GNC 2017 : 10th International ESA Conference on Guidance, Navigation & Control Systems
29 May - 2 June 2017, Salzburg, Austria
European Space Agency

A MINIMAL STATE AUGMENTATION ALGORITHM FOR VISION-BASED NAVIGATION WITHOUT USING MAPPED LANDMARKS

A. Miguel San Martin[(1)], David S. Bayard[(2)], Dylan T. Conway[(3)], Milan Mandic[(4)], Erik S. Bailey[(5)]

[(1)] *California Institute of Technology: Jet Propulsion Laboratory. 4800 Oak Grove Dr, Pasadena CA 91109. 1-818-354-3593. Alejandro.M.Sanmartin@jpl.nasa.gov*
[(2)] *California Institute of Technology: Jet Propulsion Laboratory. 4800 Oak Grove Dr, Pasadena CA 91109. 1-818-354-8208. David.S.Bayard@jpl.nasa.gov*
[(3)] *California Institute of Technology: Jet Propulsion Laboratory. 4800 Oak Grove Dr, Pasadena CA 91109. 1-818-354-1754. Dylan.T.Conway@jpl.nasa.gov*
[(4)] *California Institute of Technology: Jet Propulsion Laboratory. 4800 Oak Grove Dr, Pasadena CA 91109. 1-818-393-0235. Milan.Mandic@jpl.nasa.gov*
[(5)] *California Institute of Technology: Jet Propulsion Laboratory. 4800 Oak Grove Dr, Pasadena CA 91109. 1-818-393-3792. Erik.S.Bailey@jpl.nasa.gov*

ABSTRACT

This paper describes MAVeN (Minimal State Augmentation Algorithm for Vision-Based Navigation), which is a new algorithm for vision-based navigation that has only 21 states, yet is able to track features in successive camera images and use them to propagate estimates of the spacecraft position and velocity. The filter dimension drops to 12 if attitude information is already available. The low filter dimension makes MAVeN a very reliable and practical algorithm for real-time flight implementation. The main idea is to project observed features onto a rough shape model of the ground surface, which are then used by the filter as pseudo-landmarks. The shape model is assumed to be known beforehand, as would be obtained from prior surveillance of the landing site from orbit. MAVeN does not require pre-mapped landmarks, so it is able to navigate terrain that has not been previously observed up close. This property is especially important for close-proximity operations in small body missions where ground surface features are being seen for the first time at close range. MAVeN is also able to hover motionless above the ground without position error growth, which is unusual for this class of vision-based navigation algorithms.

1 INTRODUCTION

Several space exploration missions are currently being formulated to land and perform proximity operations on and around the surface of small and large natural bodies that make extensive, and in some cases exclusive, use of autonomous Vision Based Navigation (VBN) based on a monocular camera, an IMU, and an altimeter.

These missions make use of two sometimes-complementary VBN techniques. At high altitudes they use an on-board map of surface landmarks, developed previously during the surveying phase of the

© 2017 California Institute of Technology. Government sponsorship acknowledged.

mission, to determine the absolute localization of the spacecraft in the given body. As the spacecraft gets closer to the surface, however, Map Based Localization (MBL) has difficulty recognizing landmarks that were mapped from a much higher altitudes. This lack of robustness when operating close to the surface is worrisome because it is when the vehicle is in the most danger, and exactly when reliability is needed most.

For this reason a second VBN technique is used that relies on tracking image features from frame-to-frame to keep the position knowledge error from growing and to provide a precise ground relative velocity estimate. Most generally, this can be accomplished using SLAM (Simultaneous Localization and Mapping) algorithms from the literature [1][2]. Unfortunately, full EKF-SLAM solutions are not practical for many real-time applications due to their oversized filter state dimensions and large computational overhead. Specifically, EKF-SLAM approaches augment the Kalman Filter with 3 states for each of the n features observed, increasing the overall filter state order by $3 * n$. This leads to extremely high-order state vectors resulting in filters that are only marginally numerically stable and demand an unwieldy amount of on-board computation. In addition, SLAM type algorithms require ground relative vehicle motion during initialzation to be able to solve for the starting surface feature positions, which imposes an undesirable operational constraint.

This paper will describe MAVeN (Minimal State Augmentation Algorithm for Vision-Based Navigation), a new algorithm for vision-based navigation that estimates the position and velocity of a vehicle operating in close-proximity to a planetary or small body surface without the need of an on-board landmark map. Instead, MAVeN requires a rough shape model in the form of a Digital Elevation Map (DEM) or facet model, and information about the corresponding surface normal at each ground point. *MAVeN is based on a new conceptual approach that projects real-time observed features onto the shape model of the small body surface, which are then used as pseudo-landmarks in the filter formulation*.

MAVeN comes in two versions, the 21-state filter MAVeN-AT and the 12-state filter MAVeN-T. MAVeN-AT has more states because it propagates both attitude (A) and translational (T) states, while MAVeN-T only propagates translation. MAVeN-T assumes knowledge of the spacecraft attitude relative to the small-body, as derived from the celestial attitude knowledge of the spacecraft (provided by the Star Tracker and IMU) and a model of the natural body rotation, which are both available in these types of missions.

These algorithms are said to be "minimally augmented" in the sense that MAVeN-AT is augmented with the minimum number of 6 states needed to clone attitude and translation information, while MAVeN-T is augmented with the minimum number of 3 states needed to clone translation information. Both of these filters are significantly reduced in dimension compared to EKF-SLAM which is augmented with $3 * n$ states, where the number of features n can be several hundred in practice.

MAVeN is also novel in that the shape model's surface normals (i.e., slopes), are incorporated into the partial derivatives defining the measurement matrix that is used to update the spacecraft position.

Finally, MAVeN requires a starting knowledge of the spacecraft position and velocity relative to the small body as provided by ground navigation or high altitude on-board autonomous landmark based navigation, also a very reasonable assumption. It is important to stress at this point that any science mission, past or present, that will land on the surface of a small or planetary body is going to require extensive knowledge of the topography and other characteristics of the surface. This knowledge will

© 2017 California Institute of Technology. Government sponsorship acknowledged.

typically be acquired during the survey phase of the mission, in order to select the best landing site that maximizes science while diminishing risks. Hence, the need for MAVeN to have an on-board shape model of the terrain is not a very restrictive assumption. It is worth mentioning that the need for the shape model and for good initial position knowledge, can be relaxed for those applications where we can approximate the surface with a flat plane.

MAVeN has been shown to significantly limit the growth of position and velocity errors over a pure IMU-based propagation scheme, while greatly reducing computation and improving robustness relative to competing approaches from the literature. In addition, MAVeN does not require ground-relative vehicle motion during filter initialization, which eliminates an undesirable operational constraint.

This paper will describe the motivation and general architecture of the Navigation Filter, its mathematical derivation, and will present simulated performance results for a landing scenario.

2 ESTIMATION ARCHITECTURE

The main highlights of the MAVeN algorithm are as follows:

1. Ability to compute vehicle delta-pose by tracking features between images with a minimum state augmentation: 3 cloned states for MAVeN-T, and 6 cloned states for MAVeN-AT, where the number of states is independent of the number of features being tracked. This is enabled by projecting image features onto an on-board shape model, starting with sufficiently good position knowledge to know where on the DEM the vehicle is being initialized.

2. Delta pose measurements are created between the poses taken at the time of the present camera image (i.e. the Tracking Frame) and the earlier Base Frame image. This continues until a new Base Frame image is taken, at which time the process repeats.

3. Ability to combine delta-pose and absolute-pose measurements (e.g. altimetry, landmark based Map Relative Localization)

4. A measurement pre-processor that compresses the measurement vector dimension using the QR algorithm [3], from 2*M, where M is the number of features or landmarks being tracked, to a pseudo measurement of dimension 3 for MAVEN-T, or 8 for MAVEN-AT. This pseudo-measurement is then fed into the Kalman Filter to update the current filter state comprised of the vehicle state, Base Frame vehicle state, and sensor biases.

5. Incorporation of the Base Frame as a *clone state* in the Kalman Filter.

Highlight 1 is the main innovation behind MAVeN. It enables MAVeN to perform reliable state estimation while maintaining a completely motionless hover above the ground surface. This is discussed in more detail at the end of this section. The remaining highlights are similar to developments documented in various papers by numerous authors [3][4][5][6].

© 2017 California Institute of Technology. Government sponsorship acknowledged.

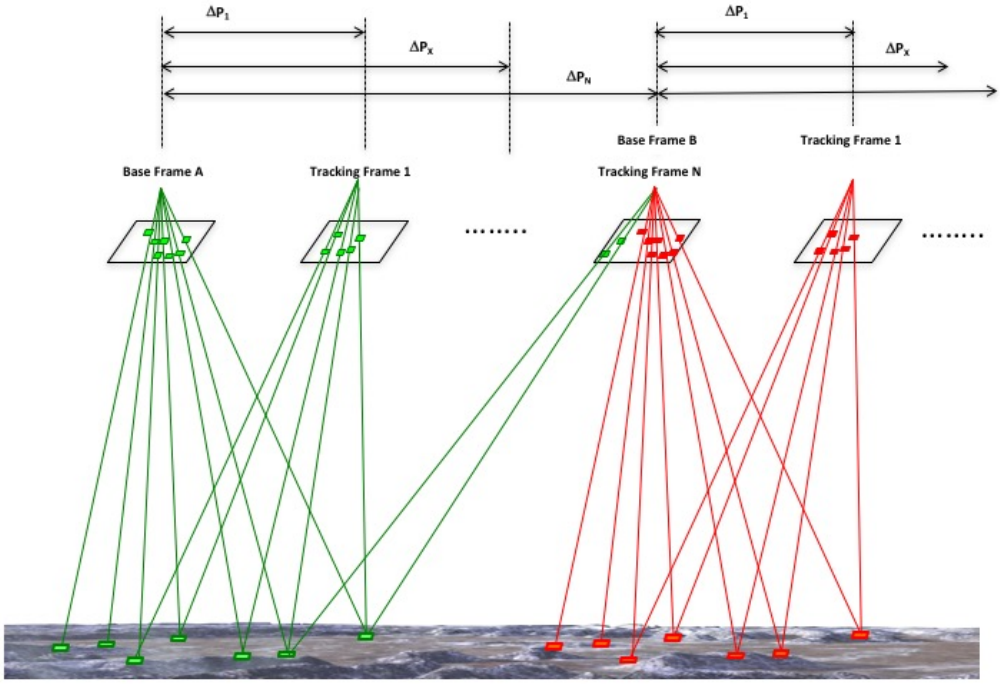

Figure 1: Feature Tracking Concept

2.1 Feature Tracking Concept

MAVeNs Feature Tracking concept is illustrated in Fig. 1. The sequence starts on the left of the figure with the first image established as Base Frame A. Every time a new Base Frame is established, the features depicted in the image are projected onto the surface of the body given the detector coordinates of the features, the current estimate of the spacecraft position and attitude, and the on-board surface DEM. The surface coordinates of these features are placed in the Base Frame Feature Catalog (BFFC) (Fig. 2) together with other ancillary data. When the next image is taken, identified in the figure as Tracking Frame 1, it is processed by a computer vision algorithm to find the correspondence between the features in the Tracking Frame and the ones in the Base Frame. Once the features are matched between the Base and the Tracking Frame, the locations of the corresponding features stored in the BFFC are used to compute the difference between the two spacecraft positions occupied when each image was taken. The resulting delta-position measurement is then fed to the Kalman Filter that fuses the measurement to the IMU and altimetry measurements in order to compute an estimate of the current spacecraft state and sensor biases (Fig. 2).

When the next image is taken the process repeats between the current Tracking Frame and the Base Frame and a new delta position between those two frames is computed and fed to the Kalman Filter. When the number of paired features between the Base Frame and Tracking Frame fall below a certain number (either because they fell out of the camera FOV or the scale has changed too much between

© 2017 California Institute of Technology. Government sponsorship acknowledged.

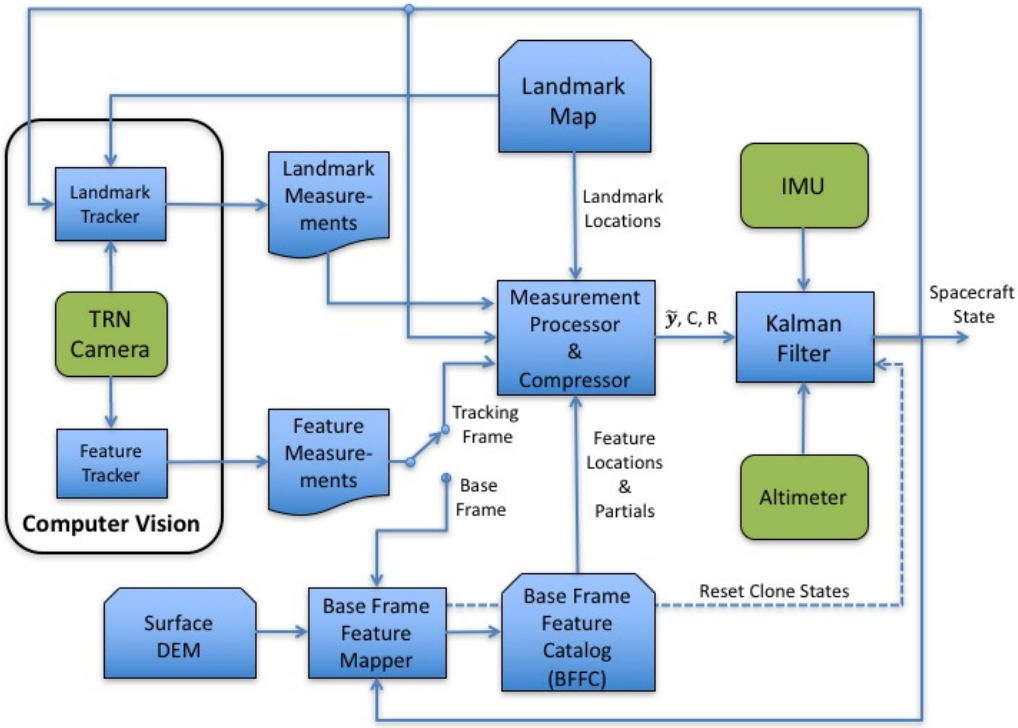

Figure 2: Block diagram of MAVeN estimation architecture

the images to be identifiable) then a new Base Frame is established by emptying the BFFC, projecting the features of the new Base Frame into the body surface, and storing its coordinates in the now empty BFFC . This illustrated in Fig. 1 where Tracking Frame N becomes new Base Frame B.

Notice that the computed delta positions are always between the current Tracking Frame and the Base Frame. This is done to reduce the unbounded growth of the position propagation errors. This error will grow in a random-walk fashion each time a new Base Frame is established. Also by extending the time as much as possible between a given Base Frame and its last associated Tracking Frame the Kalman Filter will do a better job at estimating the velocity of the vehicle and other sensor biases.

2.2 Functions and Data Flow

The MAVeNs functions and data flow are described in detailed in Fig. 2. The Computer Vision function, which is usually hosted in a computation accelerator processor (e.g. FPGA), processes the image data to identify Landmarks and Features for absolute and relative visual navigation, respectively.

The Landmark Tracker uses the spacecraft current state and an on-board Landmark Map to identify landmarks in the image and it outputs a data structure with the time of the image exposure and a list of landmark IDs with their respective 2D measurement in camera detector coordinates and measurement noise statistics.

© 2017 California Institute of Technology. Government sponsorship acknowledged.

The Feature Tracker performs a different function depending whether it is processing a Base Frame or a Tracking Frame. For a Base Frame the Feature Tracker applies first an interest operator to identify salient features in the image that it will track in successive images. The Feature Tracker then assigns an ID to each of the features and creates a data structure with a label identifying it as a Base Frame, the time of the image exposure, and a list of feature IDs with their respective 2D measurement in camera detector coordinates and measurement noise statistics. After a Base Frame is established, for each image that follows, the Feature Tracker searches the current image for the features identified in the Base Frame. For each feature found, it determines its position in the current frame and it outputs a data structure with a label identifying it as a Tracking Frame, the image exposure time, and the list of matched features with their current image position measurements and statics. When the number of features in the current image that the Feature Tracker manages to match with the ones in the Base Frame falls below a given threshold, the Feature Tracker process the current image as a new Base Frame and it starts the process anew.

The purpose of the Base Frame Feature Mapper is to process a new Base Frame output when the Feature Tracker generates one. When this is the case, the Base Frame Feature Catalog is emptied and filled with the 3D locations on the body surface of the features listed in the new Base Frame data structure, using the features sensor coordinates, the current estimate of the spacecraft state, and the on-board DEM, to project the features onto the body surface. In addition, the Feature Mapper computes for each feature the partials of their location in the body frame with respect to the state of the vehicle (position or position and attitude) at the time of the Base Frame image exposure, and places the partials also in the BFFC. Finally, the Feature Mapper sends a signal to the Kalman filter to reset its clone states associated with the Base Frame state, as will be described in the next section.

The Measurement Processor and Compressor, as its names implies, performs two functions. First it computes the measurement residuals and measurement matrix in detector coordinate space resulting in a $2n$ measurement vector where n is the number of Landmarks or Features. The computation of the measurement residuals for the Landmarks and Features is virtually identical. It uses the Landmark/Feature measurements in the data structures generated by the Computer Vision function, the location of the Landmarks/Features in the body frame as present in the Landmark-Map/Feature-Catalog databases, and the current vehicle state estimate to compute the residuals. The measurement matrix computations, on the other hand, differ somewhat between Landmarks and Features. While the partials of the measurements residuals with respect to the current vehicle state are the same for both, Feature processing requires the computation of the partials of the residuals with respect to the vehicle state at the time of the Base Frame, which requires the use of the partials stored in the BFFC.

The result of the first stage of the Measurement Processor & Compressor is a $2n$ measurement vector and a measurement matrix that is $2n \times M$, where $M = 3$ for Landmark processing if it is used to estimate the vehicle position only and 6 if it is used to estimate the vehicle position and attitude, $M = 6$ for Feature processing if it is used to estimate delta-position only, and 12 if it is used to estimate delta-position and delta-attitude. Finally, the Measure Processor also generates a $2n \times 2n$ measurement noise covariance matrix.

The second function of the Measurement Processor & Compressor is to compress the measurement equation resulting from the first step from a $2n$ dimension to an L dimension pseudo-measurement, where L is 3 for Landmark processing for position only estimation and 6 for position and attitude

estimation, or 3 for Feature processing for delta-position estimation and 9 for delta-position and delta-attitude estimation. Note that the 9 dimensions account for 3 delta-positions, plus 3 delta-attitudes, plus 3 for the absolute Base attitude for a general surface. For a flat surface, however, the dimension is 8 because the attitude around the surface perpendicular cannot be determined. Note again that the reduced dimension pseudo measurement is totally independent of the number of Features/Landmarks in the Tracking Frame.

In order to accomplish this measurement compression, the Measurement Processor and Compressor uses the QR factorization, as introduced into the vision navigation community in [3], and explained in the next section. The advantages of the measurement compression architecture are: 1) reduced number of computations, 2) improved numerical stability and 3), perhaps in a more esthetical sense, isolation of the Kalman Filter formulation from the complexity of the computer vision computations.

Finally, the function of the Kalman Filter is to fuse the measurements from the Camera, the Altimeter, and the IMU together in order to generate an estimate of the spacecraft state.

2.3 Stable Hover Property

It is known that performing state estimation during a motionless hover above the ground surface is very difficult for vision-based navigation algorithms [7]. This fact is somewhat concerning since most scenarios for small body exploration will include a hovering path segment or some approximation to it. While algorithms based on the computationally intensive EKF-SLAM are generally able to hover [1][2], it is unfortunate that there are very few reduced-order estimators with this capability. This is because simpler estimators are often based on optical flow or enforcing epipolar or multistate constraints [4][8], and as such, generate vacuous measurements under a motionless hover condition. To overcome this limitation, researchers have resorted to augmenting their filter state with persistent features [9][5] or invoking in-flight detection and special response to a hover condition [7]. While these modified methods can be made to work in practice, they generally still involve a large number of filter states and can be difficult to implement. *In contrast, the MAVeN algorithm contains only 12 filter states (21 with attitude estimation), and is able to achieve a stable, completely motionless hover.* This capability follows directly from the novel approach of projecting image features onto a shape model of the ground surface to use as pseudo-landmarks.

3 MATHEMATICAL DERIVATION

This section derives the filter equations used by MAVeN. There are three reference frames of interest. For clarity and convenience of notation, the vehicle and all sensor frames are co-located at a single body-fixed frame. The position and attitude of the body-fixed frame are estimated with respect to a surface-fixed frame. The surface-fixed frame position and attitude with respect to an inertial-frame are assumed to be known.

© 2017 California Institute of Technology. Government sponsorship acknowledged.

3.1 Error Kinematics

First define the following quantities. The vector from the surface frame to the sensor frame expressed in surface frame coordinates is $\boldsymbol{r}$. The time derivative of $\boldsymbol{r}$ is $\boldsymbol{v}$. The attitude of the sensor frame with respect to the surface frame is represented by the quaternion $\boldsymbol{q}$. The sensor frame acceleration and angular rate measurements $\tilde{\boldsymbol{a}}$ and $\tilde{\omega}$ are available as a function of time and are corrupted by biases, $\boldsymbol{\beta}_a$ and $\boldsymbol{\beta}_\omega$, and random noises.

Define the full 16-dimensional state vector at the current time as

$$\boldsymbol{x}_c = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \\ \boldsymbol{\beta}_a \\ \boldsymbol{q} \\ \boldsymbol{\beta}_\omega \end{bmatrix} \in \mathbb{R}^{16} \tag{1}$$

A stochastic kinematic model relates the acceleration and angular rate measurements to the derivative of the state, $\dot{\boldsymbol{x}}_c$ through

$$\dot{\boldsymbol{x}}_c = \mathrm{f}(\boldsymbol{x}_c, \boldsymbol{w}) \tag{2}$$

where $\boldsymbol{w}$ is random white Gaussian noise with covariance Q that corrupts the acceleration and angular rate measurements, and causes the biases to drift in time.

An estimate of the state, $\hat{\boldsymbol{x}}_c$, has incremental error $\boldsymbol{e}_c = \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)$ defined as

$$\Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c) \equiv \begin{bmatrix} \boldsymbol{r} - \hat{\boldsymbol{r}} \\ \boldsymbol{v} - \hat{\boldsymbol{v}} \\ \boldsymbol{\beta}_a - \hat{\boldsymbol{\beta}}_a \\ \Delta_q(\boldsymbol{q}, \hat{\boldsymbol{q}}) \\ \boldsymbol{\beta}_\omega - \hat{\boldsymbol{\beta}}_\omega \end{bmatrix} \in \mathbb{R}^{15} \tag{3}$$

where any variable with a hat accent is an estimate of the corresponding non-accented variable. In addition, the function $\Delta_q(\boldsymbol{q}_1, \boldsymbol{q}_2)$ is a three-dimensional representation of the rotation from $\boldsymbol{q}_2$ to $\boldsymbol{q}_1$ As an example for a small rotation, this could be represented by three Euler angles[10]. Note that $\boldsymbol{e}_c$ is 15-dimensional while $\boldsymbol{x}_c$ is 16-dimensional due to the reduction of the four-dimensional quaternion for absolute attitude to a three-dimensional vector for attitude error.

The estimate is propagated in time by the equation

$$\dot{\hat{\boldsymbol{x}}}_c = \mathrm{f}(\hat{\boldsymbol{x}}_c, \boldsymbol{0}) \tag{4}$$

which implies that the error is propagated by the equation

$$\begin{aligned} \dot{\boldsymbol{e}}_c &= \frac{d}{dt}\left(\Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)\right) && (5) \\ &= \frac{\partial \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)}{\partial \boldsymbol{x}_c}\dot{\boldsymbol{x}}_c + \frac{\partial \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)}{\partial \hat{\boldsymbol{x}}_c}\dot{\hat{\boldsymbol{x}}}_c && (6) \\ &= \frac{\partial \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)}{\partial \boldsymbol{x}_c}\mathrm{f}(\boldsymbol{x}_c, \boldsymbol{w}) + \frac{\partial \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c)}{\partial \hat{\boldsymbol{x}}_c}\mathrm{f}(\hat{\boldsymbol{x}}_c, \boldsymbol{0}) && (7) \end{aligned}$$

where Eq. 2 and Eq. 4 have been used. Note that Eq. 7 is a function of the true and estimated state $\dot{\boldsymbol{e}}_c = \dot{\boldsymbol{e}}_c(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c, \boldsymbol{w})$. Define the incremental correction function $\boldsymbol{x}_c = \mathrm{g}_c(\hat{\boldsymbol{x}}_c, \boldsymbol{e}_c)$. This is defined as the inverse of Eq. 3 such that $\boldsymbol{x}_c = \mathrm{g}_c(\hat{\boldsymbol{x}}_c, \Delta(\boldsymbol{x}_c, \hat{\boldsymbol{x}}_c))$. Substitute $\boldsymbol{x}_c = \mathrm{g}_c(\hat{\boldsymbol{x}}_c, \boldsymbol{e}_c)$ into Eq. 7 and linearize about the current estimate $\hat{\boldsymbol{x}}_c$, $\boldsymbol{e}_c = \boldsymbol{0}$, and $\boldsymbol{w} = \boldsymbol{0}$ to obtain $\dot{\boldsymbol{e}}_c(\boldsymbol{e}_c, \hat{\boldsymbol{x}}_c, \boldsymbol{w})$ to first order as

$$\dot{\boldsymbol{e}}_c = F_e \boldsymbol{e}_c + F_w \boldsymbol{w} \tag{8}$$

where

$$F_e \equiv \frac{\partial \dot{\boldsymbol{e}}_c(\boldsymbol{e}_c, \hat{\boldsymbol{x}}_c, \boldsymbol{w})}{\partial \boldsymbol{e}_c} \tag{9}$$

$$F_w \equiv \frac{\partial \dot{\boldsymbol{e}}_c(\boldsymbol{e}_c, \hat{\boldsymbol{x}}_c, \boldsymbol{w})}{\partial \boldsymbol{w}} \tag{10}$$

Let $\boldsymbol{e}_c$ have covariance $P_{cc} \equiv \mathrm{E}\left\{\boldsymbol{e}_c \boldsymbol{e}_c^{\mathrm{T}}\right\}$ where the superscript T denotes the transpose. Using the first order error model in Eq. 8, the covariance propagates as

$$\dot{P}_{cc} = F_e P_{cc} F_e^{\mathrm{T}} + F_w Q F_w^{\mathrm{T}} \tag{11}$$

3.2 Visual Feature Measurements

Image processing algorithms process camera images to produce *visual feature measurements* which are the image space pixel locations that surface fixed points project to. The image processing algorithms have two main components. The first is feature detection which is applied to *Base Frames* to detect new surface fixed points. The second is feature tracking which is applied to *Tracking Frames* to track surface fixed points detected in the most recent Base Frame. For a given camera position $\boldsymbol{r}$ and attitude $\boldsymbol{q}$, a three-dimensional surface fixed point $\boldsymbol{z}$ ideally projects to the two-dimensional image space location

$$\boldsymbol{u} = \pi(R(\boldsymbol{q})(\boldsymbol{z} - \boldsymbol{r})) \tag{12}$$

where $R(\boldsymbol{q})$ is the rotation matrix defined by the quaternion $\boldsymbol{q}$. In addition, the pinhole projection $\pi([x, y, z])$ of any three-dimensional vector $[x, y, z]$ is a two-dimensional vector

$$[x/z, y/z] = \pi([x, y, z]) \tag{13}$$

In order to use a visual feature measurement to update the state estimate $\hat{\boldsymbol{x}}_c$, the unknown surface fixed point $\boldsymbol{z}$ must be either known or marginalized out. A common approach is to augment the filter state with the locations of all tracked landmarks and jointly estimate them with the original state [1]. With this approach, no knowledge of the observed surface is used and it is assumed that all measurements are corrupted by white Gaussian noise $\boldsymbol{\nu}$ of covariance Σ_ν as

$$\tilde{\boldsymbol{u}} = \boldsymbol{u} + \boldsymbol{\nu} \tag{14}$$

MAVeN uses the measurement in Eq. 14 but takes an alternative approach. It is assumed that a Digital Elevation MAP (DEM) of the surface is available *a priori*. The DEM defines the surface elevation over a grid of two-dimensional surface frame locations on a nominal ground plane. In addition, it is

© 2017 California Institute of Technology. Government sponsorship acknowledged.

assumed that the measurement in the Base Frame is noise-free. The first assumption is valid to some degree in the intended applications where reconnaissance data can be used to produce a DEM of some resolution and accuracy. The second assumption is motivated by the image processing architecture: the Base Frame defines the surface fixed point location and appearance. All subsequent Tracking Frames are compared to that definition. Under these assumptions, the surface fixed point $\boldsymbol{z}$ is strictly a function of the corresponding measurement in the Base Frame $\boldsymbol{u}$, and the camera position $\boldsymbol{r}$ and attitude $\boldsymbol{q}$ at the time of the Base Frame (conditioned on the DEM). Therefore an estimator can be designed that augments the state $\boldsymbol{x}_c$ with the position and attitude at the time of the Base Frame, $\boldsymbol{r}_b$ and $\boldsymbol{q}_b$, instead of the location $\boldsymbol{z}$ of each tracked feature. Because $n > 100$ features may be tracked at a time, this can reduce the augmented state dimension from $3n$ to 6. The sensitivity of the resulting estimator with respect to the assumption validity is investigated in the results section of this paper.

3.3 Augmented State

The *base state* is defined as

$$\boldsymbol{x}_b \equiv \begin{bmatrix} \boldsymbol{r}_b \\ \boldsymbol{q}_b \end{bmatrix} \in \mathbb{R}^7 \tag{15}$$

The incremental error in the base state is

$$\boldsymbol{e}_b \equiv \begin{bmatrix} \boldsymbol{r}_b - \hat{\boldsymbol{r}}_b \\ \Delta_q(\boldsymbol{q}_b, \hat{\boldsymbol{q}}_b) \end{bmatrix} \in \mathbb{R}^6 \tag{16}$$

which has covariance $P_{bb} = \mathrm{E}\left\{\boldsymbol{e}_b \boldsymbol{e}_b^{\mathrm{T}}\right\}$. The augmented state is a stack of the current and base state:

$$\boldsymbol{x}_a \equiv \begin{bmatrix} \boldsymbol{x}_c \\ \boldsymbol{x}_b \end{bmatrix} \in \mathbb{R}^{23} \tag{17}$$

The augmented state incremental error $\boldsymbol{e}_a$ is defined as

$$\boldsymbol{e}_a \equiv \begin{bmatrix} \boldsymbol{e}_c \\ \boldsymbol{e}_b \end{bmatrix} \in \mathbb{R}^{21} \tag{18}$$

The cross-covariance is $P_{cb} \equiv \mathrm{E}\left\{\boldsymbol{e}_c \boldsymbol{e}_b^{\mathrm{T}}\right\}$. Therefore the augmented covariance $P_{aa} \equiv \mathrm{E}\left\{\boldsymbol{e}_a \boldsymbol{e}_a^{\mathrm{T}}\right\}$ is

$$P_{aa} \equiv \begin{bmatrix} P_{cc} & P_{cb} \\ P_{cb}^{\mathrm{T}} & P_{bb} \end{bmatrix} \tag{19}$$

At the time of a Base Frame, the estimate $\hat{\boldsymbol{x}}_b$ is updated by copying the position and quaternion portion of $\hat{\boldsymbol{x}}_c$. The error immediately after this copy operation is then

$$\boldsymbol{e}_b = S\boldsymbol{e}_c \tag{20}$$

where S is a selection matrix to extract the error of the current camera position and attitude. It is defined as

$$S \equiv \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} \end{bmatrix} \tag{21}$$

© 2017 California Institute of Technology. Government sponsorship acknowledged.

From the definitions of P_{cb} and P_{bb}, the covariances are updated at Base Frame capture time as

$$P_{cb} = P_{cc} S^{\mathrm{T}} \tag{22}$$
$$P_{bb} = S P_{cc} S^{\mathrm{T}} \tag{23}$$

The full covariance P_{aa} must be propagated using its time derivative

$$\dot{P}_{aa} \equiv \begin{bmatrix} \dot{P}_{cc} & \dot{P}_{cb} \\ \dot{P}_{cb}^{\mathrm{T}} & \dot{P}_{bb} \end{bmatrix} \tag{24}$$

The derivative of P_{cc} is given in Eq. 11. Because the base state refers to a specific instant in time, the time derivative of $\boldsymbol{x}_b$ and hence $\boldsymbol{e}_b$ is zero. Therefore the time derivative of P_{bb} is zero:

$$\dot{P}_{bb} = 0_{6\times 6} \tag{25}$$

Using Eq. 8 and the definition of P_{cb}, an expression for $\dot{P}_{cb}$ can be found as

$$\dot{P}_{cb} = \mathrm{E}\left\{\boldsymbol{e}_c \boldsymbol{e}_b^{\mathrm{T}}\right\} \tag{26}$$
$$= \mathrm{E}\left\{(F_e \boldsymbol{e}_c + F_w \boldsymbol{w}) \boldsymbol{e}_b^{\mathrm{T}}\right\} \tag{27}$$
$$= F_e P_{cb} \tag{28}$$

3.4 Base Frame Update

When a Base Frame is captured, the augmentation described above is applied. Then, a *pseudo-landmark* estimate is created for each detected feature. Let the i'th detected feature have perfect measurement $\boldsymbol{u}_i$ as defined in Eq. 12. The corresponding estimated pseudo-landmark $\hat{\boldsymbol{z}}_i$ is computed by intersecting a ray originating from the camera location $\hat{\boldsymbol{r}}_b$ with the terrain. The ray direction is a function of the attitude estimate $\hat{\boldsymbol{q}}_b$ and the measurement $\tilde{\boldsymbol{u}}_i$. The derivative of this estimate with respect to the error $\boldsymbol{e}_b$ is

$$H_{b,i} \equiv \frac{\partial \hat{\boldsymbol{z}}_i}{\partial \boldsymbol{e}_b} \tag{29}$$

which can be computed with the use of the surface normal extracted from the DEM.

3.5 Tracking Frame Update

The Tracking Frame update takes place in two steps. The first step compresses n feature measurements into a lower-dimensional measurement using a QR decomposition [3][4]. The second step processes the lower-dimensional measurement in a standard Extended Kalman Filter (EKF) update.

For the first step, let the i'th tracked feature have measurement $\tilde{\boldsymbol{u}}_i$ as defined in Eq. 14. The predicted measurement is found by substituting the estimate into Eq. 12 as

$$\hat{\boldsymbol{u}}_i = \pi(R(\hat{\boldsymbol{q}})(\hat{\boldsymbol{z}}_i - \hat{\boldsymbol{r}})) \tag{30}$$

© 2017 California Institute of Technology. Government sponsorship acknowledged.

Define the Cholesky square root $\Sigma_\nu^{1/2}$ such that $\Sigma_\nu = \Sigma_\nu^{1/2}\left(\Sigma_\nu^{1/2}\right)^{\mathrm{T}}$. The pre-processed innovation is defined as

$$\boldsymbol{\epsilon}_i \equiv \Sigma_\nu^{-1/2}\left(\tilde{\boldsymbol{u}}_i - \hat{\boldsymbol{u}}_i\right) \tag{31}$$

where $\Sigma_\nu^{-1/2}$ is the inverse of $\Sigma_\nu^{1/2}$. Expressing $\boldsymbol{\epsilon}_i$ in terms of the estimate $\hat{\boldsymbol{x}}_a$ and incremental error $\boldsymbol{e}_a$, and linearizing with respect to $\boldsymbol{e}_a$ about the current estimate and error $\boldsymbol{e}_c = 0$ yields

$$\boldsymbol{\epsilon}_i \approx H_{a,i}\boldsymbol{e}_a + \bar{\boldsymbol{\nu}}_i \tag{32}$$

where

$$H_{a,i} \equiv \Sigma_\nu^{-1/2}\frac{\partial \boldsymbol{\epsilon}_i}{\partial \boldsymbol{e}_a} \tag{33}$$

$$= \begin{bmatrix} H_{r,i} & 0_{3\times 6} & H_{q,i} & 0_{3\times 3} & \frac{\partial \boldsymbol{\epsilon}_i}{\partial \hat{\boldsymbol{z}}_i}H_{b,i} \end{bmatrix} \tag{34}$$

$$\bar{\boldsymbol{\nu}}_i \equiv \Sigma_\nu^{-1/2}\boldsymbol{\nu}_i \tag{35}$$

Here, $H_{r,i}$ and $H_{q,i}$ are the Jacobians of $\boldsymbol{\epsilon}_i$ with respect to the position and attitude portion of $\boldsymbol{e}_c$ respectively. Define a 21×12 orthogonal permutation matrix E such that

$$\mathcal{H}_{a,i} = H_{a,i}E \tag{36}$$

$$= \begin{bmatrix} H_{r,i} & H_{q,i} & \frac{\partial \boldsymbol{\epsilon}_i}{\partial \hat{\boldsymbol{z}}_i}H_{b,i} \end{bmatrix} \tag{37}$$

Stack the pre-processed innovations $\boldsymbol{\epsilon}_i$, $\mathcal{H}_{a,i}$, $H_{a,i}$ and $\bar{\boldsymbol{\nu}}_i$ to form

$$\boldsymbol{\epsilon} \equiv \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \\ \vdots \\ \boldsymbol{\epsilon}_n \end{bmatrix} \quad \mathcal{H}_a \equiv \begin{bmatrix} \mathcal{H}_{a,1} \\ \mathcal{H}_{a,2} \\ \vdots \\ \mathcal{H}_{a,n} \end{bmatrix} \quad H_a \equiv \begin{bmatrix} H_{a,1} \\ H_{a,2} \\ \vdots \\ H_{a,n} \end{bmatrix} \quad \bar{\boldsymbol{\nu}} \equiv \begin{bmatrix} \bar{\boldsymbol{\nu}}_1 \\ \bar{\boldsymbol{\nu}}_2 \\ \vdots \\ \bar{\boldsymbol{\nu}}_n \end{bmatrix} \tag{38}$$

and note that $\mathcal{H}_a = H_aE$. Also note that from the definition of $\bar{\boldsymbol{\nu}}_i$ and with the assumption that $\mathrm{E}\left\{\boldsymbol{\nu}_i\boldsymbol{\nu}_j\right\} = 0_{2\times 2}\ \forall i \neq j$, the covariance of $\bar{\boldsymbol{\nu}}$ is $I_{2n\times 2n}$.

Now perform a QR decomposition of $\mathcal{H}_a$ so that

$$\mathcal{H}_a = [Q_1\ Q_2]\begin{bmatrix} R_1 \\ 0_{2n-r\times 12} \end{bmatrix} \tag{39}$$

where r is the rank of $\mathcal{H}_a$, $[Q_1\ Q_2]$ is orthogonal, and R_1 is a $r \times 12$ upper-triangular matrix. In practice, only Q_1 and R_1 need to be computed which can save computation. For general camera motion over a ground plane, $r = 8$ which captures the fact that the absolute position and absolute heading are unobservable (change in position and attitude as well as absolute roll and pitch are observable).

Starting from Eq. 31 and the definitions in Eq. 38 and using the fact that $EE^{\mathrm{T}} = I_{12\times 12}$ yields

$$\boldsymbol{\epsilon} = H_a\boldsymbol{e}_a + \bar{\boldsymbol{\nu}} \tag{40}$$

$$= H_aEE^{\mathrm{T}}\boldsymbol{e}_a + \bar{\boldsymbol{\nu}} \tag{41}$$

$$= \mathcal{H}_aE^{\mathrm{T}}\boldsymbol{e}_a + \bar{\boldsymbol{\nu}} \tag{42}$$

$$= [Q_1\ Q_2]\begin{bmatrix} R_1 \\ 0_{2n-r\times 12} \end{bmatrix}E^{\mathrm{T}}\boldsymbol{e}_a + \bar{\boldsymbol{\nu}} \tag{43}$$

© 2017 California Institute of Technology. Government sponsorship acknowledged.

Left multiplying by Q_1^{T}, and rearranging yields

$$Q_1^{\mathrm{T}}\boldsymbol{\epsilon} = R_1 E^{\mathrm{T}}\boldsymbol{e}_a + Q_1^{\mathrm{T}}\bar{\boldsymbol{\nu}} \tag{44}$$

$$\tilde{\boldsymbol{y}} = C\boldsymbol{e}_a + \tilde{\boldsymbol{\nu}} \tag{45}$$

where $\tilde{\boldsymbol{y}} \equiv Q_1^{\mathrm{T}}\boldsymbol{\epsilon}$, $C \equiv R_1 E^{\mathrm{T}}$ and $\tilde{\boldsymbol{\nu}} \equiv Q_1^{\mathrm{T}}\bar{\boldsymbol{\nu}}$. Note that $\tilde{\boldsymbol{\nu}}$ has covariance $I_{r\times r}$ due to the orthogonality of Q_1.

The matrix C and the vector $\tilde{\boldsymbol{y}}$ are then used in a standard EKF update as follows. The Kalman gain and measurement update are

$$K = P_{aa}C^{\mathrm{T}}\left(CP_{aa}C^{\mathrm{T}} + I_{r\times r}\right)^{-1} \tag{46}$$

$$P_{aa} \leftarrow (I - KC)P_{aa}(I - KC)^{\mathrm{T}} + KK^{\mathrm{T}} \tag{47}$$

$$\hat{\boldsymbol{x}}_a \leftarrow \mathrm{g}_a(\hat{\boldsymbol{x}}_a, K\tilde{\boldsymbol{y}}) \tag{48}$$

where the function $\mathrm{g}_a(\hat{\boldsymbol{x}}_a, \hat{\boldsymbol{e}}_a)$ applies a 21-dimensional estimated incremental correction $\hat{\boldsymbol{e}}_a = K\tilde{\boldsymbol{y}}$ to update the full 23-dimensional state estimate $\hat{\boldsymbol{x}}_a$. Specifically, the function $\mathrm{g}_a(\hat{\boldsymbol{x}}_a, \hat{\boldsymbol{e}}_a)$ applies vector addition to position, velocity, and biases, and applies a small rotation to the current and base quaternion.

3.6 Laser Range Finder Update

The Laser Range Finder (LRF) measures the distance along a body-frame axis to the surface. The DEM can be used to find a predicted measurement, based on the current estimate, and the surface normal at the predicted surface intersection. This measurement can then be processed in the standard EKF formulation. Accurate altitude information from an LRF or external source is needed to initialize the scale of the problem.

3.7 Translation Only Estimation

For many space applications, accurate inertial attitude knowledge of the vehicle is available from star trackers and high quality angular rate gyroscopes. Furthermore, for planetary landings and small body proximity operations, the inertial attitude of the target body (and hence surface frame) are well known. In these applications, unmodeled errors in the assumed DEM can potentially degrade the inertial attitude estimate when visual features are used to update the state estimate. In these applications, it may be preferable to use the inertial attitude knowledge from alternative sensors and only estimate translation states. This is a special case of MAVeN. In particular, the current and augmented state vector are redefined as

$$\boldsymbol{x}_c = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \\ \boldsymbol{\beta}_a \end{bmatrix} \in \mathbb{R}^9 \qquad \boldsymbol{x}_b = \begin{bmatrix} \boldsymbol{r}_b \end{bmatrix} \in \mathbb{R}^3 \tag{49}$$

The corresponding incremental error vectors are now

$$\boldsymbol{e}_c = \begin{bmatrix} \boldsymbol{r} - \hat{\boldsymbol{r}} \\ \boldsymbol{v} - \hat{\boldsymbol{v}} \\ \boldsymbol{\beta}_a - \hat{\boldsymbol{\beta}}_a \end{bmatrix} \in \mathbb{R}^9 \qquad \boldsymbol{e}_b = \begin{bmatrix} \boldsymbol{r}_b - \hat{\boldsymbol{r}}_b \end{bmatrix} \in \mathbb{R}^3 \tag{50}$$

© 2017 California Institute of Technology. Government sponsorship acknowledged.

This alters the covariance matrices to be $P_{cc} \in \mathbb{R}^{9\times 9}$, $P_{cb} \in \mathbb{R}^{9\times 3}$, and $P_{bb} \in \mathbb{R}^{3\times 3}$. The Jacobian in Eq. 34 reduces to

$$\begin{bmatrix} H_{r,i} & 0_{3\times 6} & \frac{\partial \epsilon_i}{\partial \hat{z}_i} H_{b,i} \end{bmatrix} \tag{51}$$

where $H_{b,i}$ is now a 3×3 matrix as opposed to the original 3×6 matrix. The resulting $\mathcal{H}_{a,i}$ matrix of Eq. 36 becomes 2×6 where E is 6×15. Performing the QR decomposition on the full $2n \times 6$ $\mathcal{H}_a$ matrix yields a 3×3 upper triangular R_1 and a 3×3 orthogonal Q_1. Finally, the compressed measurement $\tilde{\boldsymbol{y}} \in \mathcal{R}^3$, Jacobian $C \in \mathcal{R}^{3\times 9}$, and covariance $I_{3\times 3}$ are processed in the standard EKF update. In this case, Eq. 48 can be written explicitly as

$$\hat{\boldsymbol{x}}_a \leftarrow \hat{\boldsymbol{x}}_a + K\tilde{\boldsymbol{y}} \tag{52}$$

4 APPLICATION: LANDING

This section presents MAVeN performance in a small body landing application. In this class of application, the motion is a vertical descent from a high altitude to the surface with near-constant attitude. In addition, attitude knowledge is expected to be highly accurate ($< 1°$ relative to the surface). The details of this test are given below.

The test simulations have several important parameters. The IMU is parameterized by Q_{g1}, Q_{g2}, Q_{a1}, and Q_{a2} which are the angle random walk, gyro rate random walk, velocity random walk, and acceleration random walk. All visual feature measurements are corrupted by an error with covariance $\Sigma_\nu = \sigma_\nu^2 I_{2\times 2}$. The laser-range finder has error with variance σ_{lrf}^2. The time intervals between images and range measurements are Δt_{img} and Δt_{lrf} respectively. The initial velocity, acceleration bias, gyroscope bias, and attitude errors have covariance $\sigma_v^2 I_{3\times 3}$, $\sigma_{ab}^2 I_{3\times 3}$, $\sigma_{gb}^2 I_{3\times 3}$, and $\sigma_\theta^2 I_{3\times 3}$ respectively. The IMU is operated at 100 Hz in both applications. The simulation parameter values are given in Table 1.

Table 1: Simulation parameters for descent and helicopter flight.

Parameter	Value	Units
Q_{a1}	7.2e-06	m^2/s^3
Q_{a2}	1.1e-08	m^2/s^5
Q_{g1}	1.9e-11	rad^2/s
Q_{g2}	2.9e-12	rad^2/s^3
σ_ν	2.05e-03	rad
Δt_{img}	2.0	s
σ_{lrf}	5.0e-01	m
Δt_{lrf}	0.2	s
σ_v	0.0e+00	m/s
σ_θ	2.9e-03	rad
σ_{ab}	6.4e-04	m/s^2
σ_{gb}	3.3e-05	rad/s

© 2017 California Institute of Technology. Government sponsorship acknowledged.

In this test, the estimator assumes that the terrain is flat while the true terrain model is given by

$$h(x,y) = \sum_{i=1}^{N} A_i \sin(\omega_i x + \phi_{xi}) + A_i \sin(\omega_i y + \phi_{yi}) \tag{53}$$

where $h(x,y)$ is the elevation above a nominal ground plane at a two-dimensional location (x,y) on a nominal ground plane. Five test scenarios are given where the amplitudes A_i are varied to show the sensitivity of the algorithm to errors in the terrain assumption. An example realization of the terrain for each of the five cases is shown in Fig. 3. Note that Fig. 3 shows a single one axis cross-section of the terrain which is defined over a two-dimensional grid. In addition, because the terrain is assumed by the estimator to be flat (zero elevation), the terrain elevation is equal to the terrain elevation error.

For each of the five test scenarios, ten simulations are run to assess performance. The phase ϕ_{xi} and ϕ_{yi} are chosen as uniform random between 0 and 2π for each simulation. For each simulation, two variants of MAVeN are used to process the data: one with attitude states, referred to as MAVEN-AT, and one without attitude states, referred to as MAVEN-T (translation only). A comparison between the two variants of MAVeN is made.

The simulated trajectory in this application, for all cases, is a vertical descent from an altitude of $1000\ \mathrm{m}$ to $10\ \mathrm{m}$ with constant attitude. The initial velocity is $20\ \mathrm{m/s}$ towards the surface and the deacceleration is constant such that the final velocity is zero.

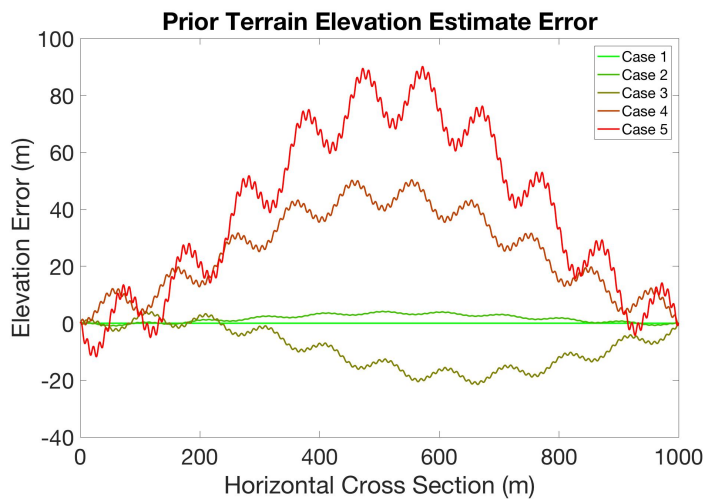

Figure 3: Instances of random terrain elevation error for five terrain cases.

Estimate error time-series for one trial of each of the five cases is shown in Fig. 4 and Fig. 5. This one trial for each case corresponds to the terrain shown in Fig. 3. The number of features tracked is also shown in these figures. MAVEN-AT is used to obtain the results of Fig. 4 while MAVEN-T is used to obtain the results of Fig. 5. Several trends should be noted. Clearly, the performance degrades as the terrain assumption degrades (more terrain elevation error) for MAVEN-AT and MAVEN-T. For both variants, the performance degradation is minimal at the end of the simulation when velocities tend to zero as can be seen in Table 2. This is especially true for MAVEN-T.

© 2017 California Institute of Technology. Government sponsorship acknowledged.

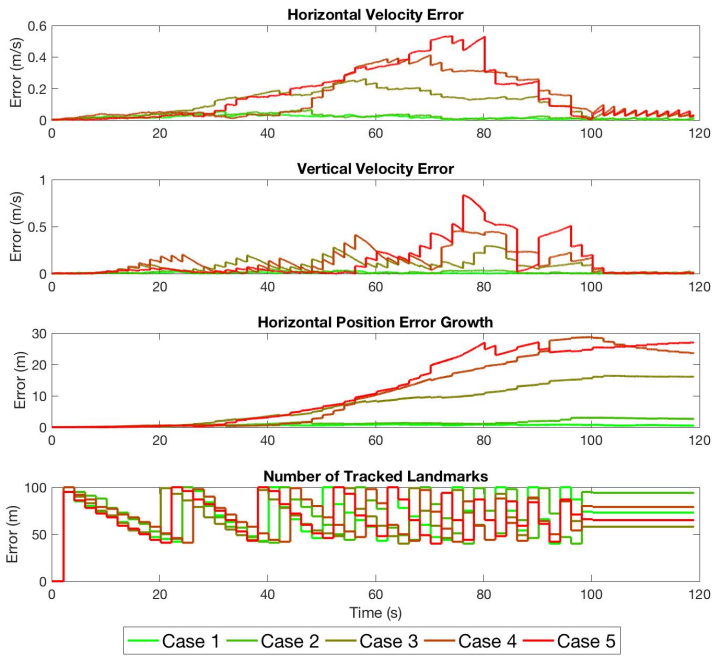

Figure 4: Performance results of MAVeN-AT for one trial on each of the five terrains in Fig. 3.

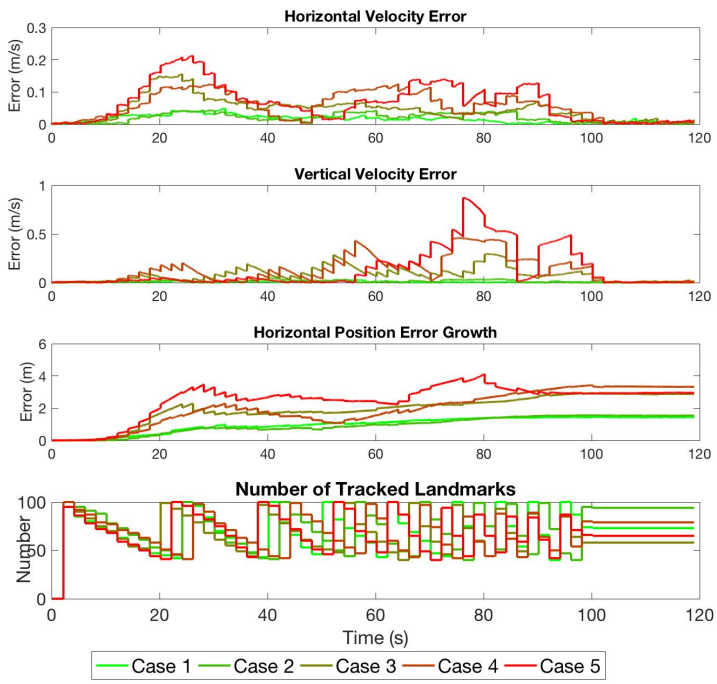

Figure 5: Performance results of MAVeN-T for one trial on each of the five terrains in Fig. 3.

© 2017 California Institute of Technology. Government sponsorship acknowledged.

Table 2 gives the mean error magnitude at the end (final time) of each run. In Case 1, for which the terrain knowledge assumption is perfect, MAVEN-AT and MAVEN-T have similar velocity errors at the final time. However, the final horizontal position error in Table 2 is much smaller for MAVEN-AT because MAVEN-AT maintained a better velocity estimate throughout the trajectory. On the other hand, MAVEN-T is much less sensitive to a degrading terrain assumption. This presents an interesting trade between accuracy at the assumed terrain model and robustness to deviations from the assumed terrain model.

The reason for the sensitivity of MAVEN-AT is that by assuming that the terrain is flat, the estimator attempts to align the attitude with a best-fit observed ground plane. The observed ground plane is not aligned with gravity and changes as the image footprint changes. This corrupts the accelerometer biases which then corrupts the velocity estimates. In Case 3, 4, and 5, the terrain slopes are much larger than the gyro-only attitude knowledge (see Table 1) which gives MAVEN-T an advantage. As a rule-of-thumb, MAVEN-AT will perform better when attitude knowledge is poor and terrain is well known, and vice-versa.

Table 2: Mean (over ten trials of each case) error magnitudes ($\frac{1}{10}\sum_{i=1}^{10}|e_i|$ for error e_i) at the final time.

Case	**Horizontal Velocity Error**		**Vertical Velocity Error**		**Horizontal Position Error**	
	MAVEN-AT	**MAVEN-T**	**MAVEN-AT**	**MAVEN-T**	**MAVEN-AT**	**MAVEN-T**
Case 1	0.71 cm/s	0.73 cm/s	0.30 cm/s	0.30 cm/s	0.57 m	3.39 m
Case 2	0.58 cm/s	0.46 cm/s	0.34 cm/s	0.34 cm/s	2.35 m	2.58 m
Case 3	1.73 cm/s	0.49 cm/s	0.44 cm/s	0.40 cm/s	10.41 m	2.48 m
Case 4	2.49 cm/s	0.60 cm/s	0.47 cm/s	0.48 cm/s	19.07 m	3.20 m
Case 5	4.62 cm/s	0.48 cm/s	0.44 cm/s	0.38 cm/s	32.85 m	8.70 m

5 CONCLUSIONS

This paper introduces the MAVeN algorithm for autonomous vision-based navigation. The main contribution is that the navigation filter has a much lower dimension than is typically required by other vision-based navigation approaches. This makes MAVeN a very reliable and practical algorithm for real-time flight implementation. Because MAVeN does not require pre-mapped landmarks, it has the advantage of being able to navigate terrain that has not been previously seen. MAVeN is also able to maintain a stable motionless hover above the ground surface without position error growth. While critical to the most basic of spacecraft exploration scenarios, this same hovering capability is suprisingly difficult to achieve for other competing approaches from the literature, and typically requires going to filters with a much larger number of states. MAVeN is currently being considered to support emerging NASA missions such as Europa Lander and Mars Helicopter (a payload on Mars 2020).

© 2017 California Institute of Technology. Government sponsorship acknowledged.

6 ACKNOWLEDGMENTS

The authors would like to thank Carl Seubert, Seth Aaron, Andrew Johnson, Nick Trawny, Geoffrey Vaughn, Jim Bodie, and Mike McHenry of JPL for technical discussions and experimental support during this development. This research was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

[1] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, 2007.

[2] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *arXiv preprint arXiv:1304.3111*, 2013.

[3] D. S. Bayard and P. B. Brugarolas, "On-board vision-based spacecraft estimation algorithm for small body exploration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 1, 2008.

[4] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Robotics and Automation, 2007 IEEE International Conference*. IEEE, 2007, pp. 3565–3572.

[5] B. Williams, N. Hudson, B. Tweddle, R. Brockers, and L. Matthies, "Feature and pose constrained visual aided inertial navigation for computationally constrained aerial vehicles," in *Robotics and Automation (ICRA), 2011 IEEE International Conference*. IEEE, 2011, pp. 431–438.

[6] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 1788–1795.

[7] D. G. Kottas, K. J. Wu, and S. I. Roumeliotis, "Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference*. IEEE, 2013, pp. 3172–3179.

[8] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments," in *Robotics and Automation (ICRA), 2012 IEEE International Conference*. IEEE, 2012, pp. 957–964.

[9] M. Li and A. I. Mourikis, "Vision-aided inertial navigation for resource-constrained systems," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference*. IEEE, 2012, pp. 1057–1063.

[10] J. L. Crassidis and J. L. Junkins, *Optimal estimation of dynamic systems*. CRC press, 2011.

© 2017 California Institute of Technology. Government sponsorship acknowledged.