Phi}_k = \mathbf{1} + \mathbf{F}T_s + \frac{\mathbf{F}^2T_s^2}{2!} + \frac{\mathbf{F}^3T_s^3}{3!} + ... \tag{B.15}$$

The discrete process noise covariance matrix, $\mathbf{Q}_k$, can be computed from the continuous-time process noise spectral density matrix, $\mathbf{Q}_c$, from the following [27],

$$\mathbf{Q}_k = \int_0^{T_s} \mathbf{\Phi}_{k-1}(\tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\mathbf{\Phi}_{k-1}^T(\tau)d\tau \tag{B.16}$$

The discrete-time form will be less computationally expensive but generally less accurate.

For the particular dynamic motion filter state in the inertial context one can often derive explicit expressions for the Jacobians, State Transition matrix, and Process Noise Covariance matrix.

JACOBIAN **F**

To evaluate the Jacobian derivatives associated with $\dot{\boldsymbol{\theta}}$, the following relationship is noted: [33]

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^\times + \frac{1}{12}\boldsymbol{\theta}^\times\boldsymbol{\theta}^\times\right]\delta\boldsymbol{\omega} \tag{B.17}$$

where $\delta\boldsymbol{\omega}$ is the relative angular velocity from the reference orientation $\mathbf{q}_{\text{ref}}$ to the net orientation $\mathbf{q}_b$. The following can be written:

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^\times + \frac{1}{12}\boldsymbol{\theta}^\times\boldsymbol{\theta}^\times\right]\left(\boldsymbol{\omega} - \delta\mathbf{C}\boldsymbol{\omega}_{\text{ref}}\right) \tag{B.18}$$

By applying the approximation,

$$\delta\mathrm{C} \approx \delta\mathrm{C}(\boldsymbol{\theta}) = \mathbf{1} - \boldsymbol{\theta}^{\times} + \frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times} \tag{B.19}$$

the following relation exists, valid to second-order,

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right]\left(\boldsymbol{\omega} - \left(\mathbf{1} - \boldsymbol{\theta}^{\times} + \frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right)\boldsymbol{\omega}_{\mathrm{ref}}\right) \tag{B.20}$$

To first-order in the small $\boldsymbol{\theta}$,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^{T}} = -\frac{1}{2}\boldsymbol{\omega}_{b}^{\times} + \frac{1}{12}\left[\boldsymbol{\omega}_{b}^{\times}\boldsymbol{\theta}^{\times} - 2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_{b}^{\times}\right] - \frac{1}{2}\boldsymbol{\omega}_{\mathrm{ref}}^{\times} - \frac{1}{12}\left[\boldsymbol{\omega}_{\mathrm{ref}}^{\times}\boldsymbol{\theta}^{\times} - 2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_{\mathrm{ref}}^{\times}\right] \tag{B.21}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_{b}^{T}} = \mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} \tag{B.22}$$

However, when implemented according to the MEKF procedure, the expressions are evaluated such that $\hat{\boldsymbol{\theta}} = \mathbf{0}$ and $\hat{\boldsymbol{\omega}}_{b} = \boldsymbol{\omega}_{\mathrm{ref}}$, hence the much simpler forms,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^{T}} = -\boldsymbol{\omega}_{\mathrm{ref}}^{\times} \tag{B.23}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_{b}^{T}} = \mathbf{1} \tag{B.24}$$

The complete Jacobian for $\mathbf{F} = \dfrac{\partial \mathbf{f}[\cdot]}{\partial \underset{\sim}{\mathbf{x}}^T}$ is as follows:

$$\begin{aligned}\mathbf{F} = \frac{\partial \mathbf{f}[\cdot]}{\partial \underset{\sim}{\mathbf{x}}^T} &= \begin{bmatrix} \dfrac{\partial \dot{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}^T} & \dfrac{\partial \dot{\boldsymbol{\theta}}}{\partial \boldsymbol{\omega}_b^T} \\ \dfrac{\partial \dot{\boldsymbol{\omega}}_b}{\partial \boldsymbol{\theta}^T} & \dfrac{\partial \dot{\boldsymbol{\omega}}_b}{\partial \boldsymbol{\omega}_b^T} \end{bmatrix} \\ &= \begin{bmatrix} -\boldsymbol{\omega}_{\mathrm{ref}}^{\times} & \mathbf{1} \\ \mathbf{0} & \mathbf{I}^{-1}\left[(\mathbf{I}\boldsymbol{\omega}_b)^{\times} - (\boldsymbol{\omega}_b^{\times}\mathbf{I})\right] \end{bmatrix}\end{aligned} \qquad \text{(B.25)}$$

<u>*JACOBIAN*</u> **L**

$$\mathbf{L} = \frac{\partial \mathbf{f}[\cdot]}{\partial \mathbf{w}^T} = \begin{bmatrix} \dfrac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{g}^T} \\ \dfrac{\partial \dot{\boldsymbol{\omega}}_b}{\partial \mathbf{g}^T} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}^{-1} \end{bmatrix} \qquad \text{(B.26)}$$

<u>*STATE TRANSITION MATRIX*</u> $\boldsymbol{\Phi}$

$$\boldsymbol{\Phi}_{k-1} \approx \mathbf{1} + \mathbf{F}T_s \qquad \text{(B.27)}$$

<u>*PROCESS NOISE COVARIANCE MATRIX*</u> $\mathbf{Q}_k$

$$\mathbf{Q}_k = \int_0^{T_s} \boldsymbol{\Phi}_{k-1}(\tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\boldsymbol{\Phi}_{k-1}^T(\tau)d\tau \qquad \text{(B.28)}$$

B.4 The Measurement and Output Equation

The measurement comes in as $\tilde{\mathbf{C}}_k$ and is converted to $\rightarrow \tilde{\mathbf{q}}_k$.

To compute the measurement error covariance we must convert from $\hat{\mathbf{q}}^{-}_{\text{ref},k}$ to $\rightarrow \hat{\mathbf{C}}^{-}_{\text{ref},k}$.

$$\begin{aligned}
\tilde{\mathbf{q}}_k &= \hat{\mathbf{q}}^{-}_{\text{ref},k} \oplus \delta\hat{\mathbf{q}}_k \\
\delta\hat{\mathbf{q}}_k &= \overline{\hat{\mathbf{q}}}^{-}_{\text{ref},k} \oplus \tilde{\mathbf{q}}_k \\
\delta\hat{\mathbf{q}}_k &= \mathbf{\Xi}(\tilde{\mathbf{q}}_k)\overline{\hat{\mathbf{q}}}^{-}_{\text{ref},k}
\end{aligned} \tag{B.29}$$

Then, extract the ε part of the corresponding Euler parameters...

$$\tilde{\boldsymbol{\varepsilon}}_k \leftarrow \delta\hat{\mathbf{q}}_k \tag{B.30}$$

and compute the measurement error covariance

$$\mathbf{R}_{\mathbf{\theta},k} = \frac{1}{4}\mathbf{C}_{\text{ref},m}\mathbf{R}_{\mathbf{\theta},m}\mathbf{C}^{T}_{\text{ref},m} \tag{B.31}$$

where

$$\mathbf{C}_{\text{ref}\,m} = \hat{\mathbf{C}}^{-}_{\text{ref},k}\mathbf{C}^{T}_{mi} \tag{B.32}$$

is the current estimate of the measurement-frame to body-frame rotation matrix.

The measurement as viewed by the filter is related as follows,

$$\mathbf{z}_{\mathbf{\theta},k} = \tilde{\boldsymbol{\varepsilon}}_k \tag{B.33}$$

The output vector used in the state update equation is as follows,

$$\mathbf{y}_{\mathbf{\theta}} = \mathbf{h}_{\mathbf{\theta}}(\mathbf{x},t) = \boldsymbol{\varepsilon} = \frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \tag{B.34}$$

where

$$K=\sqrt{1+\frac{1}{64}\theta^4} \tag{B.35}$$

The linearized version used in the gain and covariance calculations is derived as follows,

$$\mathbf{H}_{\boldsymbol{\Theta},k}=\frac{\partial\mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial\mathbf{x}^T}\bigg|_{\mathbf{x}=\hat{\mathbf{x}}_k}=\left[\frac{\partial\mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial\boldsymbol{\theta}^T}\quad\frac{\partial\mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial\boldsymbol{\omega}_b{}^T}\right] \tag{B.36}$$

From

$$\boldsymbol{\varepsilon}=\frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \tag{B.37}$$

it is found....

$$\frac{\partial\mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial\boldsymbol{\omega}_b{}^T}=\mathbf{0} \tag{B.38}$$

$$\frac{\partial\mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial\boldsymbol{\theta}^T}=\frac{1}{2}\mathbf{1} \tag{B.39}$$

Therefore,

$$\mathbf{H}_{\boldsymbol{\Theta},k}=\left[\frac{1}{2}\mathbf{1}_{3x3}\ \vdots\ \mathbf{0}_{3x3}\right]_{3x6} \tag{B.40}$$

B.5 Calculating the Filter Gain

Standard form is

$$\mathbf{K}_k=\mathbf{P}_k^-\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T+\mathbf{R}_k\right]^{-1} \tag{B.41}$$

B.6 The (Relative) State Estimate

For the state update,

$$\begin{aligned}\hat{\underset{\sim}{\mathbf{x}}}^{+}_{\boldsymbol{\theta},k} &= \hat{\underset{\sim}{\mathbf{x}}}^{-}_{\boldsymbol{\theta},k} + \mathbf{K}_{\boldsymbol{\theta},k}[\mathbf{z}_{\boldsymbol{\theta},k} - \mathbf{h}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{-}_{k}, t_k)] \\ \hat{\underset{\sim}{\mathbf{x}}}^{+}_{\boldsymbol{\theta},k} &= \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\omega}}^{-}_{b,k} \end{bmatrix} + \mathbf{K}_{\boldsymbol{\theta},k}[\mathbf{z}_{\boldsymbol{\theta},k} - \frac{1}{K}\left(\frac{1}{2}\hat{\boldsymbol{\theta}}^{-}_{k}\right)]\end{aligned} \tag{B.42}$$

So,

$$\hat{\underset{\sim}{\mathbf{x}}}^{+}_{\boldsymbol{\theta},k} = \begin{bmatrix} \hat{\boldsymbol{\theta}}^{+}_{k} \\ \hat{\boldsymbol{\omega}}^{+}_{b,k} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\omega}}^{-}_{b,k} \end{bmatrix} + \mathbf{K}_{\boldsymbol{\theta},k}\mathbf{z}_{\boldsymbol{\theta},k} \tag{B.43}$$

B.7 Update of the State Error Covariance

Shown here is the standard Joseph form often used for computation.

$$\mathbf{P}_k^{+} = [\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]\mathbf{P}_k^{-}[\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]^{T} + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{T} \tag{B.44}$$

B.8 Reset of the Absolute Orientation State

The reference orientation is moved along with the current (relative) state estimate.

Build a proper rotation matrix from the small rotation vector. The quaternion is used for this because they are easy to normalize.

$$\hat{\theta}_k^{2} = (\hat{\boldsymbol{\theta}}_k^{+})^{T}\hat{\boldsymbol{\theta}}_k^{+} \tag{B.45}$$

$$\delta\hat{\eta}_k = \frac{1}{K}\left(1 - \frac{1}{8}\hat{\theta}_k^{2}\right) \tag{B.46}$$

$$\delta\hat{\boldsymbol{\varepsilon}}_k = \frac{1}{K}\left(\frac{1}{2}\hat{\boldsymbol{\theta}}_k^{+}\right) \tag{B.47}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\delta\hat{\theta}_k^4} \tag{B.48}$$

Assembled this is,

$$\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) = \begin{bmatrix} \delta\hat{\eta}_k \\ \delta\hat{\boldsymbol{\varepsilon}}_k \end{bmatrix} \tag{B.49}$$

then

$$\begin{aligned} \hat{\mathbf{q}}_{\mathrm{ref},\,k}^+ &= \hat{\mathbf{q}}_{\mathrm{ref},\,k}^- \oplus \delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) \\ \hat{\mathbf{q}}_{\mathrm{ref},\,k}^+ &= \Xi(\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+))\ \hat{\mathbf{q}}_{\mathrm{ref},\,k}^- \end{aligned} \tag{B.50}$$

Appendix C Attitude Initialization Baseline Setup

EKF TESTING INITIALIZATION REFERENCE TABLE			
TEST REFERENCE NO. *Baseline Setup*			
Filter Type: MEKF			
Filter Sampling Interval: KALMAN_DT_SAMPLE = 1.0 [sec]			
Propagation Type: State Estimate (RK4 Solver) / Covariance (Discrete $\mathbf{P}_{k+1}^{-} = \mathbf{\Phi}_k \mathbf{P}_k^{+} \mathbf{\Phi}_k + \mathbf{Q}_k$)			
Number of Steps: State Estimate (20) / Covariance (20)			
Update Type: MEKF (with Joseph Form)			
Total Simulation Time: 10000 [sec]			
TRUTH **Variables and Parameters**		**FILTER** **Variables and Parameters**	
Symbol	***Numerical value***	***Symbol***	***Numerical value***
$\{\phi,\theta,\psi\}_{\text{ref}}$ at $(t=0)$ $\mathbf{q}_{\text{ref}}(t_0)$	*{0,0,0}* deg EULER0 = 1 0 0 0	$\{\phi,\theta,\psi\}_{\text{ref}}$ at $(t=0)$ $\hat{\mathbf{q}}_{\text{ref}}(t_0)$	*{10,0,0}* deg EULER0_ref = 0.99619 0.087156 0 0

$\boldsymbol{\omega}_b(t_0)$	`wx = 1.0 * DEG;`
	`wy = 0.0 * DEG;`
	`wz = 0.5 * DEG;`
		$\hat{\boldsymbol{\omega}}_b(t_0)$	`wx_est = 1.0 * DEG;`
			`wy_est = 0.1 * DEG;`
			`wz_est = 0.5 * DEG;`

	W_0 = 0.017453 0 0.0087266		W_0_est = 0.017453 0.0017453 0.0087266
		$\underset{\sim}{\mathbf{x}}_{\boldsymbol{\Theta}}(t_0)=\begin{bmatrix}\boldsymbol{\theta}_0\\ \boldsymbol{\omega}_{b,0}\end{bmatrix}$	KFilter_INIT_STATE = 0 0 0 0.017453 0.0017453 0.0087266
I (2nd Mom. Inertia)	>> INERTIA INERTIA = 1462 0 0 0 790.89 0 0 0 511.56	$\hat{\mathbf{I}}$ (2nd Mom. Inertia)	>> INERTIA INERTIA = 1462 0 0 0 790.89 0 0 0 511.56
		$\mathbf{P}_{\boldsymbol{\Theta}}(t_0)$	$\hat{\sigma}_{\theta_x}=10$ deg $\hat{\sigma}_{\theta_y}=10$ deg $\hat{\sigma}_{\theta_z}=10$ deg $\hat{\sigma}_{\omega_x}=5$ deg/s $\hat{\sigma}_{\omega_y}=5$ deg/s $\hat{\sigma}_{\omega_z}=5$ deg/s KFilter_P0 = 0.030462 0 0 0 0 0 0 0.030462 0 0 0 0 0 0 0.030462 0 0 0 0 0 0 0.0076154 0 0 0 0 0 0 0.0076154 0 0 0 0 0 0 0.0076154

$\mathbf{v}_{\Theta,k}$ (associated with $\delta\Theta_{mg}$)	$\sigma_{v(\phi)} = 0.2294$ deg $\sigma_{v(\theta)} = 0.6882$ deg $\sigma_{v(\psi)} = 0.6882$ deg	$\mathbf{R}_{\Theta,m}$ (associated with $\delta\Theta_{mg}$)	$\hat{\sigma}_{v(\theta_x)} = \hat{\sigma}_{v(\phi)} = 6.0$ deg $\hat{\sigma}_{v(\theta_y)} = \hat{\sigma}_{v(\theta)} = 6.0$ deg $\hat{\sigma}_{v(\theta_z)} = \hat{\sigma}_{v(\psi)} = 6.0$ deg KFilter_Rm = 0.0027416 0 0 0 0.0027416 0 0 0 0.0027416
$\mathbf{w}_{\Theta}$ (associated with $\mathbf{g}_{\text{disturb}}$)	$\mathbf{g}_{\text{disturb}}$ = 1e-5 N.m	$\mathbf{Q}_{\Theta,c}$	$\hat{\sigma}^2_{w(g_x)} = (3.2e\text{-}3)^2 (\text{N.m})^2.\ \text{s}$ $\hat{\sigma}^2_{w(g_y)} = (3.2e\text{-}3)^2 (\text{N.m})^2.\ \text{s}$ $\hat{\sigma}^2_{w(g_z)} = (3.2e\text{-}3)^2 (\text{N.m})^2.\ \text{s}$ KFilter_Qc = 1.024e-005 0 0 0 1.024e-005 0 0 0 1.024e-005
		$\mathbf{H}_{\Theta,k} = \dfrac{\partial \mathbf{h}_{\Theta}[\cdot]}{\partial \mathbf{x}^T}$	$\mathbf{H}_{\Theta,k} = \left[\frac{1}{2}\mathbf{1}_{3x3} \ \vdots \ \mathbf{0}_{3x3} \right]_{3x6}$ KFilter_H = 0.5 0 0 0 0 0 0 0.5 0 0 0 0 0 0 0.5 0 0 0
		$\mathbf{L}_{\Theta} = \dfrac{\partial \mathbf{f}_{\Theta}[\cdot]}{\partial \mathbf{w}^T}$	$\mathbf{L}_{\Theta} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}^{-1} \end{bmatrix}$ KFilter_L = 0 0 0 0 0 0

			0 0 0 0.00068401 0 0 0 0.0012644 0 0 0 0.0019548
		$\mathbf{F}_{\Theta} = \frac{\partial \mathbf{f}_{\Theta}[\cdot]}{\partial \mathbf{x}^{T}}$	$\mathbf{F}_{\Theta} = \begin{bmatrix} -\hat{\boldsymbol{\omega}}_{b}^{\times} & \vdots & \mathbf{1}_{3x3} \\ \cdots & \vdots & \cdots \\ \mathbf{0}_{3x3} & \vdots & \mathbf{I}^{-1}\left[(\mathbf{I}\hat{\boldsymbol{\omega}}_{b})^{\times} - \hat{\boldsymbol{\omega}}_{b}^{\times}\mathbf{I}\right] \end{bmatrix}$
$\mathbf{C}_{mi}(t_0)$	C_mi_0 = 1 0 0 0 1 0 0 0 1		

Appendix D Filter Tuning Literature Review

An *Extended Kalman Filter (EKF)* can be a near-optimal estimator if implemented correctly. However, to achieve acceptable performance the filter must be properly tuned to avoid *poor performance* or even *divergence.* The following provides excerpts and recommendations from various literature sources found during the author's search regarding poor EKF performance and divergence.

The author is definitely not the first to encounter *poor performance* or even *divergence* during initial testing of an EKF. Some have even chosen to develop there own methods due to frustration with the EKF. In a recent paper Julier and Uhlmann , the creators of the *Unscented Kalman Filter (UKF),* state the following,

> *"The extended Kalman filter (EKF) is probably the most widely used estimation algorithm for nonlinear systems. However, more than 35 years of experience in the estimation community has shown that it is difficult to implement, difficult to tune, and only reliable for systems that are almost linear on the time scale of the updates. Many of these difficulties arise from its use of linearization."* [34]

However, despite the pessimism in the above quotation, Markley a co-inventor of one of the most successful implementations of the EKF applied to attitude determination, namely the *Multiplicative Extended Kalman Filter (MEKF),* conducts a lengthy survey of various nonlinear attitude filtering methods and reports [19],

> *"Although the new approaches surveyed here have been shown to have some advantages, it is wise to apply the old adage "if it ain't broke don't fix it" to the standard extended Kalman filter, which has proved its worth on a multitude of spacecraft missions."*

D.1 Poor Performance and Filter Divergence

From page 467 of [18] the phenomenon of filter divergence is stated as,

"A Kalman filter achieves a steady state when the corrections to the state vector reach a consistent level and when the error covariance matrix is stable. Divergence occurs when the estimated state moves away from the true state. This is a common problem associated with Kalman filters. The most frequent causes of Kalman filter divergence are linearization errors, cumulative roundoff and truncation errors, modeling errors, and unknown noise statistics."

In [18] it is stated that linearization problems can be reduced by local iteration or more frequent selection of observations. Numerical problems may be partially solved by using a different but theoretically equivalent filter implementation such as the *U-D Factorization* implementation. Problems associated with noise statistics,

*"may be solved after extensive testing with both simulated and real data. Proper filter response will only result when the appropriate **balance** between the state noise and measurement noise covariance matrices is found."*

The state noise and measurement noise covariance matrices mentioned above are precisely, $\mathbf{Q}$ and $\mathbf{R}$. Reference [18] goes on the say,

"If, for example, state noise has been underestimated with respect to observation noise, the state estimation procedure will become less and less sensitive to the observation residuals. Divergence could then result even though the filter may have reached a steady state. Alternatively, if observation noise has been underestimated, the state estimation procedure may be incorrectly influenced by the observation errors."

In another reference, [32], the following definition is given,

"Filter divergence is often explained in terms of the calculated covariance matrix, the assertion being that this matrix becomes unrealistically small and results in placing unreasonable confidence in the estimates. Consequently, the filter gain is reduced and subsequent measurements are effectively ignored."

The author in [32] then goes on to further classify two different types of divergence,

"In some cases of divergence, the true error converges to a finite value; this is known as an Apparent Divergence. True Divergence represents the most critical type of divergence in filter design. This type of divergence means that the true error becomes arbitrarily large as time increases while the estimated error converges to a finite value."

Maybeck [35] shows an example of poor EKF performance and notes the following,

*"The preceding example revealed a bias error in the state estimate produced by an extended Kalman filter. This is very characteristic of extended Kalman filters, and it is due to the neglected higher order effects inherent in trying to exploit linear perturbation concepts. The more pronounced the nonlinearities are in a given application, the more seriously one can expect performance to be degraded by this effect. This can be compensated to some degree by "**tuning**" the filter such that its internally computed error variances match the true mean squared errors as indicated in a Monte Carlo performance analysis."*

The importance of balancing the state noise and measurement noise covariance matrices ($\mathbf{Q}$ and $\mathbf{R}$) or proper "***tuning***" of the filter will be explored in the next section.

A multitude of other references explore the KF and EKF divergence problems, these can be seen in [36-44].

D.2 Filter Tuning

Assuming the effects of numerical round-off and truncation are not an issue, the literature is exploted to find out how to attain a balancing of the state noise and measurement noise covariance matrices ($\mathbf{Q}$ and $\mathbf{R}$) such that acceptable filter performance is achieved.

From Maybeck [7],

"The basic objective of filter tuning is to achieve the best possible estimation performance from a filter of specified structural form, i.e. totally specified except for $\mathbf{P}_0$ *and the time histories of* $\mathbf{Q}$ and $\mathbf{R}$ *.....Thus, the tuning process can be considered a numerical optimization problem.... Manual "optimization" is more*

prevalent in practice. Basically, the $\mathbf{P}_0$ *matrix is the determining factor in the initial transient performance of the filter, where the* $\mathbf{Q}$ and $\mathbf{R}$ *histories dictate the longer term ("steady state" if time-invariant system and stationary noise models apply) performance and time duration of transients. "*

One might purpose that these noise covariance matrices ($\mathbf{Q}$ and $\mathbf{R}$) be based on the magnitude of physical or "actual" noise being injected into the system. Although the latter statement is often a starting point, simply using $\mathbf{Q}$ and $\mathbf{R}$ to match the "actual" noise injected into the system can often give poor performance. From Maybeck, the tuning should start by [7],

".. it can be solved by automatic search methods or by manual calculation: using physical insights to propose changes in the noise covariances, and continuing to vary them until the performance no longer improves."

This is discussed further in [45],

"..whether or not we have a rational basis for choosing the parameters, often times superior performance (statistically speaking) can be obtained by tuning the filter parameters $\mathbf{Q}$ and $\mathbf{R}$ *"*

and in [46],

"The magnitude of the noise in an EKF is therefore extremely difficult to estimate. It is therefore surprising that the issue of learning noise terms remains largely unexplored in the literature. A notable exception is the filter tuning literature.... when choosing the noise parameters for an EKF, we are interested in choosing parameters that lead to the EKF outputting accurate state estimates, rather than necessarily choosing the noise parameters that most correctly reflects each measurement's true variance.."

D.3 Why Process Noise Q is Required

In [27], a useful discussion is found regarding why process noise covariance $\mathbf{Q}$ is required,

"... setting the process noise to zero eventually causes the filter to stop paying attention to new measurements (i.e., filter bandwidth is reduced or the filter gain is too small for incorporating more measurements). In other words, a zero process noise [covariance] extended Kalman filter eventually goes to sleep!"

As mentioned above though, achieving a balanced **Q** with respect to **R** for acceptable performance is not a simple selection task. The authors in [27] re-iterate that of Maybeck by stating,

"Although the amount of process noise [covariance] to use is often determined experimentally, a good starting point is that the amount of process noise that a Kalman filter requires should reflect our estimate of our lack of knowledge of the real world."

Another discussion regarding the process noise covariance **Q** is given in [32],

"The increased noise covariance offsets modeling errors and improves filter stability. As noted in previous examples, if $\mathbf{Q} = 0$*, then the Kalman gain may approach zero and thus not use any additional measurements in the state propagation equation. The effect of* $\mathbf{Q} > 0$ *causes* $\mathbf{P} > 0$*. In other words, a nonzero lower bound implies a nonzero Kalman gain. An increase in the error estimate and poor performance will be the effects of an arbitrary selection of* $\mathbf{Q} > 0$*.*

The authors in [26] go so far as to approach that selection of **Q** and **R** by using a genetic algorithm.

D.4 General Recommendations

Finding a balance between **Q** and **R** is an essential step for a properly operating EKF. From the general recommendations in the literature this tuning process should starting by choosing noise covariance values based on physical insights, then varying them in an automatic or manual search until acceptable performance is achieved.

Appendix E Q and R Parametric Sweep for Attitude MEKF

Count	Sim Length	Qc	Rm(DEG)	max(dPHI)	mean(dPHI)	std(dPHI)
1	5000	2.00E-04	2.00E+00	17.1399	7.9622	4.1535
2	5000	2.00E-04	4.00E+00	18.8632	7.9632	4.5584
3	5000	2.00E-04	6.00E+00	10.9405	4.7325	2.6884
4	5000	4.00E-04	2.00E+00	11.2964	3.9294	2.4517
5	5000	4.00E-04	4.00E+00	13.2316	6.1598	3.1605
6	5000	4.00E-04	6.00E+00	9.5755	4.2258	2.2879
7	5000	6.00E-04	2.00E+00	9.6422	3.0674	2.0212
8	5000	6.00E-04	4.00E+00	10.1298	3.8554	2.2767
9	5000	6.00E-04	6.00E+00	7.2882	3.4316	1.6988
10	5000	8.00E-04	2.00E+00	9.2645	2.5117	1.9536
11	5000	8.00E-04	4.00E+00	8.709	3.0779	1.9175
12	5000	8.00E-04	6.00E+00	6.0565	2.5128	1.319
13	5000	1.00E-03	2.00E+00	9.496	1.8512	1.7831
14	5000	1.00E-03	4.00E+00	7.9157	2.7173	1.7489
15	5000	1.00E-03	6.00E+00	5.3813	1.9431	1.13
16	5000	1.20E-03	2.00E+00	21.234	4.3971	3.971
17	5000	1.20E-03	4.00E+00	7.8127	2.4528	1.6015
18	5000	1.20E-03	6.00E+00	4.8159	1.8139	1.0734
19	5000	1.40E-03	2.00E+00	179.6538	55.8276	48.9333
20	5000	1.40E-03	4.00E+00	8.2093	2.3851	1.7851
21	5000	1.40E-03	6.00E+00	4.5125	1.728	1.0205
22	5000	1.60E-03	2.00E+00	179.9632	63.5071	51.6025
23	5000	1.60E-03	4.00E+00	7.3961	2.0192	1.5455
24	5000	1.60E-03	6.00E+00	4.3973	1.5555	0.968
25	5000	1.80E-03	2.00E+00	179.964	63.5401	50.8256
26	5000	1.80E-03	4.00E+00	7.398	1.8091	1.5814

27	5000	1.80E-03	6.00E+00	4.8306	1.5191	0.9436
28	5000	2.00E-03	2.00E+00	179.7765	64.6716	49.511
29	5000	2.00E-03	4.00E+00	7.6446	1.4453	1.3431
30	5000	2.00E-03	6.00E+00	4.9993	1.5079	1.0549
31	5000	2.20E-03	2.00E+00	179.9046	67.3235	48.647
32	5000	2.20E-03	4.00E+00	7.922	1.3029	1.3065
33	5000	2.20E-03	6.00E+00	4.4178	1.4401	0.9974
34	5000	2.40E-03	2.00E+00	179.8547	70.2664	48.0122
35	5000	2.40E-03	4.00E+00	18.7148	4.76	4.2549
36	5000	2.40E-03	6.00E+00	4.464	1.2739	0.9011
37	5000	2.60E-03	2.00E+00	179.9311	70.3012	51.4278
38	5000	2.60E-03	4.00E+00	72.3863	10.1476	12.3909
39	5000	2.60E-03	6.00E+00	4.4072	1.2192	0.8532
40	5000	2.80E-03	2.00E+00	179.9266	84.2329	54.9036
41	5000	2.80E-03	4.00E+00	179.8486	53.0863	50.409
42	5000	2.80E-03	6.00E+00	4.5071	1.014	0.8763
43	5000	3.00E-03	2.00E+00	179.9722	70.5392	48.4778
44	5000	3.00E-03	4.00E+00	179.9507	59.9652	52.0924
45	5000	3.00E-03	6.00E+00	4.6158	0.8822	0.6967
46	5000	3.20E-03	2.00E+00	179.88	66.9261	47.9108
47	5000	3.20E-03	4.00E+00	179.975	60.5226	53.6801
48	5000	3.20E-03	6.00E+00	4.7337	0.8192	0.73
49	5000	3.40E-03	2.00E+00	179.9531	67.5224	46.9753
50	5000	3.40E-03	4.00E+00	179.8988	62.3684	49.9891
51	5000	3.40E-03	6.00E+00	5.4736	1.4107	1.1753
52	5000	3.60E-03	2.00E+00	179.9776	77.649	50.4929
53	5000	3.60E-03	4.00E+00	179.9211	62.6333	51.2863
54	5000	3.60E-03	6.00E+00	21.6635	4.7163	4.7077
55	5000	3.80E-03	2.00E+00	179.8586	66.437	45.6328
56	5000	3.80E-03	4.00E+00	179.8628	63.5019	50.6609
57	5000	3.80E-03	6.00E+00	81.013	12.1591	16.1123
58	5000	4.00E-03	2.00E+00	179.8445	66.4018	45.1449
59	5000	4.00E-03	4.00E+00	179.8835	63.8828	49.733
60	5000	4.00E-03	6.00E+00	124.4769	17.6316	24.4158
61	5000	4.20E-03	2.00E+00	179.9614	66.4861	44.633

62	5000	4.20E-03	4.00E+00	179.9759	64.2983	49.2177
63	5000	4.20E-03	6.00E+00	70.029	10.7697	13.4077

Appendix F Dynamic Motion Filter Equations – Inertial Context

The implementation of the dynamic motion filter (in an inertial context) is summarized as follows:

F.1 Initial Conditions

At time t_0 ....

The Absolute Filter State

$$\hat{\mathbf{x}}_0 = \hat{\mathbf{x}}(t_0) = \begin{bmatrix} \hat{\mathbf{q}}_b(0) \\ \hat{\boldsymbol{\omega}}_b(0) \\ \hat{\mathbf{r}}_c(0) \\ \hat{\mathbf{v}}_c(0) \\ \hat{\mathbf{r}}_{cg,g}(0) \end{bmatrix} \tag{F.1}$$

The Internal Filter State

$$\hat{\underset{\sim}{\mathbf{x}}}_0 = \hat{\underset{\sim}{\mathbf{x}}}(t_0) = \begin{bmatrix} \hat{\boldsymbol{\theta}}_b(0) \\ \hat{\boldsymbol{\omega}}_b(0) \\ \hat{\mathbf{r}}_c(0) \\ \hat{\mathbf{v}}_c(0) \\ \hat{\mathbf{r}}_{cg,g}(0) \end{bmatrix} \tag{F.2}$$

The S-function State

$$\mathbf{x}_{\text{S-func}}(t_k) = \begin{bmatrix} \underset{\sim}{\hat{\mathbf{x}}}_k^+ \\ \underset{\sim}{\hat{\mathbf{x}}}_k^- \\ \mathbf{P}_k^+ \\ \mathbf{P}_k^- \\ \hat{\mathbf{q}}_{\text{ref},k}^- \end{bmatrix} \tag{F.3}$$

where $\hat{\boldsymbol{\theta}}_k^- \equiv \mathbf{0}$ by definition.

Next...

F.2 Propagating the State Estimate

Integrate starting from $\mathbf{x}_{k-1}^+$ to get $\mathbf{x}_k^-$

The propagation is carried out using the following equations of motion,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u},\mathbf{w},t) = \begin{bmatrix} \dot{\mathbf{q}} \\ \dot{\boldsymbol{\omega}}_b \\ \dot{\mathbf{r}}_c \\ \dot{\mathbf{v}}_c \\ \dot{\mathbf{r}}_{cg,g} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Xi}(\boldsymbol{\varpi})\mathbf{q} \\ \mathbf{I}^{-1}\left[\mathbf{g}_{\text{ext}} - \boldsymbol{\omega}_b{}^{\times}\mathbf{I}\boldsymbol{\omega}_b\right] \\ \mathbf{v}_c \\ \mathbf{f}_{,i}/m \\ \mathbf{0} \end{bmatrix} \tag{F.4}$$

where, as defined earlier,

$$\boldsymbol{\varpi} = \begin{bmatrix} 0 \\ \dfrac{1}{2}\boldsymbol{\omega}_b \end{bmatrix} \tag{F.5}$$

and

$$\boldsymbol{\Xi}(\mathbf{q}) = \begin{bmatrix} \eta & -\boldsymbol{\varepsilon}^T \\ \boldsymbol{\varepsilon} & \eta\mathbf{1} - \boldsymbol{\varepsilon}^{\times} \end{bmatrix} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \tag{F.6}$$

The acting forces and torques in (F.1) can be partitioned as $\mathbf{f}_{,i} = \mathbf{f}_{\text{known}} + \mathbf{f}_{\text{disturb}}$ and $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}} + \mathbf{g}_{\text{disturb}}$. However, for the filter propagation the dynamics are driven with only known inputs such that $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}}$ and $\mathbf{f}_{,i} = \mathbf{f}_{\text{known}}$.

In terms of the reference state estimate, $\hat{\mathbf{x}}(t_k)$,

$$\hat{\mathbf{x}}_k^- = \hat{\mathbf{x}}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \mathbf{f}\{\hat{\mathbf{x}}[\tau(-)],\mathbf{u}(\tau),\tau\}d\tau \qquad \text{(F.7)}$$

F.3 Extrapolating the State-Error Covariance

Propagate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{P}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \left[\mathbf{F}(\tau)\mathbf{P}(\tau) + \mathbf{P}(\tau)\mathbf{F}^T(\tau) + \mathbf{L}(\tau)\mathbf{Q}_c(\tau)\mathbf{L}^T(\tau)\right]d\tau \qquad \text{(F.8)}$$

where,

$$\mathbf{F}(\tau) = \frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(\tau),\mathbf{u}(\tau),\tau)\bigg|_{\mathbf{x}=\hat{\mathbf{x}}} \qquad \text{(F.9)}$$

$$\mathbf{L}(\tau) = \frac{\partial}{\partial \mathbf{w}}\mathbf{f}(\mathbf{x}(\tau),\mathbf{u}(\tau),\mathbf{w}(\tau),\tau)\bigg|_{\mathbf{x}=\hat{\mathbf{x}}} \qquad \text{(F.10)}$$

or alternatively extrapolate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^+\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \qquad \text{(F.11)}$$

where the state transition matrix, $\mathbf{\Phi}_k$, is the Taylor-series expansion of $e^{\mathbf{F}T_s}$,

$$\mathbf{\Phi}_k = \mathbf{1} + \mathbf{F}T_s + \frac{\mathbf{F}^2 T_s^2}{2!} + \frac{\mathbf{F}^3 T_s^3}{3!} + ... \qquad \text{(F.12)}$$

The discrete process noise covariance matrix, $\mathbf{Q}_k$, can be computed from the continuous-time process noise spectral density matrix, $\mathbf{Q}_c$, from the following [27],

$$\mathbf{Q}_k = \int_0^{T_s} \mathbf{\Phi}_{k-1}(\tau) \mathbf{L} \mathbf{Q}_c \mathbf{L}^T \mathbf{\Phi}_{k-1}^T(\tau) d\tau \qquad \text{(F.13)}$$

The discrete-time form will be less computationally expensive but generally less accurate.

For the particular dynamic motion filter state in the inertial context one can often derive explicit expressions for the Jacobians, State Transition matrix, and Process Noise Covariance matrix.

JACOBIAN **F**

To evaluate the Jacobian derivatives associated with $\dot{\boldsymbol{\theta}}$, the following relationship is noted: [33]

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times} \right] \delta\boldsymbol{\omega} \qquad \text{(F.14)}$$

where $\delta\boldsymbol{\omega}$ is the relative angular velocity from the reference orientation $\mathbf{q}_{\text{ref}}$ to the net orientation $\mathbf{q}_b$. The following can be written:

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times} \right] (\boldsymbol{\omega} - \delta\mathbf{C}\boldsymbol{\omega}_{\text{ref}}) \qquad \text{(F.15)}$$

By applying the approximation,

$$\delta\mathbf{C} \approx \delta\mathbf{C}(\boldsymbol{\theta}) = \mathbf{1} - \boldsymbol{\theta}^{\times} + \frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times} \qquad \text{(F.16)}$$

the following relation exists, valid to second-order,

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1}+\frac{1}{2}\boldsymbol{\theta}^{\times}+\frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right]\left(\boldsymbol{\omega}-\left(\mathbf{1}-\boldsymbol{\theta}^{\times}+\frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right)\boldsymbol{\omega}_{\mathrm{ref}}\right) \qquad \text{(F.17)}$$

To first-order in the small $\boldsymbol{\theta}$,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^{T}} = -\frac{1}{2}\boldsymbol{\omega}_{b}^{\times}+\frac{1}{12}\left[\boldsymbol{\omega}_{b}^{\times}\boldsymbol{\theta}^{\times}-2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_{b}^{\times}\right]-\frac{1}{2}\boldsymbol{\omega}_{\mathrm{ref}}^{\times}-\frac{1}{12}\left[\boldsymbol{\omega}_{\mathrm{ref}}^{\times}\boldsymbol{\theta}^{\times}-2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_{\mathrm{ref}}^{\times}\right] \qquad \text{(F.18)}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_{b}^{T}} = \mathbf{1}+\frac{1}{2}\boldsymbol{\theta}^{\times} \qquad \text{(F.19)}$$

However, when implemented according to the MEKF procedure, the expressions are evaluated such that $\hat{\boldsymbol{\theta}}=\mathbf{0}$ and $\hat{\boldsymbol{\omega}}_{b}=\boldsymbol{\omega}_{\mathrm{ref}}$, hence the much simpler forms,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^{T}} = -\boldsymbol{\omega}_{\mathrm{ref}}^{\times} \qquad \text{(F.20)}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_{b}^{T}} = \mathbf{1} \qquad \text{(F.21)}$$

The complete Jacobian for $\mathbf{F}=\dfrac{\partial\mathbf{f}[\cdot]}{\partial\underset{\sim}{\mathbf{x}}^{T}}$ is as follows:

$$\mathbf{F}=\frac{\partial \mathbf{f}[\cdot]}{\partial \underset{\sim}{\mathbf{x}}^{T}}=\begin{bmatrix} \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}^{T}} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \boldsymbol{\omega}_{b}^{T}} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{r}_{c}^{T}} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{v}_{c}^{T}} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{r}_{cg,g}^{T}} \\ \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \boldsymbol{\theta}^{T}} & \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \boldsymbol{\omega}_{b}^{T}} & \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \mathbf{r}_{c}^{T}} & \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \mathbf{v}_{c}^{T}} & \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \mathbf{r}_{cg,g}^{T}} \\ \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \boldsymbol{\theta}^{T}} & \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \boldsymbol{\omega}_{b}^{T}} & \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \mathbf{r}_{c}^{T}} & \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \mathbf{v}_{c}^{T}} & \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \mathbf{r}_{cg,g}^{T}} \\ \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \boldsymbol{\theta}^{T}} & \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \boldsymbol{\omega}_{b}^{T}} & \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \mathbf{r}_{c}^{T}} & \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \mathbf{v}_{c}^{T}} & \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \mathbf{r}_{cg,g}^{T}} \\ \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \boldsymbol{\theta}^{T}} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \boldsymbol{\omega}_{b}^{T}} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{r}_{c}^{T}} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{v}_{c}^{T}} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{r}_{cg,g}^{T}} \end{bmatrix}_{15x15}$$

$$=\begin{bmatrix} -\boldsymbol{\omega}_{\mathrm{ref}}^{\times} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}^{-1}\left[(\mathbf{I}\boldsymbol{\omega}_{b})^{\times}-(\boldsymbol{\omega}_{b}^{\times}\mathbf{I})\right] & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}_{15x15} \qquad \text{(F.22)}$$

JACOBIAN **L**

$$\mathbf{L}=\frac{\partial \mathbf{f}[\cdot]}{\partial \mathbf{w}^{T}}=\begin{bmatrix} \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{g}^{T}} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{f}^{T}} \\ \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \mathbf{g}^{T}} & \frac{\partial \dot{\boldsymbol{\omega}}_{b}}{\partial \mathbf{f}^{T}} \\ \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \mathbf{g}^{T}} & \frac{\partial \dot{\mathbf{r}}_{c}}{\partial \mathbf{f}^{T}} \\ \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \mathbf{g}^{T}} & \frac{\partial \dot{\mathbf{v}}_{c}}{\partial \mathbf{f}^{T}} \\ \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{g}^{T}} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{f}^{T}} \end{bmatrix}_{15x6}=\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{I}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{1}{m}\mathbf{1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}_{15x6} \qquad \text{(F.23)}$$

STATE TRANSITION MATRIX $\mathbf{\Phi}$

$$\mathbf{\Phi}_{k-1} \approx \mathbf{1} + \mathbf{F}T_s \tag{F.24}$$

PROCESS NOISE COVARIANCE MATRIX $\mathbf{Q}_k$

$$\mathbf{Q}_k = \int_0^{T_s} \mathbf{\Phi}_{k-1}(\tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\mathbf{\Phi}_{k-1}^T(\tau)d\tau \tag{F.25}$$

F.4 The Measurement and Output Equation

The measurement provided to the filter includes the *target* pose information in the form $\tilde{\mathbf{C}}_g$ and $\tilde{\mathbf{r}}_g$.

The measurement as viewed by the filter is related as follows,

$$\mathbf{z}_k = \begin{bmatrix} \tilde{\varepsilon}_k(\delta\hat{\mathbf{q}}_k) \\ \tilde{\mathbf{r}}_g \end{bmatrix} \tag{F.26}$$

The output vector used in the state update equation is as following,

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_\theta \\ \mathbf{y}_r \end{bmatrix} = \begin{bmatrix} \mathbf{h}_\theta(\mathbf{x},t) \\ \mathbf{h}_r(\mathbf{x},t) \end{bmatrix} = \begin{bmatrix} \varepsilon \\ \mathbf{r}_{g,i} \end{bmatrix} = \begin{bmatrix} \frac{1}{K}\left(\frac{1}{2}\mathbf{\theta}\right) \\ \mathbf{r}_{c,i} - \mathbf{C}_b^T\mathbf{r}_{cg,g} \end{bmatrix} \tag{F.27}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\theta^4} \tag{F.28}$$

The linearized version used in the gain and covariance calculations is derived as follows,

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} = \begin{bmatrix} \mathbf{H}_{\boldsymbol{\Theta},k} \\ \mathbf{H}_{\mathrm{r},k} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T} \\ \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T} \end{bmatrix} \tag{F.29}$$

The partition related to the orientation is...

$$\mathbf{H}_{\boldsymbol{\Theta},k} = \begin{bmatrix} \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\theta}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{c,i}{}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{cg,g}{}^T} \end{bmatrix} \tag{F.30}$$

From,

$$\boldsymbol{\varepsilon} = \frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \tag{F.31}$$

the following is found....

$$\frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{c,i}{}^T} = \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T} = \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{cg,g}{}^T} = \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\omega}_b{}^T} = \mathbf{0} \tag{F.32}$$

$$\frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\theta}^T} = \frac{1}{2}\mathbf{1} \tag{F.33}$$

For the partition related to the translation ...

$$\mathbf{H}_{\mathrm{r},k} = \begin{bmatrix} \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\theta}^T} & \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{c,i}{}^T} & \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T} & \dfrac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{cg,g}{}^T} \end{bmatrix} \tag{F.34}$$

From,

$$\mathbf{r}_{g,i} = \mathbf{r}_{c,i} - \mathbf{C}_b^T \mathbf{r}_{cg,g} \tag{F.35}$$

the follow relations are found....

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{c,i}{}^T}=\mathbf{1} \tag{F.36}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T}=\mathbf{0} \tag{F.37}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T}=-\mathbf{C}_b^T \tag{F.38}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T}=\mathbf{C}_{\mathrm{ref}}^T\mathbf{r}_{cg,g}^{\times} \tag{F.39}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T}=\mathbf{0} \tag{F.40}$$

Finally, all assembled the linearized measurement sensitivity matrix is given by,

$$\begin{aligned}
\mathbf{H}_k &= \left.\frac{\partial \mathbf{h}(\mathbf{x}_k,t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} \\
&= \begin{bmatrix} \mathbf{H}_{\boldsymbol{\Theta},k} \\ \mathbf{H}_{\mathrm{r},k} \end{bmatrix} \\
&= \begin{bmatrix}
\frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T} & \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{c,i}{}^T} & \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T} & \frac{\partial \mathbf{h}_{\boldsymbol{\Theta}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T} \\
\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T} & \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{c,i}{}^T} & \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \dot{\mathbf{r}}_{c,i}{}^T} & \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T}
\end{bmatrix} \\
&= \begin{bmatrix}
\frac{1}{2}\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{C}_{\mathrm{ref}}^T\mathbf{r}_{cg,g}^{\times} & \mathbf{0} & \mathbf{1} & \mathbf{0} & -\mathbf{C}_b^T
\end{bmatrix}
\end{aligned} \tag{F.41}$$

F.5 Calculating the Filter Gain

Standard form is

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left[\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k \right]^{-1} \tag{F.42}$$

where $\mathbf{R}_k$ is composed of the following,

$$\mathbf{R}_k = \begin{bmatrix} \mathbf{R}_{\theta,k} & 0 \\ 0 & \mathbf{R}_{\mathrm{r},k} \end{bmatrix}_{6x6} \tag{F.43}$$

with the assumption that $\mathbf{v}_{\theta,k}$ and $\mathbf{v}_{\mathrm{r},k}$ are not correlated.

Note that,

$$\mathbf{R}_{\theta,k} = \frac{1}{4} \mathbf{C}_{\mathrm{ref},m} \mathbf{R}_{\theta,m} \mathbf{C}_{\mathrm{ref},m}^T \tag{F.44}$$

where

$$\mathbf{C}_{\mathrm{ref}\,m} = \hat{\mathbf{C}}_{\mathrm{ref},k}^- \mathbf{C}_{mi}^T \tag{F.45}$$

and

$$\mathbf{R}_{\mathrm{r},k} = \mathbf{C}_{mi}^T \mathbf{R}_{\mathrm{r},m} \mathbf{C}_{mi} \tag{F.46}$$

F.6 The Update of the State Estimate

For the state estimate update,

$$\underset{\sim}{\hat{\mathbf{x}}}_k^+ = \underset{\sim}{\hat{\mathbf{x}}}_k^- + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-, t_k)]$$

$$\underset{\sim}{\hat{\mathbf{x}}}_k^+ = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\omega}}_b^- \\ \hat{\mathbf{r}}_c^- \\ \hat{\mathbf{v}}_c^- \\ \hat{\mathbf{r}}_{cg,g}^- \end{bmatrix}_k + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-, t_k)] \tag{F.47}$$

F.7 Update of the State Error Covariance

Shown here is the standard Joseph form often used for computation.

$$\mathbf{P}_k^+ = [\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]\mathbf{P}_k^-[\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{F.48}$$

F.8 Reset of the Absolute Orientation State

The reference orientation is moved along with the current (relative) state estimate.

Build a proper rotation matrix from the small rotation vector. The quaternion is used for this because it is easy to normalize.

$$\hat{\theta}_k^2 = (\hat{\boldsymbol{\theta}}_k^+)^T \hat{\boldsymbol{\theta}}_k^+ \tag{F.49}$$

$$\delta\hat{\eta}_k = \frac{1}{K}\left(1 - \frac{1}{8}\hat{\theta}_k^2\right) \tag{F.50}$$

$$\delta\hat{\boldsymbol{\varepsilon}}_k = \frac{1}{K}\left(\frac{1}{2}\hat{\boldsymbol{\theta}}_k^+\right) \tag{F.51}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\delta\hat{\theta}_k^4} \tag{F.52}$$

Assembled this gives,

$$\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) = \begin{bmatrix} \delta\hat{\eta}_k \\ \delta\hat{\boldsymbol{\varepsilon}}_k \end{bmatrix} \tag{F.53}$$

then

$$\begin{aligned} \hat{\mathbf{q}}_{\mathrm{ref},k}^+ &= \hat{\mathbf{q}}_{\mathrm{ref},k}^- \oplus \delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) \\ \hat{\mathbf{q}}_{\mathrm{ref},k}^+ &= \Xi(\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+))\ \hat{\mathbf{q}}_{\mathrm{ref},k}^- \end{aligned} \tag{F.54}$$

Appendix G Translational Initialization Baseline Setup (Inertial)

KF TESTING INITIALIZATION REFERENCE TABLE			
TEST REFERENCE NO. *Baseline Setup*			
Filter Type: KF			
Filter Sampling Interval: KALMAN_DT_SAMPLE = 1.0 [sec]			
Propagation Type: State Estimate (RK4 Solver) / Covariance ($\mathbf{P}_{k+1}^{-} = \mathbf{\Phi}_k \mathbf{P}_k^{+} \mathbf{\Phi}_k + \mathbf{Q}_k$)			
Number of Steps: State Estimate (20) / Covariance (20)			
Update Type: KF (with Joseph Form)			
Total Simulation Time: 10000 [sec]			
TRUTH **Variables and Parameters**		**FILTER** **Variables and Parameters**	
Symbol	***Numerical value***	***Symbol***	***Numerical value***
$\mathbf{r}_c(t_0)$	*[15;0;5]* m Rc_VEC_0 = 15 0 5	$\hat{\mathbf{r}}_c(t_0)$	*[20;5;1]* m Rc_VEC_0_EST = 20 5 1

$\mathbf{v}_c(t_0)$	*[0.2;0;0]* m/s Vc_VEC_0 = 0.2 0 0	$\hat{\mathbf{v}}_c(t_0)$	*[0.1;0;0]* m/s Vc_VEC_0_EST = 0.1 0 0
$\mathbf{r}_{cg,g}(t_0)$	*[0.05;0.05;0.0]* m r_cg_g = 0.05 0.05 0	$\hat{\mathbf{r}}_{cg,g}(t_0)$	*[0.048;0.052;0.001]* m r_cg_g_EST = 0.048 0.052 0.001
m (Mass, kg)	mMASS = 100	$\hat{m}$ (Mass, kg)	mMASS = 100
		$\mathbf{P}_\mathbf{r}(t_0)$	$\hat{\sigma}_{Rc_x} = 1.0$ m $\hat{\sigma}_{Rc_y} = 1.0$ m $\hat{\sigma}_{Rc_z} = 1.0$ m $\hat{\sigma}_{Vc_x} = 0.01$ m/s $\hat{\sigma}_{Vc_y} = 0.01$ m/s $\hat{\sigma}_{Vc_z} = 0.01$ m/s $\hat{\sigma}_{rc_x} = 0.01$ m $\hat{\sigma}_{rc_y} = 0.01$ m $\hat{\sigma}_{rc_z} = 0.01$ m KFilter_P0 = 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0.0001 0 0 0 0 0 0 0 0 0 0.0001 0 0 0 0 0 0 0 0 0 0.0001 0 0 0 0 0 0 0 0 0 0.0001 0 0 0 0 0 0 0 0 0 0.0001 0 0 0 0 0 0 0 0 0 0.0001

$\mathbf{v}_{\mathbf{r},k}$	$\sigma_{\mathbf{v}_x} = 0.05$ m $\sigma_{\mathbf{v}_y} = 0.05$ m $\sigma_{\mathbf{v}_z} = 0.05$ m	$\mathbf{R}_{\mathbf{r},m}$	$\hat{\sigma}_{\mathbf{v}_x} = 0.05$ m $\hat{\sigma}_{\mathbf{v}_y} = 0.\ 05$ m $\hat{\sigma}_{\mathbf{v}_z} = 0.\ 05$ m KFilter_Rm = 0.0025 0 0 0 0.0025 0 0 0 0.0025
$\mathbf{w}_{\mathbf{r}}$ (associated with $\mathbf{f}_{\text{disturb}}$)	$\mathbf{f}_{\text{disturb}} = \mathbf{0}$ N	$\mathbf{Q}_{\mathbf{r},c}$	$\hat{\sigma}^2_{\mathrm{w}(f_x)} = (5e\text{-}4)^2\,(N)^2.\ s$ $\hat{\sigma}^2_{\mathrm{w}(f_y)} = (5e\text{-}4)^2\,(N)^2.\ s$ $\hat{\sigma}^2_{\mathrm{w}(f_z)} = (5e\text{-}4)^2\,(N)^2.\ s$ KFilter_Qc = 2.5e-007 0 0 0 2.5e-007 0 0 0 2.5e-007
$\mathbf{C}_{mi}(t_0)$	C_mi_0 = 1 0 0 0 1 0 0 0 1		
$\mathbf{r}_g(t_0)$	$\mathbf{r}_g = \mathbf{r}_c - \mathbf{C}_b^T \mathbf{r}_{cg,g}$ rg = 14.95 -0.05 5		
$\mathbf{r}_v(t_0)$	$\mathbf{r}_v = \mathbf{r}_g - \mathbf{C}_{mi}^T \mathbf{r}_{gm,m}$ rv =		

	4.95 -0.05 5		

Appendix H Dynamic Motion Filter Equations – Orbital Context

The implementation of the dynamic motion filter (in an orbital context) is summarized as follows:

H.1 Initial Conditions

At time t_0 ....

The Absolute Filter State

$$\hat{\mathbf{x}}_0 = \hat{\mathbf{x}}(t_0) = \begin{bmatrix} \hat{\mathbf{q}}_b(0) \\ \hat{\boldsymbol{\omega}}_b(0) \\ \hat{\bar{\mathbf{r}}}(0) \\ \hat{\bar{\mathbf{v}}}(0) \\ \hat{\mathbf{r}}_{cg,g}(0) \end{bmatrix} \tag{H.1}$$

The Internal Filter State

$$\hat{\underset{\sim}{\mathbf{x}}}_0 = \hat{\underset{\sim}{\mathbf{x}}}(t_0) = \begin{bmatrix} \hat{\boldsymbol{\theta}}_b(0) \\ \hat{\boldsymbol{\omega}}_b(0) \\ \hat{\bar{\mathbf{r}}}(0) \\ \hat{\bar{\mathbf{v}}}_c(0) \\ \hat{\mathbf{r}}_{cg,g}(0) \end{bmatrix} \tag{H.2}$$

The S-function State

$$\mathbf{x}_{\text{S-func}}(t_k)=\begin{bmatrix}\underset{\sim}{\hat{\mathbf{x}}}_k^+\\ \underset{\sim}{\hat{\mathbf{x}}}_k^-\\ \mathbf{P}_k^+\\ \mathbf{P}_k^-\\ \hat{\mathbf{q}}_{\text{ref},k}^-\end{bmatrix} \tag{H.3}$$

where $\hat{\boldsymbol{\theta}}_k^- \equiv \mathbf{0}$ by definition.

Next...

H.2 Propagating the State Estimate

Integrate starting from $\mathbf{x}_{k-1}^+$ to get $\mathbf{x}_k^-$

The propagation is carried out using the following equations of motion,

$$\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x},\mathbf{u},\mathbf{w},t)=\begin{bmatrix}\dot{\mathbf{q}}\\ \dot{\boldsymbol{\omega}}_b\\ \dot{\bar{\mathbf{r}}}\\ \dot{\bar{\mathbf{v}}}\\ \dot{\mathbf{r}}_{cg,g}\end{bmatrix}=\begin{bmatrix}\Xi(\boldsymbol{\varpi})\mathbf{q}\\ \mathbf{I}^{-1}\left[\mathbf{g}-\boldsymbol{\omega}_b{}^{\times}\mathbf{I}\boldsymbol{\omega}_b\right]\\ \dot{\bar{\mathbf{r}}}\\ -\Omega_*^2(\hat{\mathbf{y}}\hat{\mathbf{y}}^T-3\hat{\mathbf{z}}\hat{\mathbf{z}}^T)\bar{\mathbf{r}}+2\Omega_*\hat{\mathbf{y}}^{\times}\dot{\bar{\mathbf{r}}}\\ \mathbf{0}\end{bmatrix} \tag{H.4}$$

where, as defined earlier,

$$\boldsymbol{\varpi}=\begin{bmatrix}0\\ \frac{1}{2}\boldsymbol{\omega}_b\end{bmatrix} \tag{H.5}$$

and

$$\Xi(\mathbf{q})=\begin{bmatrix}\eta & -\boldsymbol{\varepsilon}^T\\ \boldsymbol{\varepsilon} & \eta\mathbf{1}-\boldsymbol{\varepsilon}^{\times}\end{bmatrix}=\begin{bmatrix}q_0 & -q_1 & -q_2 & -q_3\\ q_1 & q_0 & q_3 & -q_2\\ q_2 & -q_3 & q_0 & q_1\\ q_3 & q_2 & -q_1 & q_0\end{bmatrix} \tag{H.6}$$

The acting torques in (H.1) can be partitioned as $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}} + \mathbf{g}_{\text{disturb}}$. However, for the filter propagation the dynamics are driven with only known inputs such that $\mathbf{g}_{\text{ext}} = \mathbf{g}_{\text{known}}$.

Note that the use of,

$$\begin{bmatrix} \dot{\bar{\mathbf{r}}} \\ \dot{\bar{\mathbf{v}}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ -\Omega_*^2(\hat{\mathbf{y}}\hat{\mathbf{y}}^T - 3\hat{\mathbf{z}}\hat{\mathbf{z}}^T) & 2\Omega_*\hat{\mathbf{y}}^\times \end{bmatrix} \begin{bmatrix} \bar{\mathbf{r}} \\ \bar{\mathbf{v}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} [\delta\mathbf{a}] \tag{H.7}$$

is based on the small relative motion from cylindrical coordinates as derived in [20]. In (H.7) note that $\hat{\mathbf{y}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$, $\hat{\mathbf{z}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$ and $\delta\mathbf{a}$ accounts for un-modeled orbital influences including any maneuvering thrust applied by the chaser. As mentioned earlier, in this study maneuvering is not considered and $\delta\mathbf{a}$ is assumed to consist of process noise only.

In using the direct Cartesian-like vector approach, essentially the states associated with $\bar{\mathbf{r}}$ and $\bar{\mathbf{v}}$ are,

$$\begin{aligned} x_1 &= R_*\theta \\ x_2 &= y \\ x_3 &= -r \\ x_4 &= R_*\dot{\theta} \\ x_5 &= \dot{y} \\ x_6 &= -\dot{r} \end{aligned} \tag{H.8}$$

In terms of the reference state estimate, $\hat{\mathbf{x}}(t_k)$,

$$\hat{\mathbf{x}}_k^- = \hat{\mathbf{x}}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \mathbf{f}\{\hat{\mathbf{x}}[\tau(-)], \mathbf{u}(\tau), \tau\} d\tau \tag{H.9}$$

H.3 Extrapolating the State-Error Covariance

Propagate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{P}_{k-1}^+ + \int_{t_{k-1}}^{t_k} \left[\mathbf{F}(\tau)\mathbf{P}(\tau) + \mathbf{P}(\tau)\mathbf{F}^T(\tau) + \mathbf{L}(\tau)\mathbf{Q}_c(\tau)\mathbf{L}^T(\tau)\right] d\tau \qquad \text{(H.10)}$$

where,

$$\mathbf{F}(\tau) = \frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau)\Big|_{\mathbf{x}=\hat{\mathbf{x}}} \qquad \text{(H.11)}$$

$$\mathbf{L}(\tau) = \frac{\partial}{\partial \mathbf{w}} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \mathbf{w}(\tau), \tau)\Big|_{\mathbf{x}=\hat{\mathbf{x}}} \qquad \text{(H.12)}$$

or alternatively extrapolate the covariance using the form,

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^+\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \qquad \text{(H.13)}$$

where the state transition matrix, $\mathbf{\Phi}_k$, is the Taylor-series expansion of $e^{\mathbf{F}T_s}$,

$$\mathbf{\Phi}_k = \mathbf{1} + \mathbf{F}T_s + \frac{\mathbf{F}^2 T_s^{\,2}}{2!} + \frac{\mathbf{F}^3 T_s^{\,3}}{3!} + ... \qquad \text{(H.14)}$$

The discrete process noise covariance matrix, $\mathbf{Q}_k$, can be computed from the continuous-time process noise spectral density matrix, $\mathbf{Q}_c$, from the following [27],

$$\mathbf{Q}_k = \int_0^{T_s} \mathbf{\Phi}_{k-1}(\tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\mathbf{\Phi}_{k-1}^T(\tau) d\tau \qquad \text{(H.15)}$$

The discrete-time form will be less computationally expensive but generally less accurate. For this particular dynamic motion filter state in the orbital context often one can derive explicit expressions for the Jacobians, State Transition matrix, and Process Noise Covariance matrix.

JACOBIAN **F**

To evaluate the Jacobian derivatives associated with $\dot{\boldsymbol{\theta}}$, the following relationship is noted: [33]

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right]\delta\boldsymbol{\omega} \tag{H.16}$$

where $\delta\boldsymbol{\omega}$ is the relative angular velocity from the reference orientation $\mathbf{q}_{\text{ref}}$ to the net orientation $\mathbf{q}_b$. The following can be written:

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right](\boldsymbol{\omega} - \delta\mathbf{C}\boldsymbol{\omega}_{\text{ref}}) \tag{H.17}$$

By applying the approximation,

$$\delta\mathbf{C} \approx \delta\mathbf{C}(\boldsymbol{\theta}) = \mathbf{1} - \boldsymbol{\theta}^{\times} + \frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times} \tag{H.18}$$

the following relation exists, valid to second-order,

$$\dot{\boldsymbol{\theta}} = \left[\mathbf{1} + \frac{1}{2}\boldsymbol{\theta}^{\times} + \frac{1}{12}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right]\left(\boldsymbol{\omega} - \left(\mathbf{1} - \boldsymbol{\theta}^{\times} + \frac{1}{2}\boldsymbol{\theta}^{\times}\boldsymbol{\theta}^{\times}\right)\boldsymbol{\omega}_{\text{ref}}\right) \tag{H.19}$$

To first-order in the small $\boldsymbol{\theta}$,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^{T}} = -\frac{1}{2}\boldsymbol{\omega}_b^{\times} + \frac{1}{12}\left[\boldsymbol{\omega}_b^{\times}\boldsymbol{\theta}^{\times} - 2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_b^{\times}\right] - \frac{1}{2}\boldsymbol{\omega}_{\text{ref}}^{\times} - \frac{1}{12}\left[\boldsymbol{\omega}_{\text{ref}}^{\times}\boldsymbol{\theta}^{\times} - 2\boldsymbol{\theta}^{\times}\boldsymbol{\omega}_{\text{ref}}^{\times}\right] \tag{H.20}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_b^T}=\mathbf{1}+\frac{1}{2}\boldsymbol{\theta}^{\times} \tag{H.21}$$

However, when implemented according to the MEKF procedure, the expressions are evaluated such that $\hat{\boldsymbol{\theta}}=\mathbf{0}$ and $\hat{\boldsymbol{\omega}}_b=\boldsymbol{\omega}_{\text{ref}}$, hence the much simpler forms,

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^T}=-\boldsymbol{\omega}_{\text{ref}}^{\times} \tag{H.22}$$

and

$$\frac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_b^T}=\mathbf{1} \tag{H.23}$$

The complete Jacobian for $\mathbf{F}=\dfrac{\partial\mathbf{f}[\cdot]}{\partial\underset{\sim}{\mathbf{x}}^T}$ is as follows:

$$\mathbf{F}=\frac{\partial\mathbf{f}[\cdot]}{\partial\underset{\sim}{\mathbf{x}}^T}=\begin{bmatrix}
\dfrac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\theta}^T} & \dfrac{\partial\dot{\boldsymbol{\theta}}}{\partial\boldsymbol{\omega}_b^T} & \dfrac{\partial\dot{\boldsymbol{\theta}}}{\partial\bar{\mathbf{r}}^T} & \dfrac{\partial\dot{\boldsymbol{\theta}}}{\partial\bar{\mathbf{v}}^T} & \dfrac{\partial\dot{\boldsymbol{\theta}}}{\partial\mathbf{r}_{cg,g}^T} \\
\dfrac{\partial\dot{\boldsymbol{\omega}}_b}{\partial\boldsymbol{\theta}^T} & \dfrac{\partial\dot{\boldsymbol{\omega}}_b}{\partial\boldsymbol{\omega}_b^T} & \dfrac{\partial\dot{\boldsymbol{\omega}}_b}{\partial\bar{\mathbf{r}}^T} & \dfrac{\partial\dot{\boldsymbol{\omega}}_b}{\partial\bar{\mathbf{v}}^T} & \dfrac{\partial\dot{\boldsymbol{\omega}}_b}{\partial\mathbf{r}_{cg,g}^T} \\
\dfrac{\partial\dot{\bar{\mathbf{r}}}}{\partial\boldsymbol{\theta}^T} & \dfrac{\partial\dot{\bar{\mathbf{r}}}}{\partial\boldsymbol{\omega}_b^T} & \dfrac{\partial\dot{\bar{\mathbf{r}}}}{\partial\bar{\mathbf{r}}^T} & \dfrac{\partial\dot{\bar{\mathbf{r}}}}{\partial\bar{\mathbf{v}}^T} & \dfrac{\partial\dot{\bar{\mathbf{r}}}}{\partial\mathbf{r}_{cg,g}^T} \\
\dfrac{\partial\dot{\bar{\mathbf{v}}}}{\partial\boldsymbol{\theta}^T} & \dfrac{\partial\dot{\bar{\mathbf{v}}}}{\partial\boldsymbol{\omega}_b^T} & \dfrac{\partial\dot{\bar{\mathbf{v}}}}{\partial\bar{\mathbf{r}}^T} & \dfrac{\partial\dot{\bar{\mathbf{v}}}}{\partial\bar{\mathbf{v}}^T} & \dfrac{\partial\dot{\bar{\mathbf{v}}}}{\partial\mathbf{r}_{cg,g}^T} \\
\dfrac{\partial\dot{\mathbf{r}}_{cg,g}}{\partial\boldsymbol{\theta}^T} & \dfrac{\partial\dot{\mathbf{r}}_{cg,g}}{\partial\boldsymbol{\omega}_b^T} & \dfrac{\partial\dot{\mathbf{r}}_{cg,g}}{\partial\bar{\mathbf{r}}^T} & \dfrac{\partial\dot{\mathbf{r}}_{cg,g}}{\partial\bar{\mathbf{v}}^T} & \dfrac{\partial\dot{\mathbf{r}}_{cg,g}}{\partial\mathbf{r}_{cg,g}^T}
\end{bmatrix}$$

$$=\begin{bmatrix}
-\boldsymbol{\omega}_b^{\times} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{I}^{-1}\left[(\mathbf{I}\boldsymbol{\omega}_b)^{\times}-(\boldsymbol{\omega}_b^{\times}\mathbf{I})\right] & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & -\Omega_*^2(\hat{\mathbf{y}}\hat{\mathbf{y}}^T-3\hat{\mathbf{z}}\hat{\mathbf{z}}^T) & 2\Omega_*\hat{\mathbf{y}}^{\times} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}
\end{bmatrix} \tag{H.24}$$

JACOBIAN **L**

$$\mathbf{L}=\frac{\partial \mathbf{f}[\cdot]}{\partial \mathbf{w}^T}=\begin{bmatrix} \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{g}^T} & \frac{\partial \dot{\boldsymbol{\theta}}}{\partial \mathbf{a}^T} \\ \frac{\partial \dot{\boldsymbol{\omega}}_b}{\partial \mathbf{g}^T} & \frac{\partial \dot{\boldsymbol{\omega}}_b}{\partial \mathbf{a}^T} \\ \frac{\partial \dot{\bar{\mathbf{r}}}}{\partial \mathbf{g}^T} & \frac{\partial \dot{\bar{\mathbf{r}}}}{\partial \mathbf{a}^T} \\ \frac{\partial \dot{\bar{\mathbf{v}}}}{\partial \mathbf{g}^T} & \frac{\partial \dot{\bar{\mathbf{v}}}}{\partial \mathbf{a}^T} \\ \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{g}^T} & \frac{\partial \dot{\mathbf{r}}_{cg,g}}{\partial \mathbf{a}^T} \end{bmatrix}=\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{I}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{H.25}$$

STATE TRANSITION MATRIX **Φ**

$$\boldsymbol{\Phi}_{k-1} \approx \mathbf{1}+\mathbf{F}T_s \tag{H.26}$$

PROCESS NOISE COVARIANCE MATRIX $\mathbf{Q}_k$

$$\mathbf{Q}_k = \int_0^{T_s} \boldsymbol{\Phi}_{k-1}(\tau)\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\boldsymbol{\Phi}_{k-1}^T(\tau)d\tau \tag{H.27}$$

H.4 The Measurement and Output Equation

The measurement as viewed by the filter is related as follows,

$$\mathbf{z}_k=\begin{bmatrix} \tilde{\boldsymbol{\varepsilon}}_k(\delta\hat{\mathbf{q}}_k) \\ \tilde{\mathbf{r}}_{gv,o} \end{bmatrix} \tag{H.28}$$

The output vector used in the state update equation is as following,

$$\mathbf{y}=\begin{bmatrix} \mathbf{y}_{\boldsymbol{\theta}} \\ \mathbf{y}_{\mathbf{r}} \end{bmatrix}=\begin{bmatrix} \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x},t) \\ \mathbf{h}_{\mathbf{r}}(\mathbf{x},t) \end{bmatrix}=\begin{bmatrix} \boldsymbol{\varepsilon} \\ \mathbf{r}_{gv,o} \end{bmatrix}=\begin{bmatrix} \frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \\ \mathbf{C}_{\theta o}^T\mathbf{r}_{cv,\theta}-\mathbf{C}_{o\oplus}\mathbf{C}_b^T\mathbf{r}_{cg,g} \end{bmatrix} \tag{H.29}$$

where,

$$K=\sqrt{1+\frac{1}{64}\theta^4} \tag{H.30}$$

The linearized version used in the gain and covariance calculations is derived as follows,

$$\mathbf{H}_k=\left.\frac{\partial \mathbf{h}(\mathbf{x}_k,t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k}=\begin{bmatrix}\mathbf{H}_{\boldsymbol{\theta},k}\\ \mathbf{H}_{\mathbf{r},k}\end{bmatrix}=\begin{bmatrix}\dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \mathbf{x}^T}\\ \dfrac{\partial \mathbf{h}_{\mathbf{r}}(\mathbf{x}_k,t_k)}{\partial \mathbf{x}^T}\end{bmatrix} \tag{H.31}$$

The partition related to the orientation is...

$$\mathbf{H}_{\boldsymbol{\theta},k}=\begin{bmatrix}\dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \bar{\mathbf{r}}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \dot{\bar{\mathbf{r}}}^T} & \dfrac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T}\end{bmatrix} \tag{H.32}$$

From,

$$\boldsymbol{\varepsilon}=\frac{1}{K}\left(\frac{1}{2}\boldsymbol{\theta}\right) \quad ' \tag{H.33}$$

it is found that....

$$\frac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \bar{\mathbf{r}}^T}=\frac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \dot{\bar{\mathbf{r}}}^T}=\frac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T}=\frac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T}=\mathbf{0} \tag{H.34}$$

$$\frac{\partial \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T}=\frac{1}{2}\mathbf{1} \tag{H.35}$$

For the partition related to the translation ...

$$\mathbf{H}_{\mathrm{r},k} = \left[\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\theta}^T} \quad \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\omega}_b{}^T} \quad \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \overline{\mathbf{r}}^T} \quad \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \dot{\overline{\mathbf{r}}}^T} \quad \frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{cg,g}{}^T} \right] \tag{H.36}$$

From,

$$\mathbf{r}_{gv,o} = \mathbf{C}_{\theta o}^T \mathbf{r}_{cv,\theta} - \mathbf{C}_{o\oplus} \mathbf{C}_b^T \mathbf{r}_{cg,g} \tag{H.37}$$

it is found that....

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \overline{\mathbf{r}}^T} = \mathbf{C}_{\theta o}^T \begin{bmatrix} \frac{(R_v + r)}{R_*} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \frac{(R_v + r)}{R_*} \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{H.38}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \dot{\overline{\mathbf{r}}}^T} = \mathbf{0} \tag{H.39}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \mathbf{r}_{cg,g}{}^T} = -\mathbf{C}_{o\oplus} \mathbf{C}_b^T \tag{H.40}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\theta}^T} = \mathbf{C}_{o\oplus} \mathbf{C}_{\mathrm{ref}}^T \mathbf{r}_{cg,g}^{\times} \tag{H.41}$$

$$\frac{\partial \mathbf{h}_{\mathrm{r}}(\mathbf{x}_k, t_k)}{\partial \boldsymbol{\omega}_b{}^T} = \mathbf{0} \tag{H.42}$$

Finally, all assembled the linearized measurement sensitivity matrix is given by,

$$
\begin{aligned}
\mathbf{H}_k &= \left.\frac{\partial \mathbf{h}(\mathbf{x}_k, t_k)}{\partial \mathbf{x}^T}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} \\
&= \begin{bmatrix} \mathbf{H}_{\Theta,k} \\ \mathbf{H}_{\mathbf{r},k} \end{bmatrix} \\
&= \begin{bmatrix}
\frac{\partial \mathbf{h}_\Theta(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T} & \frac{\partial \mathbf{h}_\Theta(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \frac{\partial \mathbf{h}_\Theta(\mathbf{x}_k,t_k)}{\partial \overline{\mathbf{r}}^T} & \frac{\partial \mathbf{h}_\Theta(\mathbf{x}_k,t_k)}{\partial \dot{\overline{\mathbf{r}}}^T} & \frac{\partial \mathbf{h}_\Theta(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T} \\
\frac{\partial \mathbf{h}_\mathbf{r}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\theta}^T} & \frac{\partial \mathbf{h}_\mathbf{r}(\mathbf{x}_k,t_k)}{\partial \boldsymbol{\omega}_b{}^T} & \frac{\partial \mathbf{h}_\mathbf{r}(\mathbf{x}_k,t_k)}{\partial \overline{\mathbf{r}}^T} & \frac{\partial \mathbf{h}_\mathbf{r}(\mathbf{x}_k,t_k)}{\partial \dot{\overline{\mathbf{r}}}^T} & \frac{\partial \mathbf{h}_\mathbf{r}(\mathbf{x}_k,t_k)}{\partial \mathbf{r}_{cg,g}{}^T}
\end{bmatrix} \\
&= \begin{bmatrix}
\frac{1}{2}\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{C}_{o\oplus}\mathbf{C}_{\mathrm{ref}}^T\mathbf{r}_{cg,g}^{\times} & \mathbf{0} & \mathbf{C}_{\theta o}^T \begin{bmatrix} \frac{(R_v+r)}{R_*}\cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \frac{(R_v+r)}{R_*}\sin\theta & 0 & \cos\theta \end{bmatrix} & \mathbf{0} & -\mathbf{C}_{o\oplus}\mathbf{C}_b^T
\end{bmatrix}
\end{aligned}
\tag{H.43}
$$

H.5 Calculating the Filter Gain

Standard form is

$$\mathbf{K}_k = \mathbf{P}_k^{-}\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^{-}\mathbf{H}_k^T + \mathbf{R}_k\right]^{-1} \tag{H.44}$$

where $\mathbf{R}_k$ is composed of the following,

$$\mathbf{R}_k = \begin{bmatrix} \mathbf{R}_{\Theta,k} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathbf{r},k} \end{bmatrix}_{6x6} \tag{H.45}$$

with the assumption that $\mathbf{v}_{\Theta,k}$ and $\mathbf{v}_{\mathbf{r},k}$ are not correlated.

Note that,

$$\mathbf{R}_{\Theta,k} = \frac{1}{4}\mathbf{C}_{gm}\mathbf{R}_{\Theta,m}\mathbf{C}_{gm}^T \tag{H.46}$$

where

$$\mathbf{C}_{gm} = \mathbf{C}_{g\oplus}\mathbf{C}^T_{m\oplus} \tag{H.47}$$

and

$$\mathbf{R}_{\mathbf{r},k} = \mathbf{C}_{om}\mathbf{R}_{\mathbf{r},m}\mathbf{C}^T_{om} \tag{H.48}$$

where

$$\mathbf{C}_{om} = \mathbf{C}_{o\oplus}\mathbf{C}^T_{v\oplus}\mathbf{C}^T_{mv} \tag{H.49}$$

H.6 The Update of the State Estimate

For the update of the state estimate,

$$\begin{gathered}\underset{\sim}{\hat{\mathbf{x}}}{}_k^+ = \underset{\sim}{\hat{\mathbf{x}}}{}_k^- + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-, t_k)] \\ \underset{\sim}{\hat{\mathbf{x}}}{}_k^+ = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\omega}}_b^- \\ \hat{\mathbf{r}}^- \\ \hat{\mathbf{v}}^- \\ \hat{\mathbf{r}}^-_{cg,g} \end{bmatrix}_k + \mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-, t_k)]\end{gathered} \tag{H.50}$$

H.7 Update of the State Error Covariance

Shown here is the standard Joseph form often used for computation.

$$\mathbf{P}_k^{\ +} = [\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]\mathbf{P}_k^-[\mathbf{1} - \mathbf{K}_k\mathbf{H}_k]^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\ T} \tag{H.51}$$

H.8 Reset of the Absolute Orientation State

The reference orientation is moved along with the current (relative) state estimate.

Build a proper rotation matrix from the small rotation vector. The quaternion is used for this because it is easy to normalize.

$$\hat{\theta}_k^2 = (\hat{\boldsymbol{\theta}}_k^+)^T \hat{\boldsymbol{\theta}}_k^+ \tag{H.52}$$

$$\delta\hat{\eta}_k = \frac{1}{K}\left(1 - \frac{1}{8}\hat{\theta}_k^2\right) \tag{H.53}$$

$$\delta\hat{\boldsymbol{\varepsilon}}_k = \frac{1}{K}\left(\frac{1}{2}\hat{\boldsymbol{\theta}}_k^+\right) \tag{H.54}$$

where,

$$K = \sqrt{1 + \frac{1}{64}\delta\hat{\theta}_k^4} \tag{H.55}$$

Assembled this gives,

$$\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) = \begin{bmatrix} \delta\hat{\eta}_k \\ \delta\hat{\boldsymbol{\varepsilon}}_k \end{bmatrix} \tag{H.56}$$

then

$$\begin{aligned} \hat{\mathbf{q}}_{\text{ref},k}^+ &= \hat{\mathbf{q}}_{\text{ref},k}^- \oplus \delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+) \\ \hat{\mathbf{q}}_{\text{ref},k}^+ &= \Xi(\delta\mathbf{q}_k(\hat{\boldsymbol{\theta}}_k^+))\,\hat{\mathbf{q}}_{\text{ref},k}^- \end{aligned} \tag{H.57}$$

Appendix I Translational Initialization Baseline Setup (Orbital)

KF TESTING INITIALIZATION REFERENCE TABLE
TEST REFERENCE NO. *Baseline Setup*
Filter Type: KF
Filter Sampling Interval: KALMAN_DT_SAMPLE = 1.0 [sec]
Propagation Type: State Estimate (RK4 Solver) / Covariance ($\mathbf{P}^-_{k+1} = \mathbf{\Phi}_k \mathbf{P}^+_k \mathbf{\Phi}_k + \mathbf{Q}_k$)
Number of Steps: State Estimate (20) / Covariance (20)
Update Type: KF (with Joseph Form)
Total Simulation Time: 10000 [sec]

TRUTH Variables and Parameters		FILTER Variables and Parameters	
Symbol	***Numerical value***	***Symbol***	***Numerical value***
$\mathbf{r}_{cv,o}(t_0)$	*[94.25; 1.41; -40.0]* m	$\hat{\mathbf{r}}(t_0)$	*[99.25;-5;-35]* m
$\dot{\mathbf{r}}_{cv,o}(t_0)$	*[-0.077; 1.73; 1.06]* m/s	$\hat{\mathbf{v}}(t_0)$	*[0.022; -0.10; 0.099]* m/s
$\mathbf{r}_{cg,g}(t_0)$	*[0.05;0.05;0.0]* m	$\hat{\mathbf{r}}_{cg,g}(t_0)$	*[0.048;0.052;0.001]* m

	r_cg_g = 0.05 0.05 0		r_cg_g_EST = 0.048 0.052 0.001
		$\mathbf{P}_\mathbf{r}(t_0)$	$\hat{\sigma}_{r_{cv(x)}} = 1.0$ m $\hat{\sigma}_{r_{cv(y)}} = 1.0$ m $\hat{\sigma}_{r_{cv(z)}} = 1.0$ m $\hat{\sigma}_{v_{cv(x)}} = 0.5$ m/s $\hat{\sigma}_{v_{cv(y)}} = 0.5$ m/s $\hat{\sigma}_{v_{cv(z)}} = 0.5$ m/s $\hat{\sigma}_{rc_x} = 0.1$ m $\hat{\sigma}_{rc_y} = 0.1$ m $\hat{\sigma}_{rc_z} = 0.1$ m KFilter_P0 = 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0.25 0 0 0 0 0 0 0 0 0 0.25 0 0 0 0 0 0 0 0 0 0.25 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0 0 0.01
$\mathbf{v}_{\mathbf{r},k}$	$\sigma_{\mathbf{v}_x} = 0.05$ m $\sigma_{\mathbf{v}_y} = 0.05$ m $\sigma_{\mathbf{v}_z} = 0.05$ m	$\mathbf{R}_{\mathbf{r},m}$	$\hat{\sigma}_{\mathbf{v}_x} = 0.1$ m $\hat{\sigma}_{\mathbf{v}_y} = 0.1$ m $\hat{\sigma}_{\mathbf{v}_z} = 0.1$ m KFilter_Rm = 0.01 0 0 0 0.01 0 0 0 0.01

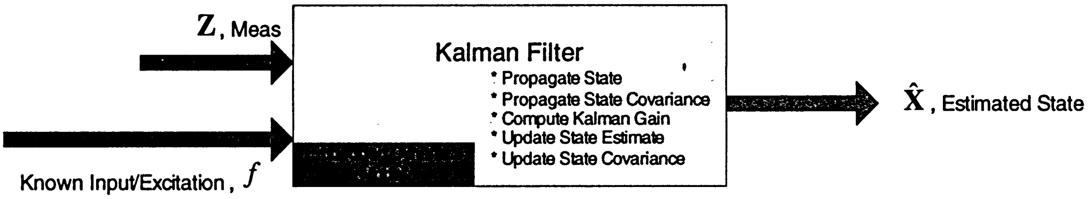
$\mathbf{w}_{\mathbf{r}}$ (associated with $\delta\mathbf{a}$)	$\delta\mathbf{a} = \mathbf{0}$ m/s^2	$\mathbf{Q}_{\mathbf{r},c}$	$\hat{\sigma}^2_{\mathrm{w}(f_x)} = (5e\text{-}4)^2\,(\mathrm{m/s}^2)^2.\ \mathrm{s}$ $\hat{\sigma}^2_{\mathrm{w}(f_y)} = (5e\text{-}4)^2\,(\mathrm{m/s}^2)^2.\ \mathrm{s}$ $\hat{\sigma}^2_{\mathrm{w}(f_z)} = (5e\text{-}4)^2\,(\mathrm{m/s}^2)^2.\ \mathrm{s}$ KFilter_Qc = 2.5e-007 0 0 0 2.5e-007 0 0 0 2.5e-007

Appendix J Software Design

The Kalman filter algorithms were developed in basic ANSI C code and simply called within the C-MEX S-function framework. The software design as described below assumes dependence on the Simulink C-MEX S-function framework for tasks such as initialization, output, and update (memory between calls) using the following interface functions: (in their order of execution)

mdlInitializeConditions()
mdlOutputs()
mdlUpdate()

A more generic software design not utilizing these built in functions, but rather using global variables, etc. could be implemented if needed for real-time applications.



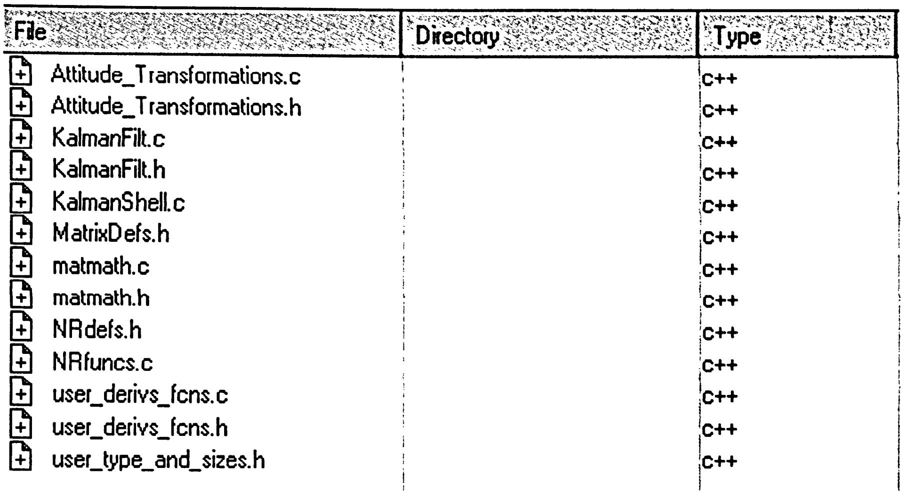
Z, Meas
Known Input/Excitation, f
Kalman Filter
• Propagate State
• Propagate State Covariance
• Compute Kalman Gain
• Update State Estimate
• Update State Covariance
$\hat{X}$, Estimated State

Figure J-1: Filter Inputs, Outputs, and Internal Processes

The basic software architecture is described in more detail in the following sections.

J.1 File Structure

The entire KalmanShell package is made up of multiple source and header files. The full list of project files can be seen in Figure J-2.



File	Directory	Type
Attitude_Transformations.c		c++
Attitude_Transformations.h		c++
KalmanFilt.c		c++
KalmanFilt.h		c++
KalmanShell.c		c++
MatrixDefs.h		c++
matmath.c		c++
matmath.h		c++
NRdefs.h		c++
NRfuncs.c		c++
user_derivs_fcns.c		c++
user_derivs_fcns.h		c++
user_type_and_sizes.h		c++

Figure J-2: Project Files

J.2 .c File Structure

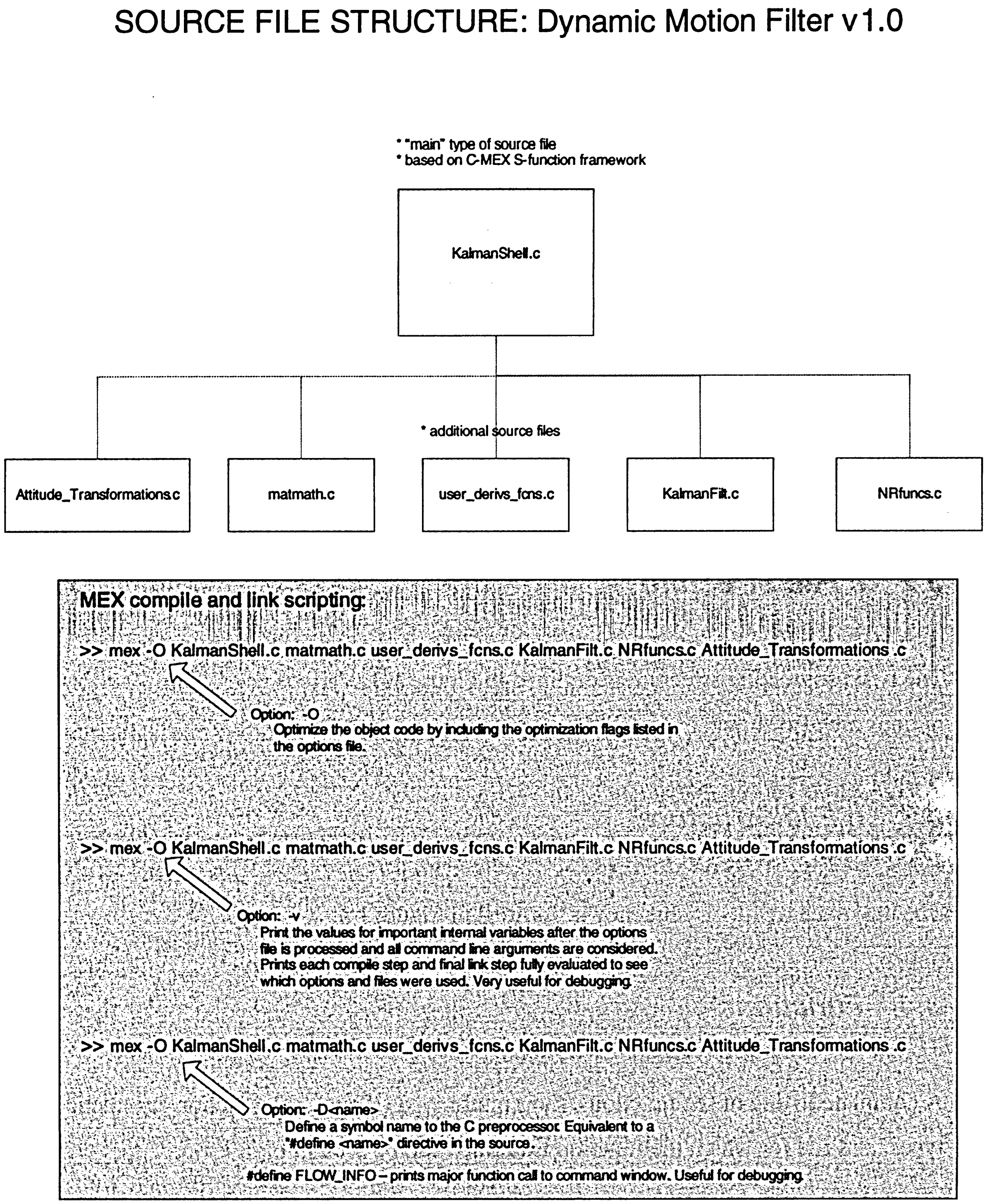
The basic source file structure is outlined in Figure J-3. Various compile/link script options are also shown in Figure J-3.

J.3 .h File Structure

The basic header file dependency is outlined in Figure J-4.

J.4 Functions

A complete listing of the functions, their return type (if any), the function argument types, and file for which the function is defined can be seen in Figure J-5.


SOURCE FILE STRUCTURE: Dynamic Motion Filter v1.0
* "main" type of source file
* based on C-MEX S-function framework
KalmanShell.c
* additional source files
Attitude_Transformations.c
matmath.c
user_derivs_fcns.c
KalmanFilt.c
NRfuncs.c

MEX compile and link scripting:

```
>> mex -O KalmanShell.c matmath.c user_derivs_fcns.c KalmanFilt.c NRfuncs.c Attitude_Transformations .c
```

Option: -O
Optimize the object code by including the optimization flags listed in the options file.

```
>> mex -O KalmanShell.c matmath.c user_derivs_fcns.c KalmanFilt.c NRfuncs.c Attitude_Transformations .c
```

Option: -v
Print the values for important internal variables after the options file is processed and all command line arguments are considered. Prints each compile step and final link step fully evaluated to see which options and files were used. Very useful for debugging.

```
>> mex -O KalmanShell.c matmath.c user_derivs_fcns.c KalmanFilt.c NRfuncs.c Attitude_Transformations .c
```

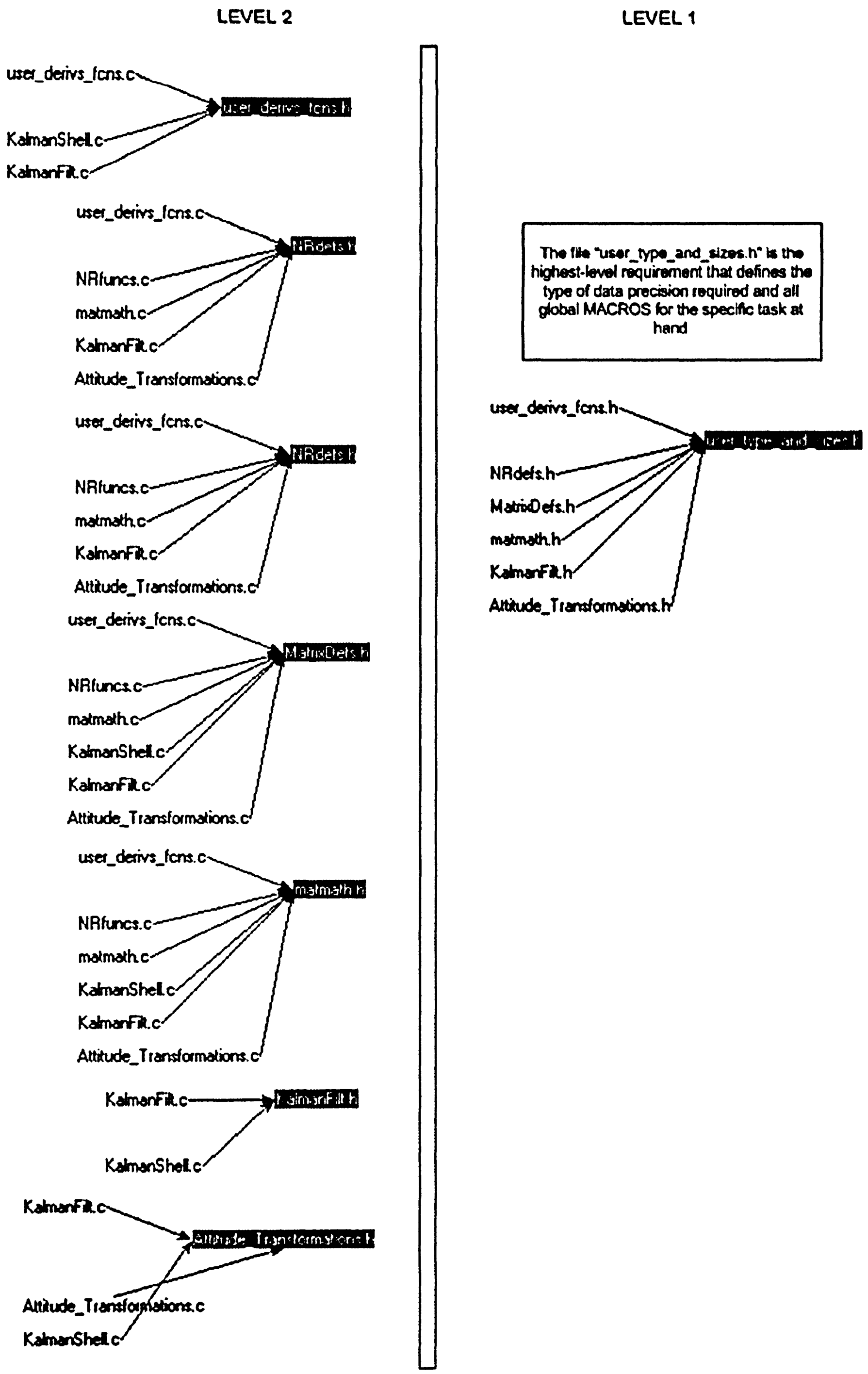
Option: -D<name>
Define a symbol name to the C preprocessor. Equivalent to a "#define <name>" directive in the source.

#define FLOW_INFO – prints major function call to command window. Useful for debugging.

Figure J-3: .c File Structure and Compile/Link Scripts

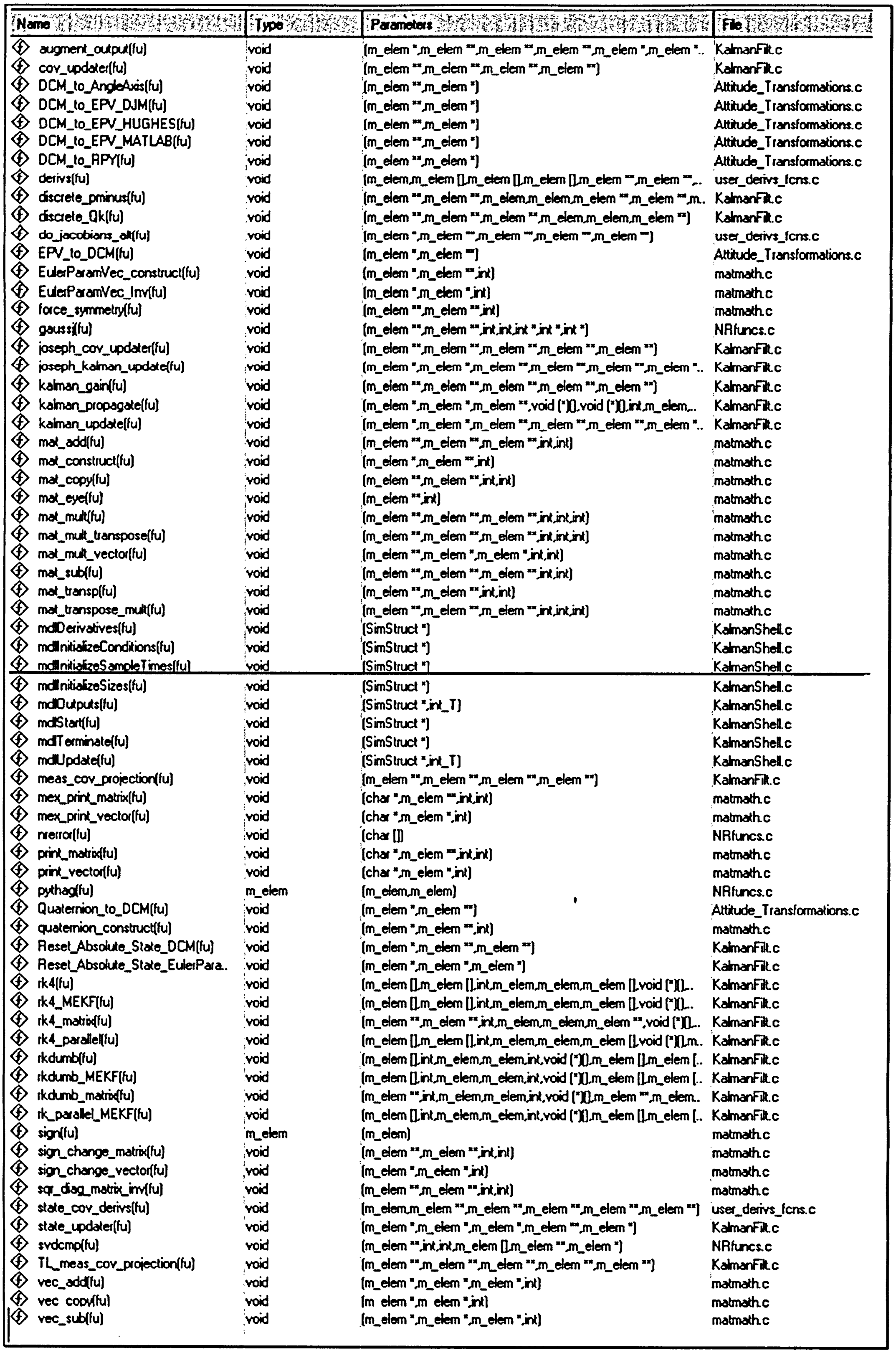
HEADER FILE STRUCTURE: Dynamic Motion Filter v1.0

Include Hierarchy – Header Files



The file "user_type_and_sizes.h" is the highest-level requirement that defines the type of data precision required and all global MACROS for the specific task at hand

Figure J-4: .h File Structure



Name	Type	Parameters	File
augment_output(fu)	void	(m_elem *,m_elem **,m_elem **,m_elem **,m_elem *,m_elem *..	KalmanFilt.c
cov_updater(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem **)	KalmanFilt.c
DCM_to_AngleAxis(fu)	void	(m_elem **,m_elem *)	Attitude_Transformations.c
DCM_to_EPV_DJM(fu)	void	(m_elem **,m_elem *)	Attitude_Transformations.c
DCM_to_EPV_HUGHES(fu)	void	(m_elem **,m_elem *)	Attitude_Transformations.c
DCM_to_EPV_MATLAB(fu)	void	(m_elem **,m_elem *)	Attitude_Transformations.c
DCM_to_RPY(fu)	void	(m_elem **,m_elem *)	Attitude_Transformations.c
derivs(fu)	void	(m_elem,m_elem [],m_elem [],m_elem [],m_elem **,m_elem **,..	user_derivs_fcns.c
discrete_pminus(fu)	void	(m_elem **,m_elem **,m_elem,m_elem,m_elem **,m_elem **,m..	KalmanFilt.c
discrete_Qk(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem,m_elem,m_elem **)	KalmanFilt.c
do_jacobians_alt(fu)	void	(m_elem *,m_elem **,m_elem **,m_elem **,m_elem **)	user_derivs_fcns.c
EPV_to_DCM(fu)	void	(m_elem *,m_elem **)	Attitude_Transformations.c
EulerParamVec_construct(fu)	void	(m_elem *,m_elem **,int)	matmath.c
EulerParamVec_Inv(fu)	void	(m_elem *,m_elem *,int)	matmath.c
force_symmetry(fu)	void	(m_elem **,m_elem **,int)	matmath.c
gaussj(fu)	void	(m_elem **,m_elem **,int,int,int *,int *,int *)	NRfuncs.c
joseph_cov_updater(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem **,m_elem **)	KalmanFilt.c
joseph_kalman_update(fu)	void	(m_elem *,m_elem *,m_elem **,m_elem **,m_elem **,m_elem *..	KalmanFilt.c
kalman_gain(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem **,m_elem **)	KalmanFilt.c
kalman_propagate(fu)	void	(m_elem *,m_elem *,m_elem **,void (*)(),void (*)(),int,m_elem,..	KalmanFilt.c
kalman_update(fu)	void	(m_elem *,m_elem *,m_elem **,m_elem **,m_elem **,m_elem *..	KalmanFilt.c
mat_add(fu)	void	(m_elem **,m_elem **,m_elem **,int,int)	matmath.c
mat_construct(fu)	void	(m_elem *,m_elem **,int)	matmath.c
mat_copy(fu)	void	(m_elem **,m_elem **,int,int)	matmath.c
mat_eye(fu)	void	(m_elem **,int)	matmath.c
mat_mult(fu)	void	(m_elem **,m_elem **,m_elem **,int,int,int)	matmath.c
mat_mult_transpose(fu)	void	(m_elem **,m_elem **,m_elem **,int,int,int)	matmath.c
mat_mult_vector(fu)	void	(m_elem **,m_elem *,m_elem *,int,int)	matmath.c
mat_sub(fu)	void	(m_elem **,m_elem **,m_elem **,int,int)	matmath.c
mat_transp(fu)	void	(m_elem **,m_elem **,int,int)	matmath.c
mat_transpose_mult(fu)	void	(m_elem **,m_elem **,m_elem **,int,int,int)	matmath.c
mdlDerivatives(fu)	void	(SimStruct *)	KalmanShell.c
mdlInitializeConditions(fu)	void	(SimStruct *)	KalmanShell.c
mdlInitializeSampleTimes(fu)	void	(SimStruct *)	KalmanShell.c
mdlInitializeSizes(fu)	void	(SimStruct *)	KalmanShell.c
mdlOutputs(fu)	void	(SimStruct *,int_T)	KalmanShell.c
mdlStart(fu)	void	(SimStruct *)	KalmanShell.c
mdlTerminate(fu)	void	(SimStruct *)	KalmanShell.c
mdlUpdate(fu)	void	(SimStruct *,int_T)	KalmanShell.c
meas_cov_projection(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem **)	KalmanFilt.c
mex_print_matrix(fu)	void	(char *,m_elem **,int,int)	matmath.c
mex_print_vector(fu)	void	(char *,m_elem *,int)	matmath.c
nrerror(fu)	void	(char [])	NRfuncs.c
print_matrix(fu)	void	(char *,m_elem **,int,int)	matmath.c
print_vector(fu)	void	(char *,m_elem *,int)	matmath.c
pythag(fu)	m_elem	(m_elem,m_elem)	NRfuncs.c
Quaternion_to_DCM(fu)	void	(m_elem *,m_elem **)	Attitude_Transformations.c
quaternion_construct(fu)	void	(m_elem *,m_elem **,int)	matmath.c
Reset_Absolute_State_DCM(fu)	void	(m_elem *,m_elem **,m_elem **)	KalmanFilt.c
Reset_Absolute_State_EulerPara..	void	(m_elem *,m_elem *,m_elem *)	KalmanFilt.c
rk4(fu)	void	(m_elem [],m_elem [],int,m_elem,m_elem,m_elem [],void (*)()..	KalmanFilt.c
rk4_MEKF(fu)	void	(m_elem [],m_elem [],int,m_elem,m_elem,m_elem [],void (*)()..	KalmanFilt.c
rk4_matrix(fu)	void	(m_elem **,m_elem **,int,m_elem,m_elem,m_elem **,void (*)()..	KalmanFilt.c
rk4_parallel(fu)	void	(m_elem [],m_elem [],int,m_elem,m_elem,m_elem [],void (*)(),m..	KalmanFilt.c
rkdumb(fu)	void	(m_elem [],int,m_elem,m_elem,int,void (*)(),m_elem [],m_elem [..	KalmanFilt.c
rkdumb_MEKF(fu)	void	(m_elem [],int,m_elem,m_elem,int,void (*)(),m_elem [],m_elem [..	KalmanFilt.c
rkdumb_matrix(fu)	void	(m_elem **,int,m_elem,m_elem,int,void (*)(),m_elem **,m_elem..	KalmanFilt.c
rk_parallel_MEKF(fu)	void	(m_elem [],int,m_elem,m_elem,int,void (*)(),m_elem [],m_elem [..	KalmanFilt.c
sign(fu)	m_elem	(m_elem)	matmath.c
sign_change_matrix(fu)	void	(m_elem **,m_elem **,int,int)	matmath.c
sign_change_vector(fu)	void	(m_elem *,m_elem *,int)	matmath.c
sqr_diag_matrix_inv(fu)	void	(m_elem **,m_elem **,int,int)	matmath.c
state_cov_derivs(fu)	void	(m_elem,m_elem **,m_elem **,m_elem **,m_elem **,m_elem **)	user_derivs_fcns.c
state_updater(fu)	void	(m_elem *,m_elem *,m_elem *,m_elem **,m_elem *)	KalmanFilt.c
svdcmp(fu)	void	(m_elem **,int,int,m_elem [],m_elem **,m_elem *)	NRfuncs.c
TL_meas_cov_projection(fu)	void	(m_elem **,m_elem **,m_elem **,m_elem **,m_elem **)	KalmanFilt.c
vec_add(fu)	void	(m_elem *,m_elem *,m_elem *,int)	matmath.c
vec_copy(fu)	void	(m_elem *,m_elem *,int)	matmath.c
vec_sub(fu)	void	(m_elem *,m_elem *,m_elem *,int)	matmath.c

Figure J-5: Complete function listing

References

1. MDA. *Orbital Robotics - Unmanned Robotic Solutions.* 2006 [cited 2006 12-Apr-2006]; Available from: http://www.mdrobotics.ca/what_we_do/aut_rend.htm.
2. Greenspan, M., Shang, L., Jasiobedzki, P. *Efficient Tracking with the Bounded Hough Transform.* in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04).* 2004.
3. Jasiobedzki, P., Greenspan, M., Roth, G. *Pose Determination and Tracking for Autonomous Satellite Capture.* in *Proceedings of the 6th International Symposium on Artificial Intelligence and Robotics & Automation in Space.* 2001. Canadian Space Agency, St. Hubert, Quebec, Canada.
4. Piedboeuf, J.-C., Dupuis, E. *Recent Canadian Activities in Space Automation & Robotics - An Overview.* in *Proceedings of the 7th International Symposium on Artificial Intelligence, Robotics, and Automation in Space: i-SAIRAS 2003.* 2003. NARA, Japan.
5. Shang, L., Jasiobedzki, P., Greenspan, M. *Discrete Pose Space Estimation to Improve ICP-Based Tracking.* in *Proceedings of the Space Vision and Advanced Robotics Workshop.* 2005. MDA Space Missions, Brampton, ON, Canada.
6. Lichter, M.D., Dubowsky, S. *State, Shape, and Parameter Estimation of Space Objects from Range Images.* in *Proceedings of the 2004 IEEE International Conference on Robotics & Automation.* 2004. New Orleans, LA.
7. Maybeck, P.S., *Stochastic Models, Estimation, and Control.* Vol. 1. 1979: Academic Press, Inc.
8. Lefferts, E.J., Markley, F.L., Shuster, M.D., *Kalman Filtering for Spacecraft Attitude Estimation.* Journal of Guidance, Control, and Dynamics, 1982. 5(5): p. 417-429.
9. Julier, S.J., Uhlmann, J.K. *A New Approach for Filtering Nonlinear Systems.* in *Proceedings of the American Control Conference, 1995.* 1995. Seattle, Washington.
10. Julier, S.J., Uhlmann, J.K. *A New Extension of the Kalman Filter to Nonlinear Systems.* in *The Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls, 1997.* 1997. Orlando, FL, USA.
11. Wan, E., van der Merwe, R. *The Unscented Kalman Filter for Nonlinear Estimation.* in *Proceedings of IEEE Symposium 2000 (AS-SPCC).* 2000. Lake Louise, Alberta, Canada.
12. Crassidis, J.L., and Markley, F.L. *Unscented Filtering for Spacecraft Attitude Estimation.* in *Proceedings of the AIAA Guidance, Navigation, and Control Conference.* 2003. Austin, TX.
13. VanDyke, M.C., Schwartz, J.L., Hall, C.D. *Unscented Kalman Filtering for Spacecraft Attitude State And Parameter Estimation.* in *Proceedings of the 2004 AAS/AIAA Space Flight Mechanics Meeting.* 2004. Maui, Hawaii.

14. Farrel, J.L., *Attitude Determination by Kalman Filtering.* Automatica, 1970. **6**(3): p. 419-430.

15. Markley, F.L., *Attitude Error Representations for Kalman Filtering.* Journal of Guidance, Control, and Dynamics, 2003. **26**(2): p. 311-317.

16. Stuelpnagel, J., *On the Parameterization of the Three-Dimensional Rotation Group.* SIAM Review, 1964. **6**(4): p. 422-430.

17. Hughes, P.C., *Spacecraft Attitude Dynamics.* 2004: Dover Publications, Inc.

18. *Spacecraft Attitude Determination and Control*, ed. J.R. Wertz. Vol. 73. 1978: D. Reidel Publishing Company.

19. Markley, F.L., Crassidis, J.L., Cheng, Y. *Nonlinear Attitude Filtering Methods.* in *AIAA Guidance, Navigation, and Control Conference and Exhibit.* 2005. San Francisco, California.

20. McTavish, D.J., *Dynamic Estimation of Body Motion and Inertia Parameters.* (Internal Report), 2006: Ryerson University.

21. McTavish, D.J., *Rotation Matrices and Angular Velocity for Body Dynamic Analysis.* (Internal Report), 2006: Ryerson University.

22. Shuster, M.D., *A Survey of Attitude Representations.* The Journal of the Astronautical Sciences, 1993. **41**(4): p. 439-517.

23. McTavish, D.J., *Three-Parameter Approaches to Small Rotation Analysis.* (Internal Report), 2006: Ryerson University.

24. Matlab. *Matlab Product Documentation.* 2006 [cited 2006 May 29 2006]; Available from: www.mathworks.com.

25. Shahid, K., *Intelligent Selection of Scanning Area for ICP Based Pose Estimation of Space Structures.* 2006, MASc Thesis, Ryerson University: Toronto. p. 1-112.

26. Clements, R., Tavares, P., Lima, P. *Small Satellite Attitude Control Based on a Kalman Filter.* in *Proceedings of the IEEE International Symposium on Intelligent Control 2000.* 2000. Patras, Greece.

27. Zarchan, P., Musoff, H., *Fundamentals of Kalman Filtering - A Practical Approach.* Progress in Astronautics and Aeronautics, ed. P. Zarchan. Vol. 190. 2000, Cambridge, Massachuetts.

28. Humphreys, T.E., *Attitude Determination for Small Satellites with Modest Pointing Constraints.* 2002, Utah State University: Utah, USA. p. 1-13.

29. Stengel, R.F., *Optimal Control and Estimation.* 1994: Dover Publications, Inc.

30. Vander Velde, W. *Lecture 24: Stochastic Estimation and Control, Fall 2004.* 2004 [cited; Available from: http://ocw.mit.edu/OcwWeb/.

31. Schumacher, R.R., *Directed Studies Report: Kalman Filtering with Applications.* 2005, Ryerson University: Toronto.

32. Lin, C.-F., *Modern Navigation, Guidance, and Control Processing.* Prentice Hall Series in Advanced Navigation, Guidance, and Control, and Their Applications. 1991, Englewood Cliffs, New Jersey: Prentice-Hall, Inc.

33. Pittelkau, M.E., *Rotation Vector in Attitude Estimation* Journal of Guidance, Control, and Dynamics, 2003. **26**(6): p. 855-860.

34. Julier, S.J., Uhlmann, J.K., *Unscented Filtering and Nonlinear Estimation.* IEEE Review, 2004. **92**(3): p. 401-422.

35. Maybeck, P.S., *Stochastic Models, Estimation, and Control.* Vol. 2. 1979: Academic Press, Inc.

36. Verhaegen, M., Van Dooren, P., *Numerical Aspects of Different Kalman Filter Implementations.* IEEE Transactions on Automatic Control, 1986. **AC-31**(10): p. 907-917.

37. Nash, R.A., Tuteir, F.B., *The Effect of Uncertainties in the Noise Covariance Matrices on the Maximum Likelihood Estimator.* IEEE Transactions on Automatic Control, 1968. **AC-13**(1): p. 86-88.

38. Fitzgerald, R.J., *Divergence of the Kalman Filter.* IEEE Transactions on Automatic Control, 1967. **AC-16**(6): p. 736-747.

39. Friedland, B., *On the Effect of Incorrect Gain in the Kalman Filter.* IEEE Transactions on Automatic Control, 1967. **AC-12**(5): p. 610.

40. Griffin, R.E., Sage, A.P., *Large and Small Scale Sensitivity Analysis of Optimum Estimation Algorithms.* IEEE Transactions on Automatic Control, 1968. **AC-13**(4): p. 320-329.

41. Price, C.F., *An Analysis of the Divergence Problem in the Kalman Filter.* IEEE Transactions on Automatic Control, 1968. **AC-13**(6): p. 699-702.

42. Schlee, F.H., Standish, C.J., Toda, N.F., *Divergence in the Kalman Filter.* AIAA Journal, 1967. **5**(6): p. 1114-1120.

43. Sorensen, H.W., *On the Error Behavior in Linear Minimum Variance Estimation Problems.* IEEE Transactions on Automatic Control, 1967. **AC-13**(5): p. 557-562.

44. Bierman, G.J., Thornton, C.L., *Numerical Comparison of Kalman Filter Algorithms: Orbit Determination Case Study.* Automatica, 1977. **13**: p. 23-35.

45. Welch, G., Bishop, G., *An Introduction to the Kalman Filter.* 2004, University of North Carolina at Chapel Hill: Chapel Hill, NC. p. 1-16.

46. Abbeel, P., Coates, A., Montemerlo, M., Ng, A.Y., Thrun, S. *Discriminative Training of Kalman Filters.* in *Proceedings of Robotics: Science and Systems.* 2005. Cambridge, USA.