AIAA Guidance, Navigation, and Control Conference
2 - 5 August 2010, Toronto, Ontario Canada

AIAA 2010-8287

Guidance and Control Algorithm Robustness Baseline Indexing

Bradley A. Steinfeldt,* Robert D. Braun†
Georgia Institute of Technology, Atlanta, GA 30332
and
Stephen C. Paschall, II‡
Charles Stark Draper Laboratory, Inc., Cambridge, MA 02139

Throughout the design process, guidance and control algorithms are continually assessed to determine their applicability to a given application. Due to this need, an equitable assessment of how various algorithms meet desired objectives is a necessity. Traditionally these comparisons rely on conducting performance assessments of the algorithms that stress the algorithms and then human judgment is used to provide recommendations on which algorithm(s) to carry forward as the design matures. This paper presents a way to develop a baseline robustness index, that is a single value that captures the desired attributes of the algorithm in the presence of uncertainty, which can be used to compare various algorithms. Due to the general formulation of the baseline index value, these indicies can be comprised of algorithm complexity metrics, algorithm consistency metrics, and algorithm performance metrics. These indicies can also be used to assess the effect of algorithm maturation on a single algorithm. In this paper, the groundwork and underlying theory for the methodology are described for use in both guidance and control algorithms. Additionally, two example uses of the methodology are provided. One demonstrates the parametric nature of the baseline robustness index and the flexibility extended during the conceptual design of a vehicle. The other demonstrates the impact on the baseline robustness index as a propulsive propulsive guidance law is matured from the theoretical foundations to an almost flight mature algorithm that is being developed for the Autonomous Landing and Hazard Avoidance Technology program.

Nomenclature

Variables

$\mathbf{a}$	Acceleration vector
$\mathbf{a_c}$	Commanded acceleration vector
a_{ij}	Preference of element i relative to element j
$\mathbf{A}$	Reciprocal matrix of preferences
C_D	Coefficient of drag
C_L	Coefficient of lift
$\mathbf{f_d}$	Force vector due to drag
$\mathbf{f_g}$	Force vector due to gravity
$\mathbf{f_l}$	Force vector due to lift
$\mathbf{f_{pert}}$	Force vector due to perturbing forces
$\mathbf{h}$	Algebraic inequality function
g	Local acceleration due to gravity
$\mathbf{g}$	Acceleration vector due to gravity

*Graduate Research Assistant, Guggenheim School of Aerospace Engineering, AIAA Student Member
†David and Andrew Lewis Professor of Space Technology, Guggenheim School of Aerospace Engineering, AIAA Fellow
‡Senior Member of the Technical Staff, Mission Design Group, AIAA Member

Copyright © 2010 by Bradley A. Steinfeldt, Robert D. Braun, and Stephen C. Paschall, II. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

g_0	Gravitational constant (=9.806 m/s^2)
$\mathbf{I_p}$	Inertia tensor in the principal frame
I_{sp}	Specific impulse
J_i	i^{th} sample of the objective function
$\hat{J}$	Objective function target
m	Vehicle mass
m_{prop}	Propellant mass
n	Number of values sampled
n_{lines}	Number of lines of code
N	Number of Monte Carlo samples
$\mathbf{q}$	Algebraic inequality value
$\bar{q}_\infty$	Free-stream dynamic pressure $\left(=\frac{1}{2}\rho v_{rel}^2\right)$
$\mathbf{r}$	Response of the analysis
$\hat{\mathbf{r}}$	Nominal response of the analysis
$\bar{\mathbf{r}}$	Distance to target vector in the target relative frame
S	Reference area
T	Thrust magnitude
t_{go}	Time-to-go before touchdown
$\dot{\mathbf{u}}$	Differential constraint value
$\bar{\mathbf{v}}$	Velocity vector in the target relative frame
v_{rel}	Atmosphere relative velocity vector magnitude
$\mathbf{v_{rel}}$	Atmosphere relative velocity vector
$\mathbf{x}$	Fixed deterministic parameters
$\mathbf{y}$	Free deterministic parameters
$\mathbf{z}$	Nominal value for probabilistic parameters
$\boldsymbol{\alpha}$	Scaled eigenvector of weights
$\boldsymbol{\alpha_c}$	Commanded angular acceleration vector
$\hat{\boldsymbol{\alpha}}$	Eigenvector consisting of attribute weights
Γ	Weighting on the time-to-go
$\mathbf{\Delta r}$	Perturbation in the response
$\mathbf{\Delta z}$	Uncertainty for probabilistic parameters
λ	Eigenvalue
ρ	Density
$\boldsymbol{\rho} = [x\ y\ z]^T$	Position vector in the inertial coordinate system
$\dot{\boldsymbol{\rho}} = [\dot{x}\ \dot{y}\ \dot{z}]^T$	Velocity vector in the inertial coordinate system
ϕ	Baseline robustness index
$\boldsymbol{\psi}$	Differential constraint function
$\boldsymbol{\omega}$	Principal body rate vector
μ	Gravitational parameter
$\boldsymbol{\theta}$	Vector of Euler angles in the principal frame
$\boldsymbol{\tau_c}$	Commanded torque in the principal frame

Subscripts

$(\cdot)_0$	Initial value of $(\cdot)$
$(\cdot)_{attitude}$	$(\cdot)$ associated with attitude motion
$(\cdot)_c$	Constraint value of $(\cdot)$
$(\cdot)_{curr}$	Current value of $(\cdot)$
$(\cdot)_f$	Final value of $(\cdot)$
$(\cdot)_i$	i^{th} component of the vector $(\cdot)$
$(\cdot)_l$	Lateral value of $(\cdot)$
$(\cdot)_{\max}$	Maximum value of $(\cdot)$
$(\cdot)_{\min}$	Minimum value of $(\cdot)$
$(\cdot)_{nom}$	Nominal value of $(\cdot)$
$(\cdot)_{\text{opt}}$	Optimal value of $(\cdot)$
$(\cdot)_t$	Target value of $(\cdot)$

$(\cdot)_{translational}$	$(\cdot)$ associated with translational motion
$(\cdot)_v$	Vertical value of $(\cdot)$

Acronyms and Abbreviations

ACS	Attitude control system
AHP	Analytic Hierarchy Process
ALHAT	Autonomous Landing and Hazard Avoidance Technology
EOM	Equation of motion
MSD	Mean square deviation
SNR	Signal-to-noise ratio
TRL	Technology Readiness Level

I. Introduction

TRADITIONALLY assessing the applicability of various guidance and control algorithms to particular applications has been done with a significant amount of human judgment (*i.e.*, where the engineer runs certain tests that stress the algorithm and then provides a recommendation based on their experience). This investigation discusses the development and implementation of a methodology that is capable of providing a baseline index value (*i.e.*, a cardinal number for comparison) for a wide variety of guidance and control algorithms. The baseline index value is a direct assessment of the algorithm's capability in a dispersed environment to achieve a desired objective. Usually, the desired objective is performance based; however, there is no inherent reason that the objective be entirely performance based. In particular, this objective can also be a function of the complexity or consistency factors associated with implementing the algorithm.

A particular focus of this investigation is propulsive descent and landing applications on planetary bodies. An example application will compare a propulsive terminal descent law applied at the Moon at two extremes of the maturity spectrum. This example application will examine the gains (and losses) associated with the maturation process of algorithms and, in particular, how applicable theoretical algorithms chosen during the conceptual design process are to flight implementation.

II. Methodology Overview

A. Objectives

The objectives of this methodology (and the resulting framework) are to:

- Equitably assess how various guidance and control algorithms behave in the presence of uncertainty.
- Provide a meaningful value (*i.e.*, a cardinal number) whereby guidance and control algorithms can be compared to one another to assess their applicability to a certain application.
- Easily be able to assess a wide variety of guidance and control algorithms, provided they have a common application.

The first objective, results from the fact that guidance and control algorithms operate in a non-deterministic environment. That is, they operate where environmental, vehicle, navigational, and state dispersions are present. Therefore, in order to obtain a true comparison of the algorithms, it is necessary that they be evaluated in the environment in which they will ultimately operate. Going along with this idea is the idea that comparison is equitable, which, fundamentally implies that the comparison is performed at the same level. For instance, where the algorithm is tuned to perform the best that it can for the given performance objective. The second objective, is simply a wish for the methodology to give more than just a ranking of how well the algorithm achieves the objective. Instead of just a ranking, it is desirable to obtain the magnitude of the difference in the baseline index for purposes of future downselection and benchmarking. Finally, the third objective, is a statement of flexibility. That is, that it is desired to be able to assess a variety of algorithms using this methodology.

B. Metrics of Interest

In order to create an equitable trade environment, appropriate metrics of interest (or figures of merit) need to be established. It is desired that each of these metrics of interest be *measurable*, *quantifiable*, and *independent* from one another. For the given problem of comparing guidance and control algorithms, three specific classes of metrics of interest can be established—algorithm complexity, algorithm consistency, and algorithm performance. The first class of metrics of interest, algorithm complexity, is a measure of the sophistication of the algorithm. Characteristics associated with the algorithm complexity include how much storage capability is required to execute the algorithm and how much computational effort is required to achieve a solution. Algorithm consistency, is a measure of how reliable the algorithm it is to produce a feasible solution. For instance, how many infeasible commands are commanded. The final class of metrics of interest, algorithm performance, is a measure of the *goodness* of the algorithm, that is, how well the algorithm achieves its objective. These are typically physical quantities such as control effort or terminal state error. As stated above, when selecting metrics of interest for an application, it is desirable that each metric of interest be independent of one another, for instance terminal state error and terminal velocity error should not both be included if velocity is a state of the system. Additionally, when forming a given objective function for comparing algorithms to one another, it is necessary that the desirement of each of the metrics of interest do not conflict (*e.g.*, it is desirable to minimize one metric of interest whereas for another metric of interest it is desirable to maximize the metric of interest). For propulsive terminal descent applications Tables 1-3 list potentially applicable complexity, consistency, and performance metrics of interest, their range of values (as defined), and the desired objective for the metric. It is important to note that the actual metrics of interest used in evaluation of the algorithm are mission dependent (*i.e.*, metrics for one application may not be appropriate for another application).

Table 1. Algorithm complexity metrics of interest.

Metric of Interest	*Definition*	*Minimum Value*	*Maximum Value*	*Objective*
Clock Time to Code	Time it takes to code the algorithm	> 0	$< \infty$	Minimize
Run-time Complexity ($\mathcal{O}$ Based)	"Big-oh" analysis of the algorithm, *i.e.* if $\lvert f(x)\rvert \leq M\lvert g(x)\rvert$ then the algorithm is said to be $\mathcal{O}(g(x))$. The value used is the result of converting $g(x)$ to a Taylor polynomial (if not already polynomial) and determining the power of the power of the leading coefficient, *e.g.*, $g(x) \sim x^a$ then the algorithm is $\mathcal{O}(x^a)$ and therefore the value for comparison is a	> 0	$< \infty$	Minimize
Run-time Complexity (# of Function Calls Based)	Maximum number of function calls it takes to compute one solution	> 0	$< \infty$	Minimize
Memory Required to Store Code	The storage size of the algorithm (excluding comments)	> 0	$< \infty$	Minimize
Memory Required to Run Code	The maximum amount of memory required by the algorithm to execute	> 0	$< \infty$	Minimize
Algorithm Heritage	Approximate TRL Level of the algorithm $9(\text{TRL})^{-1}$	1	9	Minimize
Lines of Code	Number of lines of code	1	$< \infty$	Minimize

Table 2. Algorithm consistency metrics of interest.

Metric of Interest	*Definition*	*Minimum Value*	*Maximum Value*	*Objective*
Number of Infeasible Commands	Count of Infeasible Commands	0	$< \infty$	Minimize

Table 3. Algorithm performance metrics of interest.

Metric of Interest	*Definition*	*Minimum Value*	*Maximum Value*	*Objective*
Terminal Distance Error	The terminal value of $[\frac{(x-x_t)^2}{x_t^2} + \frac{(y-y_t)^2}{y_t^2}]^{1/2}$	0	$< \infty$	Minimize
Terminal Downrange Distance Error	The terminal value of $\lvert\frac{x-x_t}{x_t}\rvert$	0	$< \infty$	Minimize
Terminal Crossrange Distance Error	The terminal value of $\lvert\frac{y-y_t}{y_t}\rvert$	0	$< \infty$	Minimize
Terminal Velocity Error	The terminal value of $[(\dot{h}-\dot{h}_t)^2 + (\dot{x}-\dot{x}_t)^2 + (\dot{y}-\dot{y}_t)^2]^{1/2}$	0	$< \infty$	Minimize
Terminal Vertical Velocity Error	The terminal value of $\lvert\dot{h}-\dot{h}_t\rvert$	0	$< \infty$	Minimize
Terminal Downrange Velocity Error	The terminal value of $\lvert\dot{x}-\dot{x}_t\rvert$	0	$< \infty$	Minimize
Terminal Crossrange Velocity Error	The terminal value of $\lvert\dot{y}-\dot{y}_t\rvert$	0	$< \infty$	Minimize
Propellant Usage	Mass of propellant used by the algorithm (both ACS and translational motion)	0	$< \infty$	Minimize
Reference Trajectory Error	The maximum over the trajectory of $[(\Sigma\frac{(x_{curr}-x_{nom})^2}{x_{nom}^2})/n]^{-1/2}$	0	1	Minimize
Attitude Constraint Distance	The maximum over the trajectory of $[(\frac{(\theta_c-\theta)^2}{\theta_c^2} + \frac{(\psi_c-\psi)^2}{\psi_c^2} + \frac{(\phi_c-\phi)^2}{\phi_c^2})/3]^{-1/2}$	0	1	Minimize
Pitch Constraint Distance	The maximum over the trajectory of $\lvert\frac{\theta_c}{\theta_c-\theta}\rvert$	0	1	Minimize
Yaw Constraint Distance	The maximum over the trajectory of $\lvert\frac{\psi_c}{\psi_c-\psi}\rvert$	0	1	Minimize
Roll Constraint Distance	The maximum over the trajectory of $\lvert\frac{\phi_c}{\phi_c-\phi}\rvert$	0	1	Minimize
Attitude Rate Constraint Distance	The maximum over the trajectory of $[(\frac{(\dot{\theta}_c-\dot{\theta})^2}{\dot{\theta}_c^2} + \frac{(\dot{\psi}_c-\dot{\psi})^2}{\dot{\psi}_c^2} + \frac{(\dot{\phi}_c-\dot{\phi})^2}{\dot{\phi}_c^2})/3]^{-1/2}$	0	1	Minimize
Pitch Rate Constraint Distance	The maximum over the trajectory of $\lvert\frac{\dot{\theta}_c}{\dot{\theta}_c-\dot{\theta}}\rvert$	0	1	Minimize
Yaw Rate Constraint Distance	The maximum over the trajectory of $\lvert\frac{\dot{\psi}_c}{\dot{\psi}_c-\dot{\psi}}\rvert$	0	1	Minimize
Roll Rate Constraint Distance	The maximum over the trajectory of $\lvert\frac{\dot{\phi}_c}{\dot{\phi}_c-\dot{\phi}}\rvert$	0	1	Minimize
Maximum Acceleration Constraint Distance	The maximum over the trajectory of $\lvert\frac{g_c}{g_c-g}\rvert$	0	1	Minimize
Terminal Vertical Acceleration Constraint Distance	The terminal value of $\lvert\frac{a_{v_c}}{a_{v_c}-a_v}\rvert$	0	1	Minimize
Terminal Lateral Acceleration Constraint Distance	The terminal value of $\lvert\frac{a_{l_c}}{a_{l_c}-a_l}\rvert$	0	1	Minimize
Minimum Altitude Constraint Distance	The maximum over the trajectory of $\lvert\frac{h_{\min_c}}{h_{\min}-h_{\min_c}}\rvert$	0	1	Minimize
Maximum Altitude Constraint Distance	The maximum over the trajectory of $\lvert\frac{h_{\max_c}}{h_{\max_c}-h_{\max}}\rvert$	0	1	Minimize
Minimum Velocity Constraint Distance	The maximum over the trajectory of $\lvert\frac{v_{\min_c}}{v_{\min}-v_{\min_c}}\rvert$	0	1	Minimize
Maximum Velocity Constraint Distance	The maximum over the trajectory of $\lvert\frac{v_{\max_c}}{v_{\max_c}-v_{\max}}\rvert$	0	1	Minimize

C. Robustness Index

1. Objective Function Definition

With the chosen metrics of interest, an objective function can be defined which captures the characteristics that are desirable in the given algorithm being baselined. Assuming a linear combination of the metrics of interest, this objective function has the form

$$J = \boldsymbol{\alpha}^T \mathbf{X} = \sum_{i=1}^{n} \alpha_i X_i \tag{1}$$

where α_i is a weighting on the importance of the i^{th} variable, X_i. The X_i are the metrics of interest adjusted for minimization or maximization. Note that this objective function is a scalar representation of a multi-dimensional space. As such, multiple solutions for the value of X_i could occur for a singular value of J, even for fixed α_i. Hence, care should be taken to define the metrics of interest such that in the domain of interest J behaves acceptably well for the application.

2. Determination of the Weights

The weighting vector, $\boldsymbol{\alpha}$, gives a relative significance (influence) of each of the metrics of interest on the objective function. Ideally, α_i takes on a value between zero and one and the sum of each of the α_i is unity. In order to determine the vector of weights, any one of a set of multi-attribute decision making techniques can be used, including solicitation of expert opinion, the Analytic Hierarchy Process (AHP), and inner product of vectors. For this methodology, it is assumed that AHP is used for the weighting, as it is a relatively simplistic to implement and results in a meaningful, cardinal weighting vector. Additionally, consistency checks are available for the pairwise comparisons.

AHP is an approach to multi-attribute decision making developed by Dr. Thomas Saaty in the 1970s which creates more tractable subproblems (attributes) for comparison.[1] These attributes are compared *pairwise* with a relative scale of 1 to 9, with 1 indicating neutral preference and 9 indicating strong preference. These pairwise comparisons are used to form a reciprocal matrix as shown in Table 4. In Table 4, element a_{ij} is interpreted as the relative preference of element i with respect to element j and $a_{ij} \in \{\frac{1}{9}, \frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ with fractions indicating the relative preference for element j over element i. Denoting this matrix as $\mathbf{A}$, the weighting vector is a scaled version of the principal eigenvector of $\mathbf{A}$, that is, the principal solution of the equation

Table 4. Reciprocal matrix resulting from pairwise comparisons using AHP.

	Attribute 1	**Attribute 2**	**Attribute 3**	**Attribute 4**
Attribute 1	1	a_{12}	a_{13}	a_{14}
Attribute 2	$\frac{1}{a_{12}}$	1	a_{23}	a_{24}
Attribute 3	$\frac{1}{a_{13}}$	$\frac{1}{a_{23}}$	1	a_{34}
Attribute 4	$\frac{1}{a_{14}}$	$\frac{1}{a_{24}}$	$\frac{1}{a_{34}}$	1

$$\mathbf{A}\hat{\boldsymbol{\alpha}} = \lambda\hat{\boldsymbol{\alpha}} \tag{2}$$

where the eigenvector, $\hat{\boldsymbol{\alpha}}$ is scaled such that

$$\boldsymbol{\alpha} = \frac{\hat{\boldsymbol{\alpha}}}{\sum_{i=1}^{N} \hat{\alpha}_i} \tag{3}$$

to produce the weighting vector. In other words, the weighting vector is a scaled dominant basis of the reciprocal matrix. Additionally, the weighting vector not only gives an ordinal ranking of each of the metrics of interest, it also gives cardinal meaning to them. For instance, if a metric of interest has a weighting that is twice another metric of interest, it is twice as significant (in the view of the assessor) as the other metric of interest.

3. Mean Square Deviation and Signal-to-Noise Ratio

Since it is of interest to determine how robust the guidance or control algorithm is, two additional concepts from statistics and robust design are used—the mean square deviation (MSD) and signal-to-noise ratio (SNR).[2] Traditionally, assessment of an algorithm under uncertainty is done using Monte Carlo simulation. For each run of the Monte Carlo simulation, a different objective function value will result, J_i. To capture the robustness in a single variable, consider that robustness is a measure both of how close a value is to a target objective function value *and* the spread of the response. The MSD captures both of these traits, and

is defined as[2]

$$\mathrm{MSD}(\theta_1, \theta_2) = \mathbb{E}\left[(\theta_1 - \theta_2)^2\right] = \frac{1}{n}\sum_{i=1}^{n}(J_{1,i} - J_{2,i})^2 \tag{4}$$

where $J_{1,i}$ is the i^{th} sampled value of the objective function and $J_{2,i}$ is the i^{th} desired value of the objective function.

Since the MSD can take on a large number of values, for practical purposes it is necessary to alter it to distinguish various values. The SNR accomplishes this task as well as provides a common objective function to optimize on, if it is desired. The definition of the SNR used in this methodology is taken from Taguchi robust design and has the general form[2]

$$\mathrm{SNR} = -10\log_{10}(\mathrm{MSD}) \tag{5}$$

Given that the desire may be to minimize the objective function, maximize the objective function, or bring the objective function to a certain target, the SNR can take on three specific forms. These are shown in Eqs. (6)-(8), where Eq. (6) would be used if the minimum value of the objective function is desired, Eq. (7) would be used if the maximum value of the objective function is desired, and Eq. (8) would be used if it is desired that the objective function take on a certain value.

$$\mathrm{SNR} = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n}J_i^2\right) \tag{6}$$

$$\mathrm{SNR} = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{J_i^2}\right) \tag{7}$$

$$\mathrm{SNR} = -10\log_{10}\left[\frac{1}{n}\sum_{i=1}^{n}\left(J_i - \hat{J}\right)^2\right] \tag{8}$$

Pictorially, the maximization of the SNR is shown in Figure 1, where it is seen that maximizing the SNR reduces the variance of the distribution and shifts the mean toward the desired target value.

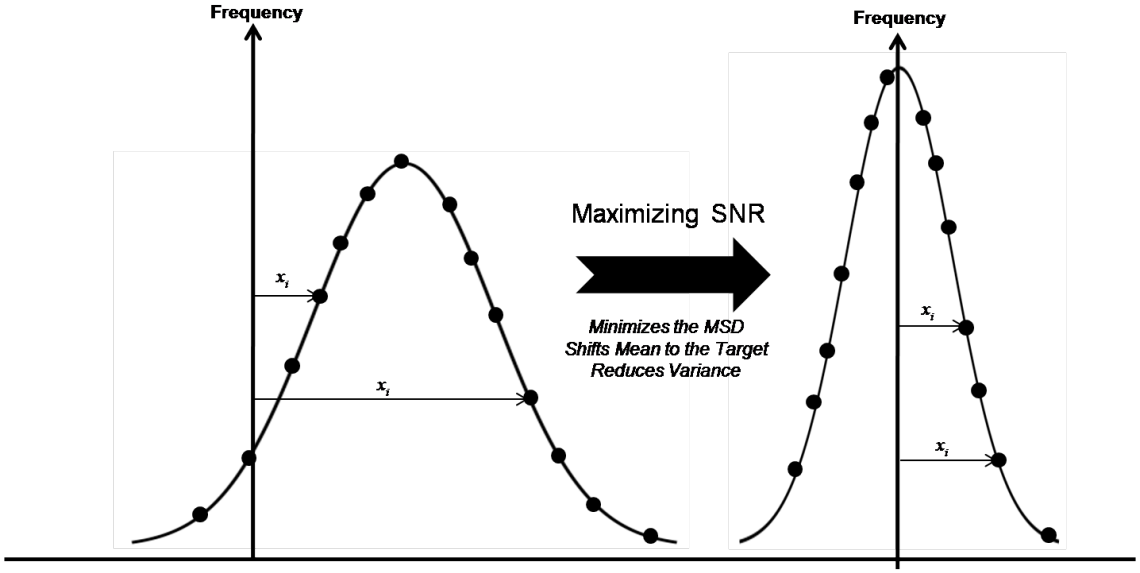

Figure 1. Signal-to-noise ratio maximization.

4. Robustness Index Definition

In light of the previous section's discussion, it is clear that the SNR captures the multi-dimensional robustness problem in a single value that is quantifiable, measurable, and cardinal. Since it is based on the logarithmic scale, it has the ability to represent large or small number equally well. Additionally, the SNR gives a suitable robustness optimization function, should that be desired. Hence, the robustness index is defined in this work as

$$\phi(J) = -10\log_{10}(\text{MSD}) \tag{9}$$

As seen in Figure 2, the robustness index, $\phi(J)$, is monotonically decreasing for increasing values of the MSD with positive values occurring only for if the MSD is less than one.

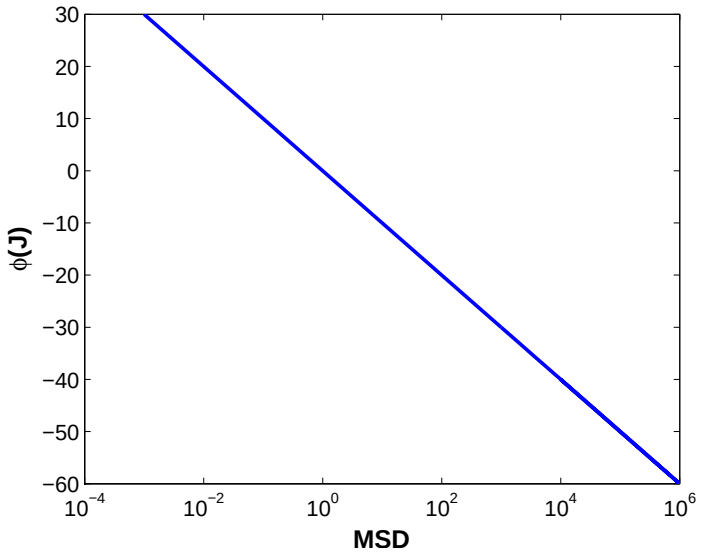

Figure 2. Robustness index behavior.

D. Creating a Robustness Index Baseline for Guidance and Control Algorithms

To create a robustness index baseline for guidance and control algorithms, first consider the analysis structure shown in Figure 3. The inputs into the analysis consist of fixed deterministic parameters ($\mathbf{x}$), free

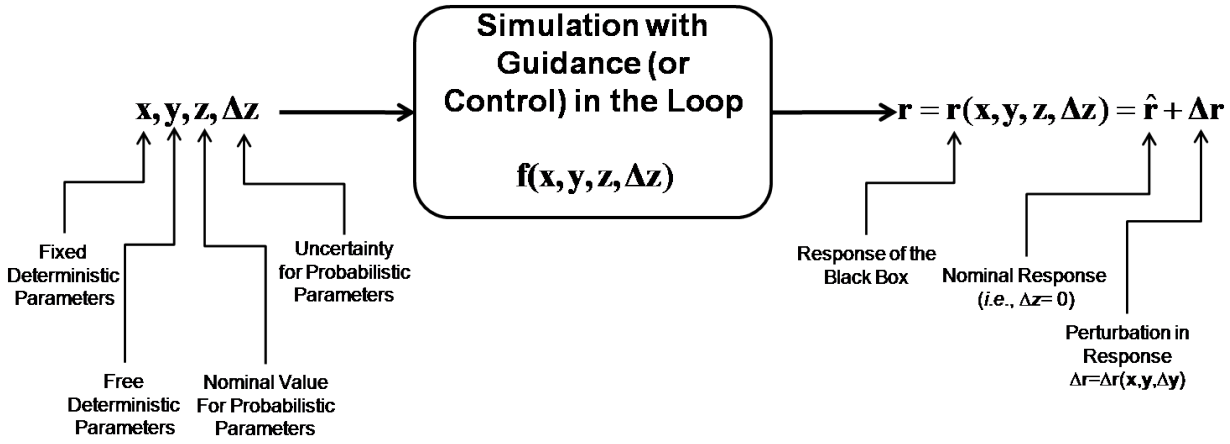

Figure 3. Analysis structure for robustness indexing.

deterministic parameters ($\mathbf{y}$), nominal values of the probabilistic parameters ($\mathbf{y}$), and variations from the nominal value of the probabilistic parameters ($\mathbf{\Delta z}$). The fixed deterministic parameters are values used in the analysis that are specified by the user but are not free to change from their specified value at the beginning of the analysis. The free deterministic parameters are values used in the analysis; however, their values can change in the analysis by some external process (*e.g.*, an external optimizer). The probabilistic parameters are variables used in the analysis which behave according to an *a priori* defined probability density function which have a nominal mean and deviation from this mean. Using these inputs, a simulation is conducted with the guidance or control algorithm being evaluated in the loop, resulting in the response of the simulation, $\mathbf{r}$ which is comprised of a nominal response ($\hat{\mathbf{r}}$) and a perturbation from the nominal response ($\mathbf{\Delta r}$).

With this framework in mind, the algorithm's robustness index can be established in the following manner:

1. Create an appropriate objective function, J
   (a) Select the desired metrics of interest for the problem
   (b) Formulate the components of the objective function such that they have a common objective (*i.e.*, minimize or maximize)
   (c) Determine the weight for each of the metrics of interest
2. Determine the distribution of the desired objective function
   (a) Identify the parameters of the problem and their distribution (if applicable)—$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ and $\Delta\mathbf{z}$
   (b) Record the value of the objective function, J, for the various combinations of the parameters
3. Obtain the SNR for objective function
   (a) Identify the form of the SNR to use depending on the objective of J
4. Tune the free parameters, $\mathbf{y}$, to give the algorithm the best performance possible for the desired objective
   (a) The optimization is formulated such that

$$\begin{aligned} \textbf{Maximize:} \quad & -10\log_{10}(\text{MSD}) \\ \textbf{Subject To:} \quad & \dot{\mathbf{u}} = \boldsymbol{\psi}(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{\Delta z}) \\ & \mathbf{q} \leq \mathbf{h}(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{\Delta z}) \\ \textbf{By Varying:} \quad & \mathbf{y} \end{aligned}$$

5. Set the baseline index value equal to the tuned SNR for the objective (*i.e.*, $\phi(J) = \text{SNR}_{\text{opt}}$)

It is important to note that should there not be any free deterministic variables ($\mathbf{y}$) then the optimization cannot proceed and Step 4 will be omitted.

III. Analysis Framework

The methodology discussed in Section II.D has been implemented in a Matlab environment. The following sections discuss details of the framework, including its organization, variable flow, and other specifics to the implemented methodology.

A. Organization

As an example of how such an analysis framework can be implemented consider the organizational diagram shown in Figure 4. For this implementation, the framework consists of six primary functions, three of which are user modified, two are user supplied, and one is user selectable. This organization allows for the tool to be extremely generic and flexible in its analysis of guidance and control algorithms, particularly because the interface between the algorithms are user supplied. This means the user ensures the variables are used correctly in the user supplied acceleration function.

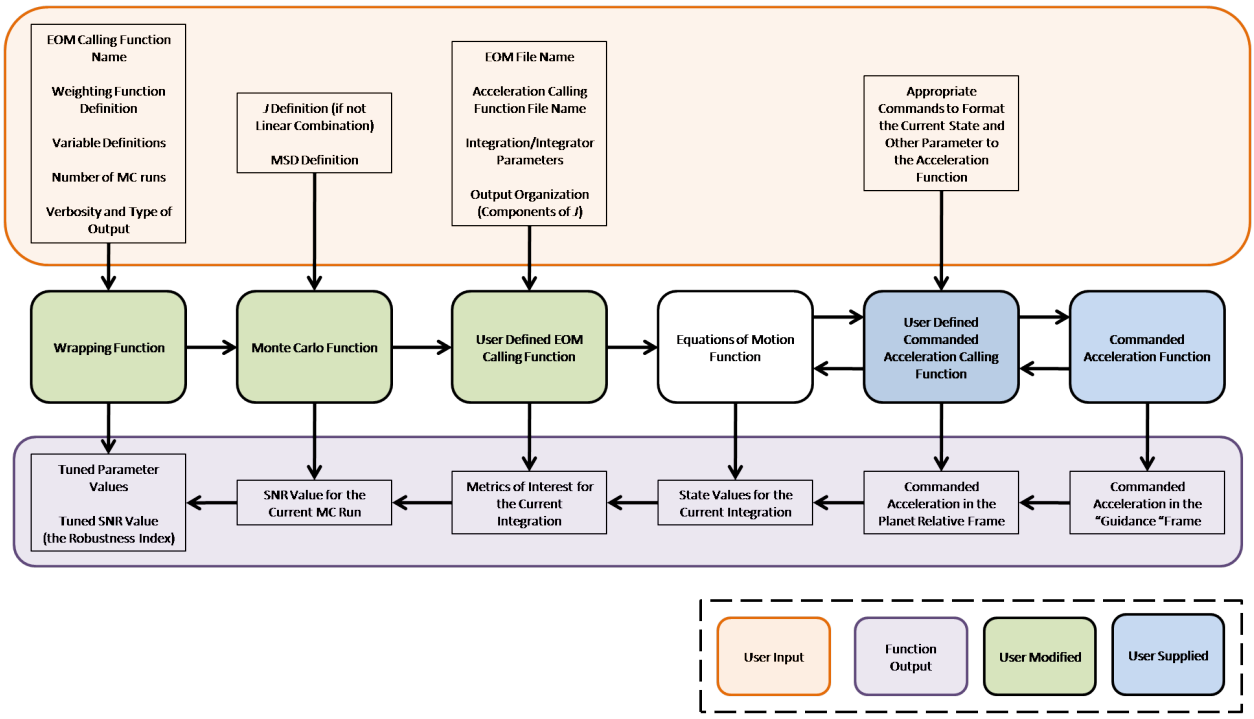

Figure 4. Organization of the analysis tool.

B. Function Descriptions

1. Wrapping Function

The wrapping function is the driving function for the entire analysis framework. Within this function the fixed and free deterministic variables are defined, distributions for each of the probabilistic parameters yield random variables for use in the Monte Carlo, and the weights for the objective function are specified. These weights can be prespecified in the function or in real-time by calling an AHP script and using the results of that analysis before completing the robustness analysis. This function also performs the tuning of the free deterministic parameters of the problem and outputs the baseline robustness index for the algorithm as well as the optimized values for the free deterministic parameters. This tuning is performed using Matlab's `fmincon` function, which uses a gradient approach to constrained multivariable minimization.

2. Monte Carlo Function

The execution of each Monte Carlo run is performed by this function. This function loops through the predefined random variables (defined in the wrapping function) then computes the MSD and SNR for the current Monte Carlo set. Assuming it is desired that the minimization form of the SNR is used and the objective function is a pure linear combination of the output from the equation of motion (EOM) calling function, no modification of this function is necessary.

3. User Defined EOM Calling Function

The user defined EOM calling function acts as an interface between the input problem parameters and the equations of motion. This interface assigns values to each of the required parameters of the equations of motion, specifies the algorithm which is being analyzed, as well as any integration parameters. The actual call to the integrator is done from this function. Currently Matlab's `ode45` is used with either an events function or maximum time terminating the propagation of the dynamic equations.

4. Equations of Motion

The equations of motion are predefined from a choice of three different sets in the equation of motion calling function. The three choices include a three degree-of-freedom (DOF) translational motion set, a three DOF attitude set in the principal frame, and a six DOF set that combines the previous two sets.

Three Degree-of-Freedom Translational Motion
The three DOF translational motion set of equations of motion is formed in inertial, Cartesian space. It accounts for spherically symmetric gravity such that

$$\mathbf{f_g} = -\frac{m\mu}{\parallel \mathbf{r} \parallel^3}\mathbf{r} + \mathbf{f_{pert}} \tag{10}$$

where the perturbation, $\mathbf{f_{pert}}$, is given by a Markov Process gravity model (if desired). In Eq. (10), m is the mass of the vehicle, μ is the gravitational parameter of the planet, and $\mathbf{r}$ is the position vector of the vehicle expressed in the inertial frame. Aerodynamic forces are accounted for when an atmosphere file is specified (this file gives the density as a function of altitude). The aerodynamic forces are given by

$$\mathbf{f_l} = \bar{q}_\infty C_L S \hat{\mathbf{l}} \tag{11}$$

$$\mathbf{f_d} = -\bar{q}_\infty C_D S \frac{\mathbf{v_{rel}}}{\parallel \mathbf{v_{rel}} \parallel} \tag{12}$$

where Eq. (11) specifies the lift force acting on the vehicle in an inertial frame and Eq. (12) specifies the drag force acting on the vehicle in an inertial frame. In these equations, $\bar{q}_\infty$ is the free-stream dynamic pressure ($=\frac{1}{2}\rho v_{rel}^2$), $\mathbf{v_{rel}}$ is the planet relative velocity vector expressed in the inertial frame, C_L is the lift coefficient, C_D is the drag coefficient, S is the projected area of the vehicle, and $\hat{\mathbf{l}}$ is the inertial direction of the lift vector. The final set of forces accounted for are those commanded by the guidance or control algorithm. These are given by

$$\mathbf{f_c} = m\mathbf{a_c} \tag{13}$$

where m is the mass of the vehicle and $\mathbf{a_c}$ is the commanded acceleration output from the algorithm. In the implementation of the EOMs, the commanded acceleration is given in the planet relative frame and then is transformed to the inertial frame. With these forces defined, the time rate of change of the state vector, defined in this case as $\mathbf{X} = \left(\mathbf{r_{3\times1}^T}\ \mathbf{v_{3\times1}^T}\ m\right)^T$, is given by

$$\dot{\mathbf{X}} = \begin{pmatrix} \boldsymbol{\dot{\rho}}_{\mathbf{3\times1}} \\ \boldsymbol{\ddot{\rho}}_{\mathbf{3\times1}} \\ \dot{m} \end{pmatrix} = \begin{pmatrix} \mathbf{v_{3\times1}} \\ \dfrac{\mathbf{f_g} + \mathbf{f_l} + \mathbf{f_d} + \mathbf{f_c}}{m} \\ -\dfrac{|T|}{g_0 I_{sp}} \end{pmatrix} \tag{14}$$

Three Degree-of-Freedom Attitude Motion
The attitude equations of motion are written in the principal body frame. That is a frame whose has its origin at the center of gravity and oriented such that the inertia tensor is diagonal for all time, that is

$$\mathbf{I_p} = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix} \tag{15}$$

Again, the guidance or control algorithm is assumed to output an acceleration, this time an angular acceleration. As such, the commanded torque can be computed as

$$\boldsymbol{\tau_c} = \mathbf{I_p}\boldsymbol{\alpha_c} \tag{16}$$

Using Euler's equations in the principal frame and defining the state vector to be $\mathbf{X} = \left(\boldsymbol{\theta}_{\mathbf{3\times1}}^{\boldsymbol{T}}\ \boldsymbol{\omega}_{\mathbf{3\times1}}^{\boldsymbol{T}}\ m\right)^T$ leads to the time rate of change of the state to be

$$\dot{\mathbf{X}} = \begin{pmatrix} \boldsymbol{\dot{\theta}}_{\mathbf{3\times1}} \\ \boldsymbol{\dot{\omega}}_{\mathbf{3\times1}} \\ \dot{m} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\omega_{3\times1}} \\ \mathbf{I_p^{-1}}\left(\boldsymbol{\tau_c} - \tilde{\boldsymbol{\omega}}\mathbf{I_p}\boldsymbol{\omega}\right) \\ -\dfrac{|T|}{g_0 I_{sp}} \end{pmatrix} \tag{17}$$

where $\tilde{\boldsymbol{\omega}}$ is the skew-symmetric matrix associated with $\boldsymbol{\omega}$.

Six Degree-of-Freedom Motion
The six degree-of-freedom equations of motion, as implemented, are just a combination of the prior two sets of equations. That is, the time rate of change of the state vector, $\mathbf{X} = \left(\mathbf{r}_{\mathbf{3\times1}}^{\mathbf{T}}\ \mathbf{v}_{\mathbf{3\times1}}^{\mathbf{T}}\ \ \boldsymbol{\theta}_{\mathbf{3\times1}}^{\boldsymbol{T}}\ \boldsymbol{\omega}_{\mathbf{3\times1}}^{\boldsymbol{T}}\ m\right)^T$, is simply given by

$$\dot{\mathbf{X}} = \begin{pmatrix} \dot{\boldsymbol{\rho}}_{\mathbf{3\times1}} \\ \ddot{\boldsymbol{\rho}}_{\mathbf{3\times1}} \\ \dot{\boldsymbol{\theta}}_{\mathbf{3\times1}} \\ \dot{\boldsymbol{\omega}}_{\mathbf{3\times1}} \\ \dot{m} \end{pmatrix} = \begin{pmatrix} \mathbf{v}_{\mathbf{3\times1}} \\ \dfrac{\mathbf{f_g} + \mathbf{f_l} + \mathbf{f_d} + \mathbf{f_c}}{m} \\ \boldsymbol{\omega}_{\mathbf{3\times1}} \\ \mathbf{I_p^{-1}}\left(\boldsymbol{\tau_c} - \tilde{\boldsymbol{\omega}}\mathbf{I_p}\boldsymbol{\omega}\right) \\ -\dfrac{|T_{translational} + T_{attitude}|}{g_0 I_{sp}} \end{pmatrix} \tag{18}$$

Note, that in this case, the mass loss is due to a combination of the thrust due to translational motion of the vehicle, $T_{translational}$, and that of the attitude motion of the vehicle, $T_{attitude}$.

5. User Defined Acceleration Calling Function

The user defined acceleration calling function acts as an interface between the equations of motion and the commanded acceleration function. The effective purpose of this function is to format the input into the commanded acceleration in a format that is appropriate to it.

6. Commanded Acceleration Function

The commanded acceleration function is the guidance or control algorithm which is supplied by the user of the framework.

IV. Guidance Algorithm Descriptions

For the example cases described subsequently, consider the following three translation guidance algorithms. Each guidance algorithm is closed-loop and returns a commanded acceleration that is used by the Commanded Acceleration Function.

A. Gravity Turn

A gravity turn, as used by the Lunar Surveyor spacecraft, provides a non-precise soft landing on a planetary body.[3] As such, it does not target a particular landing site, instead the algorithm targets an altitude at which to null the velocity (*i.e.*, the state vector in the target relative local vertical local horizontal frame is of the form $\mathbf{X_t} = \left(\bar{\mathbf{r}}^T\ \bar{\mathbf{v}}^T\right)^T = (*\ *\ 0\ 0\ 0\ 0)^T$, where * indicates an arbitrary, non-specified state). For this guidance law, the thrust is assumed to oppose the velocity vector and the commanded acceleration magnitude is constant through the descent (thrust decreases linearly with the decrease in mass).

The geometry assumed for the gravity turn guidance law is shown in Fig. 5. Assuming the planet is (1) flat, (2) non-rotating, and that it has (3) a uniform gravitational field, the commanded acceleration $\mathbf{a_c}$ is a function of three parameters:

1. $g = \| \mathbf{g} \|_2$, the acceleration magnitude due to gravity (assumed to be surface gravitational acceleration)
2. $\bar{v} = \| \mathbf{v} - \mathbf{v_t} \|_2$, the magnitude of the spacecraft's velocity relative to the target
3. $R = -h \csc\gamma = h \sec\psi$, the slant range to the target altitude at which the planet-fixed velocity is to be nullified

and is given by

$$\mathbf{a_c} = -\left(1 + \frac{\bar{v}^2}{2Rg}\right)\frac{\mathbf{v}}{v} \tag{19}$$

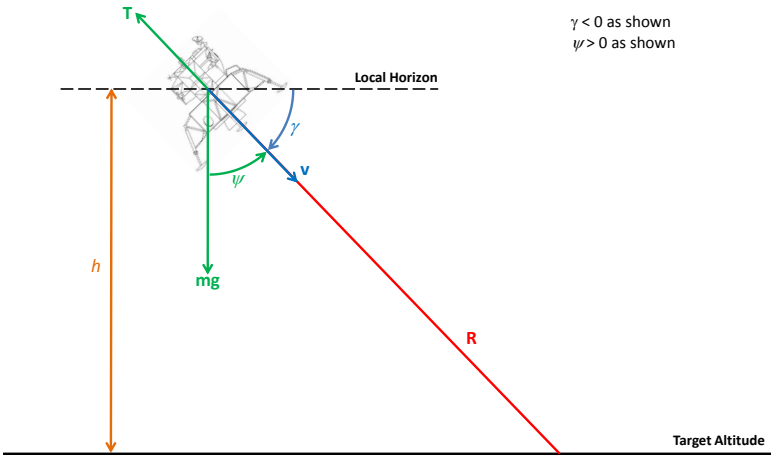

Figure 5. Gravity turn guidance law geometry

with corresponding thrust magnitude

$$T = m\left(g + \frac{\bar{v}^2}{2R}\right) \tag{20}$$

This guidance law follows from the desire to follow an idealized parabolic relation for the velocity in terms of the slant range. For values of $\psi \in [0°, 45°]$ the constant acceleration assumption is appropriate, and a closed-loop (in the dispersed sense) guidance law that satisfies the boundary conditions is obtained.

B. General Analytical Optimal Planetary Landing Law

For conceptual design with targeted landing capability (*i.e.*, both position and velocity targeting) consider the optimal planetary landing law described by D'Souza.[4] This algorithm solves the variational two-point boundary value problem for an optimal trajectory according to the cost function

$$J = \Gamma t_f + \int_0^{t_f} \mathbf{a}^T(t)\mathbf{a}(t)dt \tag{21}$$

where Γ is a weight on free final time. In the solution procedure, it is assumed that the flight is over flat, non-rotating, atmosphere-free planet. Under these assumptions, the time-to-go (t_{go}), that is, the time before touchdown, can be found by solving the transversality condition for the real, positive root of

$$t_{go}^4 - 2\frac{\bar{\mathbf{v}}^T\bar{\mathbf{v}}}{\Gamma + \frac{g^2}{2}}t_{go}^2 - 12\frac{\bar{\mathbf{v}}^T\bar{\mathbf{r}}}{\Gamma + \frac{g^2}{2}}t_{go} - 18\frac{\bar{\mathbf{r}}^T\bar{\mathbf{r}}}{\Gamma + \frac{g^2}{2}} = 0 \tag{22}$$

where g is the local acceleration magnitude due to gravity ($=\| \mathbf{g} \|$) and the vectors are given by

$$\bar{\mathbf{r}} = (\bar{r}_1 - \bar{r}_{f,1}\ \bar{r}_2 - \bar{r}_{f,2}\ \bar{r}_3 - \bar{r}_{f,3})^T \tag{23}$$

$$\bar{\mathbf{v}} = (\bar{v}_1 - \bar{v}_{f,1}\ \bar{v}_2 - \bar{v}_{f,2}\ \bar{v}_3 - \bar{v}_{f,3})^T \tag{24}$$

$$\mathbf{g} = (0\ 0\ g)^T \tag{25}$$

where the f subscript indicates the terminal state value. The commanded acceleration is then given by

$$\mathbf{a_c} = -4\frac{\bar{\mathbf{v}}}{t_{go}} - 6\frac{\bar{\mathbf{r}}}{t_{go}^2} - \mathbf{g} \tag{26}$$

Note that there are no state constraints placed on the system which could result in infeasible subterranean trajectories. However, by increasing the weighting on the free time-to-go, Γ, a pseudo-constrained trajectory can be obtained. Additionally, this weighting gives a free deterministic parameter for the optimizer to act upon during the optimization process.

C. ALHAT Terminal Phase Algorithm

The Autonomous Landing and Hazard Avoidance Technology (ALHAT) guidance algorithm is an algorithm designed for technology planning and development for NASA's lunar exploration campaign. It is comprised of four maneuvers, a braking maneuver, a pitch-up maneuver, an approach maneuver, and a vertical descent maneuver.[5] These phases are shown in Figure 6.

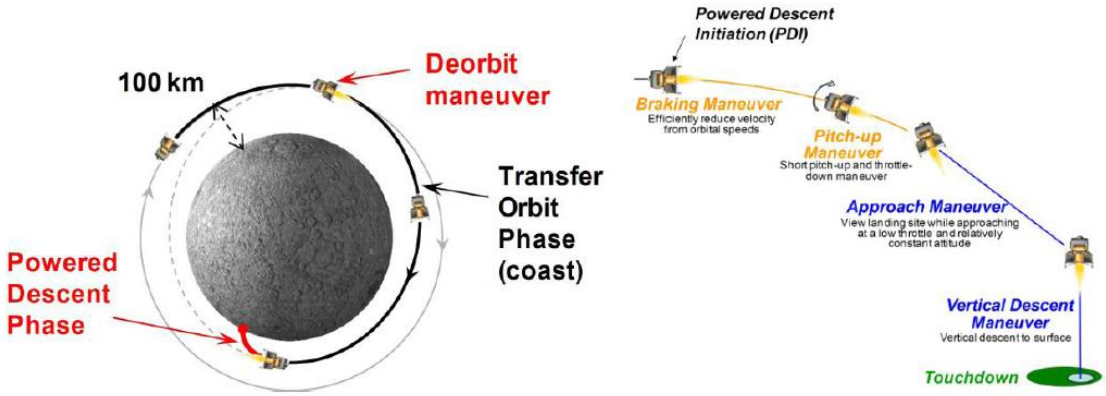

Figure 6. ALHAT descent phases and powered descent sub-phases[5]

Like the D'Souza algorithm described previously, the approach maneuver of ALHAT's guidance algorithm provides a solution to the variable thrust two-point boundary value problem for the minimum control effort solution. However, unlike the D'Souza algorithm, the ALHAT algorithm has practical constraints placed on it for things such as the initial trajectory shape, the vehicle's attitude, the touchdown glide slope, control system implementation, and the rate that commands are output. With these constraints removed, the optimal solution to the unconstrained variational problem is precisely that of an optimal linear profile for a given approach time. This is the same solution as derived by D'Souza. It should be noted, that unlike the general analytical optimal planetary landing law, there are no *useful* parameters to tune for this algorithm. Therefore, when obtaining the baseline robustness index, it is not tuned (optimized).

V. Using the Baseline Robustness Index in Algorithm Selection

As an example application of the baseline robustness index during the conceptual design process, consider the selection of the propulsive terminal descent algorithm for a robotic lander on the Moon. For this application it is desired to achieve a soft landing at a desired landing spot. However, it is recognized that mass may become a larger driving factor in the design than precision landing capability. Hence, two algorithms are under consideration, the non-targeted gravity turn guidance law and the targeted general analytical optimal landing law.

A. Objective Function Definition

Based on the selection criterion described above, the metrics of interest are: the mass of the propellant used during the descent (m_{prop}) and the landed state accuracy ($\| \mathbf{r_f} - \mathbf{r_t} \|$ and $\| \mathbf{v_f} - \mathbf{v_t} \|$). Therefore the objective function is of the form

$$J = \alpha_1 m_{prop} + (1 - \alpha_1)\sqrt{1 \times 10^{-3} \| \mathbf{r_f} - \mathbf{r_t} \| + \| \mathbf{v_f} - \mathbf{v_t} \|} \tag{27}$$

where the weight on the propellant mass, α_1 is varied parametrically in order to gauge the impact of the precision landing capability on the landed mass. With values of α_1 less than 0.5 there is a bias towards landed accuracy, while values greater than 0.5 biases the objective towards propellant mass.

B. Problem Parameters

For the analysis, it is assumed that the dispersions shown in Table 5 can be used. The three different variable distributions used are denoted as follows: (1) uniform ($\mathcal{U}(x_{\min}, x_{\max})$), (2) Gaussian ($\mathcal{N}(\mu, \sigma^2)$), and (3) triangular ($\mathcal{T}(x_{\min}, x_{\max}, x_{\text{mode}})$). Notice the uncertainty in the landed position effectively represents a map-tie error, that is, an uncertainty in the mapping of terrain features to the inertial position.

Table 5. Parameters used in the analysis.

Parameter	Nominal Value	Distribution	Units
r_{planet}	1,737,100	-	m
μ	4.9027779×10^{12}	$\mathcal{T}\left(4.80472 \times 10^{12}, 5.00083 \times 10^{12}, 4.9027779 \times 10^{12}\right)$	m^3/s^2
$\mathbf{r_t}$	$(1,746,660\ 0\ 0)^T$	$(\mathcal{N}(1,746,660, 0)\ \mathcal{N}(0, 100)\ \mathcal{N}(0, 100))^T$	m
$\mathbf{v_t}$	$(0\ 0\ 0)^T$	-	m/s
$\mathbf{r_0}$	$(1,738,000\ 0\ 0)^T$	$(\mathcal{N}(1,738,000, 100)\ \mathcal{N}(0, 500)\ \mathcal{N}(0, 500))^T$	m
$\mathbf{v_0}$	$(-173.2\ 100\ 0)^T$	$(\mathcal{N}(-173.2, 100)\ \mathcal{N}(100, 100)\ \mathcal{N}(0, 10))^T$	m/s
m_0	300	$\mathcal{U}(298, 302)$	kg
I_{sp}	218	$\mathcal{U}(345, 355)$	s

C. Results

Using the analysis framework on each of the algorithms the results shown in Figure 7 were obtained.

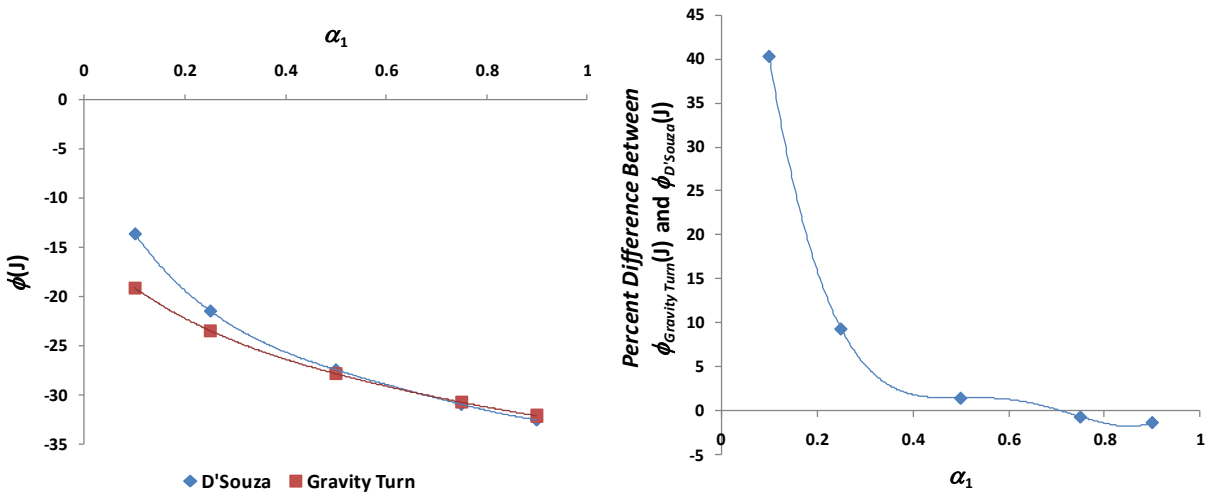

Figure 7. Robustness index variation with importance of propellant mass, α_1

As is expected, there is a crossover between the two algorithms depending on the significance on the landed accuracy, with a crossover point occurring when the propellant mass is $\sim 133\%$ more important than the landed accuracy (*i.e.*, $\alpha_1 \approx 0.7$). With the robustness index presented parametrically, it allows the designers the flexibility to choose the guidance algorithm based on what they feel the current design drivers are while retaining information regarding the performance at other conditions. In this instance, with the general analytical optimal planetary landing law's robustness baseline being no worse than 1.3% different than gravity turn's robustness baseline index across the range of α_1 investigated, it would be to the designers' advantage to downselect to the analytical optimal planetary landing law for development for the metrics of interest considered in this application.

VI. Effect of Guidance Algorithm Maturity on the Baseline Robustness Index

Using the analysis framework described, the effect of the maturation of a propulsive terminal descent guidance law was also investigated. Coupled with this analysis is a sensitivity analysis of the size of the Monte Carlo as well as whether or not the methodology is able to achieve a known minimum.

A. Approach

Two of the propulsive terminal descent guidance algorithms described above—the general analytical optimal planetary landing law as well as the ALHAT terminal phase algorithm, were used in the analysis of the impact of algorithm maturation. As previously described, these algorithms demonstrate a natural progression from the theoretical development of an algorithm (the general analytical optimal planetary landing law) to an algorithm that takes into account practical considerations and is effectively flight ready (ALHAT's terminal phase). The general approach to these analyses is outlined below

- Develop a set of common metrics of interest for the problem
- Using these metrics of interest, develop appropriate weights for each metric, in order to form an objective function
- Using the ALHAT DAC2 vehicle and dispersions, obtain each baseline robustness index using
  - The general analytical optimal planetary landing law, for three different Monte Carlo sizes—500 samples, 1000 samples, and 1500 samples
  - The terminal phase of ALHAT's algorithm for the 1000 case sample size

With these data, the performance of the optimizer coupled with the baseline robustness index, the sensitivity to Monte Carlo size, and the effect of algorithm maturation can be analyzed.

B. Objective Function Definition

The three objective functions used in this analysis are listed below:

1. $J = m_{prop}$
2. $J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.5 m_{prop}$
3. $J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.2 n_{lines} + 0.3 m_{prop}$

These three objective functions were chosen with the overall objective of the investigation, to incorporate non-performance based metrics of interest into the analysis of guidance and control algorithms. The first two objective functions are purely performance based objectives, whereas the third includes a measure of algorithm complexity. Additionally, for the general analytical optimal planetary landing law algorithm, each of the objective functions have a known deterministic solution, namely $\Gamma = 0$.

C. Problem Parameters

The vehicle parameters (m_0 and I_{sp}) listed in Table 6 correspond to those of the ALHAT DAC2 vehicle. Note that the initial and target states are chosen arbitrarily based on those used by D'Souza in his example application problem.[4] For shorthand, the different variable distributions used for the dispersions are identified as: uniform ($\mathcal{U}(x_{\min}, x_{\max})$), Gaussian ($\mathcal{N}(\mu, \sigma^2)$), and triangular ($\mathcal{T}(x_{\min}, x_{\max}, x_{\text{mode}})$).

D. Results

Using the analysis framework on each of the algorithms the results shown in Table 7 were obtained. In Table 7, the column denoted as "D'Souza" corresponds to the general analytical optimal planetary landing law and the column denoted as "ALHAT" corresponds to the ALHAT terminal phase algorithm. Considering the performance of the optimizer on the SNR, the results show that increasing control effort (*i.e.*, $\Gamma > 0$) will improve robustness (less variation in the response). This is expected as increasing the control effort reduces the dispersion in the landing state error. For each of the objective functions, it is seen there is very little sensitivity to the size of the Monte Carlo simulation. In particular, the tuned baseline robustness index varies by less than 0.1% for all cases examined. Maturing the algorithm also did not significantly alter the

Table 6. Parameters used in the analysis.

Parameter	Nominal Value	Distribution	Units
r_{planet}	1,737,100	-	m
μ	4.9027779×10^{12}	$\mathcal{T}\left(4.80472 \times 10^{12}, 5.00083 \times 10^{12}, 4.9027779 \times 10^{12}\right)$	m^3/s^2
$\mathbf{r_t}$	$(0\ 0\ r_{planet})^T$	-	m
$\mathbf{v_t}$	$(0\ 0\ 0)^T$	-	m/s
$\mathbf{r_0}$	$(152,400\ 30,480\ 1,752,340)^T$	$(\mathcal{N}(152400,100)\ \mathcal{N}(30480,100)\ \mathcal{N}(1752340,10))^T$	m
$\mathbf{v_0}$	$(-914.4\ 0\ 0)^T$	$(\mathcal{N}(-914.4,0.01)\ \mathcal{N}(0,0.1)\ \mathcal{N}(0,0.8))^T$	m/s
m_0	32,240	$\mathcal{U}(32456,34024)$	kg
I_{sp}	446.9	$\mathcal{U}(444.9,448.9)$	s

Table 7. Analysis results.

		D'Souza		*ALHAT*	
Objective Function	N	$\mathbf{\Gamma}$	$\phi(J)$	$\phi(J)$	% Difference
$J = m_{prop}$	500	0.3834	-77.4275	-	-
$J = m_{prop}$	1000	0.1738	-77.4135	-77.6197	0.266
$J = m_{prop}$	1500	0.1731	-77.4134	-	-
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.5 m_{prop}$	500	12.5032	-71.3856	-	-
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.5 m_{prop}$	1000	16.4777	-71.3959	-71.5817	0.260
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.5 m_{prop}$	1500	12.0278	-71.3894	-	-
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.2 n_{lines} + 0.3 m_{prop}$	500	13.8326	-66.9603	-	-
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.2 n_{lines} + 0.3 m_{prop}$	1000	16.8814	-66.9731	-68.6076	2.441
$J = 0.25 \parallel \mathbf{r_f} - \mathbf{r_t} \parallel + 0.25 \parallel \mathbf{v_f} - \mathbf{v_t} \parallel + 0.2 n_{lines} + 0.3 m_{prop}$	1500	14.7131	-66.9655	-	-

robustness of the algorithm. Due to the constraints maintained in the algorithm, slightly higher propellant usage ($\sim 2\%$) is observed across all of the cases. A major penalty is taken for the the number of lines of code for the ALHAT algorithm since the algorithm's code contains approximately two orders of magnitude more lines of code than that of the theoretical algorithm. It is important to recognize, however, that the number of lines of code is directly related to the maturity of the algorithm. Therefore, if desired by the stakeholders, a factor accounting for the maturity of the algorithm could be included in the objective function, which would offset the tremendous penalty seen in this example.

VII. Altering the Analysis Framework for Control

Since control algorithms are typically designed to follow a *desired* acceleration profile, as opposed to command an acceleration profile, slight modification to the analysis framework is necessary. As the framework is setup, the commanded acceleration function can remain the same or be changed to give an acceleration profile solely as a function of time, (*i.e.*, $\mathbf{a}(t)$). Assuming this modification has taken place, an additional function should then be added to interpret the desired acceleration slew and apply the appropriate vehicle dynamics. This function has combined control and vehicle dynamics functionalities since the output from this block is an actual acceleration imparted on the vehicle from the control subsystem, $\mathbf{a_c}$, which is then interpreted as if it were a command from guidance. These modifications are outlined in Figure 8.

VIII. Conclusions and Future Work

A methodology that identifies a baseline robustness index for both guidance and control algorithms has been developed. The baseline robustness index allows equitable comparison of various guidance and control algorithms in a dispersed environment and is capable of including multiple classes of algorithm metrics (complexity, consistency, and performance) in the assessment process. The baseline robustness index employs concepts concepts from robust design, in particular the SNR and MSD, and allows for tuning of the parameters in the algorithm in order to maximize the performance of the algorithm (for an equal basis of comparison). The implementation of this methodology for both guidance and control applications

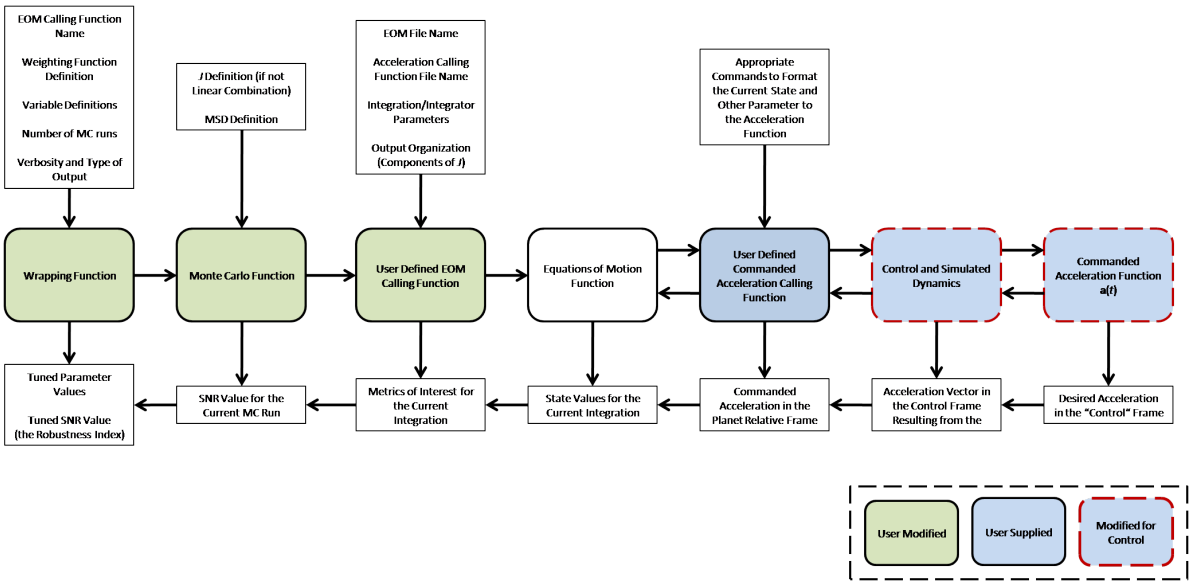

Figure 8. Analysis structure changes required for analyzing control.

has been discussed as well as two example applications provided. The first design example emphasized the use of the baseline robustness index in the selection of a propulsive descent guidance algorithm for soft-landing on the Moon's surface. This example showed the parametric nature of the robustness baseline index and the how a different algorithm may be selected should there be a different emphasis in the same general formulation of the objective function. The other example evaluated an immature algorithm (the general analytical optimal planetary landing law) and a more mature version of the same algorithm (the terminal phase of ALHAT's terminal descent guidance law) to three different objective functions, which were comprised of both performance and complexity metrics of interest. It was seen that the specialized performance of the mature ALHAT algorithm only slightly compromised the performance relative to each of the objective functions considered. Additionally, it was observed that the results were relatively insensitive to Monte Carlo sample size.

Acknowledgments

The authors would like to acknowledge the C.S. Draper Laboratory, Inc. where the bulk of this work was performed. In particular, Thomas Fill and Gregg Barton have both provided invaluable insight into the guidance robustness assessment problem.

References

[1]Saaty, T. L., "Decision Making with the Analytic Hierarchy Process," *International Journal of Services Sciences*, Vol. 1, No. 1, 2008, pp. 83–98.

[2]Taguchi, G., Chowdhury, S., and Wu, Y., *Taguchi's Quality Engineering Handbook*, Wiley, 2004.

[3]Cheng, R. K., "Surveyor Terminal Guidance," *NASA CR-57550*, 1966.

[4]D'Souza, C. N., "An Optimal Guidance Law for Planetary Landing," *AIAA Paper 1997–3709*, New Orleans, LA, Aug. 1997.

[5]Fill, T., "Lunar Landing and Ascent Trajectory Guidance Design for the Autonomous Landing and Hazard Avoidance Technology (ALHAT) Program," *AAS Paper 10-257*, San Diego, CA, Feb. 2010.