PROXIMITY NAVIGATION OF HIGHLY CONSTRAINED SPACECRAFT

S. Scarritt, M. Swartwout
Washington University in St. Louis

Bandit is a 3-kg automated spacecraft in development at Washington University in St. Louis. Bandit's primary mission is to demonstrate proximity navigation, including docking, around a 25-kg student-built host spacecraft. However, because of extreme constraints in mass, power and volume, traditional sensing and actuation methods are not available. In particular, Bandit carries only 8 fixed-magnitude cold-gas thrusters to control its 6 DOF motion. Bandit lacks true inertial sensing, and the ability to sense position relative to the host has error bounds that approach the size of the Bandit itself.

Some of the navigation problems are addressed through an extremely robust, error-tolerant soft dock. In addition, we have identified a control methodology that performs well in this constrained environment: behavior-based velocity potential functions, which use a minimum-seeking method similar to Lyapunov functions. We have also adapted the discrete Kalman filter for use on Bandit for position estimation and have developed a similar measurement vs. propagation weighting algorithm for attitude estimation.

This paper provides an overview of Bandit and describes the control and estimation approach. Results using our 6DOF flight simulator are provided, demonstrating that these methods show promise for flight use.

INTRODUCTION

Close proximity operations between spacecraft is an area of growing interest within the space community. This technology, defined for this paper as the navigation of one spacecraft within 100 meters of another, including docking, allows for several useful activities, including inspection or surveillance of a target vehicle, on-orbit assembly or repair of the target vehicle, protection of a target from space debris or other vehicles, or even transfer of fuel, power, data, or other resources. Very small spacecraft – under 20 kg – are well-suited for these types of proximity activities. They are easy to maneuver, inexpensive to build and operate, and have a minimal mass cost for launch. However, their advantages are offset somewhat by the constraints of their small size: they are limited in the amount of power they can generate and store, they cannot carry traditional precision navigations systems, and they do not have sufficient power or antenna gain to establish high-data-rate ground communication. These constraints pose a significant navigation challenge.

Recent and current proximity operations programs include Surrey Satellite's SNAP-1, MIT's SPHERES [1,2], AFRL's ANGELS project, and several others. While these projects cover an impressive breadth, there is still significant room for research in the aspects of very close proximity operations (under 10 m, including docking), multi-vehicle operations, extremely small vehicles (under 10 kg), and responsiveness. Especially interesting is the combination of several (or all) of these elements. The intent of Washington University's Bandit-C project is to increase the body of research in this area. The Bandit mission concept addresses the constraints of small spacecraft operation by decoupling orbital functions between the 3 kg Bandit drone and larger (25 kg) dedicated host vehicle. The host vehicle, Akoya, will be responsible for all ground communication and power generation, enabling Bandit to be stripped to its essentials: imaging, short-term power, short-range communications and navigation. In addition, behavior-based potential functions have shown themselves to be promising control methods for Bandit due to their insensitivity to uncertainties, disturbances, and errors in the system model.

In this paper, we will provide an overview of the Bandit mission concept and design. We will also describe the behavior-based potential function controller and outline the reinforcement learning method. Results from our 6DOF simulation testbed for the potential function approach will be provided, showing the good performance of this method despite external disturbances and unmodeled variances in the actuators. Finally, we will conclude by outlining a plan for future development work.

BANDIT SYSTEM

Design Concept

The impetus for the overall Bandit design concept comes from the distinction of orbital functions as either "short-period" or "long-period." Short-period functions last only minutes to hours and include most servicing functions. These functions require short-range maneuverability, the ability to avoid collisions, and effective sensing and actuation. Long-period functions sustain a spacecraft for months or even years (justifying the cost of manufacture and launch): power generation and storage, ground communications, momentum management, orbit maintenance (and possibly transfer), and so on. Often, these two types of functions undermine each other – for example, the size needed to perform long-period functions reduces the maneuverability of the vehicle. Bandit seeks to eliminate this conflict by decoupling these functions; short-period functions are performed by the 3 kg reusable service drones, while the mass- and power-intensive long-period functions are assigned to the larger host vehicle. The host releases the drone or drones for brief servicing excursions, and the drones then re-dock with the host to recharge their batteries or refuel. This design philosophy allows for a minimalist service drone that carries nothing more than is absolutely essential for its survival. Bandit requires only 6DOF propulsion, short-term power, and a short-range communication system for communicating with the host vehicle. Thus, the drone can be kept small, agile, and inexpensive [3]. This approach enables the deployment of many drones (even dozens) from the same host.

The enabling technology for this decoupling of functions is Bandit's ability to re-dock with its host vehicle. Because the two vehicles are only separated for a few hours at most, orbital instabilities such as atmospheric drag, relative orbital mechanics, and sensor drift do not have the chance to build up. Bandit begins and ends every sortie attached to the host, and so does not face the problems of target acquisition

and rendezvous. Using low-thrust actuators and a highly error-tolerant dock, described below, diminishes the effects of docking errors, further simplifying the navigation problem.

System Design

Flight operations for Bandit-C are constrained to take place only when the host vehicle is in line-of-sight contact of Washington University's ground station in St. Louis, meaning that initial drone sorties must take less than four minutes from release to re-dock. Minimum mission success requires that a single Bandit drone be activated, released from dock, and successfully re-docked. If this is achieved, the subsequent sorties will be incrementally more complex, backing the drone further from the dock over sorties lasting longer than four minutes and eventually performing a full inspection orbit of the host using only Bandit's on-board navigation.

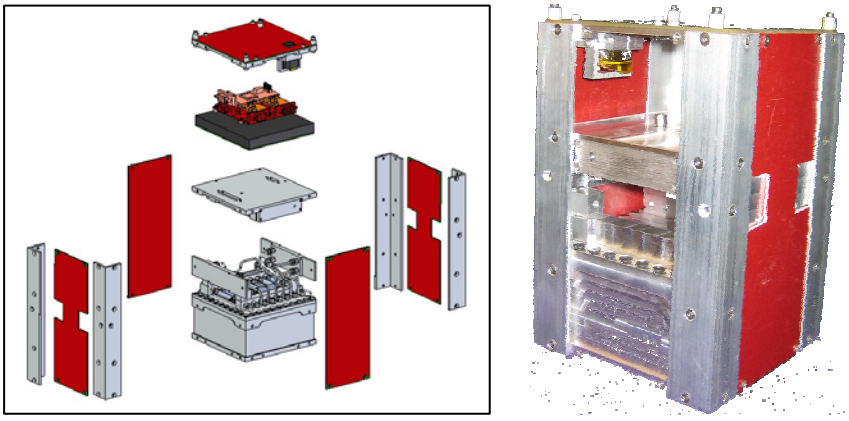

Figure 1: Exploded Schematic of Bandit and Engineering Demonstration Unit

The Bandit drone consists of an integrated propellant tank and electronics box; an exploded view is shown in Figure 1. The cold-gas propulsion system consists of eight thrusters arranged to provide decoupled translation or rotation about any single axis; the number of thrusters was limited by the size of the valves (which were themselves limited by program budget).

Bandit is navigated through a combination of inertial and vision sensors. MEMS accelerometers and rate gyros provide high-bandwidth information about Bandit's motion, although sensitivity limitations in the accelerometers may render Bandit's small translational accelerations indistinguishable from sensor noise. An on-board vision system converts images of the host to relative position and attitude information at a slower frame rate, providing a way to calibrate the inertial sensors. The exterior of the host vehicle is instrumented with color-coded LEDs to aid the image-based navigation solution, and each Bandit vehicle is also outfitted with LEDs for the host-mounted vision system. We expect to process images at speeds up to 30 frames per second, but at present the processing speed is only 1 frame approximately every 2-4 seconds.

It is worth repeating that the Bandit mission concept itself is a significant aid to navigation; service vehicles start and end each sortie atop their docks, service vehicles are designed to stay within 5 meters of the host vehicle, and the typical sortie lasts less than 15 minutes. Over those relative distances and time scales, the primary error sources (sensor noise, thruster calibration errors and relative orbital dynamics) build up to only a few centimeters, and the docking system is tolerant to a few centimeters' position error.

Bandit's docking mechanism is a hook-and-loop-fastener pushrod. The pushrod tip is a 2 cm diameter hemisphere with the hooks; the entire surface of Bandit is arrayed with squares of the loops. This arrangement gives Bandit tremendous margin in docking angle and offset; in air-bearing tabletop test, we have shown consistent, successful docking for approach angles of up to 75° and offsets of up to 3 cm. The hook-and-loop fastener acts as both an impact damper and a low-speed snag; again demonstrated for relative closing velocities as high as 100 cm/s and as low as 0.5 cm/s.

We have ground-tested Bandit flight systems in three ways, including a NASA C-9 microgravity experiment. Testbeds on the Washington University campus include a 3DOF air-bearing sled atop a precision granite surface and a Java-based 6DOF dynamic simulator that models orbital mechanics, vehicle translation and rotation, propulsion, and local disturbances.

POTENTIAL FUNCTION CONTROL

The use of a finite number of fixed-impulse actuators limits the applicability of published potential function control methods, which assume actuators with infinitely-variable magnitude and/or direction. Therefore, a specialized approach has been developed [4] in which the potential function, Φ, is weighted sum of generalized velocity errors, rather than position error (Eq. 1); firing thruster set j causes instantaneous changes in the generalized velocities and thus an instantaneous change in the potential, $\Delta\Phi_j$ (Eq. 2).

$$\Phi = \left(\mathbf{w} - \mathbf{w}_d\right)^T N^T N\left(\mathbf{w} - \mathbf{w}_d\right) = \left(N\mathbf{w}_e\right)^T N\mathbf{w}_e \tag{1}$$

$$\Delta\Phi_j = 2\left(N\mathbf{w}_e\right)^T\left(N\Delta\mathbf{w}_j\right) + \left(N\Delta\mathbf{w}_j\right)^T\left(N\Delta\mathbf{w}_j\right) \tag{2}$$

Defining the weighted impulse vector created from firing thruster set j (Eq. 3) results in a simplified form of Eq. 2:

$$\mathbf{T}_j = N\Delta\mathbf{w}_j = T_j\hat{\mathbf{t}}_j \tag{3}$$

$$\Delta\Phi_j = 2\left(N\mathbf{w}_e\right)^T\mathbf{T}_j + T_j^2 \tag{4}$$

Control actions are selected by evaluating a triggering function (Eq. 5), which weighs the benefit of each impulse set against a cost function σ_j , which takes into account factors such as the propellant and power cost of firing the thruster as well as plume impingement from thrusters facing the target vehicle. We consider σ_j to be a positive constant, indicative of the relative cost of firing thruster set j. The thruster option which satisfies and minimizes Eq. 5 is selected; if no thruster set can satisfy this relation, no control action is taken.

$$2\left(N\mathbf{w}_e\right)^T\mathbf{T}_j \le -T_j^2 - \sigma_j \tag{5}$$

Scenarios for Firing Thrusters

Recognizing that the right-hand side of Eq. 5 is always negative, there are two cases for which a thruster will not be fired. The first case is when the residual error is smaller than the control authority of the impulsive thrusters and/or the cost of firing a thruster (the "deadband threshold," described in References 4 and 5). The second case is when the available impulses do not span the control space, and thus there exist a range of weighted velocity errors for which the left-hand side is always positive.

This scenario is avoided by the proper thruster configuration. The controllability of a set of thruster configurations can be measured by comparing the scaled dot product of each configuration with an arbitrary unit error vector. For the thruster option that gives the best result, we then find the worst value across all unit error vectors. The result is a degree of controllability metric, α, with a range of -1 to 1:

$$\alpha = -\max_{\hat{w}_e}\left[\min_j\left(\frac{\hat{w}_e{}^T\,\mathbf{T}_j}{T_j}\right)\right] \tag{6}$$

A positive value of α indicates that there exists at least one thruster configuration to reduce the magnitude of an arbitrary generalized velocity error; if α is negative, then the thruster configurations do not span the error space, and the system is not controllable. Furthermore, the magnitude of α indicates how well the worst-case error configuration can be managed; a value of 1 indicates that the thruster

configurations have uniform effect on the weighted generalized errors, whereas a value near 0 indicates that there are one or more directions over which the thrusters have only marginal impact.

Potential Function Primitives

For translational control, we create a desired velocity opposite to the error vector:

$$\mathbf{v}_d = -f(\mathbf{r}, \mathbf{r}_d)\hat{\mathbf{r}}_e \tag{7}$$

Where $-f(\mathbf{r}, \mathbf{r}_d)$ is always non-negative. Similarly, for rotational control, we define a desired look vector, $\hat{\mathbf{n}}_d$, and regulate angular velocity to align some body axis, $\hat{\mathbf{n}}$, with that chosen direction.

$$\boldsymbol{\omega}_d = -g(\hat{\mathbf{n}}, \hat{\mathbf{n}}_d)\left(\frac{\hat{\mathbf{n}}_d \times \hat{\mathbf{n}}}{|\hat{\mathbf{n}}_d \times \hat{\mathbf{n}}|}\right) \tag{6}$$

We can now define primitives in the forms of Eqs. 7 and 8 which can be combined to create complex behaviors. We first examine a basic attitude-control potential that aligns a spacecraft camera with a desired vector; because the image would be useful regardless of roll about the desired direction, this angle is not constrained, but the roll rate is controlled. The desired angular velocity will align the body axis $\hat{\mathbf{n}}$ with the desired axis $\hat{\mathbf{n}}_d$, with a magnitude that is proportional to error when errors are small, but has a maximum allowed angular velocity $\mu_\omega$to reduce propellant consumption for large errors:

$$\omega_{d,CAMERA} = \mu_\omega \sqrt{\frac{1-\hat{\mathbf{n}}\cdot\hat{\mathbf{n}}_d}{2}}\left(\frac{\hat{\mathbf{n}}\times\hat{\mathbf{n}}_d}{|\hat{\mathbf{n}}\times\hat{\mathbf{n}}_d|}\right) \tag{9}$$

Our first translational primitive maintains a set separation distance between the vehicle and a target, regardless of the relative position vectors. In this primitive, the unit position error $\hat{\mathbf{r}}_e$ is the relative position vector $\hat{\mathbf{r}}$. As with the rotational primitive, the desired velocity is proportional to error magnitude for small errors, and μ_{SEP} for large errors.

$$\mathbf{v}_{d,SEP} = \mu_{SEP}\frac{5(r_{SEP}-r)}{r_{SEP}+5|r_{SEP}-r|}\hat{\mathbf{r}} \tag{10}$$

Eq. 10 is not concerned with the specific directions of the actual and desired position vectors. For other situations, we may want to adjust velocity and position, $\mathbf{r}_d$. We first define ϕ as the angle between the actual and desired radius vectors.

$$\phi = \cos^{-1}\left(\frac{\mathbf{r}\cdot\mathbf{r}_d}{rr_d}\right) \tag{11}$$

For close-up inspection or docking, the vehicle must maintain a safe distance from the target except for the region near the dock/imaging target. For that situation, we define $\mathbf{r}_d$ in the direction of the region of interest with magnitude at the desired separation distance.

$$\mathbf{v}_{d,SEP2} = \mu_{SEP2}\frac{5(r_{SEP}-r)}{r_{SEP}+5|r_{SEP}-r|}\left(1-e^{-\lambda_{sep2}\phi}\right)\hat{\mathbf{r}} \tag{12}$$

where the maximum translation speed during collision avoidance is μ_{SEP2} and the decay constant λ_{SEP2} determines the width of the safe zone for proximity operations. Similarly, to approach a target along a specific direction (such as docking), we want a complementary function to drive the vehicle to the dock while in the zone of interest, but have no effect outside the zone:

$$\mathbf{v}_{d,APP} = \mu_{APP}e^{-\lambda_{APP}\phi}\hat{\mathbf{r}}_e \tag{13}$$

Note that on perfect approach, the closing velocity between vehicle and dock is μ_{APP}, the velocity does not diminish with reduced error because of the need for finite relative velocity in docking.

In addition to these radial controls, a circumferential primitive with maximum velocity μ_{CIRC} is shown in Eq. 14. This primitive has the effect of directing the vehicle around the target onto to a specified line emanating from the target; separation distance is not regulated.

$$\mathbf{v}_{d,CIRC} = \frac{2\mu_{CIRC}\phi}{1+(\phi/\pi)}\left(\frac{(\mathbf{r}\times\mathbf{r}_d)\times\mathbf{r}}{|(\mathbf{r}\times\mathbf{r}_d)\times\mathbf{r}|}\right) \tag{14}$$

Building Behaviors from Primitives: Docking

A docking potential is achieved by combining the docking approach function (Eq. 13) with circumferential control (Eq. 14) and the separation potential with zone-of-interest (Eq. 12).

$$\mathbf{v}_{d,DOCK} = \mathbf{v}_{d,SEP2} + \mathbf{v}_{d,CIRC} + \mathbf{v}_{d,APP} \tag{15}$$

As shown in Figure 2, each of these primitives has a region of space for which they dominate the potential function; the net effect is to cause the drone to retreat to a safe distance, circulate around the host and follow a predefined approach to dock with a finite closing speed. This simple example creates a potential function with a spherical keep-out zone, but it would be straightforward to add potentials to define more complex shapes (e.g., to avoid a boom).

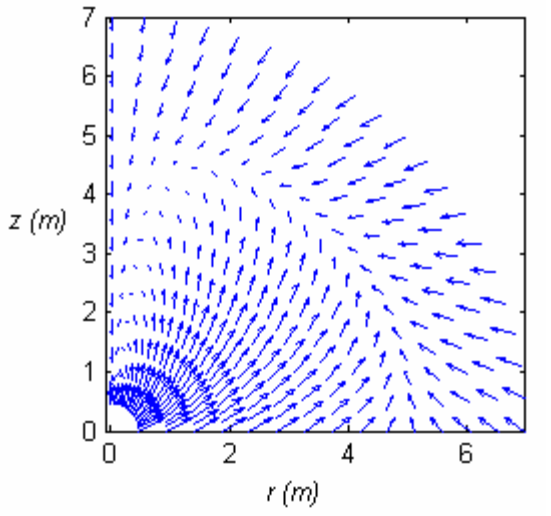

Figure 2: Desired Velocity Field for Docking Along Z-axis

POSITION AND ATTITUDE ESTIMATION

Translation

Bandit's translational dynamics are written in the Clohessy-Wiltshire coordinate frame, shown in Figure 3. This yields the following relative equations of motion with respect to the host vehicle:

$$\ddot{x} = 3\Omega_{ref}{}^2 x + 2\Omega_{ref}\dot{x} + a_x \tag{16}$$

$$\ddot{y} = -2\Omega_{ref}\dot{y} + a_y \tag{17}$$

$$\ddot{z} = -\Omega_{ref}{}^2 z + a_z \tag{18}$$

where Ω_{ref} is the angular velocity of the reference orbit. Defining the state vector $\vec{r}$ as

$$\vec{r} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T \tag{19}$$

these equations of motion can be written in the linear state space form

$$\ddot{\vec{r}} = A\vec{r} + Bu + w_t \tag{20}$$

where u is the total control input from the eight-thruster propulsion system and w_t is the process noise associated with the state propagation. For this paper, we assume variance in the thrusters to be the primary source of process noise. Because the visual navigation system can only measure position, and not velocity, the measured state is written as

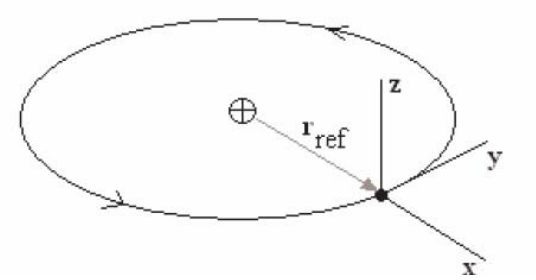

Figure 3: Clohessy-Wiltshire Relative Coordinate Frame

$$\tilde{\ddot{\vec{r}}} = H\vec{r} + \vec{v} \tag{21}$$

where

$$H = \begin{bmatrix} I_{3x3} & 0_{3x3} \end{bmatrix} \tag{22}$$

and $\vec{v}$ is the measurement noise. Since the noise in the visual navigation is a function of pixel uncertainty, and the distance represented by each pixel varies with distance from the host vehicle, the measurement noise term will be a function of the total distance between the Bandit drone and Akoya. We model this term as a quadratic function of total distance r (Eq. 23), based on the maximum error equation developed by Tsai [7] for camera pose estimation.

$$v_{\max} = \eta_1 r^2 + \eta_2 r + \eta_3 \tag{23}$$

Assuming a normal noise distribution, the variance of the measurement noise as a function of r can be determined from this equation:

$$\sigma^2 = f(r) \tag{24}$$

The linearity of the relative translation dynamics allows for Kalman filtering techniques to be used for estimating Bandit's position. We use the discrete Kalman filter equations [6] for propagating and updating Bandit's position. The state is first propagated with the assumption of no process noise (no thruster variance), so that the estimated state at time step k is written as

$$\hat{\vec{r}}_k^- = A\hat{\vec{r}}_{k-1} + B\vec{u}_k \tag{25}$$

For all time steps in which Bandit does not receive an update of the measured position, this will be the final estimated state. Next, the covariance matrix P is propagated as follows:

$$P_k^- = AP_{k-1}A^T + Q \tag{26}$$

where Q is the diagonal matrix representing the uncertainty in the process. The propagated covariance matrix is then used to calculate the Kalman gain K:

$$K_k = P_k^- H^T \left(H P_k^- H^T + R \right)^{-1} \tag{27}$$

For this case, R is a time-varying diagonal matrix that depends on the measurement variance σ^2:

$$R = \sigma^2 I_{3x3} \tag{28}$$

The Kalman gain is used to optimally weight the contributions of the propagated state and the measured position (when a position measurement exists) to the final estimated state, and also to update the covariance matrix P:

$$\hat{\vec{r}}_k = \hat{\vec{r}}_k^- + K_k \left(\tilde{\vec{r}}_k - H \hat{\vec{r}}_k^- \right) \tag{29}$$

$$P_k = \left(I - K_k H \right) P_k^- \tag{30}$$

Because of Bandit's limited computational ability, these equations must be taken out of matrix form for the final estimation algorithm. Because of the structure of the A, Q, R, and P matrices, we know that several of the elements of P and K will always be zero. Thus, we need only concern ourselves with the nonzero elements of these matrices. We find that the final estimate of each element of the state vector $\vec{r}$ can be expressed as a function of certain elements of P and R and of the propagated and measured values of that element. For example, the equation for $\hat{x}$ is found to be:

$$\hat{x}_k = \hat{x}_k^- + \frac{P_{k11}^-}{P_{k11}^- + R_{k11}} \left(\tilde{x}_k - \hat{x}_k \right) \tag{31}$$

The other elements can be similarly expressed. The necessary elements of P are written as functions of the elements of the A, R, Q, and previous P matrices.

Rotation

The equations of motion for Bandit's rotation come from rigid body dynamics (Eq. 32) and quaternion kinematics (Eq. 33):

$$J \dot{\vec{\omega}} = \vec{\omega} \times \left(J \vec{\omega} \right) + \vec{u}_{rot} + \vec{w}_r \tag{32}$$

$$\dot{\vec{q}} = \begin{bmatrix} q_4 \vec{\omega} + \vec{q}_{13} \times \vec{\omega} \\ - \vec{q}_{13}{}^T \vec{\omega} \end{bmatrix} \tag{33}$$

where $\vec{w}_r$ is the process noise, again assumed to come primarily from thruster variance. Because the equations are nonlinear, we cannot use the discrete Kalman filter equations as we did for the translational motion. The current attitude estimation algorithm for Bandit uses the same concept as the Kalman filter, in that the final estimate is a weighted average of the ideally propagated attitude and the measured attitude (Eqs. 34 and 35), but the gains are chosen arbitrarily rather than calculated for optimality.

$$\hat{\vec{q}} = \frac{k_p \hat{\vec{q}}^- + k_v \tilde{\vec{q}}}{k_p + k_v} \tag{34}$$

$$\hat{\vec{\omega}} = \frac{k_{\omega p} \hat{\vec{\omega}}^- + k_{\omega v} \tilde{\vec{\omega}}}{k_{\omega p} + k_{\omega v}} \tag{35}$$

Although the quaternion measurement will come from both the visual navigation system and integration of the rate gyro measurements, it is currently assumed to come only from the visual navigation. Thus, the

weightings k_p and k_v for the quaternion estimation algorithm are functions of the relative distance *r*. The angular velocity measurement, however, comes from the rate gyros, and so the weightings are constants whose values are determined by the noise properties of the sensors.

SIMULATION AND RESULTS

The estimation and control algorithms were tested using our 6DOF Java simulation. Simulation parameters are shown in Table 1. For the first five simulations, Bandit was commanded to release from dock to a specified waypoint coordinate relative to the host vehicle, then return to dock. For the final two simulations, Bandit was to hold its position at the given waypoint. Figures 4-10 show the distances of the actual and estimated Bandits from the host vehicle and the error between the estimated and actual position for each simulation.

Table 1: Simulation Parameters

Orbit Radius	Pinitial	Q	k_p	k_v	$k_{\omega p}$	$k_{\omega v}$	Waypoint 1 (m)	Waypoint 2 (m)
6778 km	5e-4*I_{6x6}	1.5e-5*I_{6x6}	r^2	0.015/r^2	0.001	1.0	(2.0, 0.0, 0.0)	(1.25, 1.25, 1.25)

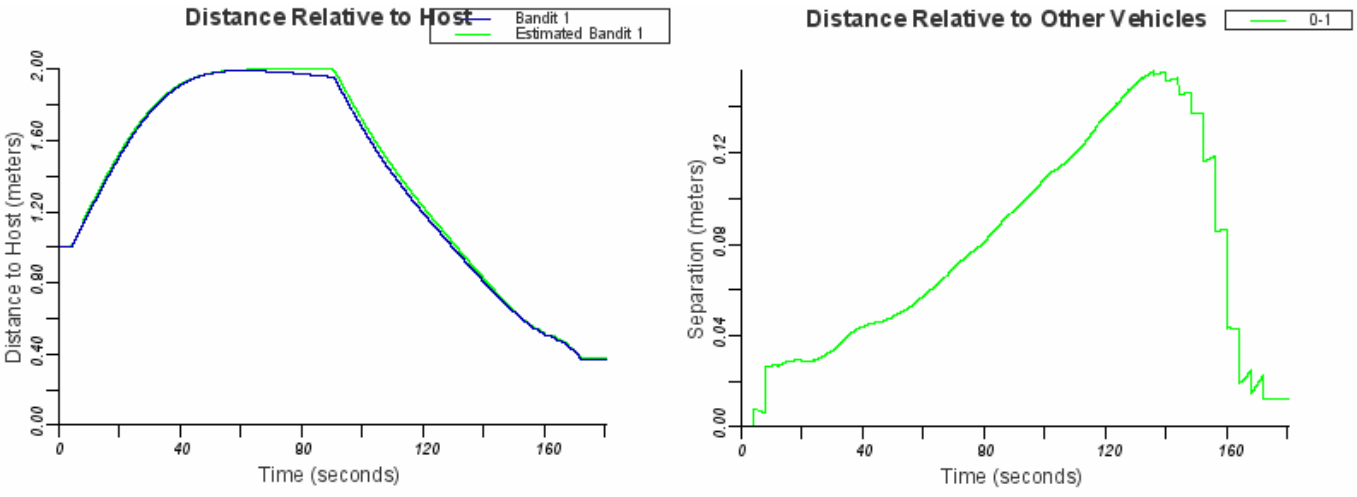

Figure 4: Distance from Host and Separation Distance, Waypoint 1, 10% Variance

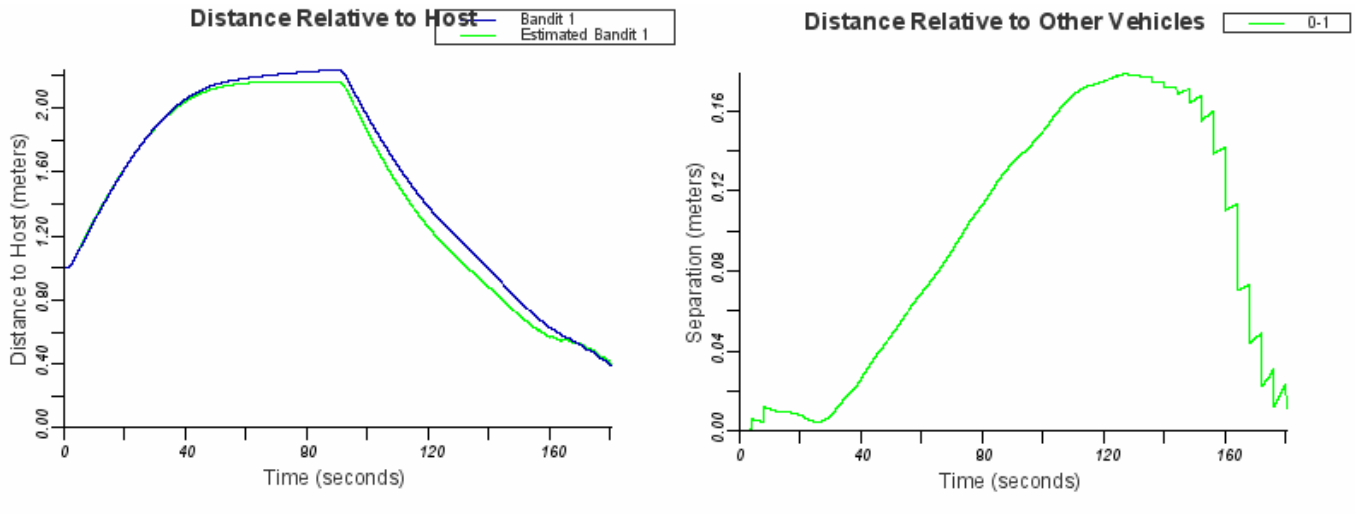

Figure 5: Distance from Host and Separation Distance, Waypoint 2, 10% Variance

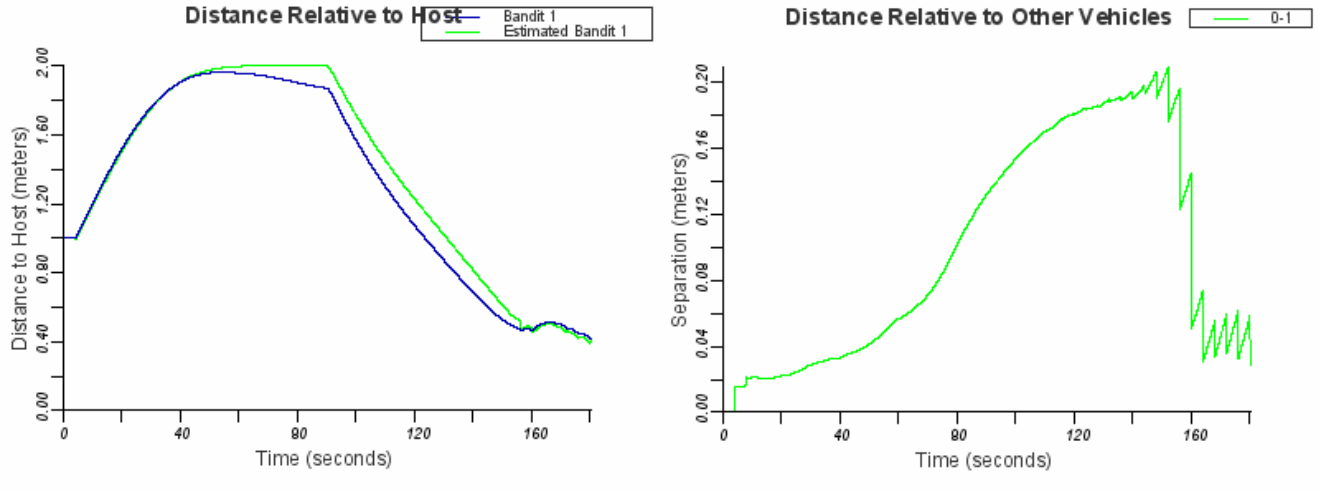

Figure 6: Distance from Host and Separation Distance, Waypoint 1, 15% Variance

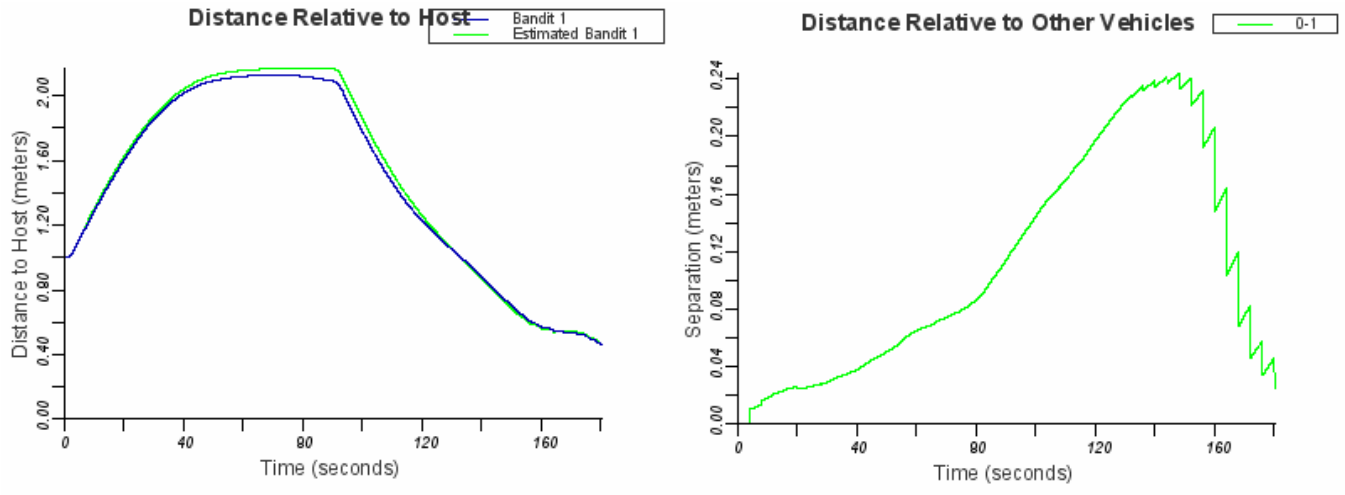

Figure 7: Distance from Host and Separation Distance, Waypoint 2, 15% Variance

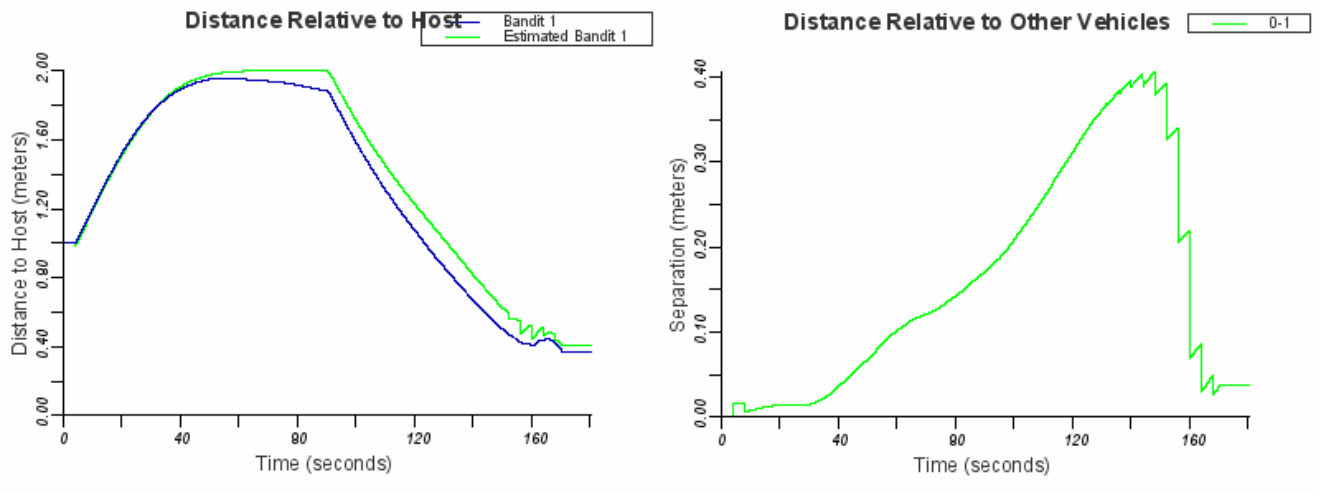

Figure 8: Distance from Host and Separation Distance, Waypoint 1, 20% Variance

As can be seen from Figures 4-8, the performance of the estimator declined noticeably the farther Bandit was from the host vehicle. However, as Bandit approached to redock, the estimator was able to correct its errors in time to ensure a safe docking. Also noticeable in these Figures is the effect of the increase of thruster variance on the estimator's performance. While the algorithm worked well at 10% and 15% variance, at 20% variance, Bandit was barely able to redock successfully from Waypoint 1. Based on these results, it was deemed more useful to move on to the Waypoint-Holding scenario rather than simulate the more difficult redocking from Waypoint 2. The results of these holding scenarios can be seen in Figures 9 and 10. The estimator and potential function controller are easily able to guide Bandit to the desired waypoint, but at that distance the estimator cannot maintain the correct position indefinitely. With no reliable measurement data to calibrate its propagation, the estimated position drifts irrecoverably away from the actual position within a few minutes. Again, the performance declines as the thruster variance increases.

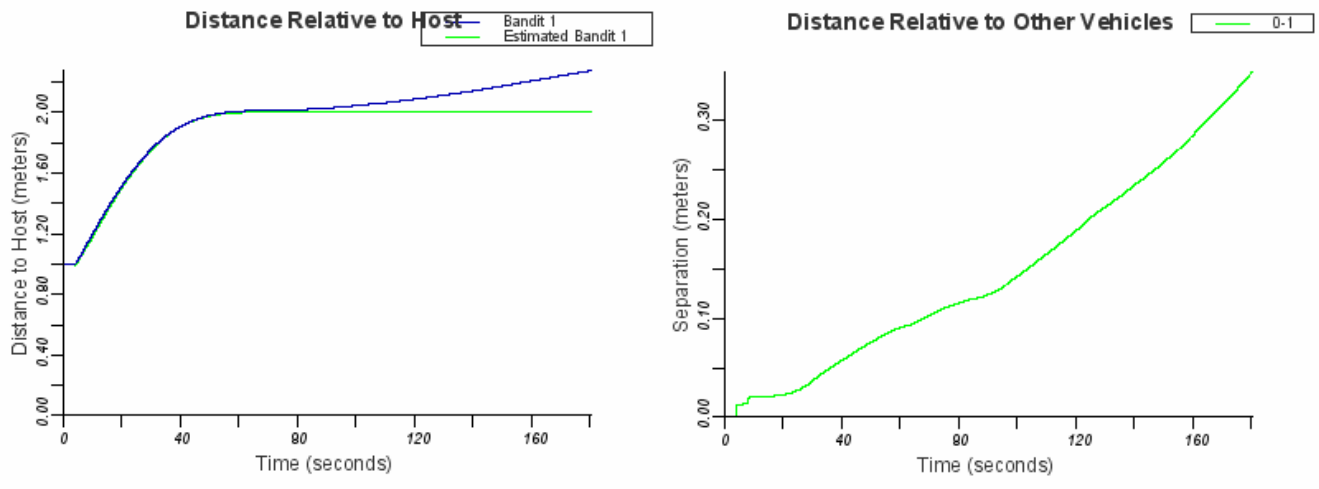

Figure 9: Distance from Host and Separation Distance, Waypoint 1, 10% Variance

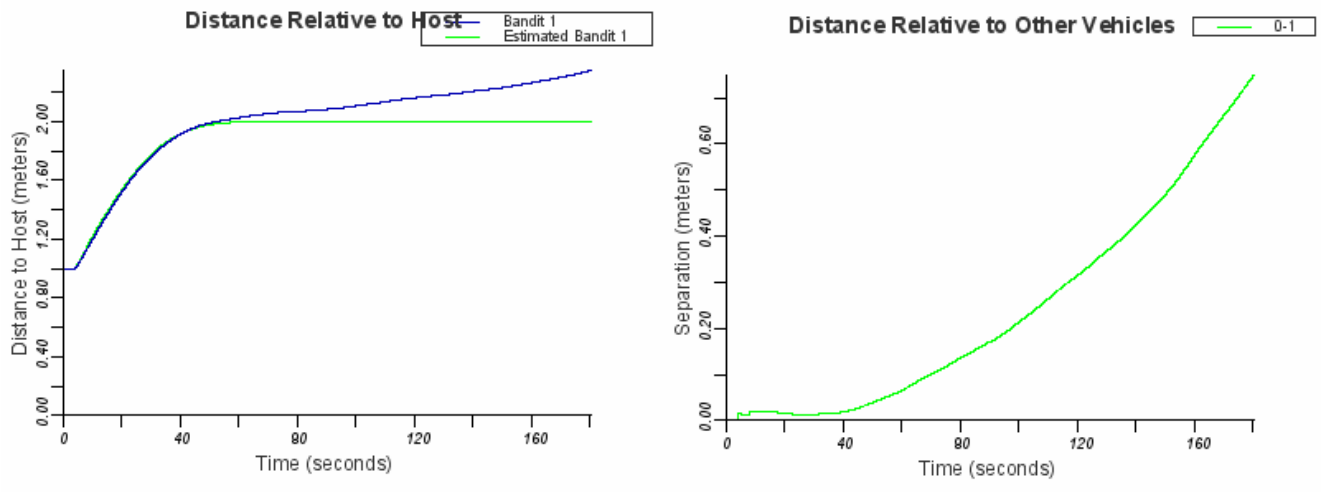

Figure 10: Distance from Host and Separation Distance, Waypoint 1, 15% Variance

CONCLUSIONS AND FUTURE WORK

Very small service vehicles can play a role in future proximity operations missions. However, to do so, the navigation challenges of these tiny spacecraft must be addressed. The Bandit mission to proposes to do in three ways: decoupled design, a robust "soft dock" and potential function control methods coupled with a computationally inexpensive estimation algorithm.

The decoupled design of the Bandit service vehicle provides a simple and efficient solution to many of the navigation problems inherent in proximity operations. By relegating all "long-period" orbital functions to a larger host vehicle, Bandit can be kept small and easily maneuverable. Furthermore, the use of a dock allowing Bandit to begin and end each of its sorties attached to the host vehicle. The extremely error-tolerant "soft dock" used in the Bandit mission further simplifies the navigation problem by enabling positive docking under a wide range of approach angles and relative speeds.

Because Bandit's thrusters are highly constrained (fixed magnitude and direction), a method of potential function control has been developed to control the Bandit system. Based on velocity error, rather than position error, this method creates a desired velocity field over the state space and defines a potential function by the error between the actual and desired velocities. The instantaneous change in potential due to firing a thruster is weighed against the cost of firing the thruster. For position and attitude estimation, a simplified version of the linear discrete Kalman filter has been applied to the translational motion and a similar, though non-optimal, process has been applied to the rotational motion. The simplified Kalman filter calculates each matrix element used in its computation separately and ignores those elements which are always zero, and the attitude estimator simply uses arbitrary weightings on the measured and propagated state vectors to obtain a final estimate. The potential function controller and estimation algorithms have been shown to work well in simulation for thruster variances of up to 15% of the nominal value, though performance of the estimator begins to drop off at 20% variance. The estimator also has difficulty holding a position far from the host vehicle for more than a few minutes, but has demonstrated the ability to recover from position errors larger than Bandit itself in order to enable successful docking.

Future work includes examining nonlinear versions of the Kalman filter (such as the Extended Kalman Filter and Unscented Kalman Filter) for use in attitude estimation. We would also like to address the drift problem for holding scenarios and attempt to extend the duration for which the estimated position remains within an acceptable distance of the actual position. Because there will likely be a great deal of uncertainty in our knowledge of Bandit's inertia parameter, we would also like to add an adaptive process to the current potential function controller.

NOTATION

f	=	desired translational velocity function
g	=	desired rotational velocity function
j	=	thruster number index
N	=	total number of thrusters
$\hat{\mathbf{n}}$	=	vector opposite camera look vector
$\mathbf{r}$	=	position vector
$\hat{\mathbf{r}}$	=	normalized position vector
r_d	=	desired target to inspector distance
$\mathbf{r}_d$	=	desired position vector
$\mathbf{r}_e$	=	position error vector
$r_{e,ss}$	=	maximum steady state translational error
$\mathbf{T}$	=	thrust matrix
$\mathbf{v}$	=	velocity vector
$\mathbf{v}_d$	=	desired velocity vector
α	=	total thruster configuration factor
$\Delta\mathbf{v}$	=	instantaneous change in translational velocity
$\Delta\boldsymbol{\omega}$	=	instantaneous change in rotational velocity
$\Delta\Phi$	=	instantaneous change in potential function
$\theta_{e,ss}$	=	maximum steady state rotational error
μ_T	=	translational velocity factor
μ_ω	=	rotational velocity factor
σ	=	thruster cost factor
Φ	=	potential function
$\boldsymbol{\omega}$	=	rotational velocity vector
$\boldsymbol{\omega}_d$	=	desired rotational velocity vector
$\boldsymbol{\omega}_o$	=	initial rotational velocity vector
Ω_{ref}	=	angular velocity of reference orbit

REFERENCES

1. Steyn, W.H. and Hashida, Y. "In-Orbit Attitude Performance of the 3-axis Stabilized SNAP-1 Nanosatellite," Proceedings of the 15th Annual AIAA/USU Conference on Small Satellites, Logan, UT, August 2001

2. Saenz-Otero, A. and Miller, D.W. "Spheres: A Platform for Formation-Flight Research," Proceedings of SPIE – The International Society for Optical Engineering, 5419:57-65. 2005

3. M. Swartwout, "Bandit: A Platform for Responsive Educational and Research Activities," 4th Responsive Space Conference, Los Angeles, CA, 26 April 2006.

4. J. Neubauer, “Controlling Swarms of Bandit Inspector Spacecraft,” Proceedings of the 20th Annual AIAA/USU Conference on Small Satellites, Logan, UT, August 2006

5. Swartwout, Michael, Sara Scarritt and Jeremy Neubauer, “Potential Function Controllers for Proximity Navigation of Underactuated Spacecraft,” Proceedings of the 30th Annual AAS Guidance and Control Conference, Breckenridge, CO, February 2007

6. Welch, G. and Bishop, G. “An Introduction to the Kalman Filter,” Technical Report. UMI Order Number: TR95-041., University of North Carolina at Chapel Hill. 1995.

7. Tsai, Roger Y. “A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses,” IEEE Journal of Robotics and Automation RA-3 (4) August 1987