AutoNav Mark3:

Engineering the Next Generation of Autonomous Onboard Navigation and Guidance

J. Ed Riedel, Shyam Bhaskaran, Dan B. Eldred, Robert A. Gaskell, Christopher A. Grasso, Brian Kennedy, Daniel Kubitscheck, Nickolaos Mastrodemos, Stephen. P. Synnott, Andrew Vaughan, Robert A. Werner
Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Dr., Pasadena, CA. 91101, USA, joseph.e.riedel@jpl.nasa.gov

The success of JPL's AutoNav system at comet Tempel-1 on July 4, 2005, demonstrated the power of autonomous navigation technology for the Deep Impact Mission. This software is being planned for use as the onboard navigation, tracking and rendezvous system for a Mars Sample Return Mission technology demonstration, and several mission proposals are evaluating its use for rendezvous with, and landing on asteroids. Before this however, extensive re-engineering of AutoNav will take place. This paper describes the AutoNav systems-engineering effort in several areas: extending the capabilities, improving operability, utilizing new hardware elements, and demonstrating the new possibilities of AutoNav in simulations.

I. Introduction

Autonomous onboard deep-space navigation is an inevitable progression of robotic, and undoubtedly manned, space exploration technology. The success of JPL's AutoNav system during the Comet Temple impact and flyby on July 4 (Fig. 1, Ref. 1) 2005, following the equally successful encounters at the comets Borrelly (Fig. 2, Ref. 2, Ref. 3) and Wild-2 (Fig. 3, Ref. 4), also utilizing this system, demonstrates the power of this technology. Indeed, plans have been made to use the AutoNav system as the onboard navigation, tracking and rendezvous system for an at-Mars demonstration of autonomous rendezvous for purposes of proving technologies for Mars Sample Return Mission (MSR). Prior to a potential MSR use however, several mission proposals are evaluating the use of the system for purposes of rendezvous with, and landing and sample-taking on small bodies. This is an application for which the DS1/DI AutoNav system was not originally intended. Consequently, engineering evaluation of the existing system is ongoing, and has produced some encouraging results. This paper will describe this AutoNav systems engineering effort in several areas: extending the capabilities, improving operability, utilizing and interfacing to new hardware, and integration to attitude guidance and control elements.

II. AutoNav Software Systems Engineering

A. The Current State of the AutoNav System

Dynamic Modeling

The DS1/DI AutoNav system was conceived as a means of effecting deep space cruise between Earth and an asteroid flyby, and to effect the flyby of that asteroid (or comet). This was the system used during parts of DS1 cruise and for the DS1 and DI flybys. As such, the DS1 and DI AutoNav systems were equipped with limited capabilities in a number of subsystems. In the dynamic modeling and trajectory integration subsystem, notable limitations of capability are exemplified by point-mass gravity models, single-flat-plate solar pressure models, and limited flexibility in the modeling of engine operation (although

Figure 1: July 4, 2005 Deep Impact at Comet Tempel-1

the DS1 version of AutoNav was capable of, and did successfully model and operate DS1's low thrust propulsion system.)

Observable Modeling

The current AutoNav is also limited to optical observables, i.e. images of solar system bodies. Data partial derivatives of such observations are purely geometrical (Refs. 5, 6), and are easily and reliably computed onboard a spacecraft. Calibrations for such observables are readily made by taking images of star fields and calibrating the distortion in the camera. Such calibrations are generally stable over time, and needn't be often repeated. The data associated with a single picture comes in two general forms, the two dimensional position of a solar system object in a picture alone, or that position and the position of one or more stars or other inertial reference objects. In the latter case, the star images are used to determine the pointing of the camera, in large measure removing errors associated with uncertainties in the camera pointing, and greatly enhancing the power of the data. Otherwise, onboard resident knowledge about the spacecraft - and therefore camera - attitude are obtained from the spacecraft's Attitude Control System (ACS). AutoNav currently has no physical body modeling capability, in the sense of predicting the appearance (as in shape and color) of an object, images of which it needs to process. For the cruise phase of a mission, this is not problematic, as the typical bodies used are so distant as to appear as point sources, which allows the simple star-like processing, or smeared-star processing (Ref. 2). For larger objects, simple center-of-brightness calculations are performed by the current system. Deep Impact supplemented these calculations with somewhat elaborate corrections intended to maximize the likelihood of the DI Impactor spacecraft impacting the comet surface at a lit location, on the "flyby" side of the comet (Ref. 1).

Figure 2: September 22, 2001, DS1 Encounter with Comet Borrelly

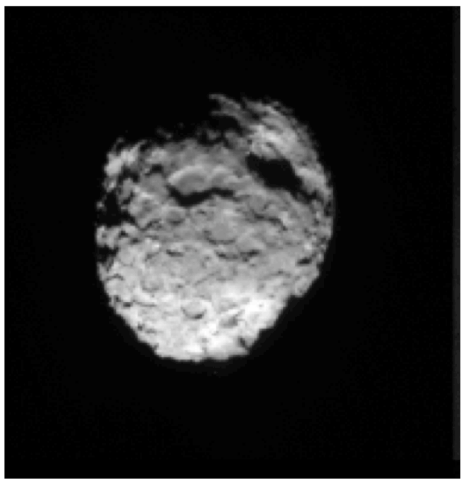

Figure 3: AutoNav tracking of Comet Wild, January 2, 2004

The Navigation Filter

The DS1 AutoNav navigation filter (referred to as the "estimator") is also fairly limited. Fig. 4, taken from Ref. 7, provides an overview of the DS1 AutoNav estimator. The parameters subject to estimation are the spacecraft state (position and velocity), bias acceleration parameters, and low-thrust proportionality term (a scale factor on the thrust magnitude of the main ion engine). During encounter, a simplified estimator solved only for relative (to target) state, and attitude bias (to account for gyro offsets near encounter when star images were not discernable as independent means of determining pointing) Ref. 3. During Deep Impact, the system was similar, with the added simplification that there was no low-thrust model invoked. The bias accelerations were not used for Deep Impact (since the data arc was so short) and attitude offsets were not estimated, but attitude drifts were. Additionally, the state that was estimated throughout encounter was not target relative, but inertial heliocentric. In the actual operations, the lack of an attitude bias estimate capability, such as the DS1 system had, was sorely missed due to the surprising actual performance of the mission's star trackers observed on approach.

Onboard Data Management

The DS1 and DI AutoNav systems used nearly identical file-based data management systems. For these missions there were 20 or so files involved in making AutoNav operate. These files came in two types. The first type was text-based "name-list-like" parameter files, for example camera parameters, and integration and estimation control parameters. The second type was binary files, for example the maneuver profile file - containing the schedule and values for thrust events, and ephemeris files. The ephemeris files included the trajectory of the spacecraft, and the positions of asteroids and planets. Most of the approximately 20 files had to be loaded individually from the ground, and about one

Figure 4 a,b,c: Overview of the DS1/DI AutoNav Orbit Determination Kalman Filter (Ref. 7)

Dynamical equations of motion

- **Includes central body acceleration, 3rd body perturbations from other planets, solar radiation pressure, thrust from the ion engines, and miscellaneous accelerations**
- **2nd order differential equation modeled as two 1st order differential equations**

$$\dot{\mathbf{r}} = \mathbf{v}$$

$$\dot{\mathbf{v}} = -\frac{\mu_s}{r^3}\mathbf{r} + \sum_{i=1}^{n_p}\mu_i\left[\frac{\mathbf{r}_{ri}}{r_{ri}^3} - \frac{\mathbf{r}_{pi}}{r_{pi}^3}\right] + \frac{AG}{mr^3}\mathbf{r} + \frac{k}{m}\mathbf{T} + \mathbf{a}$$

where

$\mathbf{r}$ = the heliocentric cartesian position vector of the spacecraft

$\mathbf{v}$ = the heliocentric cartesian velocity vector of the spacecraft

$\mathbf{r}_{pi}$ = the heliocentric cartesian position vector of the ith perturbing planetary body

$\mathbf{r}_{ri}$ = the position of the spacecraft relative to the ith perturbing body

μ_s = the gravitational constant of the sun

μ_i = the gravitational constant of the i_{th} perturbing planet

n_p = the number of perturbing planets

A = the cross -sectional area of the spacecraft

G = the solar flux constant

T = the thrust vector from the ion engine

k = the thrust scale factor

m = the spacecraft mass

$\mathbf{a}$ = miscellaneous accelerations acting on the spacecraft

Given $\mathbf{q}^*$, the nominal trajectory parameters, as

$$\mathbf{q}^* = \begin{bmatrix}\mathbf{r} & \mathbf{v} & k & \mathbf{a}\end{bmatrix}$$

Filter estimates corrections, $\mathbf{q}$, to nominal trajectory parameters

$$\mathbf{q}(t) = \begin{bmatrix}\Delta x & \Delta y & \Delta z & \Delta\dot{x} & \Delta\dot{y} & \Delta\dot{z} & \Delta k & \Delta a_x & \Delta a_y & \Delta a_z\end{bmatrix}$$

The correction at time t is a linear mapping of the correction from time t_0

$$\mathbf{q}(t) = \Phi\mathbf{q}(t_0)$$

where Φ, the state transition matrix, is defined as

$$\Phi(\mathbf{t}) = \frac{\partial\mathbf{q}^*(t)}{\partial\mathbf{q}^*(t_0)}$$

The partial derivatives of the observed pixel and line locations, p, l, with respect to the state, at time t is

$$\mathbf{H}(t) = \begin{bmatrix}\partial p/\partial\mathbf{r} & 0_{1x7}\\ \partial l/\partial\mathbf{r} & 0_{1x7}\end{bmatrix}$$

This can be mapped back to the epoch, t_0, via the state transition matrix

$$\tilde{\mathbf{H}}(t_0) = \mathbf{H}(\mathbf{t})\Phi$$

The minimum variance least squares solution to the epoch state corrections is

$$\hat{\mathbf{q}} = \left[\mathbf{P}_0 + \tilde{\mathbf{H}}^T\mathbf{W}\tilde{\mathbf{H}}\right]^{-1}\tilde{\mathbf{H}}^T\mathbf{W}\mathbf{Y}$$

where

$\mathbf{P}_0$ = the a - priori covariance of the state parameters

$\mathbf{W}$ = the weighting values of the pixel and line observables

$\mathbf{Y}$ = the residual vector between the observed pixel/line locations and their predicted values

third of these were updated automatically onboard by AutoNav. Organizing and configuring these files on the two missions involved very different approaches due to the very different way each mission did onboard file management. Table 1 gives an overview of the file interfaces used for DS1, and the nature of their volatility onboard; this table is taken directly from Ref. 7, and applied to the deep space cruise phase of the DS1 mission.

Table 1: DS1 AutoNav Files, Sizes, Autonomy Status, Locations and Update Frequency (From Ref. 7)

File Description	File Size (KB)	File Update Frequency		Onboard Location
		From Ground	Auto-Onboard	
Star Catalog	>1000	1/mission	Never	EEPROM
Planetary Ephemeris	<100	1/mission	Never	EEPROM
TCM Params	<10	4/year	Never	EEPROM
Encounter Params	<1	2/encounter	Never	EEPROM
Encounter Star Catalog	<1	2/encounter	Never	EEPROM
FrankenKenny Params	<1	2/encounter	Never	EEPROM
CCD Camera Params	<1	2/year	Never	EEPROM
APS Camera Params	<10	1/encounter	Never	EEPROM
Beacon Ephemeris File	<10	2/year	Never	EEPROM
Mass Profile	<100	4/year	Never	EEPROM
Picture plan	<100	4/year	Never	EEPROM
Control Params	<100	4/year	Never	EEPROM
Photo-Op Params	<10	2/year	Never	EEPROM
IPSburn Params	<1	2/year	Never	EEPROM
Nongrav Params	<1	2/year	Never	EEPROM
Imageproc Params	<1	2/year	Never	EEPROM
File of Filenames	<10	4/year	1/month	EEPROM
Maneuver	<100	4/year	Weekly	EEPROM
OD	<100	2/year	Weekly	EEPROM
Spacecraft Ephemeris	<100	1/year	Weekly	EEPROM
OpNav	>1000	Never	Weekly	RAM
Nongrav History	<100	Never	Several/day	EEPROM

In neither mission was it an easy task to configure the AutoNav system, but somewhat different philosophies toward managing files onboard Deep Impact made the task much more labor intensive for that mission than for DS1. Since the current monolithic AutoNav files control so many parameters, maintaining consistency between the two spacecraft that comprised the Deep Impact Mission (Flyby and Impactor), multiple ground test installations, and between various sequential activities, was problematic and costly in manpower. In addition to the mechanical complexity of establishing and maintaining a suite of partly volatile data files onboard a spacecraft, there are the critical issues of validation and verification of these data before uplink at numerous times throughout the mission, especially prior to critical operations. These updates were made considerably more challenging when minimal parameter changes necessitated an uplink of a complete set of related - but not changing - parameters in a monolithic data file.

Part of the control mechanism for DS1 and DI AutoNav was not file-based, but memory and command-based, and therefore much more operationally tractable and malleable. AutoNav has a suite of mode states. These modes control the basic behavior of the AutoNav system, such as enabling certain capabilities, or altering the values of limited data locations that might require quick and ready update. A specific AutoNav command would update these modes, or alternatively downlink the entire state of the mode suite. This handy and flexible means of control became the model for the re-engineered AutoNav data management system currently under development.

B. Future Operational Challenges and System Requirements

Dynamic Modeling

Future missions that may utilize AutoNav will face mission scenarios with navigational challenges well beyond those very significant challenges met by DS1 and DI. These future missions may include orbit insertion and orbital operations around small bodies and planets, especially Mars. They also include unprecedented navigation operations such as autonomous on-orbit rendezvous between vehicles, and/or sample canisters; landings with pinpoint accuracy on airless bodies; and guidance through atmospheres for entry or even skip-out. These missions, notably in the case of Mars or Lunar infrastructure missions, may entail the autonomous navigation of one spacecraft by another. Capabilities and needs for all of these cases, except those dealing with atmospheric interaction with a vehicle lifting body, will be met in whole or in large part by the currently active AutoNav developments and upgrades to be outlined below.

In the area of dynamic modeling and trajectory integration necessary for the next generation AutoNav, one of the most needed upgrades is high-order gravitational modeling for planetary bodies, both large and small. Many prototypes exist within navigational ground software for modeling high-order gravitational fields, and computation of associated dynamic partial derivatives (perturbations of a vehicle state as a result of the gravity field). The developments currently ongoing involve incorporating variations and upgrades of this code into robust flight software elements of AutoNav. Other dynamic model upgrades are also necessary, for example, generalizations of the solar pressure model may be required for some missions such as those utilizing solar electric propulsion ion-drive spacecraft with very large solar panels. A key area of development is the model of spacecraft propulsive forces, including stochastic accelerations due to modeling errors in those propulsive forces. Though optical data is inherently insensitive to small accelerations (unlike Doppler and range measurements), potential addition of these random forces may be necessitated by the addition of local radiometric observables, as discussed below. Additionally, stochastic forces may prove useful for orbital operations with low thrust propulsion.

One unique provision being built into the prototype next generation AutoNav is the ability to integrate the trajectory, and indeed estimate, the state of, another spacecraft. This feature will have applicability to rendezvous, for example, the Mars Sample Return Mission, which will have to recover a soil sample lofted into Mars orbit from a lander. Technology demonstrations of the AutoNav capabilities necessary for MSR was to occur on the now cancelled Mars Telesat Orbiter Mission, (Ref. 8), and the CNES PREMIER Mission (Ref. 9). A more general vehicle-to-vehicle rendezvous capability is also envisioned with the other-spacecraft-state-estimation capability. For both of these capabilities, however, it is not absolutely necessary to estimate the position of the other spacecraft (or, orbital sample), as relative state estimation generally is adequate, wherein the absolute errors in position of the two vehicle system are ignored. Though this is very much true in a system that only includes observations within and between the two-vehicle system, once measurements are taken relative to an inertial frame (e.g. Earth, another body, or a well known spacecraft position, such as a lander) the need to correct the absolute position of the two-vehicle frame is presented. Additionally, one advanced, and as yet rather speculative need for onboard other-vehicle-navigation comes from the idea of having an infrastructure capability at another body (e.g. the Moon or Mars) that will estimate the state of approaching or in-orbit spacecraft, as a navigation service for those vehicles. This would have special applicability for small and inexpensive probes that need precise atmospheric entry and landing at Mars, or pinpoint landing on the Moon.

New Data Types

AutoNav requires new or expanded capability of both optical and non-optical measurements. The most important new observables being added to AutoNav are surface landmarks. Using a system developed over the last 15 years (Ref. 10), a prototype landmark tracking capability has been integrated to the existing DS1/DI AutoNav system, as a feasibility demonstration (to be discussed below.) This demonstration proved the feasibility of providing AutoNav with the ability to utilize proximity observations of known bodies as data. High accuracy proximity observations of the surfaces of bodies enables a host of capabilities, including very high accuracy orbit insertion or atmospheric entry, orbital operations, and high precision landing.

Accurately modeling radiometric observables across long stretches of space and/or through atmospheres to planetary surfaces requires complex and computationally intensive operations (Ref. 11), and therefore it is unlikely that onboard navigation will soon receive radio signals from Earth for purposes of interplanetary navigation. This is so because one of the primary reasons to use autonomous onboard navigation is to reduce Deep Space Network (DSN) antenna-time requirements, which such a strategy clearly does not do. However, over short ranges - for example between orbiting spacecraft or between a landed beacon and an orbiter - it is quite possible that relatively low-precision calculations of Doppler and range observables will find a purpose and a place in a generalized AutoNav system. One particular application that seems most likely is an MSR rendezvous where the lofted sample canister is transmitting a beacon - perhaps even a coherent Doppler beacon. Though the nominal plan for MSR is to perform the rendezvous with the canister using passive optical, in an extreme contingency - with the loss of all imaging - it may well be possible to autonomously rendezvous using a 2-way radio beacon, especially when proximity ranging is added. AutoNav may well need to be able to deal with these types of radio observables.

LIDAR and RADAR observables are primarily range observations, however some scanning LIDARs also give cross-axis information. These non-passive measurement methods are often used in close proximity to a target for terminal landing and rendezvous navigation, and the structure of AutoNav needs to provide for their use. The AutoNav system currently utilizes inputs from an IMU directly in its propagation of the trajectory from the past estimated starting state to the last available datum. Such a strategy assumes that the IMU inputs are perfect, or otherwise just absorbs whatever inaccuracies might be present. An alternative strategy would be to take IMU inputs as data, appropriately weighted, and with error sources modeled and estimated. Such a future capability is being enabled by the ongoing redesign of the AutoNav system.

Estimation Strategies

AutoNav currently uses a batch-sequential estimation scheme, as opposed to a strict Kalman current-state filter strategy. The principal difference is that the latter simultaneously maps the information matrix to the present in order to fold in the information matrix of the current datum and estimate the current state, whereas the batch-sequential filter divides the data arc into batches, and produces an information matrix across this batch relative to an estimated fixed-epoch state (typically at the beginning of the batch), Ref. 12. The batch-sequential filter is very useful for data analysis, particularly for identifying trends across residuals to determine stray outliers, or patches of data suffering systematic anomalies. After sufficient data is accumulated in the batch, or sufficient batches are obtained, the epoch is advanced in a process very analogous to the current state Kalman filter, the data arc shortened, and the epoch state time is advanced and the covariance mapped accordingly. This strategy is outlined in Figure 4.

However, within the batch-sequential filter strategy are a variety of architectural choices. One of the principal choices is whether to compute the observational partial derivatives inside or outside the filter itself. In other words, even within a batch-sequential filter, a Kalman-like strategy could be adopted wherein the partial derivatives for each datum could be computed immediately before combining into the information matrix. This is in fact how the existing AutoNav filter operates. The disadvantage of this approach is that if multiple data editing passes are made, redundant data-partial calculations are required. If, on the other hand, data partials are computed across the data arc *en masse,* and the information combined in a single operation, the computation can be much more efficient, and especially so if multiple data editing passes are required. An additional advantage of such a strategy is a greater modularity in the software structure with modules associated with different data types being called outside of the core filter algorithm. One disadvantage of this latter approach is the necessity to store the data and the partial derivatives. Despite this latter disadvantage, the new AutoNav system is adopting this second batch-sequential strategy. It is a strategy that is consistent with existing heritage optical navigation ground infrastructure, the advantages of which will be discussed below.

Perhaps the most important upgrades to the AutoNav filter is generalizing and expanding the ability to choose among a reasonably rich set of estimable parameters. As the new AutoNav filter is developed, provisions are being made to estimate or consider a wide range of dynamic parameters, including gravity field coefficients, target body ephemeris terms, including those of another spacecraft, and parameters associated with execution of maneuvers. Though the IMU record of propulsive events is often good enough for operations, there are certain cases, especially involving small maneuvers - such as for rendezvous - where estimation of the IMU errors for those burns may be desirable or necessary especially if local Doppler and range is reduced onboard as an AutoNav observable.

C. The Next Generation AutoNav Data and Command Management System

The AutoNav Blackboard Architecture

The issues discussed above with regards to the operational challenges posed by a file-based data management system and to a great deal of increased flexibility in the estimator, are being addressed with a completely overhauled data management and command system in the next generation AutoNav. The core of this revamped system is notionally referred to as the "AutoNav Sea Of Memory," or ASOM. This concept is commonly referred to as a "blackboard architecture:" "A blackboard architecture contains a

hierarchically organized global memory or database called a blackboard which saves the solutions generated by the knowledge sources; a collection of knowledge sources that generate independent solutions on the blackboard using expert systems, neural networks, and numerical analysis; and a separate control module or scheduler which reviews the knowledge sources and selects the most appropriate one.

The advantages of a blackboard include separation of knowledge into independent modules with each module being free to use the appropriate technology to arrive at the best solution with the most efficiency. An additional advantage of the independent modules is the potential for using separate computing units for the independent knowledge sources, thus allowing distributed computing. This approach allows for rapid prototyping of complex problems and simplifies long-term system maintenance," Ref. 13 .

Implementation of the AutoNav Blackboard

The advantages of the ASOM blackboard architecture begin during the development phase, as alluded to in the reference just above. An AutoNav Master Control Document (MCD) has been created which contains a set of definitions of all parameters that need to be uplinked to operate AutoNav and that are passed between AutoNav Computation Elements (ACE's). The ACEs are the core compute engines of AutoNav, and include functions such as image processing, orbit determination, trajectory integration, and maneuver calculation. In the DS1/DI implementation of AutoNav, the ACE's had to perform their own data collection and archiving, via reads and writes to the data files discussed earlier; however, in the next generation implementation the ACEs have access to the ASOM blackboard, and push and pull information

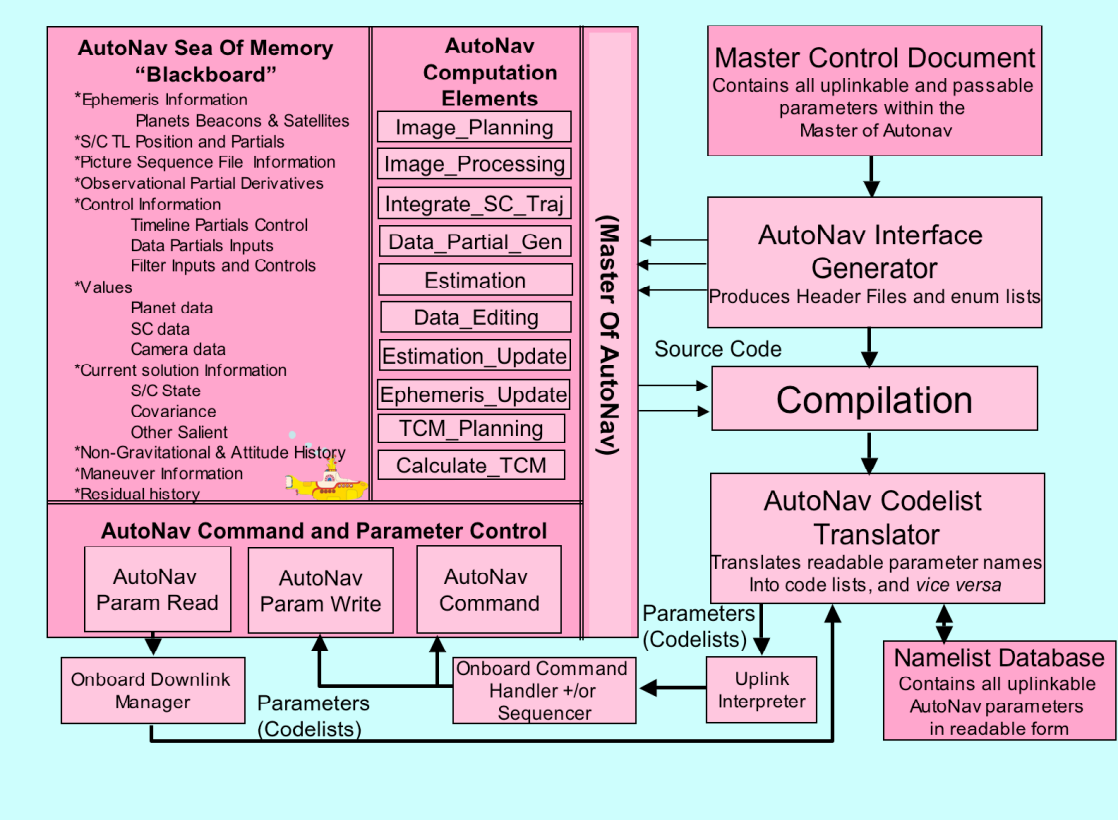

Figure 5: AutoNav Sea of Memory Blackboard Software Architecture

as required to obtain controlling parameters or to post results to be used by the next ACE to be called. Additionally, the AutoNav Executive, to be discussed later, can push and pull, and manipulate ACE parameters as needed, by the same method, even though the Executive is not in the same thread as the ACEs. The ACEs are callable routines in the Master of AutoNav (MOA) thread that also contains the input

and output data commands. Figure 5 displays the interconnection of these architectural elements, and those discussed below.

Before compilation of the ACEs and the MOA, the AutoNav Interface Generator is run using the MCD as input, which generates an extensive array of header files and enumeration lists - the system is coded in C - which allow all code elements to automatically be aware of all of the parameters in the ASOM. On top of an extensive list of user-friendly text names for the AutoNav parameters, there are associated code numbers established in the MCD to enable uplink of those numbers instead of inefficient long name lists. At run time, an AutoNav Codelist Translator (ACT) is available to convert the human-friendly form of AutoNav inputs into uplink-efficient command form. Also, the ACT will translate downlink telemetry from the ASOM into readily readable command input form. Figure 5 shows these connections. Access to the ASOM parameters is through a limited set of commands, the parameter input and output commands mentioned above, and other more specialized functions, for example, for arrays and the 2-D array inputs and parameter dumps of covariance matrices.

As a mission operates, the MOA will offer a convenient means of maintaining a database of current onboard AutoNav states. Though the MOA operates onboard, a parallel flight-like AutoNav operation will exist on the ground for test and validation purposes. All uplinked parameter commands are processed through this configuration ground version of MOA. Ground MOA will then be capable of performing necessary AutoNav tests, or simulations, and dumping complete sets of memory values for safe archive. These telemetry dumps of Ground MOA memory states can be converted via ACT into input commands that could be turned around and applied back to the ground or flight MOA as needed.

D. Prototyping the AutoNav Filter with Ground Software Architecture and Elements

Generalized navigation filters, such as described above as planned for the next generation AutoNav, are large and complex software developments, often entailing several man-years of work to develop and test. One of the more difficult aspects of these developments is the creation of versatile, easy to use, and reliable means of specifying and tracking the multiple input options for estimate and consider lists, and associated varied means of specifying *a priori* uncertainties. Additionally, navigation estimation filters are almost always specifically designed around the predominant data types to be processed, especially with regard to data noise modeling, and a redesigned AutoNav filter would have to be tuned primarily with optical data in mind. Fortunately, a set of programs exists that has solved these problems and has a heritage going back to the Voyager project of the late 1970's.

This program set was used to perform navigation functions using optical data throughout the Voyager mission, and has been in continuous use since 1977, (Ref. 14.) The program set is composed of a suite of tools, two principal elements of which are the an observable partials generator, and an estimator. The first of these is the means of generating the optical data partials, an example of which would be $\partial Phobos\_pixel / \partial Spacecraft\_x_{now}$, or the sensitivity of the x-position of Phobos as viewed in an image with respect to the inertial position of the spacecraft "now," i.e. at the time of the picture. In the batch sequential filter strategy, another set of partial derivatives is necessary, namely those commonly associated with the state transition matrix (STM), for example, $\partial Spacecraft\_x_{now} / \partial Spacecraft\_x_{t\_initial}$. Of course, this is normally a 6x6 matrix. The STM is a trivial example of a dynamic partial derivative, which is the derivative of the current state of the spacecraft with respect to any parameter that affects that state. Examples beyond the STM include the mass of the central body (e.g. the sun and the body the spacecraft is orbiting), maneuver parameters, or solar pressure.

The optical navigation software does not compute the STM or other dynamic parameters. Instead, it relies on other elements of the navigation software set for that information. The means of computing the STM and other dynamic partials is well documented in many texts, but the method of economically storing a continuous description of such partials is an art, and consumes much effort of such partials-generating software, since in general, the dynamic partials must be obtained at thousands of points, with often high frequency, such as for processing radiometric data. This methodology generally produces very large data

files, as full precision computation of all the partials is provided at all times. Optical data however tends to be quite sparse, with at most a few hundred, as opposed to thousands of observations, which is common for radiometric data. This sparseness allows for the cataloging of the dynamic partials in a straight-forward tabular fashion for each and every optical observable. This "time-line" based partials generator is notionally referred to then as the TimeLine Partials Generator.

This team of existing, heritage estimation elements, and one new element forms the emerging prototype of the next generation AutoNav estimator. The two heritage elements are written in Fortran. Nevertheless, they are being configured as callable elements from the MOA, through temporary interfaces that read from the ASOM blackboard and write transient Fortran data files necessary for the ONP's operation. Then, these are read, and the results written back into the ASOM. As the prototype is tested, and upgraded, the internal elements of ASOM will be re-written into C language, and incorporated as ACE's in the MOA.

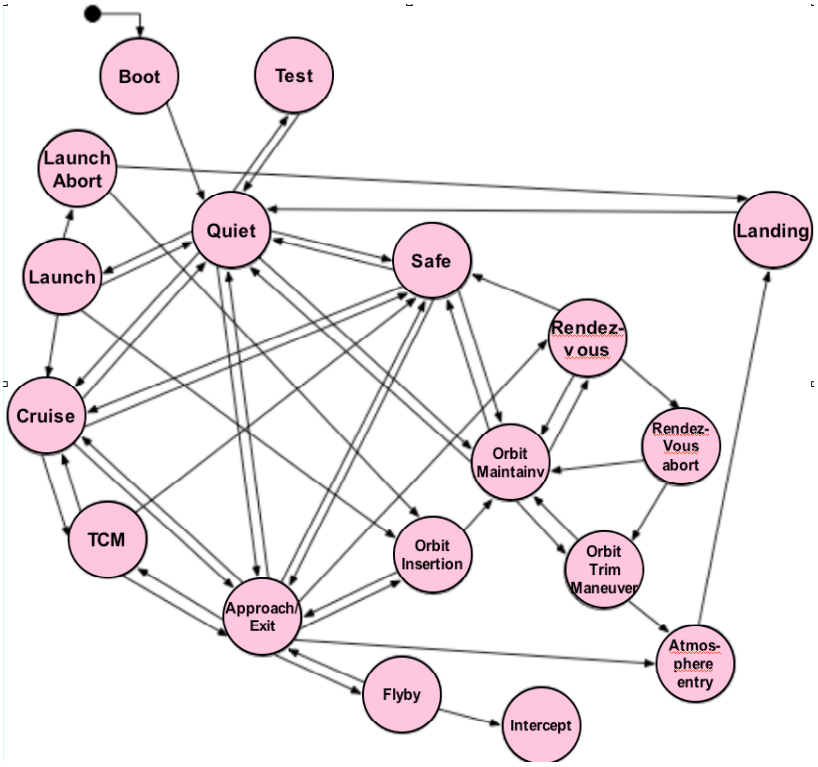

Figure 6: Notional AutoNav VML Executive Mode Transitions, Showing AutoNav Mode Functional Progression Through Generic Mission Phases

E. The VML AutoNav Executive

AutoNav Executive Concepts

One final important element of the next generation AutoNav design is the "Executive." For both DS1 and DI, the Exec was a C-language routine that planned and directed AutoNav activity, and interfaced with the rest of the spacecraft flight software. The DS1 Exec was reasonably skilled; it planned extensive activities, including operation of DS1's low thrust engine, Trajectory Correction Maneuvers (TCM's), and picture-taking sequences which were a series of attitude changes and camera activations over a period of hours. Deep Impact's Exec was much more limited in scope, as that AutoNav system only operated over a period of two hours for the entire mission. But missions of the future will demand much more planning capability and intelligence than even DS1 exhibited, and DS1's Exec probably exhausted the reasonable

scope of a custom-tooled state-machine-based executive. Consequently, the next generation AutoNav is building its executive in a general sequencing language that has been adopted as the operating standard for JPL's, and some commercial, space missions. This language is VML for "Virtual Machine Language," (Refs. 15,16.)

Introduction to VML

VML is an advanced procedural sequencing language that simplifies spacecraft operations, minimizes uplink product size, and allows autonomous operations aboard a mission without the development of autonomous flight software. VML is used, or will be used, on a variety of missions, including Mars Reconnaissance Orbiter, Mars Odyssey, the Spitzer Space Telescope, Dawn, and Phoenix. The language is a mission-independent, high level, human-readable script interpreted by flight software. VML sequences are implemented as a set of named functions, allowing operations to be abstracted. Files containing named functions are loaded onto specific sequencing "engines," which serve both to contain the script and to provide a thread of execution. VML features a rich set of data types (including integers, doubles, and strings), named functions, parameters to functions, IF, FOR, and WHILE control structures, polymorphism, and on-the-fly creation of spacecraft commands from calculated values. All statements are time-tagged with either absolute or relative times. Programmable time tags to delay by a calculated amount or delay until a calculated time are provided. The language also features event-driven sequencing through the use of sequence global variables, which contain spacecraft state information visible to all functions and accessible to flight software through accessor routines. Global variables computed and stored by expert systems accessed by VML such as AutoNav may be stored and reasserted. These programming, and conditional sequencing constructs form an invaluable resource from which the AutoNav Exec can draw to perform the complex and varying tasks required of autonomous onboard navigation.

The VML AutoNav Executive Architecture

The VML AutoNav Executive is hierarchically structured within VML sequence engines, and consists of a number of blocks that variously run continuously or discreetly as "one-off" function calls. The AutoNav Computational Elements (ACE's), which do the actual numerical analysis of navigation, are command-able calls within MOA (see above). Data transfers between MOA and VML are via the MOA PSET and specialized PDUMP_EXEC commands, which make AutoNav variables available to VML, and allow VML to alter and set parameters, as discussed earlier. At the highest level of the Executive is the AutoNav Master Scheduler (AMS). The AMS establishes the fundamental modes in which AutoNav operates. Modes are an abstraction that intrinsically limit the kinds of activities that may occur to a

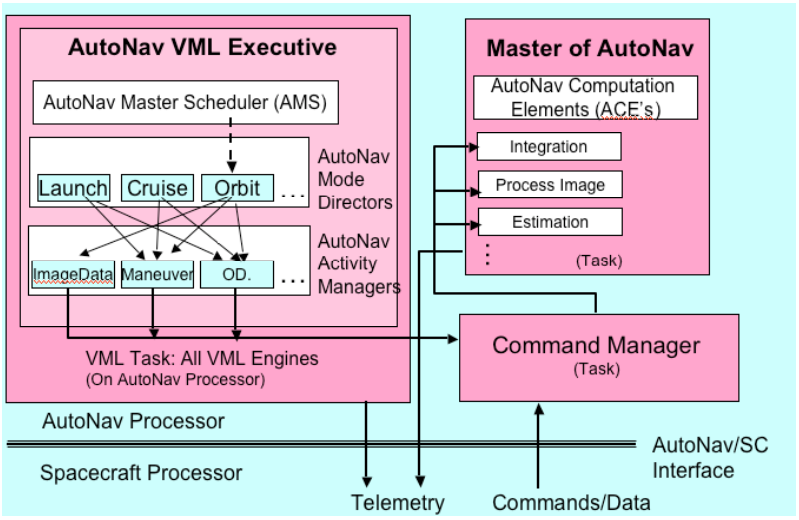

Figure 7: VML Executive Interaction with Master of AutoNav Task

specifically focused navigation activity. The modes are designed into the AutoNav executive in order to capture typical spacecraft activities for a wide variety of different kinds of missions. Valid mode transitions are shown in Figure 6. All mode transitions are fundamentally allowed in order to simplify emergency operations, but commanding an invalid transition will generate warning telemetry and may cause an invalid internal state within the ACE. Hard locks to prevent invalid mode transitions can be enabled or disabled as desired during operations. Examples of the modes are Test, Safe, Launch, Launch_Abort, Cruise, TCM, Approach/Exit_Target, Orbit_Insertion, OTM, Rendezvous, Rendezvous_Abort, Entry and Landing.

For each AutoNav Mode, there is a Director. For example, there is a Launch Director, and a Cruise Director, and one of the most important tasks for the AMS is to initiate the Director for a particular mode. The Directors are blocks that remain active throughout the Mode, until terminated by the AMS. The Directors will invoke subsidiary Managers of AutoNav activity, such as the Orbit Determination Manager, and Image Data Manager, and the Maneuver Manager. These are also VML blocks that will generally run for finite durations, but are invoked only periodically as needed by the Directors. The AutoNav Managers are invoked by many Directors, for example the Cruise Director and the Orbit Director will both invoke the Orbit Determination and Image Data Managers. Finally, the Managers will invoke ACEs within MOA to perform the computational navigation work necessary. ACE calls can be invoked from any of the Managers, for example the Integration ACE is called by the Maneuver Computation Manager as well as the Orbit Determination Manager. The invocation of an ACE is through the command manager, which is a separate thread within the operating system. Figure 7 shows this hierarchy and architecture.

	MRO ONC	Cassini Imaging
Pixel Resolution	<30 microradians	<10 microradians
Field of View	>1.5 degrees	>0.5 degrees
Mass	<3 kg	<20kg
Power	<3 Watts	<20Watts
Volume	<8000cm^3	<80000cm^3
Noise Level	>10e$^-$	>30e$^-$
Full Well	>150,000e$^-$	<100,000e$^-$
Dynamic Range	>1000:1	<500:1
Navigation Acc'y	<2 microradians	<2 microradians

Table 3: Comparison of Navigation Attributes of the MRO ONC and the Cassini Imaging System

III. Hardware Elements of AutoNav Capability

A. General Considerations of Hardware Systems for Onboard Optical Navigation

On the three missions that have flown AutoNav, the onboard science imaging instruments were used for taking the navigation images. Though this was entirely satisfactory, judging by the success of these missions, performance of AutoNav was limited in many ways. Also, the nature of these three missions -flyby missions - lent itself to the dual use of the science instrument, which was configured for long-range remote sensing of bright objects. Future missions will, however, likely not have such limited operational navigational regimes as did these missions. Imaging instruments that more balance the needs for distant and near field observations will be necessary, perhaps even requiring multiple navigation instruments. Fixed, body-mounted cameras present substantial challenges to a mission and AutoNav operations. The navigation function naturally "desires" to obtain the maximum amount of data possible, while the mission "desires" the least perturbation possible to the spacecraft in the

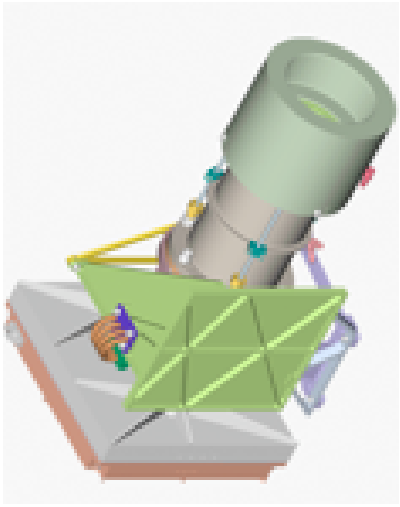

Figure 8: Preliminary Design Concept for the MRO Optical Navigaiton Camera (ONC)

way of turns. This is, among other things, an issue of a spacecraft-time-available resource. Spacecraft resources are also subject to competition with regard to the availability of computer cycles. AutoNav functions can be very computationally intensive. For DS1, at the height of operations, AutoNav was consuming close to half of the available RAD6K CPU cycles. On Deep Impact, maximum use of the RAD750 by AutoNav was comparable. To address these resource issues it is necessary to consider different hardware options that could comprise a suite of integrated hardware elements constituting an AutoNav "instrument," including a dedicated CPU. As this analysis of potential AutoNav-applicable hardware has been performed, it is clear that there is a crossover of function and capability between the types of imaging systems and software that are normally used for attitude control purposes, e.g. star trackers, and what is needed for many navigation scenarios calling for AutoNav. Furthermore, the tendency to dedicate specific computational hardware to attitude control systems presents an opportunity to integrate not only AutoNav software and hardware, but to combine these functions with attitude control systems as well, thereby potentially saving missions substantial amounts of development and integration costs.

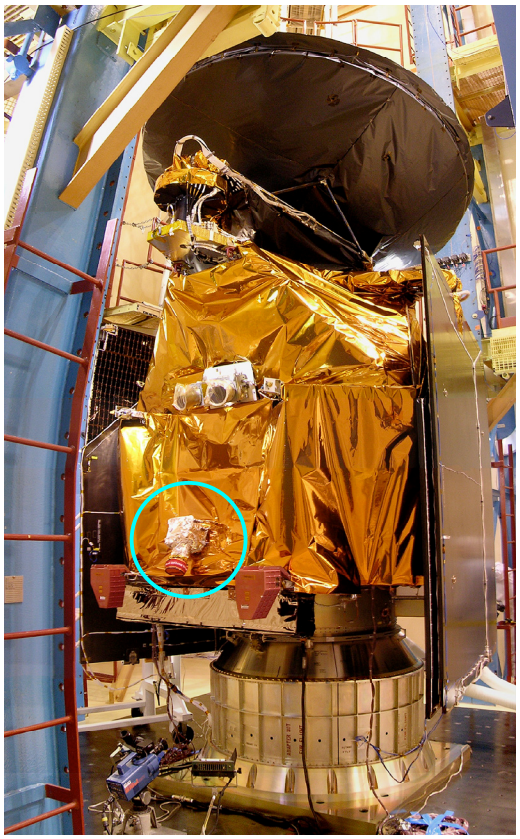

Figure 9: Optical Navigation Camera (circled) Ready to Fly on MRO

B. The MRO ONC

Of the hardware resources to draw upon for developing an integrated AutoNav instrument, the recently flown and flight-tested MRO Optical Navigation Camera (ONC) is the principal element. Designed over a period of six years with the DS1 AutoNav experience fresh in mind, and the then developing Mars Sample Return mission being vigorously investigated, this instrument was specifically designed to image the dim objects (e.g. stars, distant asteroids, and orbiting sample canisters) in the proximity of bright near-field planetary bodies. As such, it has a large dynamic range and low noise. Table 2 compares the attributes of this instrument with the Cassini imaging system from a navigation standpoint. The MRO ONC is a small, compact, high-power instrument and offers such a wide range of navigation capability, it will likely form the core of the navigation system for a broad spectrum of upcoming missions, and therefore is the ideal candidate to form the center of an integrated AutoNav instrument package. Figure 8 shows an early

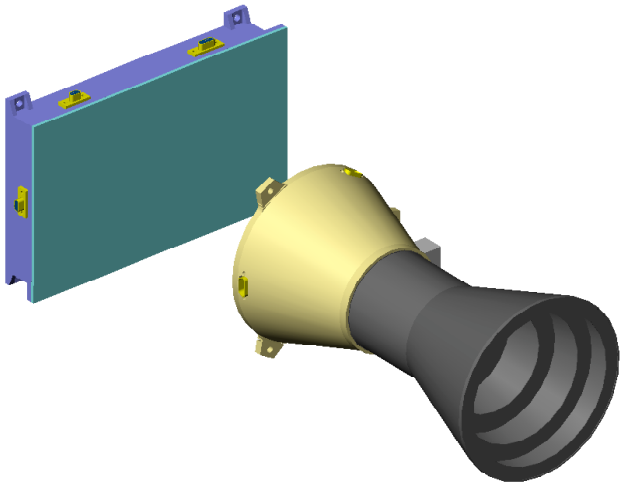

Figure 10: The Charles Stark Draper Laboratory Inertial Stellar Compass (from Reference 20)

Figure 11: The MOOG/Schaefer Type 11 Biaxial Gimbal (from Reference 18)

configuration notion of this camera, and Figure 9 shows the instrument mounted on the MRO spacecraft, and well demonstrates the diminutive size of this powerful camera.

C. The CSCL ISC

The Charles Stark Draper Laboratory has developed an instrument, called the "Inertial Stellar Compass", or ISC, (Fig. 10). The ISC "is a real-time, miniature low-power stellar inertial attitude determination system composed of a wide field-of-view active pixel star camera and a microelectromechanical system (MEMS) gyro assembly" Ref. 17. The ISC solves the general attitude estimation, and "lost-in-space," or tumbling spacecraft attitude problem in a compact integrated hardware and software package, of low mass and power (~2.5kg and ~3.5W respectively - IBID). One of the key innovations of the ISC is to combine the ability of a camera to not only determine a fixed attitude, but rates as well, with the MEMS gyro to enable maintenance of attitude knowledge during fast turns when star tracking would not be possible. Another key innovation was the pre-integration of the hardware and software elements, including hardware drivers, into a single unit with simple high-level spacecraft interfaces. The ISC utilizes an ERC-32 CPU, which is a SPARC-based architecture, and hosts on this processor the RTEMS Real-Time Operating System (RTOS). The ISC represents an excellent model for an integrated software and package which potentially greatly eases a mission's development and integrations costs. Additionally, equipped as it is with the CPU and wide angle camera, the ISC represents a very logical choice for a core element of the integrated AutoNav instrument suite, and will serve in that role as a point design for further discussion and analysis. The ISC is soon to be flight tested as part of the New Millennium space technology program ST6 mission.

D. Integrating and Gimbaling the Instrument Suite

As mentioned above, one of the mission resources that is in conflict when AutoNav is operating is time, in particular, pointing time. Turning the vehicle to orient instruments in order to obtain navigation data, whether radio or optical, almost always presents a conflict with other mission objectives when those instruments are body-fixed. The means of greatly reducing this conflict is to provide a self-pointable camera by mounting the instrument cluster on a gimbal. There is a potentially wide range of capabilities to choose from for a camera gimbal, but the MOOG/Schaefer Model 11 2-dimensional gimbal offers a good strawman design for the AutoNav instrument (Ref. 18). Figure 11 shows the instrument in its standard configuration. Finally, Figure 12 shows a notional assembly of the ONC, ISC, and MOOG gimbal. This concept could provide a mission with a 7kg 7Watt package of 0.04 cubic meters that provides full navigation and attitude estimation capability for

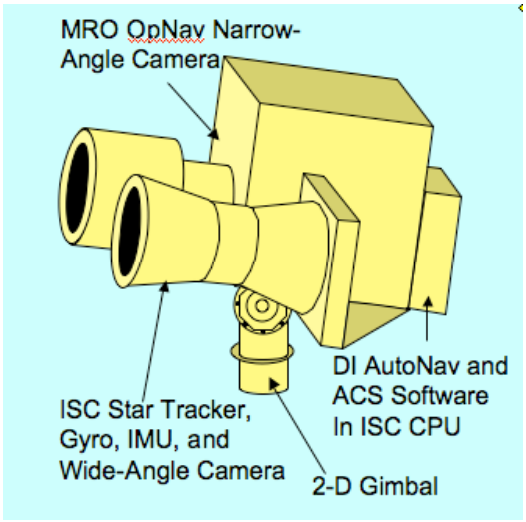

Figure 12: A Concept for the AutoNav Gimbaled Camera Assembly Instrument Suite

the cost of a typical small science instrument. One of the principal costs that a space exploration mission undertakes is the integration of hardware and software elements to each other, and those elements to the spacecraft overall. Guidance Navigation and Control (GN&C) software and hardware elements are often the most difficult, costly and risk-prone to integrate. A pre-integrated suite of imaging instruments, gimbal and AutoNav and attitude estimation software could potentially save substantial funds and retire much mission risk for a wide range of future missions. Costs could be further reduced by allowing a series of missions to share the responsibility for this development.

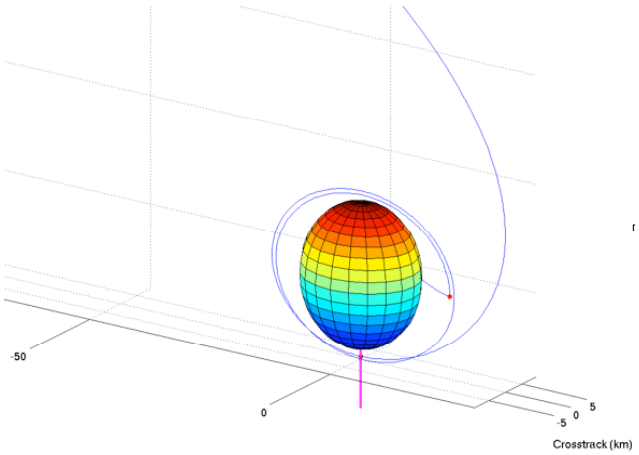

Figure 13: Phobos Rendezvous and Landing Trajectory for AutoNav Simulation

IV. Demonstrations of Future AutoNav Capability

A. Phobos Rendezvous and Landing

As discussed, future small body missions, missions to the planets, and rendezvous with man-made targets will require a navigation system that can carry and utilize extensive knowledge of the object of attention. Modeling of portions of, or entire planetary bodies, will be necessary, as well as modeling of specific "landmark locations" of spacecraft. Much of this modeling effort has been progressing in a parallel but separate effort, but nevertheless one focused on navigation (Ref. 10). In a prototype for the next generation AutoNav design, this body modeling and landmark tracking system was interfaced to the Deep Impact AutoNav flight software. This prototype was configured to operate in the Mars environment and implement a rendezvous and landing with the moon Phobos. Figure 13 shows the Phobos approach and rendezvous trajectory and Figures 14 and 15 show two scenes from the simulation, showing the images received from the simulated spacecraft camera, identified and processed landmarks, and the current accuracy of the navigation system, in both knowledge and control. The development of this prototype system has revealed important lessons in the nature of interfaces and simulation capabilities required. Most importantly, it has provided insight into the capabilities of the eventual flight AutoNav system, providing a landing control accuracy capability on a relatively massive small body on the order of one meter using only passive optical data and requiring relatively modest instruments. Figure 16 contains a history of the approach from the beginning of the simulation at 11.5 hours before landing, to touchdown, showing the altitude, the knowledge and control error, and the accumulated deterministic and statistical delta-v at discrete propulsion event opportunities - of which there were 16. For this simulation, a pair of cameras was assumed, one of moderate Field of View

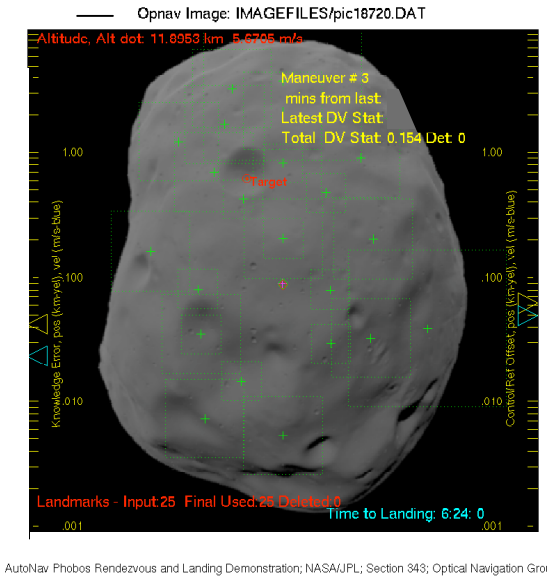

Figure 14: The Phobos rendezvous and landing demonstration: ~6h before touchdown, showing landmarks, and landing site.

(FOV) of 20 degrees, the other very wide at 70 degrees. These wide fields were chosen largely to enhance the visualization during the demonstration - the net effect on the navigation performance of choosing these vs. the instruments discussed above, is a large attenuation of data strength, which makes this a very conservative simulation indeed. The ONC/ISC combination discussed above would provide even better performance than that displayed in Figure 16.

The initial position and velocity error in the Phobos simulation is roughly one-half kilometer, and several cm/sec relative to Phobos, both in the initial offset from the desired trajectory and in initial estimate - in different directions. This is a very conservative error, assuming a mission would have been doing extensive optical navigation up to the beginning of the rendezvous and landing. As was the process for Deep Impact onboard the Impactor, a long data arc is accumulated before the first Orbit Determination (OD) is done. The simulation starts at about landing minus half a day, and the first OD is at about 9 hours before landing. Image processing error sources are created realistically by processing "real" simulated images, and letting the realistic distortions (due to orbit errors) and pixelization make their natural effects. No pointing errors are invoked, for the obvious reasons that the fields are so wide, and it is reasonable to assume that star-trackers or equivalent would be working nominally. Sixteen maneuver opportunities exist, with three of these having initial deterministic components. There are more than 20m/s expended for this demonstration deterministically, and a small fraction of that used for statistical corrections. Many opportunities for statistical corrections are not utilized by the system because the good levels of orbit control. Maneuvers experience noise errors of about one percent of magnitude and an equivalent amount in direction errors, and bias errors of several mm/. Phobos is assumed to be point-source so no gravity field errors are invoked, nor is solar pressure. These would be added to a flight system, of course, as discussed above. However, the system does adapt to errors in the position of the landing site - the guidance system refers to an inertial reference trajectory until the landing site is visible on descent, and then switches to a visual lock on the landing site - and thereby adjusts for several 10's of meters error in the predicted position of the landing site relative to the center of mass, potentially absorbing a wide range of modeling errors. Additionally, optical data is not greatly sensitive to gravity perturbations on short time scales, for obvious reasons - it's a direct-observation-of-geometry data type and does not infer perturbations in cross-line-of-sight as do Doppler and range radiometrics.

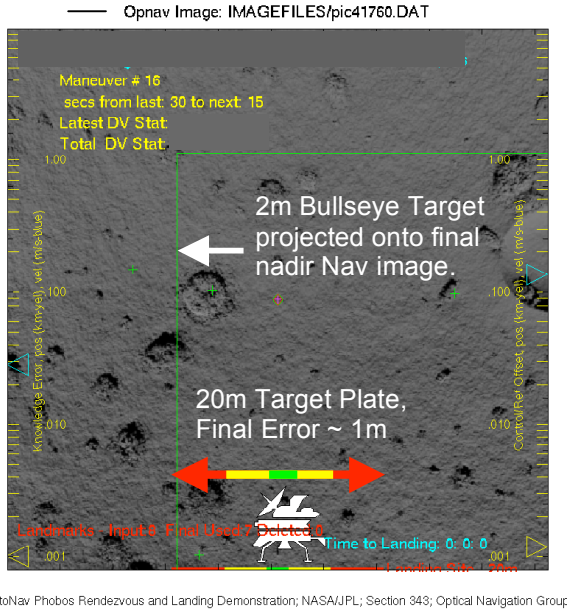

Figure 15: Phobos rendezvous and landing demonstration: touchdown, with sub-meter target-relative navigation.

B. Rendezvous and Docking with MRO

Two additional simulations are currently being developed based in part on the Phobos simulation. The first of these is a rendezvous and docking, not with a natural satellite or asteroid, but with a spacecraft. Much time, effort and thought is currently being expended throughout NASA's various programs, especially the manned program, on the subject of vehicle-to-vehicle rendezvous. Many observational strategies have been developed that can be applied to a vehicle on vehicle rendezvous, including radiometrics, active optical methods, such as LIDAR, and RADAR. All of these methods require some sort of cooperation on the part of the target vehicle, either actively by way of transponders for radio data, or passively by way of corner reflectors for LIDAR and RADAR, in order to eliminate the ambiguity of the reflection of signal from various surfaces of a complex spacecraft. This AutoNav demonstration is endeavoring to show that an accurately controlled rendezvous can be performed with passive optical, with nothing more cooperative required of

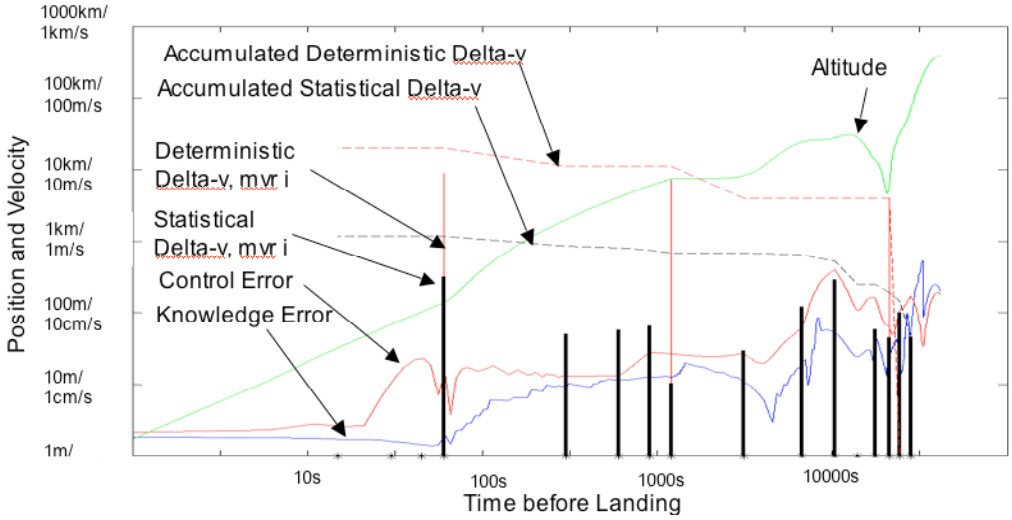

Figure 16: Phobos rendezvous and landing simulation navigation performance, showing the progression of knowledge and control error from L-11:30 to Landing.

the target vehicle than flat-painted targets on the bus, for example, a NASA decal. To be sure, to enable rendezvous in all lighting conditions, the target vehicle will probably require navigation lights, which manned craft generally come equipped with anyway.

The core difficulty in performing such a rendezvous with passive optical, and especially proving its viability via a computerized simulation is the portrayal of the target spacecraft. The spacecraft must be portrayed two ways for such a simulation, "externally" from the navigation system proper, for purposes of generating a "truth," *ie*, the real-world image pretending to be that taken from the chasing vehicle, and "internally" to the navigation system for purposes of extracting the data. The former is much more difficult than the latter, as the whole of the spacecraft, in general, must be rendered. There are many different means of generating simulated views of spacecraft, but the software means of performing these renderings is quite CPU intensive, making the demonstration of real-time rendezvous operations – let alone their real implementation – very difficult, and the simulations for multiple Monte Carlo analyses prohibitively time consuming. Fortunately, the Model-based Systems Engineering and Architectures Section at JPL has developed a means of very realistically rendering vehicle models in PC-based hardware, and allowed the OpNav group to use these tools for the simulation. Figures 17a and b show two views of the MRO spacecraft as generated by these tools. Of note in Figure 17b is a round "docking-device" and NASA and JPL logos on the X-face of the spacecraft, which are not on the actual MRO vehicle of course, but were added as necessary for a rendezvous and docking simulation. Though only the docking mechanism would be mechanically necessary in a real rendezvous, the logos form very useful targets for the navigation system. The same techniques and software that are used in the Phobos demonstration to model the landmarks on that body are used to model these three patches on MRO (the docking mechanism and logos.) The other areas of the spacecraft, and typically of unmanned spacecraft, are covered with dark or specularly reflecting blankets, or glassy solar panels, which would not lend themselves well to modeling. These three patches are fit to digital terrain and albedo maps, and used to precisely locate the target vehicle for the rendezvous and docking. At the time of writing this demonstration is not yet complete, but expectations based on navigation performance simulations for Mars Sample Return are that the navigation will be accurate to the centimeter level and several mm/sec (Ref. 9).

C. Lunar Landing

The second new demonstration that is in the process of development is a Lunar landing demonstration. This simulation more closely mimics that of the Phobos rendezvous and landing, with the exception of a

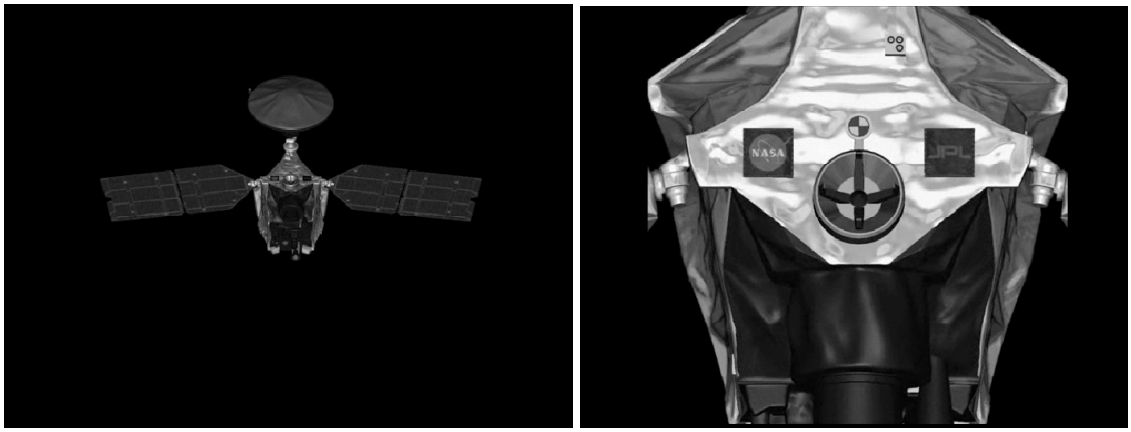

Figure 17a: OpenGL-rendered model of the MRO spacecraft: AutoNav processing method from this range will be "center of brightness." b: spacecraft model rendered at sufficient range for AutoNav to acquire and perform precision location of rendezvous targets.

much stronger gravitational potential, and much faster relative velocities. Additionally, the simulation will utilize a newly developed global terrain and topography model based on images taken by the Clementine mission. Like Clementine, the Lunar Lander simulation will start in a polar orbit. The system assumes that a detailed map (to sub-meter resolution) has been made using a small telescope, such as the ONC, from a staging orbit of 10 or 20 km. The final detailed maps can be made in a matter of a few hours manually on Earth, or even autonomously onboard, in a few minutes. Armed with these last detailed images at the landing site, the simulation will take one last half-rev over the lit Lunar surface, and commence a long powered descent being guided by the landmarks. The landing site chosen is in Amundsen crater near the south Lunar pole, where regions of perpetual shadow offer the potential for ice deposits. Clementine measurements offer tantalizing hints of high proton flux over this region, indicating the potential presence of water. Though only lit regions are used as landmarks, in the event of an actual mission landing in this region, areas of shadow, and even perpetual shadow could be used, utilizing Earth-shine or alpine-glow (the reflected light from nearby peaks, some of which are in perpetual sunlight). Fig.18 shows one of the Clementine images used for the model development and several of the landmarks being used by the simulation in this region.

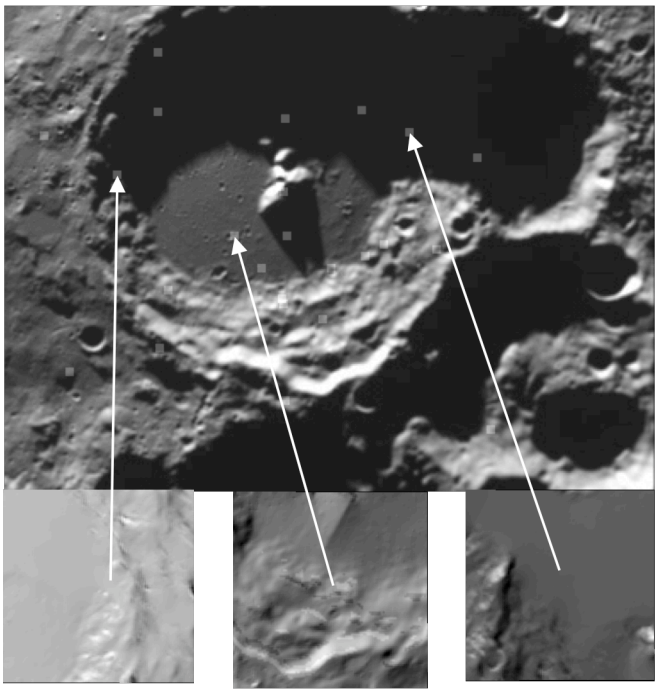

Figure 18: Amundsen Crater on the Lunar South Pole, image from Clementine, and three sample landmarks being used for simulated Lunar landing, using passive optical data only.

V. Concluding Remarks and Acknowledgements

There will be many opportunities for AutoNav/AutoGNC systems to enhance operations and increase science return for future missions. The early versions of these systems have proven themselves to be very successful in the

recent past, but reaching their full future potential will require careful systems engineering. That work is now beginning.

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. Several individuals and organizations at JPL were very important in providing support for this work, including Frank Jordan of the Advanced Studies office of the Mars Program, Robert J. Cesarone of the Architecture and Strategic Planning Office of the Interplanetary Network Directorate, and Krishna M. Koliwad of the Research and Technology Program. Special thanks go to Kevin Hussey, John Howard, and Paul Upchurch in the model-based systems engineering and architectures section at JPL for extensive assistance in the OpenGL rendering of the MRO spacecraft. Thanks go also to Johnny Chang of the Flight System Simulation Section for assistance in the similar hardware rendering of Phobos, and Bill Owen who is leading the integration of some of the advanced AutoNav concepts discussed here into the existing Voyager heritage navigation software. Finally, thanks go to Chuck Acton and David Berry for reviewing this text and offering many helpful suggestions.

References

1. Kubitschek, D. G., et al, "Deep Impact Autonomous Navigation: The Trials of Targeting the Unknown", AAS 06-081

2. Riedel, J.E., Bhaskaran, S., Desai, S.D., Han, D., Kennedy, B., McElrath, T., Null, G.W., Ryne, M., Synnott, S.P., Wang, T.C., Werener, R.A., "Using Autonomous Navigation for Interplanetary Missions: The Validation of Deep Space 1 AutoNav", IAA Paper L-0807, Fourth IAA International Conference on Low-Cost Planetary Missions, Laurel, Maryland, May 2000

3. Bhaskaran, S., J. E. Riedel, B. Kennedy, T. C. Wang, "Navigation of the Deep Space 1 Spacecraft at Borrelly," AIAA paper 2002-4815, AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Monterey, CA, August5-8, 2002.

4. Bhaskaran, S., Mastrodemos, N., Riedel, J., Synnott, S., "Optical Navigation for the Stardust Wild 2 Encounter", 18th International Symposium of Space Flight Dynamics, 11-15 October, 2004, Munich Germany.

5. S. P. Synnott, A. J. Donegan, J. E. Riedel, J. A. Stuve, "Interplanetary Optical Navigation: Voyager Uranus Encounter", AIAA-86-2113, AIAA Conference, Williamsburg Virginia, August 1986.

6. Riedel, J. E., W. Owen, J.Stuve, S. Synnott, "Optical Navigation During the Voyager Neptune Encounter", AIAA paper 90-2877-CP, AIAA/AAS Astordynamics Conference, Portland Oregon, August 1990.

7. Riedel, J. E., Desai, S., Han,D., Bhaskaran,S., Kennedy, B., McElrath, T., Null, G.W., Ryne, M., Synnott, S.P., Wang, T.C., Werner, R.A., and Zamani, E.B., "Autonomous Optical Navigation (AutoNav) Technology Validation Report" in Deep Space 1 Technology Validation Reports, JPL Publication 00-10 (Jet Propulsion Laboratory, Pasadena, CA, October 2000).

8. Franklin, S., Slonski, J., Kerrdige, S., Noreen, G., Riedel, J, Komarek, T., Stosic, D., Racho, C., Edwards, B., Austin, R., Boroson, D., "The 2009 Mars Telecom Orbiter Mission", IEEE paper IEEEAC#1597, IEEE Aerospace Conference, March 5-12, 2005, Big Sky, Montana.

9. Riedel, J.E., Guinn, J., Delpech, M., Dubois, J.B., Geller, D., Kachmar, P., "A Combined Open-Loop and Autonomous Search and Rendezvous Navigation System For the CNES/NASA Mars Premier Orbiter Mission, 26th Annual AAS Guidance and Control Conference, February 5-9, 2003, Breckenridge, Colorado.

10. Gaskell, Robert A., "Landmark Navigation and Target Characterization in a Simulated Itokawa Encounter", 2005 AAS/AIAA Astrodynamics Specialist Conference, Lake Tahoe, California

11. Moyer, T. M., Formulation for Observed and Computed Values of Deep Space Network Data Types for Navigation, Monograph 2, Deep Space Communications and Navigation Series, 2000, JPL Publication 00-7.

12. Bierman, G.J., Factorization Methods for Discrete Sequential Estimation, Academic Press, 1977.

13. "http://www.pcai.com/web/ai_info/blackboard_technology.html"

14. J. K. Campbell, S. P. Synnott, J. E. Riedel, S. Mandell, L. A. Morabito, G. C. Rinker, "Voyager 1 and Voyager 2 Jupiter Encounter Orbit Determination", AIAA-80-0241, AIAA Aerospace Sciences Meeting, January 14-16, 1980, Pasadena, California

15. Grasso, C. A., "The Fully Programmable Spacecraft: Procedural Sequencing for JPL Deep Space Missions Using VML (Virtual Machine Language)," 0-7803-7231-XIEEE, IEEEAC paper #187, September 28, 2001

16. Peer, Scott, Grasso, C. A., "Spitzer Space Telescope Use of the Virtual Machine Language," 0-7803-8870-4/05, 2005 IEEE, IEEEAC Paper #1068, December 1, 2004

17. Brady, T., Tillier, C., Brown, R., Jimenez, A., Kourepenis, A., "The Inertial Stellar Compass: A New Direction in Spacecraft Attitude Determination", 16th Annual AIAA?USU Conference on Small Satellites, Logan, Utah, August 12-15, 2002.

18. "http://www.moog.com/Media/1/Biax%20Intro.pdf"