# On-Orbit Inspection of an Unknown, Tumbling Target using NASA's Astrobee Robotic Free-Flyers

Charles Oestreich, Antonio Terán Espinoza, Jessica Todd, Keenan Albee, Richard Linares
Massachusetts Institute of Technology
coestrei@mit.edu

## Abstract

*Autonomous spacecraft critically depend on on-orbit inspection (i.e., relative navigation and inertial properties estimation) to intercept tumbling debris objects or defunct satellites. This work presents a practical method for on-orbit inspection and demonstrates its performance in simulation using NASA's Astrobee robotic free-flyers. The problem is formulated as a simultaneous localization and mapping task, utilizing IMU data from an observing "chaser" spacecraft and point clouds of the observed "target" spacecraft obtained via a 3D time-of-flight camera. The relative navigation between the chaser and target is solved via a factor graph-based approach. The target's principal axes of inertia are then estimated via a conic fit optimization procedure using a polhode analysis. Simulation results indicate the accuracy of the proposed method in preparation for hardware experiments on the International Space Station.*

## 1. Introduction

The accumulation of debris and defunct satellites in Earth orbit poses a serious problem for spacecraft operations. There are several missions in development to address this issue, ranging from servicing existing satellites [16, 26] to actively de-orbiting defunct satellites and space junk [6]. As the scope of these missions continues to expand, they will increasingly rely on highly autonomous spacecraft to intercept the target object prior to servicing or de-orbiting operations [4]. This intercept maneuver is especially complex when the target is tumbling in an unknown manner, a common occurrence for uncontrolled space objects. An example of this case is Envisat, a large, inactive satellite whose size poses a risk for on-orbit collisions [13].

As such, it is important to obtain accurate and reliable solutions to the relative navigation problem between the target and the servicing spacecraft (i.e., the "chaser"). Furthermore, for the tumbling object case, the target's inertial properties must be estimated to predict its motion and plan an intercept trajectory [9, 24]. Together, estimating the relative navigation and inertial properties of the target formulate the *on-orbit inspection* problem.

There has been a steady progression in recent years regarding relative navigation for autonomous spacecraft. Early techniques can largely be grouped into the *cooperative*, *model-based* category. Cooperative scenarios refer to the case where the target spacecraft is actively faciliating the relative navigation problem, such as exhibiting distinct patterns for visual sensors or retroreflectors to assist active sensors. Model-based techniques assume that there is a computer-aided design (CAD) model of the target spacecraft readily available (i.e., the target is known). As an example, Howard et al. [10] successfully tested relative navigation in space via the Orbital Express mission, utilizing active laser sensing with retroreflectors on the target alongside visual imagery to estimate the target's pose.

Generalizing to the case where the target is *non-cooperative*, the RAVEN (space-based) NASA mission has tested model-based estimation using visual imagery in space [25], using the visiting vehicles to the International Space Station (ISS) as "non-cooperative" targets. This method first detects edges and features in a visual image, then aligns them with a 3D CAD model projected onto the image to measure the relative pose (which is subsequently passed to a navigational filter). Sharma et al. [21] developed a related method that utilizes more robust feature detection techniques and does not require an accurate initial guess for the alignment problem.

For relative navigation via 3D point cloud data (which can potentially be more robust to lighting variability), Ruel et al. [17] utilized a Light Detection and Ranging (LiDAR) sensor to generate point clouds of the target and align them with the known CAD model using the iterative closest points (ICP) method [2]. This method was also validated in space using the Space Shuttle as the chaser and the International Space Station as the "non-cooperative" target. More recently, Aghili and Su [1] developed a robust, filtering-based approach for relative navigation that also utilized the ICP method for registering point cloud data of the target

with a CAD model.

Recent advances in simultaneous localization and mapping (SLAM) techniques have removed the need for a CAD model of the target and can thus address situations where the target is unknown. Often formulated for terrestrial robotics applications, these methods are focused in the idea of creating a map of the environment and simultaneously localizing the robot (i.e., estimating pose) within that map using estimated landmarks [22, 14]. These ideas have been extended towards spacecraft relative navigation. Tweddle et al. [29] formulated a factor graph-based SLAM approach to resolve both the chaser's relative pose and the target's inertial properties for the case of a stationary observer, thereby being able to remove the need of a known target model, but unable to run in real-time due to the high time complexity of the approach. Setterfield et al. [19] built upon this approach and extended it to cases where the chaser exhibits arbitrary motion (e.g., following an inspection trajectory relative to the tumbling target). However, the estimation of the target's inertial properties was removed from the factor graph to address the computational bottleneck, and instead solved for via a batch optimization method based on polhode[1] analysis [20]. Terán-Espinoza [27] improved the computational tractability of these approaches by exploiting the underlying Bayes tree structure of the incremental smoothing and mapping problem and preserving the joint estimation of both the chaser's relative pose and the target's inertial properties as a single online algorithm. It is important to note that while the previous approaches have been demonstrated using real world data acquired inside the ISS by the SPHERES and VERTIGO testbed [8, 30], an on-orbit inspection approach capable of dealing with unknown and uncooperative targets is yet to be demonstrated as part of a real-time system.

The purpose of this work is to formulate a practical on-orbit inspection SLAM approach utilizing advances from [27] and [20], and to demonstrate its practicality on NASA's Astrobee robotic free-flyers. The approach incorporates chaser IMU measurements and target pose odometry measurements obtained via a 3D time-of-flight (ToF) camera into a factor graph, which is solved via incremental smoothing and mapping to yield a solution to the relative navigation problem. The target state estimates are subsequently utilized for polhode analysis to determine the target's principal axes of inertia. Astrobee simulation results demonstrate the accuracy and reliability of the proposed method of on-orbit inspection. Hardware demonstrations on the Astrobee ground testbed and the International Space Station (ISS) are scheduled in the near future.

The structure of this paper is as follows: Section 2 outlines the on-orbit inspection problem and provides a brief description of the Astrobee robots. Section 3 presents the front-end measurements to the SLAM problem. Section 4 outlines the factor graph approach for relative navigation along with the conic fit optimization used to estimate the principal axes of the target. Section 5 provides details on the ISS Astrobee simulation environment and software implementation of the full algorithm. Section 6 shares results obtained for a sample on-orbit inspection trajectory in the simulator. Finally, Section 7 provides a conclusion and discussion of future work, including planned hardware testing.

[1]The polhode is the path traced by the angular velocity vector of a rotating body on its inertia ellipsoid.

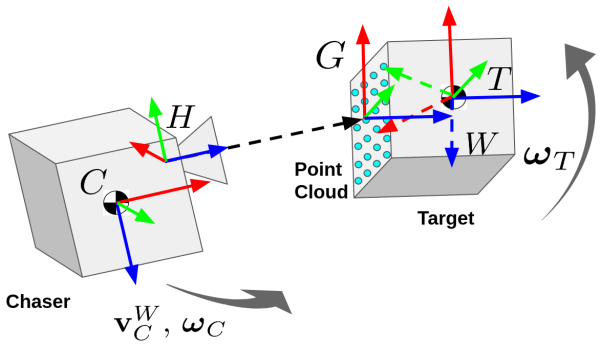


Figure 1. Overview of coordinate frames and variables for on-orbit inspection.

## 2. Problem Overview

The on-orbit inspection problem is defined as estimating the states of both the chaser and target in the world frame $W$ as well as the target's inertial properties. The state of the chaser is defined as $\mathbf{x}_C^W \triangleq \left(\mathbf{T}_C^W, \mathbf{v}_C^W, \boldsymbol{\omega}_C\right)$, where $\mathbf{T}_C^W \in SE(3)$ is the homogeneous transformation representing the chaser pose (orientation and translation) w/r to the $W$ frame, $\mathbf{v}_C^W \in \mathbb{R}^3$ is the velocity w/r to the $W$ frame, and $\boldsymbol{\omega}_C \in \mathbb{R}^3$ is the chaser angular velocity expressed in the chaser body frame $C$. The state of the target is defined as $\mathbf{x}_T^W \triangleq \left(\mathbf{R}_T^W, \boldsymbol{\omega}_T\right)$, where $\mathbf{R}_T^W \in SO(3)$ is the target orientation w/r to the $W$ frame and $\boldsymbol{\omega}_T \in \mathbb{R}^3$ is the target angular velocity expressed in the target frame $T$. The $T$ frame is aligned with the target's principal axes of inertia.

It is assumed that the target is stationary (with respect to translation) with its center of mass located at the origin of the $W$ frame. The $H$ frame is the camera frame of the chaser's sensor that generates point clouds of the target. The $G$ frame, termed the geometric frame, is an arbitrary frame attached to the surface of the target. The target's inertial properties that must be determined are defined as the orientation of its principal axes with respect to the $G$ frame, $\mathbf{R}_T^G$, and its inertia ratios $J_1 = I_{xx}/I_{zz}$ and $J_2 = I_{yy}/I_{zz}$, where $\mathbf{I} = \text{diag}(I_{xx}, I_{yy}, I_{zz})$ are the moments of inertia expressed in principal axes (diagonalized). These frames are depicted in Figure 1.

The NASA Astrobee robots are a set of robotic free-flyers operating on the ISS with the purpose of (a) astro-

naut assistance, and (b) microgravity autonomy research [3] [23]. As such, they provide a suitable testbed for experiments involving autonomous, on-orbit inspection in a realistic micro-gravity environment. Astrobee utilizes two impellers which use vents to release pressurized air to provide full, holonomic propulsion capabilities on the interior of the ISS [23]. For this work, one Astrobee robot represents the chaser and follows a pre-programmed inspection trajectory. Another Astrobee robot acts as the target and follows a tumbling trajectory.

## 3. Front-end Measurements

Astrobee is outfitted with several sensors that provide front-end measurements to the SLAM problem. The "HazCam" is a 3D ToF camera on the chaser used to generate point clouds of the tumbling target; in simulation, it operates at 1 Hz with a resolution of 224 $\times$ 171 pixels. Point clouds are truncated to remove excess background data and isolate the target. The IMU is utilized to provide chaser acceleration and angular velocity measurements at a rate of 62.5 Hz.

The geometric frame $G$ is defined at the point cloud's centroid in the first keyframe (i.e., first available point cloud measurement) (Figure 1). This places it roughly on the surface of the target with a constant but unknown translation vector $\mathbf{t}_T^G$ that relates it to the $T$ frame. The orientation between the $G$ frame and the principal axes, $\mathbf{R}_T^G$, is initially undetermined. The orientation of the $G$ frame is arbitrarily initialized relative to the chaser as $\mathbf{R}_{C_0}^G = I_{3\times3}$, since it is impossible to define a non-arbitrary target frame until the target's principal axes are determined.

Relative target-chaser pose odometry is provided by registering successive point cloud measurements. For a pair of target point clouds $\mathbf{z}_i^\nu$ and $\mathbf{z}_j^\nu$ obtained at keyframes $i$ and $j$, the registration process follows a three-step pipeline: **1.** detecting 3D features in $\mathbf{z}_j^\nu$ using fast point feature histograms (FPFH) as the feature descriptor [18]; **2.** matching the 3D features between $\mathbf{z}_i^\nu$ and $\mathbf{z}_j^\nu$ using the Fast Library for Approximate Nearest Neighbors (FLANN) search method [15]; and finally **3.** registering the correspondences to a relative pose estimate between the two point clouds via the Teaser++ robust registration solver [31].

The result is a pose odometry measurement

$$\mathbf{z}_{ij}^\nu = \left\{ \mathbf{z}_{\mathbf{R}_{ij}}^\nu, \mathbf{z}_{\mathbf{t}_{ij}}^\nu \right\} \in SE(3) \tag{1}$$

of the chaser with respect to the target's $G$ frame. A depiction of the feature matching and registration process is shown in Figure 2.

The HazCam is also used to provide range and bearing measurements between the point cloud centroids and the chaser at each keyframe $j$. This measurement is defined as

$$\mathbf{z}_j^\rho = \left\{ z_j^{\rho d} \in \mathbb{R}^+, \mathbf{z}_j^{\rho b} \in S^2 \right\}, \tag{2}$$

where $z^{\rho d}$ and $\mathbf{z}^{\rho b}$ are assumed to be range and bearing measurements to the target's center of mass, respectively, with $S^2$ representing the 2-sphere manifold.

Since Astrobee's IMU operates at a much faster rate than the HazCam, IMU preintegration theory [7] is leveraged to bundle the set of measurements $\mathbf{z}_{ij}^\psi = \{\mathbf{z}_{a_\tau}, \mathbf{z}_{\boldsymbol{\omega}_\tau}\}_{\tau=i}^j$ as a single measurement between keyframes $i$ and $j$:

$$\Delta\mathbf{x}_{C_{ij}}^W = \left\{ \Delta\mathbf{T}_{C_{ij}}^W, \Delta\mathbf{v}_{C_{ij}}^W, \Delta\mathbf{b}_{ij} \right\} \tag{3}$$

where $\Delta\mathbf{x}_{C_{ij}}^W$ represents the odometry of the chaser's navigational state and $\mathbf{b}$ is the estimate of the IMU biases.

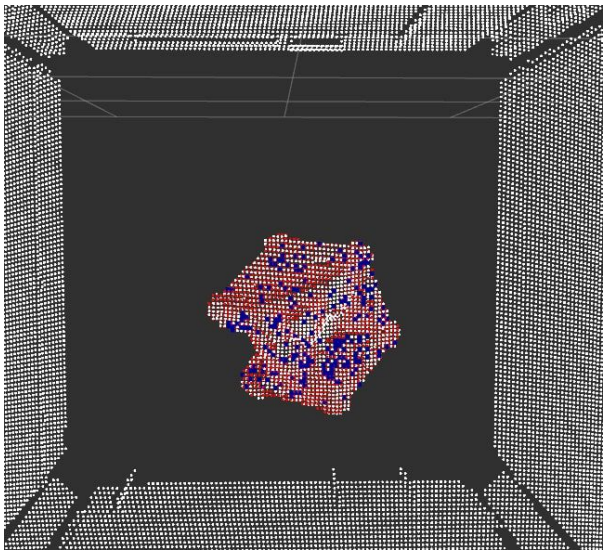

Figure 2. Depiction of the target point cloud matching and registration process in the ISS Astrobee simulator. The white point cloud is the raw HazCam data. Blue points indicate matched 3D features between succesive frames. The red point cloud is produced by applying the obtained pose odometry measurement to the previous frame; close alignment with the raw data indicates accurate odometry.

## 4. Relative Navigation via Factor Graphs and Conic Optimization

A factor graph for the relative navigation problem connects all relevant variables via probabilistic factors that arise from the available measurements. The goal is to find the maximum *a posteriori* state history that best satisfies the graph's factors. This nonlinear optimization problem is efficiently solved using incremental smoothing and mapping, yielding online navigational state updates at each keyframe. In this work, the factor graph largely consists of two pose chains that are connected by various measurement factors.

The target pose chain consists of unknown target poses at each keyframe connected by relative pose factors that arise from the target odometry measurements in equation (1). Each relative pose factor between the pose variables $\mathbf{T}_{C_i}^G$ and $\mathbf{T}_{C_j}^G$ (chaser w/r to $G$ frame) at keyframes $i$ and $j$ is written as

$$\nu_{ij} \propto \exp\left\{ -\frac{1}{2} \left\| \left(\mathbf{T}_{C_i}^G\right)^{-1} \mathbf{T}_{C_j}^G \ominus \mathbf{z}_{ij}^{\nu} \right\|_{\mathbf{\Sigma}_\nu}^2 \right\} \tag{4}$$

where $\ominus$ denotes on-manifold subtraction and $||(\cdot)||_{\mathbf{\Sigma}_\nu}^2$ is the weighted Mahalanobis distance using the noise model parameter $\Sigma_\nu$.

Loop closures in target odometry can also be added to the factor graph in the same manner when $i$ and $j$ are non-subsequent keyframes. At each time step, loop closures are attempted by randomly selecting a previous point cloud from a moving window of past frames and attempting to find 3D feature matches. If the number of matches is above a design threshold, the loop closure is successful and an additional factor is added to the graph between the matched keyframes.

The range and bearing measurements between the chaser and the (approximate) target center of mass, defined in equation (2), are implemented as binary factors between the chaser's navigational state $\mathbf{x}_{C_j}^W$ and a variable representing a constant target center of mass offset $\mathbf{t}_T^W$. Since the target center of mass is assumed to be centered at the $W$ frame origin, this extra variable simply accounts for any errors while modeling. The range and bearing factor is formulated as

$$\rho_j \propto \exp\left\{ -\frac{1}{2} \left\| \begin{bmatrix} h_d(\mathbf{x}_{C_j}^W, \mathbf{t}_T^W) - z_j^{\rho d} \\ h_b(\mathbf{x}_{C_j}^W, \mathbf{t}_T^W) \ominus \mathbf{z}_j^{\rho b} \end{bmatrix} \right\|_{\mathbf{\Sigma}_\rho}^2 \right\} \tag{5}$$

where $h_d$ and $h_b$ are the measurement models for range and bearing, respectively, and $\Sigma_\rho$ is the noise model.

Chaser pose odometry factors between subsequent keyframes $i$ and $j$ are formulated from the preintegrated IMU measurements in equation (3). This factor is defined as

$$\psi_{ij} \propto \exp\left\{ -\frac{1}{2} \left\| \left( \mathbf{r}_{\Delta \mathbf{T}_{C_{ij}}^W}^\top, \mathbf{r}_{\Delta \mathbf{v}_{C_{ij}}^W}^\top, \mathbf{r}_{\Delta \mathbf{b}_{ij}}^\top \right) \right\|_{\Sigma_\psi}^2 \right\} \tag{6}$$

where $\mathbf{r}$ are the residual error vectors for the respective components of the chaser navigational state and $\Sigma_\psi$ is the noise model.

Additional factors, termed rotation kinematic factors, are added to the graph in order to link the target and chaser pose chains and disambiguate the relative motion between the two spacecraft. These factors represent a zero sum vector addition constraint between chaser/target poses in subsequent key frames, given the assumption that the target is unperturbed and stationary in terms of translation (Figure 3). Note that this factor utilizes the constant but unknown variable $\mathbf{t}_T^G$ relating the $G$ frame to the $T$ frame. This factor is implemented as

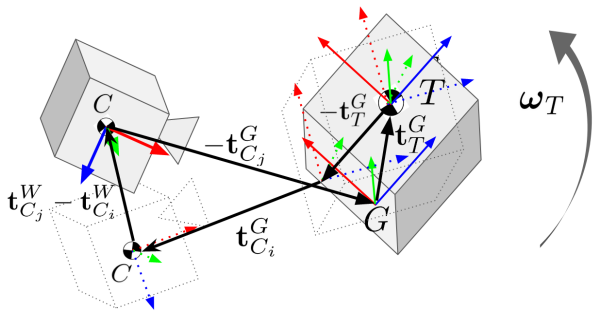


Figure 3. Depiction of the rotation kinematic factors that link the otherwise separate target and chaser pose chains. The bold vectors in the image must sum to zero to satisfy the factor's constraint. The wireframe chaser and target indicate the poses at the previous keyframe, whereas the solid chaser and target indicate the poses at the current keyframe.

$$\begin{aligned} \kappa_{ij} \propto \exp\Bigg\{ -\frac{1}{2} \Bigg\| & \mathbf{R}_{C_i}^W \left(\mathbf{R}_{C_i}^G\right)^\top \left(\mathbf{t}_{C_i}^G - \mathbf{t}_T^G\right) \\ & + \mathbf{R}_{C_j}^W \left(\mathbf{R}_{C_j}^G\right)^\top \left(\mathbf{t}_T^G - \mathbf{t}_{C_j}^G\right) \\ & + \left(\mathbf{t}_{C_j}^W - \mathbf{t}_{C_i}^W\right) \Bigg\|_{\Sigma_\kappa}^2 \Bigg\} \end{aligned} \tag{7}$$

where $\mathbf{\Sigma}_\kappa$ is the noise model.

The factor graph is now complete and contains enough factors to solve for the relative navigation variables. Figure 4 displays how the factors defined in equations (4 - 7) connect the variables at each subsequent keyframe. The actual online solution to optimize the graph is computed using a different, but directly related, data structure called the Bayes tree [12]. By recycling computations at each time step, a full SLAM solution is efficiently and accurately computed every time new information is received.

The final step for relative navigation is to compute the angular velocities of both spacecraft using the estimated orientations between successive time steps $i$ and $j$:

$$\boldsymbol{\omega}_{G_j} = \frac{1}{\Delta t_{ij}} \mathrm{Log}\left( \left( \mathbf{R}_{C_i}^W (\mathbf{R}_{C_i}^G)^\top \right)^\top \mathbf{R}_{C_j}^W (\mathbf{R}_{C_j}^G)^\top \right) \tag{8a}$$

$$\boldsymbol{\omega}_{C_j} = \frac{1}{\Delta t_{ij}} \mathrm{Log}\left( (\mathbf{R}_{C_i}^W)^\top \mathbf{R}_{C_j}^W \right) \tag{8b}$$

where $\Delta t_{ij}$ is the time length in between keyframes and Log is shorthand for the $SO(3)$ logarithimic map $\log : SO(3) \to \mathfrak{so}(3)$ follow by the inverse skew symmetric matrix operator $\vee : \mathfrak{so}(3) \to \mathbb{R}^3$. The target's angular velocity estimates are naturally noisy as a consequence of the

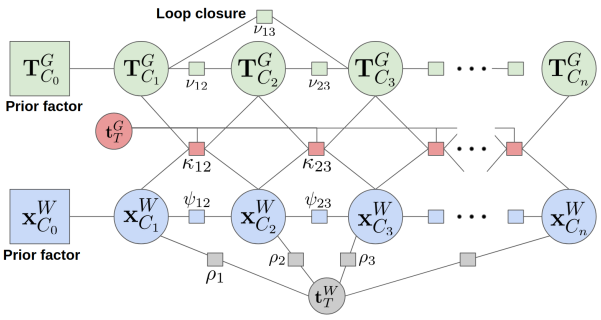


Figure 4. Full factor graph for relative navigation. Green indicates the target pose chain, blue indicates the chaser pose chain, red indicates the rotation kinematic factors, and grey indicates the range and bearing factors. Large squares are prior factors that initialize each pose chain, circles represent the variables to be estimated, and small squares are factors that encode local constraints between adjacent variables.

point cloud odometry. Thus, smoothed estimates $\bar{\boldsymbol{\omega}}_{G_j}$ are obtained via a recursive algorithm:

$$\beta = \exp(\alpha - 1) \tag{9a}$$

$$\gamma_j = \frac{\beta\left[1 - \beta^j\right]}{1 - \beta} \tag{9b}$$

$$\bar{\boldsymbol{\omega}}_{G_j} = \frac{\beta}{\gamma_j}\left(\gamma_i \bar{\boldsymbol{\omega}}_{G_i} + \boldsymbol{\omega}_{G_i}\right) \tag{9c}$$

where $\alpha \in (0, 1)$ is a constant design parameter termed the "forgetting factor".

Note that the target's orientation and angular velocity estimates are with respect to the arbitrarily initialized $G$ frame. While the $\mathbf{t}_T^G$ variable is solved for in the factor graph, the orientation $\mathbf{R}_T^G$ is still undetermined. This issue is resolved by estimating the principal axes of the target's inertia tensor and establishing a non-arbitrary $T$ frame. Using the smoothed angular velocity measurements from (9c) of the target object with respect to the arbitrary frame $G$, the procedure outlined in [20] is followed to rotate this misaligned polhode such that the curve's new orientation produces central conic projections onto the XY, YZ, and XZ planes (see Figure 7).

The specific combination of conic types for each plane further specifies the convention and estimate for the principal axes orientation. For instance, a tumbling target with a tri-axial inertia tensor and low rotational energy will result in two ellipses and one hyperbola [20]. The final result is an optimized orientation $\mathbf{R}_T^G$ that best aligns the measured angular velocity values into the newly defined $T$ frame based upon the target's principal axes of inertia. As such, it is now possible to determine the target's orientation with respect to the $W$ frame and its angular velocity with respect to the $T$ frame:

$$\mathbf{R}_T^W = \mathbf{R}_C^W \left(\mathbf{R}_C^G\right)^\top \mathbf{R}_T^G \tag{10a}$$

$$\boldsymbol{\omega}_T = \left(\mathbf{R}_T^G\right)^\top \boldsymbol{\omega}_G \tag{10b}$$

Once the angular velocity profile has been correctly aligned with the principal axes, it is possible to proceed with the estimation of the target's inertia ratios. By leveraging the closed-form solution for rigid body motion based on the Jacobi elliptic functions [11], a second procedure from [20] can be employed to create a constrained nonlinear optimization problem and solve for physically consistent values for the inertia ratios $J_1$ and $J_2$. This would allow for the prediction of the target's tumbling motion, thus enabling the planning of an intercept trajectory [24] and the communication between the chaser's estimation and guidance modules [28].

## 5. Implementation on Astrobee

The Astrobee free-flyer is a cube-shaped robot, measuring 32 cm per side [3]. Its holonomic propulsion system draws in air through two central impellers, which is expelled precisely through 12 exhaust nozzles for thrust [23]. The Astrobee Robot Software uses the Robotic Operating System (ROS) as middleware for communication, with 46 nodelets grouped into approximately 14 processes running on two ARM processors [5]. The Astrobee Robot Software also contains a Gazebo-based[2] simulator, which enables testing of developed algorithms before implementation on hardware. This simulation environment includes extensive modeling of Astrobee including its impeller propulsion system, onboard visual navigation, environmental disturbances, and many more true-to-life models [5].

The proposed framework for relative navigation and inertial property estimation was tested in the ISS Astrobee simulation prior to eventual hardware testing. The chaser was commanded to follow a sample inspection trajectory consisting of a) a lateral arc maneuver, b) a vertical arc maneuver, and c) an approach/recede maneuver along the viewing axis. Chaser attitude commands kept the target in the HazCam field of view. Meanwhile, the target was commanded to follow attitude setpoints representing a tumble produced by the tri-axial inertia tensor of the Envisat satellite. The chaser started its inspection trajectory 1.5 meters away from the target. The target's initial orientation was $\mathbf{R}_T^W = I_{3\times3}$ with an initial angular velocity $\boldsymbol{\omega}_T = [0,\ 3.53,\ 3.53]$ deg/s. Figure 5 shows the chaser stationed at its initial condition in the simulator, ready to begin the inspection maneuver of the tumbling target.

The relative navigation framework was packaged as a ROS nodelet to interact with Astrobee and the simulator. The main computational bottleneck was the point cloud registration pipeline; as such, voxel grid filtering of the raw

[2] http://gazebosim.org/

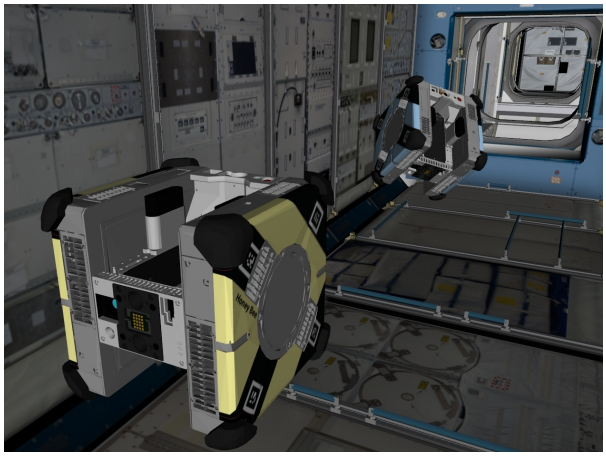

Figure 5. The chaser (yellow) waiting to begin its inspection trajectory as the target (blue) begins its tumble in the ISS simulation environment. While Astrobee's inertia tensor is nearly spherical, the target can simulate the tumble of an arbitrary rigid body.

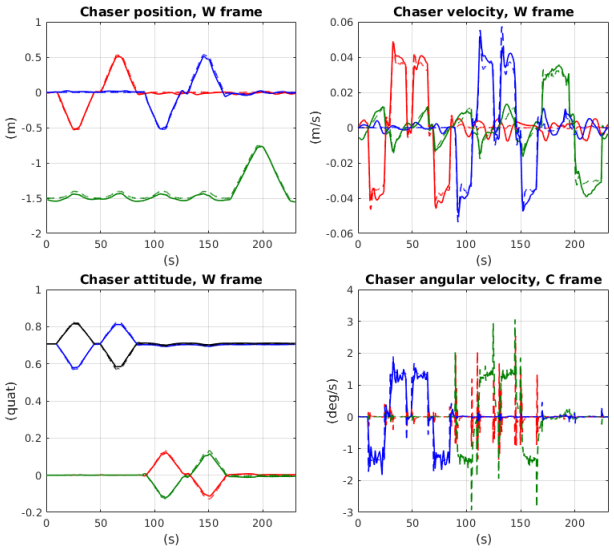


Figure 6. Chaser state estimation results for the simulated inspection trajectory. Solid lines indicate estimates, dashed lines indicate truth values. R/G/B indicate the x/y/z axes, respectively, and black indicates the scalar part of the attitude quaternion.

HazCam data was implemented to significantly improve computational performance. The Georgia Tech Smoothing and Mapping (GTSAM)[3] library was used for efficiently building and solving the factor graph. Inertial property estimation was performed by post-processing the estimated target angular velocity data.

## 6. Simulation Results

Figure 6 shows the estimated chaser navigational state throughout the inspection trajectory. Truth values from the simulator serve as a metric to evaluate the proposed approach and provide estimation error statistics. The estimates were smooth and closely corresponded to the true values, demonstrating the ability to disambiguate chaser motion from the tumbling target motion. Slight oscillatory drift on the order of 5 mm/s is evident. This is likely due to the slight oscillation of the point cloud's centroid that arises during from the target Astrobee's tumbling, cubic shape.

The target orientation and angular velocity estimates were estimated in the factor graph with respect to the arbitrary $G$ frame. As such, there is a coordinate frame offset between the initially estimated values and the simulator truth values. These "unaligned" values were used to perform the polhode analysis in order to determine the principal axes of the target and its inertia tensor ratios (see Figure 8, where "Measured" denotes "unaligned").

Figures 7, 8, and 9 show the results for the principal axes estimation portion of the pipeline. The optimized value of $\mathbf{R}_T^G$ produced the best conic projection fits shown in Figure 7. Since the Envisat's inertia tensor is tri-axial, the projection of the polhode produced central ellipses in the XY and YZ planes, and a hyperbola in the XZ plane. This orientation is then used to rotate the "measured" target angular velocity values (defined in the $G$ frame) into the $T$ frame (polhodes shown in Figure 8). The corresponding time history representation for the same three curves from Figure 8 is subsequently shown in Figure 9, along with the per-measurement error between the estimated alignment using $\mathbf{R}_T^G$ and the ground truth alignment of the angular velocity measurement. The value of $\mathbf{R}_T^G$ also allows us to obtain target attitude estimates in the $\mathbf{R}_T^W$ frame, thus completing the relative navigation problem.

Table 1 shares estimation error statistics for the navigational states. The low magnitude of these values indicate successful performance of the on-orbit inspection task, which in turn would enable the subsequent motion planning and trajectory tracking phases that are required to intercept the target.

In summary, the proposed approach demonstrates suitable estimation accuracy that compares well with existing methods. Moreover, the only major assumption that this approach makes is the lack of external forces acting upon the target object (that is, that the target is stationary with respect to translation and tumbling with torque-free dynamics); otherwise, the target is truly unknown and no shape models are required. Finally, this method is specifically designed for experiments on hardware: the full pipeline can reliably run faster than 1 Hz on embedded processors.

[3] https://github.com/borglab/gtsam

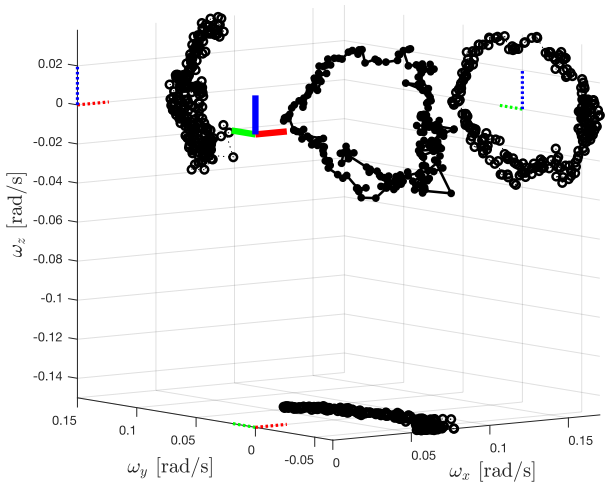


Figure 7. Resulting polhode after alignment to the target's estimated principal axes. The polhode's plane projections produce central conics (not rotated nor displaced from their respective origins). Solid circles indicate the polhode, open circles indicate the projected conics.

| | |
|---|---|
| $\widetilde{\mathbf{t}}_C^W$ | 4.1 cm |
| $\widetilde{\mathbf{v}}_C^W$ | 5.3 mm/s |
| $\widetilde{\mathbf{a}}_C^W, \widetilde{\theta}$ | $[-0.1°,\ 0.001°,\ 0.61°]^\top$, 1.00° |
| $\widetilde{\boldsymbol{\omega}}_C$ | 0.08 °/s |
| $\widetilde{\mathbf{a}}_T^W, \widetilde{\theta}$ | $[6.26°,\ -4.24°,\ 0.85°]^\top$, 7.25° |
| $\widetilde{\boldsymbol{\omega}}_T$ | 0.6199 °/s |

Table 1. Estimation error statistics for the navigational states. All error values are the median L2 norm between the estimated and true values, except for $\widetilde{\mathbf{a}}$, which is computed as $\widetilde{\mathbf{a}} = \text{Log}(\bar{\mathbf{R}}^\top \hat{\mathbf{R}}) = [\delta a_x,\ \delta a_y,\ \delta a_z]^\top$ and $\widetilde{\theta} = ||\widetilde{\mathbf{a}}||$, where $\bar{\mathbf{R}}$ and $\hat{\mathbf{R}}$ correspond to true and estimated orientations, respectively.

## 7. Conclusion

This work presents a complete, practical algorithm for performing autonomous on-orbit inspection of an unknown, tumbling target. Utilizing the Astrobee robotic free-flyer's ToF camera and IMU measurements, the factor graph-based approach with conic fit optimization efficiently solves for the navigational states of both the chaser and target. Designed practically with the eventual goal of actual hardware testing, the algorithm was successfully tested in a sample inspection trajectory within the realistic ISS Astrobee simulator. Despite a complex, tri-axial target tumble, the chaser is able to accurately estimate the target's state while also maintaining a clear estimate of its own dynamic state throughout the trajectory. Future work will demonstrate the algorithms introduced in this paper on the Astrobee robots onboard the International Space Station, thus validating the method in a real, 6-DOF, microgravity environment.

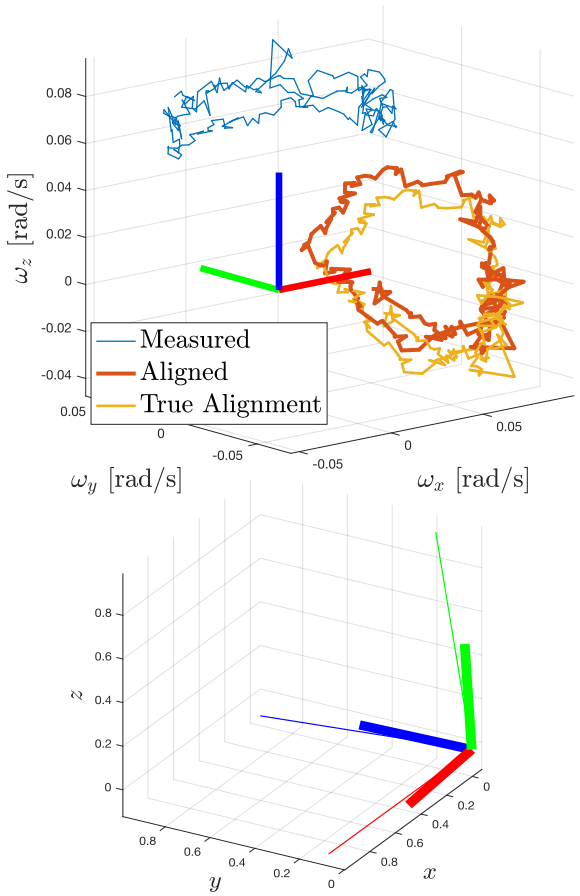


Figure 8. Top: polhode comparison between the measured (blue, in geometric frame), aligned (red, after rotating by the estimated orientation from geometric to principal axes), and ground truth-aligned (orange) curves. Bottom: visual comparison of the offset between the ground truth principal axes (bold RGB triad) against the estimated orientation (thin, longer RGB triad).

## References


[1] Farhad Aghili and Chun Yi Su. Robust relative navigation by integration of ICP and adaptive Kalman filter using laser scanner and IMU. *IEEE/ASME Transactions on Mechatronics*, 21(4), 2016. 1

[2] Paul J. Besl and Neil D. McKay. A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 1992. 1

[3] Maria G. Bualat, Trey Smith, Terrence W. Fong, Ernest E. Smith, and D. W. Wheeler. Astrobee: A new tool for ISS operations. In *15th International Conference on Space Operations, 2018*, 2018. 3, 5

[4] Angel Flores-Abad, Ou Ma, Khanh Pham, and Steve Ulrich. A review of space robotics technologies for on-orbit servicing. *Progress in Aerospace Sciences*, 68, 2014. 1

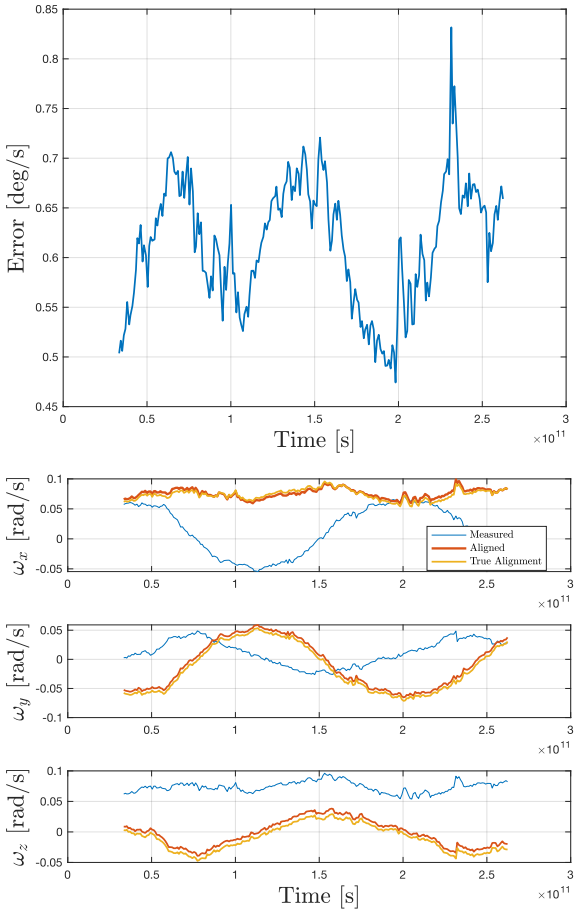


Figure 9. Top: per time step magnitude error of the estimated angular velocity in principal axes. Bottom: measured (blue), estimated-aligned (red), and ground truth-aligned (orange) time history view of the polhodes, separated into its individual components.


[5] Lorenzo Flückiger, Kathryn Browne, Brian Coltin, Jesse Fusco, Theodore Morse, and Andrew Symington. Astrobee Robot Software: Enabling Mobile Autonomy on the ISS. Technical report, 2018. 5

[6] Jason L. Forshaw, Guglielmo S. Aglietti, Nimal Navarathinam, Haval Kadhem, Thierry Salmon, Aurélien Pisseloup, Eric Joffre, Thomas Chabot, Ingo Retat, Robert Axthelm, Simon Barraclough, Andrew Ratcliffe, Cesar Bernal, François Chaumette, Alexandre Pollini, and Willem H. Steyn. RemoveDEBRIS: An in-orbit active debris removal demonstration mission. *Acta Astronautica*, 127, 2016. 1

[7] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Transactions on Robotics*, 33(1), 2017. 3

[8] D. Fourie, B. E. Tweddle, S. Ulrich, and A. Saenz-Otero. Flight results of vision-based navigation for autonomous spacecraft inspection of unknown objects. *Journal of Spacecraft and Rockets*, 51(6):2016–2026, 2014. 2

[9] Ulrich Hillenbrand and Roberto Lampariello. Motion and parameter estimation of a free-floating space object from range data for motion prediction. *European Space Agency, (Special Publication) ESA SP*, (603), 2005. 1

[10] R. T. Howard, A. F. Heaton, R. M. Pinson, and C. K. Carrington. Orbital express advanced video guidance sensor. In *IEEE Aerospace Conference Proceedings*, 2008. 1

[11] J. E. Hurtado and N. Satak. Poinsot motion attitude. In *International Conference on Computational & Experimental Engineering and Sciences*, volume 16, 2011. 5

[12] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J. Leonard, and Frank Dellaert. ISAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 31(2), 2012. 4

[13] Daniel Kucharski, Georg Kirchner, Franz Koidl, Cunbo Fan, Randall Carman, Christopher Moore, Andriy Dmytrotsa, Martin Ploner, Giuseppe Bianco, Mikhailo Medvedskij, Andriy Makeyev, Graham Appleby, Michihiro Suzuki, Jean Marie Torre, Zhang Zhongping, Ludwig Grunwaldt, and Qu Feng. Attitude and spin period of space debris envisat measured by satellite laser ranging. *IEEE Transactions on Geoscience and Remote Sensing*, (12), 2014. 1

[14] J. J. Leonard and H. F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*, pages 1442–1447 vol.3, 1991. 2

[15] Marius Muja and David G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11), 2014. 3

[16] Benjamin B. Reed, Robert C. Smith, Bo Naasz, Joseph Pellegrino, and Charles Bacon. The Restore-L servicing mission. In *AIAA Space and Astronautics Forum and Exposition, SPACE 2016*, 2016. 1

[17] Stéphane Ruel, Tim Luu, and Andrew Berube. Space shuttle testing of the TriDAR 3D rendezvous and docking sensor. *Journal of Field Robotics*, 29(4), 2012. 1

[18] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. In *2009 IEEE International Conference on Robotics and Automation*, 2009. 3

[19] Timothy P. Setterfield, David W. Miller, John J. Leonard, and Alvar Saenz-Otero. Mapping and determining the center of mass of a rotating object using a moving observer. *International Journal of Robotics Research*, 37(1), 2018. 2

[20] Timothy P. Setterfield, David W. Miller, Alvar Saenz-Otero, Emilio Frazzoli, and John J. Leonard. Inertial properties estimation of a passive on-orbit object using polhode analysis. *Journal of Guidance, Control, and Dynamics*, 41(10), 2018. 2, 5

[21] S. Sharma, J. Ventura, and S. D'Amico. Robust model-based monocular pose initialization for noncooperative spacecraft rendezvous. *Journal of Spacecraft and Rockets*, 55(6), 2018. 1

[22] Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5:56–68, 1986. 2

[23] Trey Smith, Jonathan Barlow, Maria Bualat, Terrence Fong, Christopher Provencher, Hugo Sanchez, and Ernest Smith.

Astrobee: A New Platform for Free-Flying Robotics on the ISS. In *iSAIRAS 2016*, 2016. 3, 5

[24] Samantha Stoneman and Roberto Lampariello. A Nonlinear Optimization Method to Provide Real-Time Feasible Reference Trajectories to Approach a Tumbling Target Satellite. *Isairas 2016*, 13, 2016. 1, 5

[25] Matthew Strube, Ross Henry, Eugene Skelton, John Van Eepoel, Nat Gill, and Reed McKenna. Raven: An on-orbit relative navigation demonstration using international space station visiting vehicles. volume 154, 2015. 1

[26] Brook Sullivan, Bernard Kelm, Gordon Roesler, and Carl Glen Henshaw. DARPA robotic space servicer: On-demand capabilities in GEO. In *AIAA SPACE 2015 Conference and Exposition*, 2015. 1

[27] Antonio Terán Espinoza. *Versatile Inference Algorithms using the Bayes Tree for Robot Navigation*. PhD thesis, Massachusetts Institute of Technology, 2021. 2

[28] A. Terán Espinoza, H. Hettrick, K. Albee, A. Cabrales Hernandez, and R. Linares. End-to-end framework for close proximity in-space robotic missions. In *70th International Astronautical Congress (IAC)*, Washington, DC, USA, Oct. 2019. International Astronautical Federation. 5

[29] Brent E. Tweddle, Alvar Saenz-Otero, John J. Leonard, and David W. Miller. Factor Graph Modeling of Rigid-body Dynamics for Localization, Mapping, and Parameter Estimation of a Spinning Object in Space. *Journal of Field Robotics*, 32(6), 2015. 2

[30] B. E. Tweddle, T. P. Setterfield, A. Saenz-Otero, and D. W. Miller. An open research facility for vision-based navigation onboard the international space station. *Journal of Field Robotics*, 33(2):157–186, 2016. 2

[31] Heng Yang, Jingnan Shi, and Luca Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Transactions on Robotics*, 2020. 3