ORIGINAL RESEARCH
published: 17 September 2021
doi: 10.3389/frobt.2021.641338

A Robust Observation, Planning, and Control Pipeline for Autonomous Rendezvous with Tumbling Targets

Keenan Albee[1]†, Charles Oestreich[1,2]*†, Caroline Specht[3], Antonio Terán Espinoza[1], Jessica Todd[4], Ian Hokaj[1], Roberto Lampariello[3] and Richard Linares[1]*

[1]Space Systems Laboratory (SSL) and Astrodynamics, Space Robotics and Controls Lab (ARCLab), Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, United States, [2]Guidance & Control Group, The Charles Stark Draper Laboratory, Inc., Cambridge, MA, United States, [3]Autonomy and Teleoperation, Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Oberpfaffenhofen, Germany, [4]Human Systems Laboratory (HSL), Department of Aeronautics and Astronautics and Department of Applied Ocean Physics and Engineering, Massachusetts Institute of Technology/Woods Hole Oceanographic Institute, Cambridge, MA, United States

Accumulating space debris edges the space domain ever closer to cascading Kessler syndrome, a chain reaction of debris generation that could dramatically inhibit the practical use of space. Meanwhile, a growing number of retired satellites, particularly in higher orbits like geostationary orbit, remain nearly functional except for minor but critical malfunctions or fuel depletion. Servicing these ailing satellites and cleaning up "high-value" space debris remains a formidable challenge, but active interception of these targets with autonomous repair and deorbit spacecraft is inching closer toward reality as shown through a variety of rendezvous demonstration missions. However, some practical challenges are still unsolved and undemonstrated. Devoid of station-keeping ability, space debris and fuel-depleted satellites often enter uncontrolled tumbles on-orbit. In order to perform on-orbit servicing or active debris removal, docking spacecraft (the "Chaser") must account for the tumbling motion of these targets (the "Target"), which is oftentimes not known *a priori*. Accounting for the tumbling dynamics of the Target, the Chaser spacecraft must have an algorithmic approach to identifying the state of the Target's tumble, then use this information to produce useful motion planning and control. Furthermore, careful consideration of the inherent uncertainty of any maneuvers must be accounted for in order to provide guarantees on system performance. This study proposes the complete pipeline of rendezvous with such a Target, starting from a standoff estimation point to a mating point fixed in the rotating Target's body frame. A novel visual estimation algorithm is applied using a 3D time-of-flight camera to perform remote standoff estimation of the Target's rotational state and its principal axes of rotation. A novel motion planning algorithm is employed, making use of offline simulation of potential Target tumble types to produce a look-up table that is parsed on-orbit using the estimation data. This nonlinear programming-based algorithm accounts for known Target geometry and important practical constraints such as field of view requirements, producing a motion plan in the Target's rotating body frame. Meanwhile, an uncertainty characterization method is demonstrated which propagates uncertainty in the Target's tumble uncertainty to provide disturbance bounds on the motion plan's reference trajectory in the inertial

OPEN ACCESS

Edited by:
Behcet Acikmese, University of Washington, United States

Reviewed by:
Saptarshi Bandyopadhyay, NASA Jet Propulsion Laboratory (JPL), United States
Tommaso Guffanti, Stanford University, United States

***Correspondence:**
Keenan Albee
albee@mit.edu
Charles Oestreich
coestrei@mit.edu

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Space Robotics, a section of the journal Frontiers in Robotics and AI

Received: *14 December 2020*
Accepted: *12 July 2021*
Published: *17 September 2021*

Citation:
Albee K, Oestreich C, Specht C, Terán Espinoza A, Todd J, Hokaj I, Lampariello R and Linares R (2021) A Robust Observation, Planning, and Control Pipeline for Autonomous Rendezvous with Tumbling Targets. Front. Robot. AI 8:641338. doi: 10.3389/frobt.2021.641338

frame. Finally, this uncertainty bound is provided to a robust tube model predictive controller, which provides tube-based robustness guarantees on the system's ability to follow the reference trajectory translationally. The combination and interfaces of these methods are shown, and some of the practical implications of their use on a planned demonstration on NASA's Astrobee free-flyer are additionally discussed. Simulation results of each of the components individually and in a complete case study example of the full pipeline are presented as the study prepares to move toward demonstration on the International Space Station.

Keywords: space robotics, motion planning, robust tube MPC, visual estimation, on-orbit servicing, planning under uncertainty

1 INTRODUCTION

Tumbling objects are commonplace on-orbit. Spent rocket bodies, fuel-exhausted satellites, and space debris are all examples of potential free-tumbling objects. In a variety of sub-fields of space robotics, including on-orbit servicing and repair, active debris removal, and on-orbit assembly, the ability to dock with arbitrary tumbling objects given the limited initial knowledge of the Target object is a key capability (Flores-Abad et al., 2014). Often, these tasks are safety critical but cannot allow human-in-the-loop oversight due to the complexity of the maneuvers involved and the lack of reliable teleoperation and communication. As a result, robotic autonomous execution of rendezvous activities is a desirable capability in order to repair satellites, de-orbit debris, and more.

Consequently, autonomous docking with tumbling Targets is an active area of literature in the space robotics community, with a variety of individual algorithmic contributions over the past two decades. Some of the earliest studies dealt with motion and parameter estimation of tumbling Targets, the first step in preparing for rendezvous (Hillenbrand and Lampariello, 2005). Assuming Target knowledge, multiple early studies also proposed a variety of motion planning and control techniques, including deterministic approximate analytical trajectory optimization (Fejzi and Miller, 2008), detailed optimal control formulations (Aghili, 2008), and nonlinear optimization-based formulations (Lampariello, 2010). Work in this area progressively added greater complexity, including increased sources of uncertainty (Aghili, 2012; Sternberg and Miller, 2018) and more advanced motion planning approaches (Aghili, 2020).

Recently, some work has begun to explore the significant robotics systems integration considerations required for the integration of multiple elements of the autonomous rendezvous phase: estimation, motion planning, and control under uncertainty, to name a few (Terán Espinoza et al., 2019). Leveraging recent work in robust control and planning, the ability to provide guarantees on system performance in the real-time setting of autonomous rendezvous has brought full-fledged autonomous docking frameworks within reach (Limon et al., 2008; Majumdar and Tedrake, 2017; Buckner and Lampariello, 2018; Mammarella et al., 2018). However, the need remains for a complete autonomy pipeline for such a maneuver that is additionally robust to the most significant uncertainty sources of autonomous docking with tumbling Targets and that can operate with automatic visual estimation and motion planning components.

This study details such a framework that can account for some of the key uncertainty sources of tumbling rendezvous; specifically, the unknown Target tumble state. This framework, currently scheduled for a series of International Space Station (ISS) tests in 2021, introduces the algorithmic approach to connect these submodules to make autonomous close proximity rendezvous a reality, while additionally considering uncertainty due to imperfect knowledge of these tumbling Targets. Furthermore, this study describes the unique way in which this form of uncertainty results in an uncertain reference trajectory and discusses the limits on providing robustness guarantees. Initial results on all parts of the autonomy pipeline are presented independently, along with a full case study example of an autonomous rendezvous in a detailed simulation environment.

The remainder of this article is formulated as follows: **Section 2** outlines the autonomous rendezvous problem; **Section 3** details the varied methods needed to form the full autonomy pipeline and how these segments interact; **Section 4** presents results of individual components as well as a case study example of the full pipeline algorithm in a detailed simulation environment; and finally, **Section 5** includes a discussion of the proposed pipeline and plans for integration and future experimental testing.

2 PROBLEM FORMULATION

The problem considers a close proximity rendezvous maneuver within the last ~20–40 m of the approach operation, with the goal of safely approaching a tumbling free-floating object (the "Target") and reaching a predefined offset point fixed in the Target's body frame called the mating point (MP). It is assumed that the Target is non-cooperative and passive. The Chaser spacecraft that will perform the rendezvous begins at some initial standoff offset distance $\mathbf{r}_{\text{off}}$ and is equipped to perform visual estimation of the Target. Geometric and velocity state constraints $\mathbb{X}$ on the Chaser motion are given by the geometry and motion of the Target and the velocity and operational limits of the Chaser. The Chaser also has input limits $\mathbb{U}$ due to its thruster capability. The Chaser has rough knowledge of the

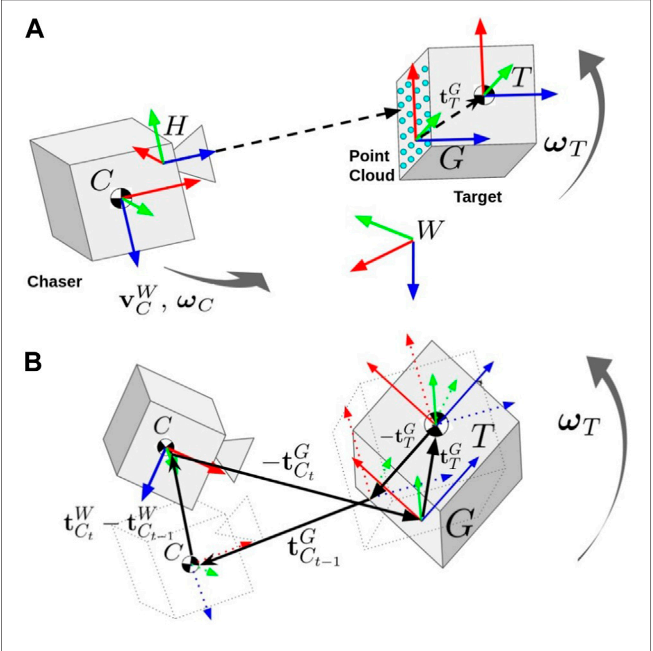


FIGURE 1 | (A) The relevant coordinate frames for estimation of the Chaser and Target states and Target inertial parameters. The Chaser's body frame ***C*** moves while the 3D ToF camera (frame ***H***) collects point cloud measurements of the Target. The Target's body frame (***T***) tumbles around its center of mass. The geometric frame (***G***) origin is defined at the centroid of the first point cloud measurement with an orientation matching the Target's body frame. There is a constant translation vector $\mathbf{t}_T^G$ between the Target's body frame and geometric frame which must be estimated. **(B)** The rotation kinematic factors that link the otherwise separate Target and Chaser pose chains in the factor graph formulation are also depicted. The bold vectors in the image must sum to zero to satisfy the factor's constraint.

Target's inertial parameters, but not its spin. A strategy is required to perform ingress to the uncertain tumbling Target and sync up with a safe and predefined MP. Ideally, any motion planning will be optimal with respect to some performance measure and maintaining some level of robust execution with respect to any plan is desirable.

2.1 Representative Satellite System of Interest

The Astrobee robots are a series of free-flying robots operating aboard the ISS with the purpose of 1) astronaut assistance and 2) microgravity autonomy research (Albee, Ekal, and Oestreich, 2020). In this way, the Astrobee program serves as a successor to the Synchronized Position Hold, Engage, and Reorient Experimental Satellites (SPHERES) in terms of guidance, navigation, and control experiments in microgravity Saenz-Otero and Miller (2005). The development of the autonomy pipeline proposed in this study has been produced in conjunction with scheduled ISS testing on the Astrobee hardware, and so uses this representative satellite system for demonstration purposes.

Astrobee is one of the first reusable microgravity testbeds capable of providing the hardware needed to test an autonomous tumbling rendezvous framework. Astrobee utilizes two impellers which are throttled in each direction to provide full holonomic propulsion capability. The robots use multiple sensors for navigation, including 2D visual cameras, 3D time-of-flight (ToF) cameras, and an inertial measurement unit (IMU). The flight software is implemented on two general-purpose processors, each running Ubuntu 16.04 and the Robotic Operating System (ROS). Two Astrobees are used in this experiment: one designated as the Chaser satellite, and the other designated as the Target object. The Chaser's forward-facing 3D ToF camera ("HazCam") and IMU provide the front-end measurements to the relative state and parameter estimation problem.

2.2 System Dynamics

Two sets of dynamics are at work: those of a partially uncharacterized/uncertain tumbling rigid body Target and of a rigid body Chaser. The Newton–Euler equations can be used to fully describe the 6 degree of freedom (DOF) motion of both systems. The translational dynamics are of particular interest for the Chaser with respect to motion robustness (i.e., collision avoidance).

2.2.1 Reference Frames

The frames of reference used in this study are defined as follows:

$$\begin{aligned}
&T\text{: Target Body Frame (centred at the centre of mass of the Target Satellite),}\\
&C\text{: Chaser Body Frame (centred at the centre of mass of the Chaser Satellite),}\\
&G\text{: Geometric Frame (arbitrary fixed body frame on the surface of the Target),}\\
&W\text{: Inertial Frame of the International Space Station.}
\end{aligned}$$

These are depicted in context of the relative state estimation problem in **Figure 1**. Note that frames are also referred to using ${}^{T}\mathcal{F}$ notation, and left superscripts indicate the frame of a vector quantity when the frame is not obvious.

2.2.2 Tumbling Target Rigid Body Dynamics

The Target is assumed to be a rigid body in a free tumble, the dynamics of which are described by the Newton–Euler equations. The full equations of motion are given by the following:

$$\begin{aligned}
\mathbf{r} &\triangleq \begin{bmatrix} r_x & r_y & r_z \end{bmatrix}^\top \\
\mathbf{q} &\triangleq \begin{bmatrix} q_x & q_y & q_z & q_\theta \end{bmatrix}^\top \\
\mathbf{v} &\triangleq \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^\top \\
\boldsymbol{\omega} &\triangleq \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^\top
\end{aligned}
\qquad \mathbf{x} \triangleq \begin{bmatrix} \mathbf{r} \\ \mathbf{q} \\ \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix}, \tag{1}$$

$$\begin{aligned}
\dot{\mathbf{r}}_{CoM} &= \mathbf{v}, \\
\dot{\mathbf{v}}_{CoM} &= \frac{\mathbf{F}}{m}, \\
\dot{\boldsymbol{\omega}} &= -\mathbf{I}^{-1}\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} + \mathbf{I}^{-1}\boldsymbol{\tau}, \\
{}^{I}_{T}\dot{\mathbf{q}} &= \frac{1}{2}\overline{H}\left({}^{I}_{T}\mathbf{q}\right)^{\top}\,{}^{T}\boldsymbol{\omega}_{IB},
\end{aligned} \tag{2}$$

where the rotational terms are of interest, namely, $\boldsymbol{\omega}$ and ${}^{I}_{T}\mathbf{q}$ describing the Target's angular velocity and attitude with respect to the local orbital (inertial) frame $\boldsymbol{W}$, respectively. The input torques $\boldsymbol{\tau}$ are set to zero for the free-tumbling scenario of interest. The inertia tensor $\mathbf{I}$ is given fully as:

$$\mathbf{I} \triangleq \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}, \tag{3}$$

where the products of inertia I_{xy}, I_{xz}, and I_{yz} are 0 in a principal axes frame. In order to solve the initial value problem (IVP) of future Target states for an inertia tensor in any arbitrary Target frame, it is necessary to know $\mathbf{q}(t_0), \boldsymbol{\omega}(t_0)$, and $\mathbf{I}$.

The inertia tensor and initial angular velocity can yield a large variety of future system behaviors. A classic example is the flat spin, where $\boldsymbol{\omega}$ is aligned with the maximum moment of inertia axis. In this case, the tumble is simply one-dimensional about this axis. In a more extreme case, the angular velocity changes drastically and suddenly when aligned with an intermediate moment of inertia axis, flipping approximately one and a half revolutions from one axis alignment to the other (Harris, 1994).

The Target is free to tumble, and dissipative effects are considered negligible over the timescales of rendezvous. Uncertainty in the Target's tumbling dynamics will result in an uncertain solution to the IVP, addressed in **Section 2.4**.

2.2.3 Chaser Translational Dynamics

The Chaser is also a 6 DOF rigid body and obeys the same set of dynamics as the Target, albeit with its own unique parameters. Since the translational dynamics of interest are purely linear they may be written in state space form, as follows:

$$\mathbf{z}^+ = \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{u}, \tag{4}$$

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

$$\mathbf{B} = \begin{bmatrix} \frac{dt^2}{2m} & 0 & 0 \\ 0 & \frac{dt^2}{2m} & 0 \\ 0 & 0 & \frac{dt^2}{2m} \\ \frac{dt}{m} & 0 & \\ 0 & \frac{dt}{m} & 0 \\ 0 & 0 & \frac{dt}{m} \end{bmatrix}, \tag{6}$$

$$\mathbf{z} = \begin{bmatrix} r_1 & r_2 & r_3 & v_1 & v_2 & v_3 \end{bmatrix}^\top, \tag{7}$$

where $\mathbf{z}$ indicates a nominal state which is derived from a deterministic dynamics model of the Chaser ($\mathcal{M}_C$), $\mathbf{z}^+$ indicates a discrete state update, and dt is the timestep. Note that orbital dynamics effects (e.g., the Clohessy–Wiltshire equations) are not considered in this particular problem statement due to the short timescales and distances under consideration in the system of interest (1.5 meters and 3 minutes). However, the linear CHW dynamics could easily be substituted into the system dynamics models of multiple components of the pipeline detailed in subsequent sections, which is intended for use in the range of ~20–40 m of the Target, and with motion plan timescales of up to approximately 10 [minutes], as demonstrated in Stoneman and Lampariello (2016). If desired, the CHW equations are additionally applicable for distances up to a few kilometers Fehse (2003). However, the control forces at play noted in **Section 2.3** are dominating with respect to any orbital mechanics or perturbation forces.

The Chaser and Target Astrobees each have a mass of 9.58 kg and inertial parameters $I_{xx} = 0.153$, $I_{yy} = 0.143$, and $I_{zz} = 0.162\,\mathrm{kg\,m^2}$. The robots are capable of providing up to 0.4 N of force and up to 0.04 Nm of torque. Additionally, the tumbling motion of the Envisat satellite is of interest ESA (2015). The inertial parameters of Envisat are also provided in the Envisat body frame:

$$\mathbf{I}_{ES} = \begin{bmatrix} 17023.3 & 397.1 & -2171.4 \\ 397.1 & 124825.7 & 344.2 \\ -2171.4 & 344.2 & 129112.2 \end{bmatrix} [\mathrm{kg/m^2}]. \tag{8}$$

The Envisat tumble can be mimicked by an actively controlled Astrobee with a different inertia tensor—precomputed tumble types can be actively tracked to simulate the tumble of a different object.

2.3 Chaser Motion Constraints

As it is typical in on-orbit missions, the maneuver is constrained by the Target geometry, as well as the Chaser's limits on velocity and input. Constraints are additionally drawn from safety considerations including collision avoidance and plume impingement of the Target. The thruster authority available to the Chaser from the on-board thrusters dictates the available Chaser linear force and rotational torque.

2.3.1 Constraints for the System of Interest

For the particular scenario of interest involving Astrobee, the motion constraints also result from the restrictions on Chaser movement dictated by the space available in the ISS Japanese Experiment Module (JEM) which houses the Astrobee platform, resulting in additional position and velocity constraints on the Chaser.

Considering that the approach tasks are restricted to the JEM, the robots are confined to move within a workspace of dimensions of approximately $1.4 \times 2.8 \times 3.4$ m. The Astrobee specifications indicate that robots are restricted to move at a rate of 0.1 m/s in each spatial dimension, and the thrusters can provide forces $\mathbf{f} = [f_x, f_y, f_z]^\top = [0.452, 0.216, 0.257]^\top$ N and torques $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^\mathrm{T}$ Nm approximately equal to one-tenth of the force available in each spatial dimension (NASA, 2015).

While the Astrobee robots have a fairly simple shape, the inertia of the Target satellite has been simulated, as discussed in the previous section, to match the more complex shape of the Envisat satellite. It is further assumed that this complex shape is the actual shape of the Target spacecraft. The Target Astrobee robot, therefore, wears a virtual geometry, similar to the real geometry of Envisat. As can be seen in **Figure 2**, the virtual

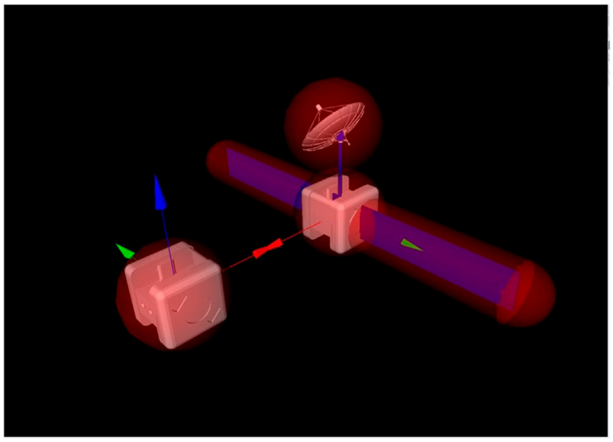


FIGURE 2 | Scene depicting the approach of the Chaser to the Target Astrobee. The virtual geometry of the Target Astrobee is shown.

geometry of the Target Astrobee includes a large solar panel and antenna appendages. This makes the collision avoidance task non-trivial. This complexity is tackled by creating a geometric model ($\mathcal{M}$) for each of the Chaser and Target shapes and using these to conduct collision detection. This geometric model appears in **Figure 2** as translucent red shapes encapsulating the Target and Chaser geometries. The penetration depth (d) is then used to formulate the collision avoidance constraint (see **Section 3.3**). **Figure 2** additionally shows how the x-axis of the Chaser body frame is required to point toward the center of mass of the Target robot, a field of view constraint. The Chaser must, therefore, rotate its attitude as well as translate in the course of the approach maneuver. This finally indicates that a torque constraint is present in tandem with the force limitations, limiting how fast the Chaser can reorient itself to maintain the field of view constraint.

2.4 Uncertainty Sources

Uncertainty enters the problem in a unique fashion: normally, additive uncertainty is thought of as a representation of unknowns in a system's dynamics, environmental disturbances, and "noise." However, in this problem the uncertainty of interest is due primarily to the reference trajectory itself (Buckner and Lampariello, 2018). As the motion planner must use the predicted Target motion to successfully reach the MP and avoid collisions, deviations in the Target motion due to parameter uncertainty in the Target's initial tumbling state must be accounted for.

First, an approach trajectory in the Target's body frame ${}^{T}\mathbf{x}_{ref}(k = 0, 1, \ldots N)$ is provided by a motion planner, guaranteeing collision avoidance. This trajectory is rotated into the inertial frame using the nominal predicted Target attitude ${}^{I}_{T}\bar{\mathbf{R}}(k)$ at each timestep, as given below:

$$ {}^{I}\mathbf{x}_{ref}(k) = {}^{I}_{T}\bar{\mathbf{R}}\left({}^{I}_{T}\bar{\mathbf{q}}(k), \bar{\boldsymbol{\omega}}_{IT}(k), \bar{\mathbf{I}}\right)^{T}\mathbf{x}_{ref}(k), \tag{9} $$

which is a function of the initial Target state estimate $({}^{I}_{T}\bar{\mathbf{q}}(0), \bar{\boldsymbol{\omega}}_{IT}(0))$, inertial parameter estimates or prior knowledge $\bar{\mathbf{I}}$, and the tumbling dynamics.

The nominal inertial trajectory ${}^{I}\mathbf{x}_{ref,0}(k = 0, 1, \ldots N)$ provided by the motion planner guarantees feasibility and collision avoidance based on the nominal predicted Target motion. However, this predicted motion is subject to initial estimation errors and Target inertia parameter uncertainty. As the controller performs the approach maneuver, the real collision-free trajectory in the inertial frame

$$ {}^{I}\mathbf{x}_{ref}(k) = {}^{I}_{T}\hat{\mathbf{R}}\left({}^{I}_{T}\mathbf{q}(k), \boldsymbol{\omega}_{IT}(k), \mathbf{I}\right)^{T}\mathbf{x}_{ref}(k) \tag{10} $$

will differ from the nominal trajectory due to the real Target motion. This creates an additive disturbance to the system in the inertial frame dependent on initial estimate inaccuracy, the inertial parameter estimates or prior knowledge, and the rate and accuracy of online updates,

$$ {}^{I}\mathbf{w}(k) = {}^{I}\mathbf{x}_{ref}(k) - {}^{I}\mathbf{x}_{ref,0}(k). \tag{11} $$

As such, the controller should strive to track ${}^{I}\mathbf{x}_{ref}(k)$ in order to maintain the guarantees of the nominal body frame trajectory. However, for high levels of ${}^{I}\mathbf{w}(k)$, the Chaser's available actuation will not be capable enough to stabilize this disturbance, thus removing robustness guarantees from the docking trajectory—this is an interesting case noted in **Section 3.5**.

3 APPROACH AND METHODS

3.1 The Autonomy Pipeline: Concept of Operations

The approach taken can be broken into several distinct phases, depicted pictorially in **Figure 3**. First, the Chaser vehicle uses its time in the holding standoff position within approximately ~20–40 m of the Target to gather observations of the Target tumble state and produce estimates of its angular velocity and attitude as well as its principal axes of rotation, as discussed in **Section 3.2**. Next, the Target tumble may be propagated using the nominal Target model, enabling the motion planning process described in **Section 3.3** and the calculation of an uncertainty bound using the statistics of these estimates, as in **Section 3.4**. In **Section 3.5**, the maneuver is executed using tube-based robust model predictive control to retain performance guarantees. Finally, the MP is reached and the mission concludes. The algorithmic components in each of these phases are depicted in **Figure 4**, delineating the flow of inputs and outputs to each component of the algorithm.

3.2 Relative Navigation and Target Characterization

As outlined in **Section 3.1**, on-orbit inspection of an unknown spacecraft is not only a required component of currently proposed missions (Sullivan et al., 2015) but also of the more general spacecraft close-proximity operations (Terán Espinoza et al., 2019). There have been numerous research efforts to implement simultaneous localization and mapping (SLAM) approaches for estimating the Chaser and Target states as well as the Target's inertial parameters (Tweddle et al., 2015;

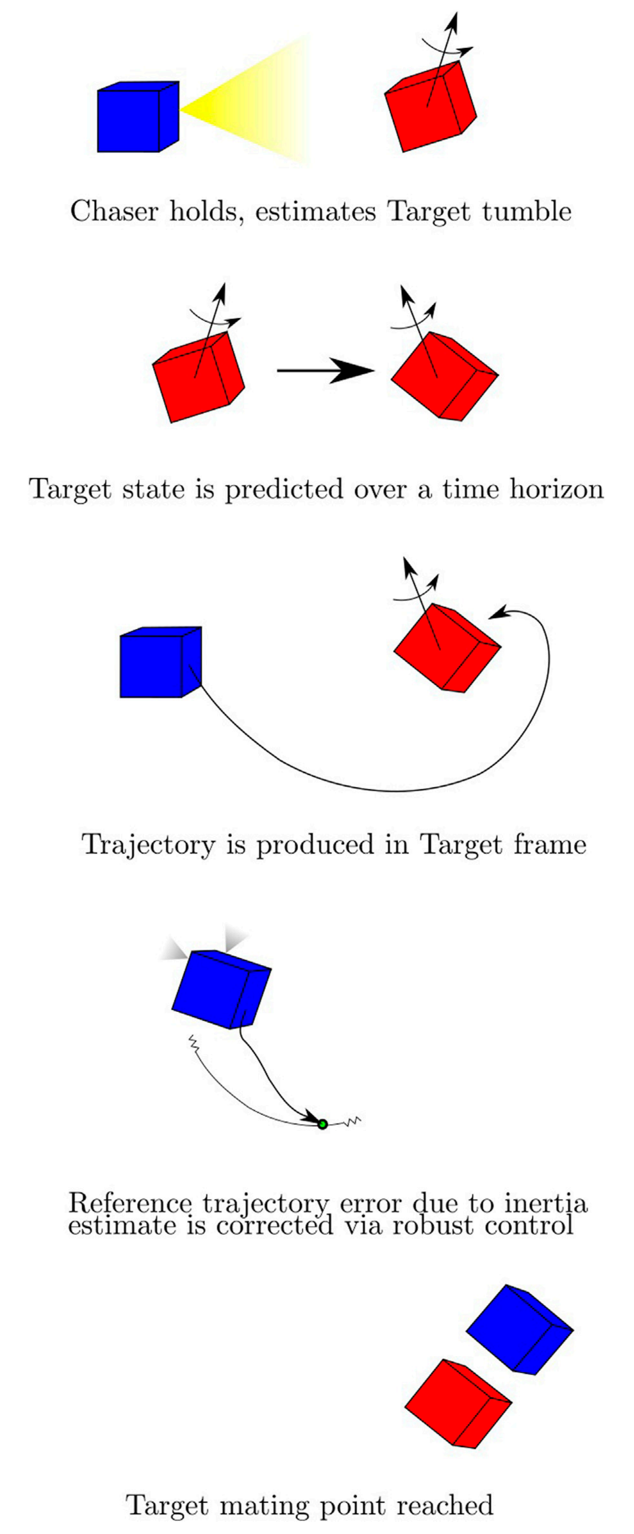

Chaser holds, estimates Target tumble
Target state is predicted over a time horizon
Trajectory is produced in Target frame
Reference trajectory error due to inertia estimate is corrected via robust control
Target mating point reached

FIGURE 3 | The concept of operations for the autonomous docking procedure with Chaser (blue) and Target (red). Following a standoff estimation phase a motion planner uses an estimate of future Target attitude to create a plan in the inertial frame, which is tracked by a robust controller. Uncertainty is converted from Target parameter uncertainty to Chaser reference trajectory uncertainty bounds.

Setterfield et al., 2018a; Terán Espinoza, 2020). These studies utilize incremental smoothing and mapping (iSAM) (Kaess et al., 2008; Kaess et al., 2012) with stereo camera and IMU measurements to recover both the inspector and the Target state estimates in real time. Microgravity experiments on the ISS-based SPHERES testbed indicated the success and potential of these algorithms (Fourie et al., 2014; Tweddle et al., 2016). As such, this pipeline adapts the framework of Terán Espinoza (2020) for the inspection of an unknown tumbling Target using Astrobee's sensor suite to provide SLAM front-end measurements. The full SLAM framework is implemented in Astrobee's simulation environment via ROS.

The remainder of this section describes the specific details of the portion of the scenario handled herein (e.g., sensor suite), the mathematical formulation used to formalize the problem (factor graphs), the front-end components developed for information extraction (e.g., point cloud registration), the back-end components used for sensor fusion (e.g., iSAM), and the accompanying problems resolved in incorporation with the pipeline.

3.2.1 Scenario Description

It is assumed that the Chaser satellite has a sensor suite capable of providing internal IMU measurements and external measurements that can be interpreted by a front-end as the Target pose. The Chaser Astrobee is equipped with the following sensors that fulfill these requirements: an IMU for ego-motion estimation, received at 62.5 Hz; and a 3D time-of-flight (ToF) camera to return 3D information of the Target for feature extraction and tracking purposes, received at 5 Hz. The 3D ToF camera is forward-facing on Astrobee and has a resolution of 224 × 171 pixels.

Using the acquired per-frame IMU and 3D depth information, the goal is to incrementally estimate the dynamic states of the Target and Chaser.

$$\begin{aligned}\text{Target} &: ({}^{W}\mathbf{T}_{T}, {}^{T}\boldsymbol{\omega}_{T}),\\ \text{Chaser} &: ({}^{W}\mathbf{T}_{C}, {}^{W}\mathbf{v}_{C}, {}^{C}\boldsymbol{\omega}_{C}),\end{aligned}$$

where $\mathbf{T}$ represents complete pose (position and attitude) information. The problem is formulated as a factor graph SLAM problem using odometry measurements from the ToF camera and the Chaser's IMU. An overview of relevant coordinate frames for the SLAM problem is given in **Figure 1**.

3.2.2 Factor Graphs and Problem Formulation

A bipartite factor graph connects all relevant variables and measurement factors for the relative state estimation problem (**Figure 5**). The graph is composed of independent Target and Chaser pose chains. Nodes represent unknown variables (the Chaser and the Target poses) and are connected by probabilistic factors that arise from odometry measurements. The Target motion ${}^{G}\mathbf{T}_{C_n}$ is measured using a depth-based odometry, while the Chaser motion ${}^{W}\mathbf{T}_{C_n}$ is measured using an IMU-based odometry. The two chains are connected via rotational kinematic factors that disambiguate the Chaser and Target's motion, even while the Chaser is moving. Each measurement

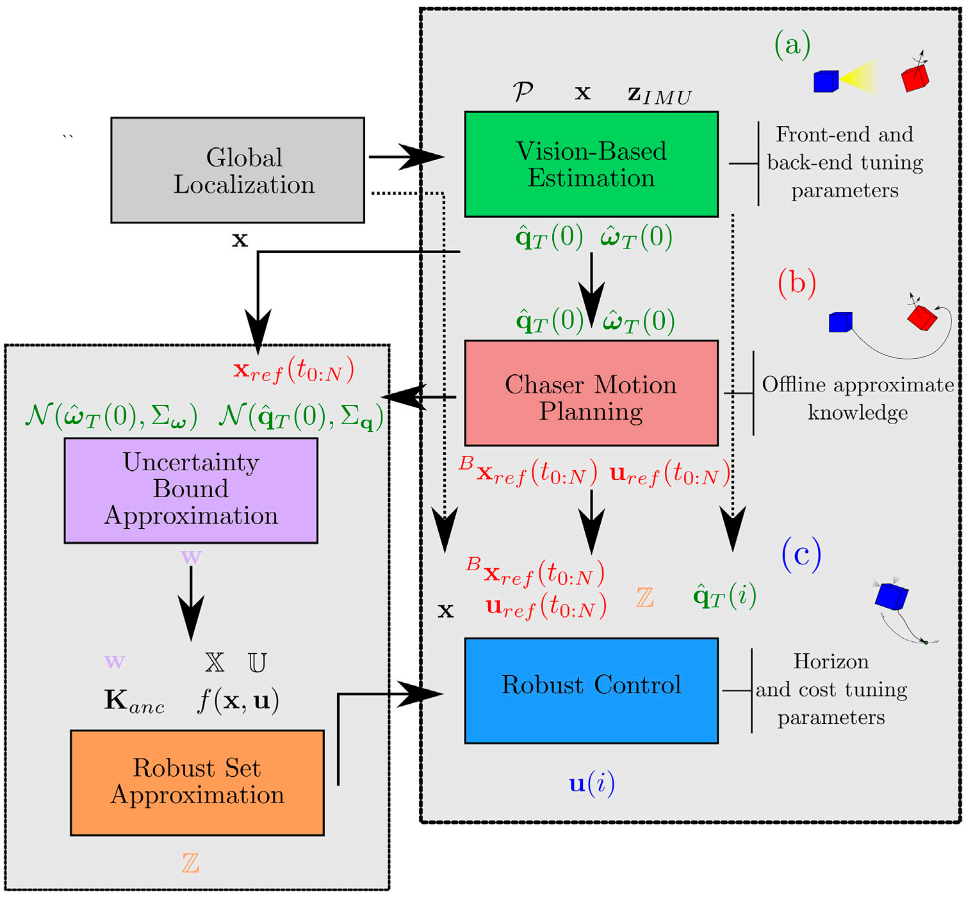

(a)
$\mathcal{P}$ $\mathbf{x}$ $\mathbf{z}_{IMU}$
Global Localization
Vision-Based Estimation
Front-end and back-end tuning parameters
$\mathbf{x}$
$\hat{\mathbf{q}}_T(0)$ $\hat{\boldsymbol{\omega}}_T(0)$
(b)
Chaser Motion Planning
Offline approximate knowledge
$\mathbf{x}_{ref}(t_{0:N})$
$\mathcal{N}(\hat{\boldsymbol{\omega}}_T(0),\Sigma_{\boldsymbol{\omega}})$ $\mathcal{N}(\hat{\mathbf{q}}_T(0),\Sigma_{\mathbf{q}})$
Uncertainty Bound Approximation
$^B\mathbf{x}_{ref}(t_{0:N})$ $\mathbf{u}_{ref}(t_{0:N})$
$\mathbb{W}$
(c)
$\mathbf{x}$ $\mathbb{Z}$ $\hat{\mathbf{q}}_T(i)$
$\mathbb{W}$ $\mathbb{X}$ $\mathbb{U}$
$\mathbf{K}_{anc}$ $f(\mathbf{x},\mathbf{u})$
Robust Control
Horizon and cost tuning parameters
Robust Set Approximation
$\mathbf{u}(i)$
$\mathbb{Z}$

FIGURE 4 | The full algorithmic pipeline, showing major inputs and outputs of each component algorithm. Inputs are shown above, and outputs are shown below. **(A)** shows the SLAM and polhode analysis visual estimation of the Target, **(B)** shows the online motion planning component, and **(C)** shows the robust model predictive controller and its supporting algorithms.

is associated with a predefined noise model based on the sensor's characteristics. Through probabilistic inference on the factor graph, one can minimize the error associated with the factors and smooth all state estimates over time.

3.2.2.1 Front-End Modules for Information Extraction

The Chaser's ToF camera provides a 3D point cloud of the scene at regular intervals. This depth data provides measurements of the Target motion relative to the Chaser which are then used in the Target's pose chain within the factor graph. The Teaser++ solver (Yang et al., 2020) is used to solve the point cloud registration problem between frames, thus providing pose odometry measurements with respect to the geometric frame $\Delta^G\mathbf{T}_C(t-1;t)$. The geometric frame G is an arbitrary body-fixed frame on the Target with the origin placed at the centroid of the initial features gathered, and thus has some fixed offset from frame T. The registration problem is carried out via three steps: 1) detection of 3D features in the current frame, 2) matching of features between the previous and current frames, and 3) robustly solving the registration problem using the found matches. Fast point feature histograms (FPFH) are selected as the 3D feature descriptor for detecting and matching (Rusu et al., 2009). For a pair of point clouds $\mathbf{z}_i^\gamma$ and $\mathbf{z}_j^\gamma$ at timesteps i and j, the result is a pose odometry measurement

$$\mathbf{z}_{ij}^\gamma = \left\{\mathbf{z}_{\mathbf{R}_{ij}}^\gamma, \mathbf{z}_{\mathbf{t}_{ij}}^\gamma\right\} \in SE(3), \tag{12}$$

of the Chaser with respect to the Target's $\boldsymbol{G}$ frame.

The specific front-end for the Astrobee system of interest will be addressed here, though similar front-ends could be substituted for different satellite sensor suites. The Chaser's IMU serves as the source of measurements for the Chaser's pose chain in the factor graph. Since Astrobee's IMU operates at a much faster rate than the ToF camera, the IMU preintegration theory from (Lupton and Sukkarieh, 2011; Carlone et al., 2014; Forster et al., 2016) is leveraged. The set of IMU readings $\mathbf{z}_{\mathrm{IMU}_{ij}} = \{\mathbf{z}_{\omega_\tau}, \mathbf{z}_{a_\tau}\}_{\tau=i}^{j}$ are bundled between keyframes at t_i and t_j as a relative transformation between the navigation states, allowing the collected information to be summarized as a single measurement of the following form:

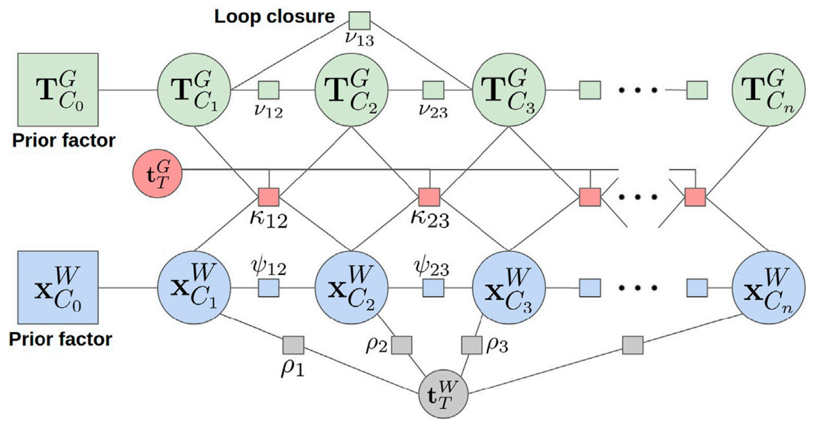


FIGURE 5 | The full factor graph used for Chaser and Target state estimation. Nodes represent pose variables and are connected by ToF camera pose odometry, IMU odometry, range/bearing measurements, and rotation kinematic factors.

$$\Delta \boldsymbol{x}_{ij} = \left\{\Delta \boldsymbol{R}_{ij}, \Delta \boldsymbol{t}_{ij}, \Delta \boldsymbol{v}_{ij}, \Delta \boldsymbol{b}_{ij}\right\}, \tag{13}$$

where $\boldsymbol{R}$ denotes the rotational part of the Chaser's state, $\boldsymbol{t}$ the translation, $\boldsymbol{v}$ the velocity, and $\boldsymbol{b}$ the bias variables for the IMU.

Finally, range and bearing measurements of the Target are provided by the ToF camera point clouds. These measurements are defined as:

$$\mathbf{z}_j^{\rho} = \left\{z_j^{\rho d} \in \mathbb{R}^+, \mathbf{z}_j^{\rho b} \in S^2\right\}, \tag{14}$$

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 two-sphere manifold.

3.2.2.2 Factor Graphs for Probabilistic Inference

As stated above, the Target's pose chain is established using features from the Target's depth information, correlated across subsequent frames. The point cloud-based odometry factor ν added to the graph is formulated as a relative pose measurement constraint, as given below:

$$\nu_{ij}\left({}^G\mathbf{T}_{C_i}, {}^G\mathbf{T}_{C_j}\right) \propto \exp\left\{\frac{1}{2}\left\|\left({}^G\mathbf{T}^G_{C_i} \cdot {}^G\mathbf{T}_{C_j}\right) \ominus \mathbf{z}^{\nu}_{ij}\right\|^2_{\Sigma_{\nu_i}}\right\}, \tag{15}$$

where $\|(\cdot)\|^2_{\Sigma_\nu}$ is the weighted Mahalanobis distance using the noise model parameter Σ_ν.

Loop closures on this pose chain can be implemented in the same manner by taking matching point cloud features across non-successive frames. Practically speaking, a randomly chosen past frame can be checked for loop closures with the current frame. If there are enough matches, the odometry is computed and added to the graph.

The Chaser's pose chain is generated from IMU factors. To carry out inertial navigation using the IMU's information, a binary IMU factor ψ is created of the form given below:

$$\psi_{ij}\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \propto \exp\left\{-\frac{1}{2}\left\|\left(\boldsymbol{r}^\top_{\Delta R_{ij}}, \boldsymbol{r}^\top_{\Delta t_{ij}}, \boldsymbol{r}^\top_{\Delta v_{ij}}, \boldsymbol{r}^\top_{\Delta b_{ij}}\right)\right\|\Sigma_{\psi_{ij}}\right\}, \tag{16}$$

where $\boldsymbol{r}$ represents the residual errors for each of the terms shown in **Eq. 13**. This factor is placed between two subsequent navigation states $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in the inertial navigation pose graph chain to constrain their relative motion using the IMU information.

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

$$\rho_j \propto \exp\left\{-\frac{1}{2}\left\|\begin{bmatrix} h_d\left({}^W\mathbf{x}_{C_j}, {}^W\mathbf{t}_T\right) - z_j^{\rho d} \\ h_b\left({}^W\mathbf{x}_{C_j}, {}^W\mathbf{t}_T\right) \ominus \mathbf{z}_j^{\rho b}\end{bmatrix}\right\|^2_{\Sigma_\rho}\right\}, \tag{17}$$

where h_d and h_b are the measurement models for range and bearing.

Rotation kinematic constraints are used to disambiguate the motion of the Target and Chaser satellites. The rotation kinematic factor enforces a zero sum vector addition between temporally equivalent time step pairs in the Target and Chaser pose chains, given the assumption that the Target is unperturbed and translationally stationary. The rotation kinematic factor is built by following the vector addition shown in **Figure 1**, to give:

$$\kappa_{t-1,t}\left(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t, {}^G\mathbf{T}_{C_{t-1}}, {}^G\mathbf{T}_{C_t}\right) \propto \exp\left\{-\frac{1}{2}\left\|{}^W\boldsymbol{R}_{C_{t-1}} {}^{G_{t-1}}\boldsymbol{R}^\top_{C_{t-1}}\left({}^{G_{t-1}}\boldsymbol{t}_{C_{t-1}/G_{t-1}} - {}^G\boldsymbol{t}_{C/G}\right) + {}^W\boldsymbol{R}_C {}^{G_t}\boldsymbol{R}^\top_{C_t}\left({}^G\boldsymbol{t}_{C/G} - {}^{G_t}\boldsymbol{t}_{C_t/G_t}\right) + \left({}^W\boldsymbol{t}_{C_t/W} - {}^W\boldsymbol{t}_{C_{t-1}/W}\right)\right\|^2_{\Sigma_{\rho_{t-1}}}\right\}. \tag{18}$$

Notice that the rotation kinematic factor is dependent on estimating the constant offset between the geometric frame and Target's body frame, ${}^G\mathbf{t}_T$ (3). This translation is added as a single unknown variable to the factor graph that is connected to all rotation kinematic factors. The full factor graph, including the Target's pose chain, Chaser's pose chain, and rotation kinematic factors is shown in **Figure 5**.

While the factor graph shown thus far exactly represents the mathematical formulation of the problem being solved, the actual online solution is computed using a different, but directly related, data structure called the Bayes tree (Kaess et al., 2010; Fourie et al., 2020; Terán Espinoza, 2020). By recycling computations at each time step, a full SLAM solution can be computed fast and accurately every time new information is received.

3.2.3 Estimation of Angular Velocity and Target Principal Axes

The above factor graph formulation allows the Chaser to estimate its own navigational state and the Target's attitude at each timestep. These estimates are then processed to provide angular velocity measurements of each spacecraft and also determine the Target's principal axes of rotation.

The instantaneous angular velocity estimates ${}^{G}\boldsymbol{\omega}_G$ and ${}^{C}\boldsymbol{\omega}_C$ are extracted at each timestep via the relative rotation between temporally subsequent poses in the world frame. Using the factor graph variables,

$$ {}^{C}\boldsymbol{\omega}_{C_i} = \frac{1}{\Delta t_{ij}} \mathrm{Log}\left(\left({}^{W}\mathbf{R}_{C_i} \right)^{\top} {}^{W}\mathbf{R}_{C_j} \right), \tag{19} $$

$$ {}^{G}\boldsymbol{\omega}_{G_i} = \frac{1}{\Delta t_{ij}} \mathrm{Log}\left(\left({}^{W}\mathbf{R}_{T_i} \left({}^{G}\mathbf{R}_{T_i} \right)^{\top} \right)^{\top} {}^{W}\mathbf{R}_{T_i} \left({}^{G}\mathbf{R}_{T_i} \right)^{\top} \right), \tag{20} $$

where Log acts as shorthand for the SO(3) logarithmic map log followed by the inverse skew symmetric matrix operator.

Note that the Target's orientation and angular velocity estimates are with respect to the arbitrarily initialized $\boldsymbol{G}$ frame. While the ${}^{G}\mathbf{t}_T$ variable is solved for in the factor graph, the orientation ${}^{G}\mathbf{R}_T$ 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. Through polhode analysis,[1] by Setterfield et al. (2018b), this misaligned angular velocity vector in the $\boldsymbol{G}$ frame is rotated such that the polhode's new orientation produces central conic projections onto the XY, YZ, and XZ planes.

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 Setterfield et al. (2018b). The final result is an optimized orientation ${}^{G}\mathbf{R}_P$ that best aligns the measured angular velocity values into the newly defined $\boldsymbol{P}$ frame based upon the Target's principal axes of inertia. The rotation between the principal axes frame and the defined Target frame $\boldsymbol{T}$ where Envisat's inertia tensor is defined, ${}^{T}\mathbf{R}_P$, is now used to determine the Target's orientation and angular velocity with respect to the $\boldsymbol{T}$ frame, as given below:

$$ {}^{T}\mathbf{R}_W = {}^{W}\mathbf{R}_C \left({}^{G}\mathbf{R}_C \right)^{\top} {}^{G}\mathbf{R}_P \left({}^{T}\mathbf{R}_P \right)^{\top}, \tag{21a} $$

$$ {}^{T}\boldsymbol{\omega}_T = \left({}^{G}\mathbf{R}_P \right) \left({}^{T}\mathbf{R}_P \right)^{\top} {}^{G}\boldsymbol{\omega}_G. \tag{21b} $$

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

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 if they are not already known. By leveraging the closed-form solution for rigid body motion based on the Jacobi elliptic functions Hurtado and Satak (2011), a second procedure from Setterfield et al. (2018b) 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. An alternative approach for identifying the inertia parameters, as well as the state parameters of a tumbling Target, can be found in Lampariello et al. (2021). The same study also presents a method for performing long-term motion prediction, while accounting for the state and inertial parameter dispersity which results from the parameter identification. This allows for robust motion prediction of the Target's tumbling motion even if there is no available inertia tensor. The inclusion of such methods in the pipeline proposed here is a subject of future study.

3.3 Chaser Motion Planning

The motion planner considers only the nominal motion of the Target (propagated within the motion planner using Boost (Mascellani, 2019)) to generate nominal trajectories to control the Chaser. The method is based on Stoneman and Lampariello (2016), in which it is made evident that, if present, large appendages play an important role in the motion planning task. Similar approaches can be found in the literature, such as (Ventura et al., 2015; Park et al., 2017; Sternberg and Miller, 2018). In these studies, the convex optimization-based approach does not retain convergence guarantees. In Virgili-Llop et al. (2019), a covexification of the nonlinear collision avoidance constraint allows fast online planning for any relevant tumbling state of the Target, while retaining convergence guarantees. It is noted, however, that the method therein is based on a three-stage planning approach, necessary to account for the case in which the MP is inside the convex hull, as well as on repeated replanning, possibly for a complete period of motion of the Target, to identify the global minimum for the query at hand.

The Chaser trajectory $\mathbf{x}_{ref}(\mathbf{p}, t)$ is composed of three b-splines and their first three derivatives, one b-spline for each spatial dimension. Each spline has n free parameters $\mathbf{p}$ and is sampled at m via points. The Chaser pose $\mathbf{x}_{ref}(t)$ considers the Chaser's position and orientation such that $\mathbf{x}_{ref}(\mathbf{p}, t) = [\mathbf{x}_{ref,p}, \mathbf{x}_{ref,o}] = [x_{ref,x}, x_{ref,y}, x_{ref,z}, q_{ref,x}, q_{ref,y}, q_{ref,z}, q_{ref,\theta}]$. At the final via point, the Chaser is required to meet the MP, matching the position and velocity of the MP in the inertial frame, and must be oriented such that the Chaser body frame's positive x-axis points towards the Target's center of mass.

The approach maneuver is formulated here as a nonlinear optimization problem (NLP) to ensure feasibility of the generated trajectories with respect to the motion constraints. The NLP minimizes the mechanical energy cost $J(\mathbf{p})$, for a fixed final time t_f, and as a function of the free parameters of these b-splines subject to position, velocity, actuation, plume impingement, and collision avoidance constraints at each via point:

$$
\begin{aligned}
&\min_{\mathbf{p}} J(\mathbf{p}), \\
&\quad s.t. \quad c_{position}\left(\mathbf{x}_{ref,p}(\mathbf{p}), t\right) \leq 0, \\
&\qquad\quad c_{velocity}\left(\dot{\mathbf{x}}_{ref,p}(\mathbf{p}), t\right) \leq 0, \\
&\qquad\quad c_{force}\left(\ddot{\mathbf{x}}_{ref,p}(\mathbf{p}), t\right) \leq 0, \\
&\qquad\quad c_{torque}\left(\mathbf{x}_{ref}(\mathbf{p}), \dot{\mathbf{x}}_{ref}(\mathbf{p}), \ddot{\mathbf{x}}_{ref}(\mathbf{p}), t\right) \leq 0, \\
&\qquad\quad c_{collision}\left(\mathbf{x}_{ref,p}(\mathbf{p}), t\right) \leq 0, \\
&\qquad\quad c_{plume}\left(\mathbf{r}, \mathbf{q}, \mathbf{x}_{ref}(\mathbf{p}), \ddot{\mathbf{x}}_{ref,p}(\mathbf{p}), t\right) \leq 0, \\
&\quad for\ t = 0, \ldots, m-1.
\end{aligned}
\tag{22}
$$

In this NLP, the first four constraints in the list are box constraints using the values outlined in **Section 2.3**. Additionally, $\mathbb{X}$ and $\mathbb{U}$ constraints are further tightened to provide a conservative bound on potential tightening required by the robust MPC nominal controller. It should also be noted that, while an optimal trajectory in the spatial dimensions is sought $\mathbf{x}_{ref,p}(\mathbf{p}, t)$, the orientation of the Chaser along this optimal trajectory must also be sought $\mathbf{x}_{ref,o}(\mathbf{p}, t)$, to ensure that its x-axis always points in the direction of the Target's center of mass (field of view constraint). It can thence be determined if the necessary torque related to this pointing rotational motion is permitted by the Chaser thrusters.

Recalling the discussion of the collision avoidance constraint from **Section 2.3**, the collision constraint is given by the penetration depth d such that

$$c_{collision}\left(\mathbf{x}_{ref,p}(\mathbf{p}, t)\right) = d\left(\mathbf{q}, \mathbf{x}_{ref,p}(\mathbf{p}), \mathbf{M}, t\right) \leq 0, \tag{23}$$

which is an iterative, nonlinear function. Note that the Chaser is modeled as a sphere, therefore relieving the need to consider its orientation in this constraint. Geometric modeling and collision detection is implemented using the Open Dynamics Engine, and additionally tested using the Gazebo simulation environment, detailed in **Section 4** (Smith, 2009).

In order to reduce the computation time, an online planning method has been devised which provides a warm start to the optimization problem. This method makes use of a precompiled look-up table (LUT) of corresponding pairs of initial conditions for the Target's motion and composite optimization parameter solutions $\mathbf{p}$ for the three b-splines to identify a suitable initial guess for the planner. By inspecting the members of the LUT to select the initial guess, it is possible to quickly reach the local minimum in solving (**Eq. 22**).

This LUT must be generated offline, due to the computational burden involved. Its generation requires approximate knowledge of the inertia of the Target (up to a constant multiplying factor) and geometry of the Target. If the Target is initially entirely unknown with respect to these criteria, this missing information must first be identified and relayed to ground. The inertia can be identified using one of a variety of methods, including those detailed in Setterfield and Miller (2017); Teran, (2021); Lampariello et al. (2021) and the geometry can be determined using 3D reconstruction methods. However, aside from this modeling information (which is often given ahead of time, at least approximately) the motion planning method is entirely online-computable. The resulting NLP is implemented and solved using the SLSQP algorithm provided in the nonlinear optimization package NLopt (Johnson, 2017).

3.4 Uncertainty Bound Definition

The disturbance level of major uncertainty sources must be defined in order to provide a form of robustness guarantee. The tube model predictive control (MPC) relies on a known bound of the additive disturbances; in the context of the tumbling rendezvous maneuver, this means that limits must be determined for the magnitude of disturbances defined by **Eq. 11**, the primary uncertainty source of interest. Given a nominal reference trajectory provided by the motion planner and uncertainty levels of the Target state estimates a series of Monte Carlo trials is computed to approximate the maximum disturbance over the course of the trajectory in a manner similar to (Buckner and Lampariello, 2018). Additionally, the effect of online-updating (and the frequency of online updating) is accounted for in the approximation procedure. For each trial, the nominal initial Target state is perturbed within the state estimate uncertainty levels, based on the statistics of the state estimates. This creates a Target tumbling trajectory that differs from the predicted motion used by the motion planner. The disturbance is then measured at each timestep of the Monte Carlo trial trajectory by comparing the nominal reference trajectory to the "real" trajectory that arises from **Eq 10**. This difference constitutes the defined disturbance in **Eq. 11**. The repeated Monte Carlo trials build up a sampling of disturbance values seen across all trajectories and, like all Monte Carlo evaluations, increase the modeling accuracy of the true distribution with a greater number of trials. From there, the uncertainty bound is defined via the largest disturbance magnitudes in each state dimension. This is a conservative approximation, and can be relieved if a lower σ value is desired. The full algorithm to determine the uncertainty bound is detailed in the algorithm of **Figure 6**. Note that the current implementation assumes all disturbances arise from initial state estimate errors. This algorithm can be easily modified to include disturbances from inertia parameter error as well, if desired. The Monte Carlo uncertainty bound determination process is shown in **Figure 6**.

3.5 Robust Tube Model Predictive Control

Model predictive control (MPC) is a commonly used control technique which is particularly notable for its ability to approximately optimally control general dynamical systems under constraints. While proofs for MPC stability under certain conditions (Mayne et al., 2011) abound, guarantees for stochastic systems are generally lacking. Tube MPC is a notable exception, providing robustness guarantees when bounded additive uncertainty is encountered in the system dynamics. Using tube MPC, a portion of control authority is reserved for robust actuation, often in a simple feedback form to counter disturbances. The guarantee obtained is one of tube robustness—if a system starts in a tube of possible states, it remains within a tube around a nominal trajectory. These tubes are formed around the nominally planned MPC trajectory, and the motion of the system can be thought of as a composition of these planned "safe sets."

Tube MPC methods exist for both nonlinear and linear systems with additive bounded uncertainty—for this particular problem linear tube MPC is of interest for the linear translational satellite (double integrator) dynamics. (Attitude control is not considered under this robustness paradigm: the Euler equations are nonlinear and pointing constraints under the tumbling uncertainty are not as important—line of sight constraints are somewhat generous and it is assumed that motion plans produce collision-free trajectories for a max-radius ball about the system's center of mass.) In essence, two controllers must be produced: 1) a nominal MPC, operating under a modified set of constraints to account for worst-case uncertainty; 2) an ancillary or "disturbance rejection" controller that provides robustness to aleatoric uncertainty. Given a reference trajectory, tube MPC will repeatedly plan the nominal trajectory online, and append ancillary controller actuation to the system inputs, resulting in *tube robustness*.

This framework is now explained for the trajectory tracking of the uncertain rendezvous reference trajectory, requiring a few ingredients: a reference trajectory $\mathbf{x}_{ref}(i)$, a reference input $\mathbf{u}_{ref}(i)$, a Chaser dynamics model $\mathcal{M}_C$, and a Target dynamics model which includes parameter and angular velocity estimates and prior knowledge $\mathcal{M}_T$: $\{\mathbf{I}, \hat{\mathbf{q}}, \hat{\boldsymbol{\omega}}\}$ (for solving the IVP for pose propagation). The tube robustness guarantee for the robust rendezvous problem is depicted in **Figure 7**.

$\mathcal{M}_T$, which is updated every time a pose observation is available, allows for continuous inertial frame updates of ${}^I x_{ref}(i)$. Further, the algorithm of **Figure 6** is used to convert the state Target uncertainty into a *predicted* bound on reference trajectory disturbance. The resulting Target orientation uncertainty is used to express an additive disturbance on the Chaser reference trajectory error dynamics, ${}^I\mathbf{w}$,

$$\mathbb{W} = \left\{ {}^I\mathbf{w} \in \mathbb{R}^n \colon \begin{bmatrix} \mathbf{I}_6 \\ -\mathbf{I}_6 \end{bmatrix} \mathbf{w} \le \begin{bmatrix} {}^I\mathbf{w}_{max} \\ {}^I\mathbf{w}_{max} \end{bmatrix} \right\}. \tag{24}$$

The inertial frame Chaser translational error dynamics are then,

$${}^I\mathbf{x}^+_{err} = \mathbf{A}{}^I\mathbf{x}_{err} + \mathbf{B}{}^I\mathbf{u}_{err} + \mathbf{w}, \tag{25}$$

where $\mathbf{x}_{err} = \mathbf{z} - \mathbf{x}_{ref,i}$ (where $\mathbf{x}_{ref,i}$ is updated based on the latest pose estimate) and $\mathbf{u}_{err} = \mathbf{u} - \mathbf{u}_{ref,i}$. $\mathbf{w}$ can be thought of as the worst possible disturbance in the reference trajectory that can be caused by a Target pose update at any given timestep.

The stochastic dynamics are now in a suitable format for linear robust tube MPC. First, a disturbance rejection controller called the ancillary controller must be used to reject disturbance from a nominal MPC trajectory. The ancillary controller takes the form below and is added onto a nominal MPC input, $\mathbf{v}$:

$$\mathbf{u}_{anc} = \mathbf{K_{anc}}(\mathbf{x} - \bar{\mathbf{z}}), \tag{26}$$

$$\mathbf{u} = \mathbf{v} + \mathbf{u}_{anc}, \tag{27}$$

where $\mathbf{v}$ is a nominal actuation determined by a deterministic MPC and $\mathbf{K}_{anc}$ indicates the ancillary controller disturbance rejection gain, and $\bar{z}$ is a nominal state not necessarily the same as the real initial state. $\mathbf{K}_{anc}$ can be determined through a simple LQR procedure on the nominal dynamics, but provides optimal performance and better robustness guarantees when determined via a tube minimization procedure (Buckner and Lampariello, 2018). That is, if the robust positively invariant set (RPI) can be minimized through the choice of $\mathbf{K}_{anc}$ then more rigorous guarantees exist; namely, the tube robustness guarantee that if the system state $\mathbf{x}$ starts within a set $\mathbb{Z}$ centered around a planned control trajectory $\mathbf{z}$, under the given uncertainty and ancillary controller it will remain with a tube around this trajectory for all possible $\mathbf{w}$ disturbances. The tube robustness guarantee can be thought of simply as "if you start in the tube, you stay in the tube."

Secondarily, the actual nominal MPC trajectory $\mathbf{z}$ used by the ancillary controller must be determined. The error dynamics of **Eq. 25** are used. $\mathbb{U}$ and $\mathbb{X}$ are converted to tightened constraints, effectively giving up control authority to the ancillary controller for disturbance rejection. These tightened constraints are indicated as $\mathbf{x} \in \overline{\mathbb{X}} \subset \mathbb{X}$ and $\mathbf{u} \in \overline{\mathbb{U}} \subset \mathbb{U}$ and are derived from the nominal box constraints, $\mathbb{X}$ and $\mathbb{U}$. The exact constraint tightening procedure can be found in both Buckner and Limon (Limon et al., 2008; Buckner and Lampariello, 2018). A notable feature observed in this constraint tightening procedure is the fact that large uncertainty bounds will make constraint tightening infeasible. This serves as a notification that the considered uncertainty levels are beyond the system's actuation and/or dynamics capability to adequately counter. It is possible to compromise and settle for a lower level of robustness with a lower σ for the procedure of Section 3.4, or to even require replanning at prior stages of the pipeline.

An additional modification to the MPC can be added: a parameterization of steady state values for the nominal system, captured in $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$ as defined in (Buckner and Lampariello, 2018). Finally, the nominal MPC (which does not necessarily align with the initial real state, $\mathbf{x}_i$) can be found by solving:

$$\min_{\mathbf{u}(i),\bar{\mathbf{z}}_0,\bar{\boldsymbol{\theta}}} \quad J = \sum_{i=0}^{N-1} \left[\bar{\mathbf{z}}_i - \mathbf{x}_{i,ref}\right]^\top \mathbf{Q}\left[\bar{\mathbf{x}}_i - \mathbf{x}_{i,ref}\right] + \left[\bar{\mathbf{v}}_i - \mathbf{u}_{i,ref}\right]^\top \mathbf{R}\left[\bar{\mathbf{v}}_i - \mathbf{u}_{i,ref}\right] + \left[\bar{\mathbf{z}}(N) - \mathbf{x}_{ref}(N)\right]^\top \mathbf{H}[\bar{\mathbf{z}}(N) - \mathbf{x}_{des}(N)] + \left[\bar{\boldsymbol{\theta}} - \boldsymbol{\theta}_{ref}\right]^\top \mathbf{T}\left[\bar{\boldsymbol{\theta}} - \boldsymbol{\theta}_{ref}\right]$$

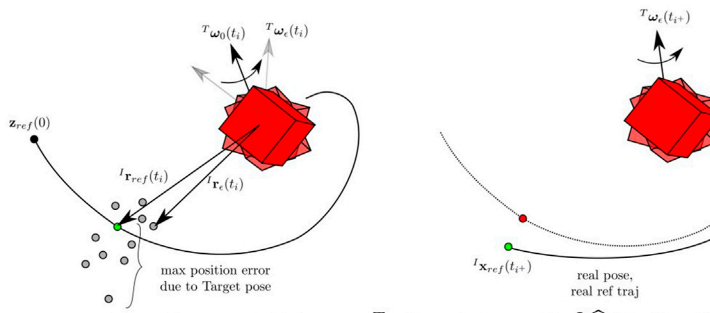
subject to

$$\begin{gathered} \mathbf{z}_i^+ = f(\mathbf{z}_i, \mathbf{v}_i). \\ \bar{\mathbf{z}} \in \overline{\mathbb{X}} \\ \bar{\mathbf{v}} \in \overline{\mathbb{U}} \\ \bar{\mathbf{v}} \in \mathbf{x} \oplus (-\mathbb{Z}) \\ \left(\bar{\mathbf{z}}(N), \bar{\boldsymbol{\theta}}\right) \in \Omega^e \end{gathered}$$

Eq. 27 is executed for the first timestep of the nominal MPC solution until the solution can be recomputed at i^+.

4 RESULTS

The components of the full rendezvous pipeline were implemented and tested using Astrobee's simulation environment. Astrobee's software is based on the Robotic Operating System (ROS) framework, where modular software components (nodes)



```
1:  procedure TUMBLEUCBOUND(${}^T\mathbf{x}(k = 0, 1, \dots N)$, ${}^I_T\widehat{\mathbf{R}}(0)$, $\widehat{\boldsymbol{\omega}}_{IT}(0)$, $\Sigma(\mathbf{R}, \boldsymbol{\omega})$, $\mathbf{I}$, NumTrials)
2:      ${}^I\mathbf{z}(k = 0, 1, \dots N) \leftarrow$ Propagate(${}^I_T\widehat{\mathbf{R}}(0), \widehat{\boldsymbol{\omega}}_{IT}(0), \mathbf{I}$)
3:      for $i = 1, 2, \dots,$ NumTrials do
4:          ${}^I_T\mathbf{R}(0), \boldsymbol{\omega}_{IT}(0) \sim \mathcal{N}\left(\left[{}^I_T\widehat{\mathbf{R}}(0), \widehat{\boldsymbol{\omega}}_{IT}(0)\right], \Sigma(\mathbf{R}, \boldsymbol{\omega})\right)$
5:          for $k = 0, 1, \dots, N$ do
6:              ${}^I\mathbf{x}(k) = {}^I_T\mathbf{R}(k)\, {}^T\mathbf{x}(k)$
7:              ${}^I\mathbf{w}(k) = {}^I\mathbf{x}(k) - {}^I\mathbf{z}(k)$
8:              ${}^I_T\mathbf{R}(k+1), \boldsymbol{\omega}_{IT}(k+1) \leftarrow$ Propagate(${}^I_T\mathbf{R}(k), \boldsymbol{\omega}_{IT}(k), \mathbf{I}$)
9:          end for
10:     end for
11:     return max ${}^I\mathbf{w}$ (over all trials and in each state dimension)
12: end procedure
```

FIGURE 6 | The Propagate step of the uncertainty bound procedure **(left)**. Many Monte Carlo runs over potential initial conditions and parameters for the Target result in many possible reference trajectory updates from $[t_0, t_f]$. At each point in time, there is a maximum possible error from the nominal reference trajectory. A reference trajectory update resulting from a pose update **(right)**. When new Target pose estimates are available, ${}^T\mathbf{x}_{ref}$ can be repropagated in the inertial frame. The algorithm to compute the uncertainty bound for tube MPC is shown below. The maximum disturbance magnitudes are found over the course of a number of Monte Carlo trials.

exchange messages over topics for complex multi-threaded code coordination. Each part of the proposed autonomy pipeline constitutes a ROS node that is added to Astrobee's core flight software and in some cases overrides default behavior.

The Target coordinator provides torque-free tumbling motion setpoints that are tracked by a custom standard MPC controller. In this way, the Target Astrobee can follow correct tumbling trajectories for various inertia tensor test cases, such as Envisat's. The Chaser coordinator orchestrates all parts of the autonomy pipeline and activates each step based on timing parameters and completion status. The full ROS architecture will be used in experimental testing on the ISS. Additional detail is provided in **Section 4.4** on the full pipeline setup and testing.

Each component of the pipeline is first demonstrated against individual performance tests, in preparation for combined experimental testing. The full composition of algorithms is provided in **Figure 4**. The results of the vision-based state and parameter estimation are summarized in **Section 4.1**, the motion planning in **Section 4.2**, and the uncertainty bound and tube MPC in **Section 4.3**. In **Section 4.4**, a case study is presented for the full operations pipeline, providing a complete example of the full autonomous rendezvous pipeline operating in the high-fidelity Astrobee simulation environment.

4.1 State and Principal Axes Estimation Performance

In testing the relative state and principal axes estimation framework, the Chaser was initially situated 1.5 m away from the Target Astrobee, which had an initial angular velocity $\boldsymbol{\omega}_{IT}(0) = [0, 3.53, 3.53]$ deg/s. In this case, the Target performed a tumble that abides by Envisat's inertia tensor. The Chaser was commanded to follow a sample inspection trajectory consisting of 1) a lateral arc maneuver, 2) a vertical arc maneuver, and 3) an approach/recede maneuver along the viewing axis. Chaser attitude commands kept the Target in the HazCam field of view. The factor graph was solved at a rate of 1 [Hz]; most of the computation run-time was taken up by point cloud matching and registration, this fairly slow rate ensures that the factor graph updates are achievable on the robot hardware.

Downsampling and background elimination was applied to the point clouds for more efficient processing. Sufficient numbers of matches from frame to frame were found, which in turn enabled reliable pose registration solutions. In general, the Teaser++ pose registration solver provided robust pose odometry estimates of the Target's geometric frame with low noise levels along with several loop closures. Truth values from the simulator serve as a metric to evaluate the proposed approach and provide estimation error statistics.

In terms of Chaser navigation, the estimates were smooth and closely corresponded to the true values, demonstrating the ability to disambiguate Chaser motion from the tumbling Target motion. 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 (if applicable).

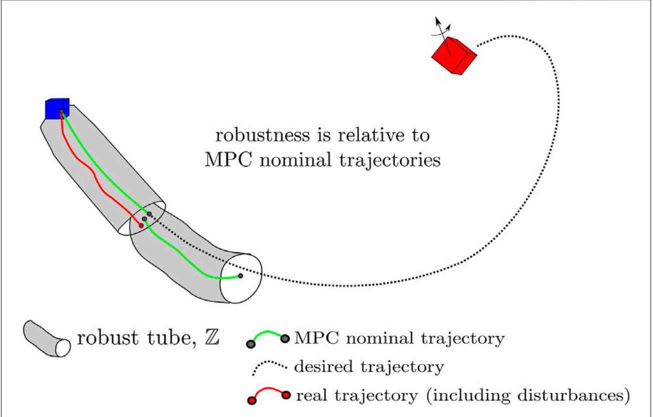

robustness is relative to
MPC nominal trajectories
robust tube, $\mathbb{Z}$
MPC nominal trajectory
desired trajectory
real trajectory (including disturbances)

FIGURE 7 | A demonstration of the tube robustness guarantee. Robust tubes are produced around nominal reference trajectories which need not necessarily coincide with the real initial state and are in turn made to track a motion planning reference trajectory, $\mathbf{x}_{ref}$ (black). The maximum possible deviation of $\mathbf{x}_{ref}$ in any given timestep is treated as an additive uncertainty for tracking, and is dealt with via the tube robustness guarantee.

Figure 8 shows the results for the principal axes estimation portion of the pipeline. The optimized value of ${}^{G}\mathbf{R}_{P}$ produced the best conic projection fits shown in the figure. 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 $\boldsymbol{G}$ frame) into the $\boldsymbol{P}$ frame.

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 with online-updating phases that are required to intercept the Target.

4.2 Chaser Motion Planning Performance

The motion planner was validated and analyzed for performance. A few statistics are note-worthy within the above context.

First, for the chosen tuning of the NLP implementation in terms of the cost function accuracy (1e-8) and constraint gradient finite-difference step size (1e-10) and with a warm start provided by the LUT and nominal initial conditions for the Chaser, it is found that the planner produces motion plans for random queries of the Target parameters with a mean runtime of 0.84 s with a 3σ of 2.03 s. Without this warm start, the mean planner runtime is 1.68 s and the 3σ is 5.56 s.

For 1,000 queries with a warm start of the motion planner and the Chaser located at the nominal initial position, there is a 100% success rate in the optimizer converging to a feasible trajectory. For comparison, for the same sample size and Chaser initial position, but with a cold start, the success rate is 90%.

As the Chaser will be placed at its initial position manually, the effect of placement error has also been investigated. For 10, 000 samples of perturbation only in the x-direction of up to

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 $\tilde{a}$, which is computed as $\tilde{a} = Log(\overline{R}^{\top}\widehat{R}) = [\delta a_x, \delta a_y, \delta a_z]^{\top}$ and $\tilde{\theta} = ||\tilde{a}||$, where $\overline{R}$ and $\widehat{R}$ correspond to true and estimated orientations, respectively.

$\tilde{t}_{C}^{W}$	4.1 cm
$\tilde{v}_{C}^{W}$	5.3 mm/s
$\tilde{a}_{C}^{W}$, $\tilde{\theta}$	$[-0.1, 0.001, 0.61]^{\top}$, 1.00 deg
$\tilde{\boldsymbol{\omega}}_{C}$	0.08 deg/s
$\tilde{a}_{T}^{W}$, $\tilde{\theta}$	$[6.26, -4.24, 0.85]^{\top}$, 7.25 deg
$\tilde{\boldsymbol{\omega}}_{T}$	0.6199 deg/s

10 cm, 10, 000 samples of perturbation in the y-direction only of up to 10 cm, 10, 000 samples of perturbation in the z-direction only of up to 10 cm, and 10, 000 samples where the modulus of the perturbation in the x-, y-, and z-direction is up to 10 cm, the failure rate for the optimization to converge to a feasible trajectory is only 0.007%. When the initial conditions of the Chaser are within boundaries and a warm start is provided to the motion planner, the mean runtime is 0.92 s with a 3σ of 1.51 s.

4.3 Uncertainty Bound and Robust Tube MPC Performance

A reference trajectory $\mathbf{x}_{ref}$ and reference input $\mathbf{u}_{ref}$ were provided using the nonlinear programming-based motion planning method of **Section 3.3**. Using these references, the uncertainty bound method of **Section 3.4** was used to produce $\mathbb{W}$. Finally, using the tube MPC method of **Section 3.5**, an ancillary controller and nominal MPC were then created to robustly guide the Chaser along the reference trajectory in the inertial frame, with knowledge of the potential disturbances induced by inaccurate Target pose knowledge.

The reference trajectory used is a long looping motion occurring over $t_f = 120$ s. Providing this reference trajectory to `TumbleUCBound` produces a $\mathbb{W}$ box constraint of $\mathbf{w}_{max} = [0.147, 0.0385, 0.1236, 0.0053, 0.0049, 0.0016]$ for the duration of the entire maneuver, the result of $n = 100$ Monte Carlo propagations of inertia disturbances derived from approximate statistics of the visually estimated parameters. In practice for this sample study, a smaller $\mathbf{w}_{max}$ was required for tightened constraint feasibility. This can be viewed as a feature of robust tube MPC—if constraints are tightened beyond feasibility, this is a sign that uncertainty levels must be improved and that robust control (given system constraints) is not enough to robustly counteract uncertainty. A smaller, less conservative stepwise bounded uncertainty is used in this sample analysis: $\mathbf{w}_{max} = [0.05, 0.05, 0.05, 0.00005, 0.00005, 0.00005]$. Note that reference trajectory velocity uncertainty is less impacted by Target uncertainty.

An ancillary gain matrix $\mathbf{K}_{anc}$ was computed using an LQR formulation for $\mathbf{Q} = diag([1, 1, 1, 10, 10, 10])$ and $\mathbf{R} = diag([1000, 1000, 1000])$, and the discrete dynamics of **Eq. 4** with $dt = 0.2$ s. An example of the constraint tightening procedure can be shown for $\overline{\mathbb{U}} \in \mathbb{U}$. Initial constraints are provided by an input box constraint $\mathbb{U}$, with $\mathbf{u}_{max} = 0.4$ [N]. Other parameters are set as in **Section 2.3**. After computation of the robust positively invariant set (RPI), constraint tightening via

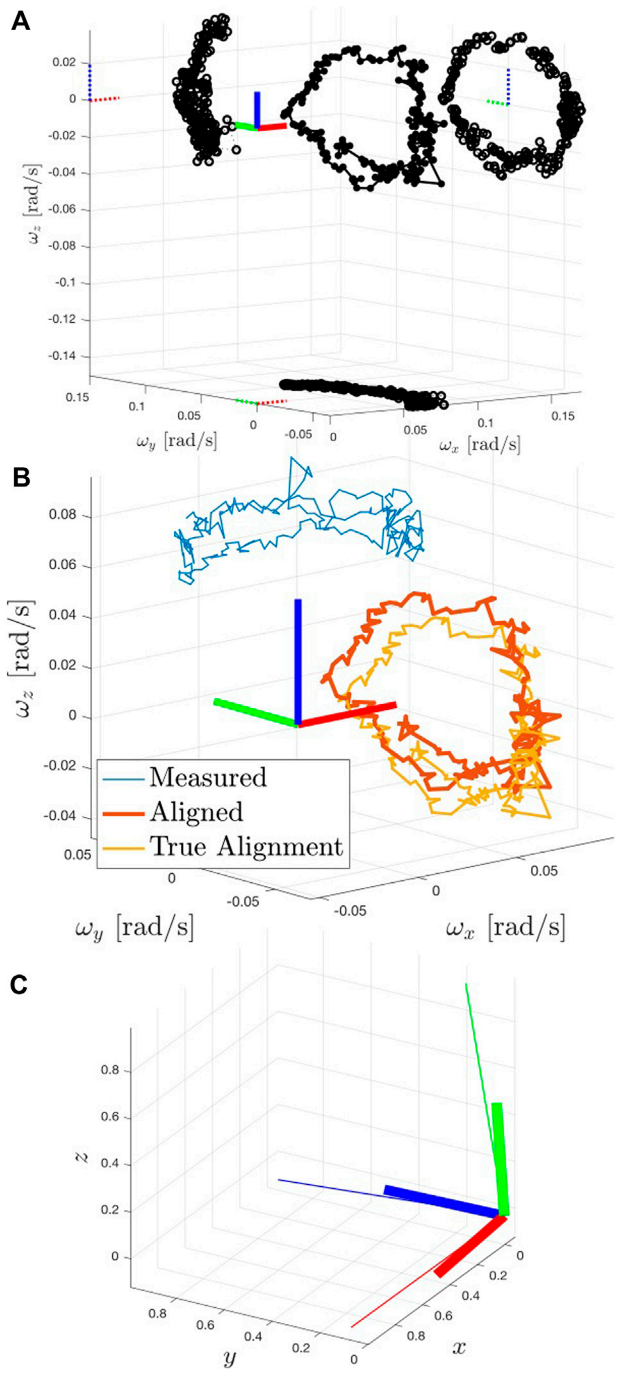


FIGURE 8 | (A) Resulting polhode after alignment to the Target's estimated principal axes. The polhode's plane projections produce central conics (not rotated nor displaced from their respective origins). Solid circles indicate the polhode, and open circles indicate the projected conics. **(B)** Polhode comparison between the measured (blue, in geometric frame), aligned (red, after rotating by the estimated orientation from geometric to principal axes), and groundtruth-aligned (orange) curves. **(C)** Visual comparison of the offset between the groundtruth principal axes (bold RGB triad) against the estimated orientation (thin, longer RGB triad).

the procedure in (Limon et al., 2008) is performed, resulting in a new set of polyhedral constraints:

$$\mathbf{A}_u\mathbf{u} = \mathbf{b}_u, \tag{28}$$

$$\mathbf{A}_u = \begin{bmatrix} \mathbf{I}_{3\times3} \\ -\mathbf{I}_{3\times3} \end{bmatrix}, \tag{29}$$

$$\mathbf{b}_u = \begin{bmatrix} 0.239 & 0.239 & 0.239 & 0.239 & 0.239 & 0.239 \end{bmatrix}^\top. \tag{30}$$

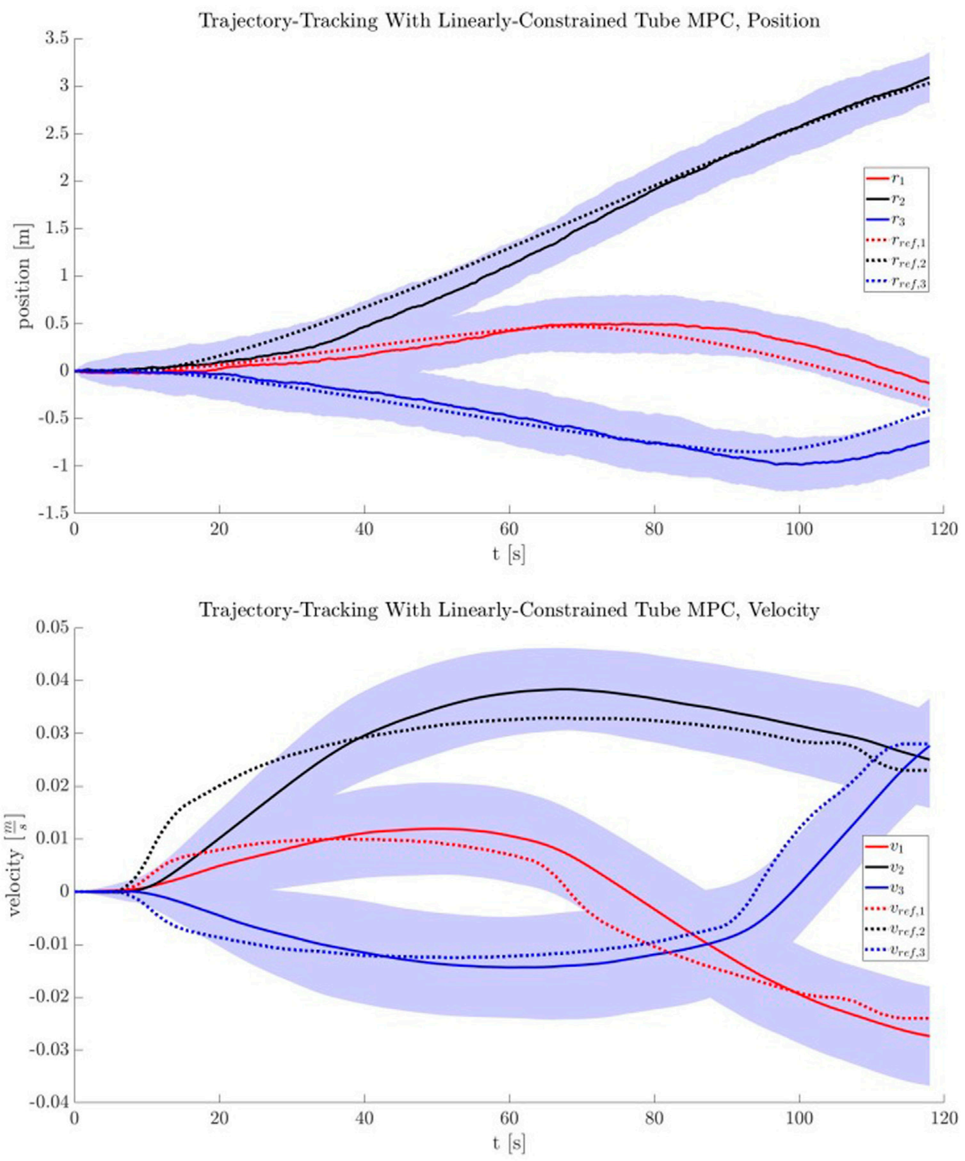


FIGURE 9 | The position and velocity tracking performance for the robust tube MPC, with shading indicating 1-σ bounds of n = 100 Monte Carlo trials. Note that nominal MPC and ancillary controller tuning can have a significant impact on controller performance (as with any MPC).

The tightened constraints are passed off to the robust tube MPC, which computes a nominal MPC solution at each time step, supplemented by the action of the ancillary controller. A series of $n = 100$ Monte Carlo trials using this reference trajectory and uncertainty bound were performed to evaluate the tube MPC's performance. To approximate **w**, a uniform distribution of $\mathbf{w} \sim \mathcal{U}(-\mathbf{w}_{max}, \mathbf{w}_{max})$ was used to simulate the stepwise reference trajectory uncertainty. In reality, this trajectory uncertainty is caused by the differing Target tumble dynamics coupled with online trajectory updates (and is shown in **Section 4.4**). The averaged tracking results of this procedure are shown in **Figure 9** for position and velocity tracking. The robust tube MPC was implemented using the CasADi optimal control framework (Andersson et al., 2019). Runtimes for a CasADi-based implementation with $N = 10$, $dt = 0.2$ were $\mu_t = 0.014$ s, $\sigma_t^2 = 1.89 \times 10^{-5}$ [s^2] per step on a standard quad core Intel Core i7-4700MQ desktop system.

4.4 The Full Pipeline: A Case Study Result

A case study is presented for a full pipeline run of the proposed rendezvous algorithm for a scenario representing planned testing on the ISS using NASA's Astrobee robotic satellites. To provide additional context on the computing environment, the Astrobee Robot Software uses the Robotic Operating System (ROS) as middleware for communication, with approximately 50 nodelets running on two ARM processors

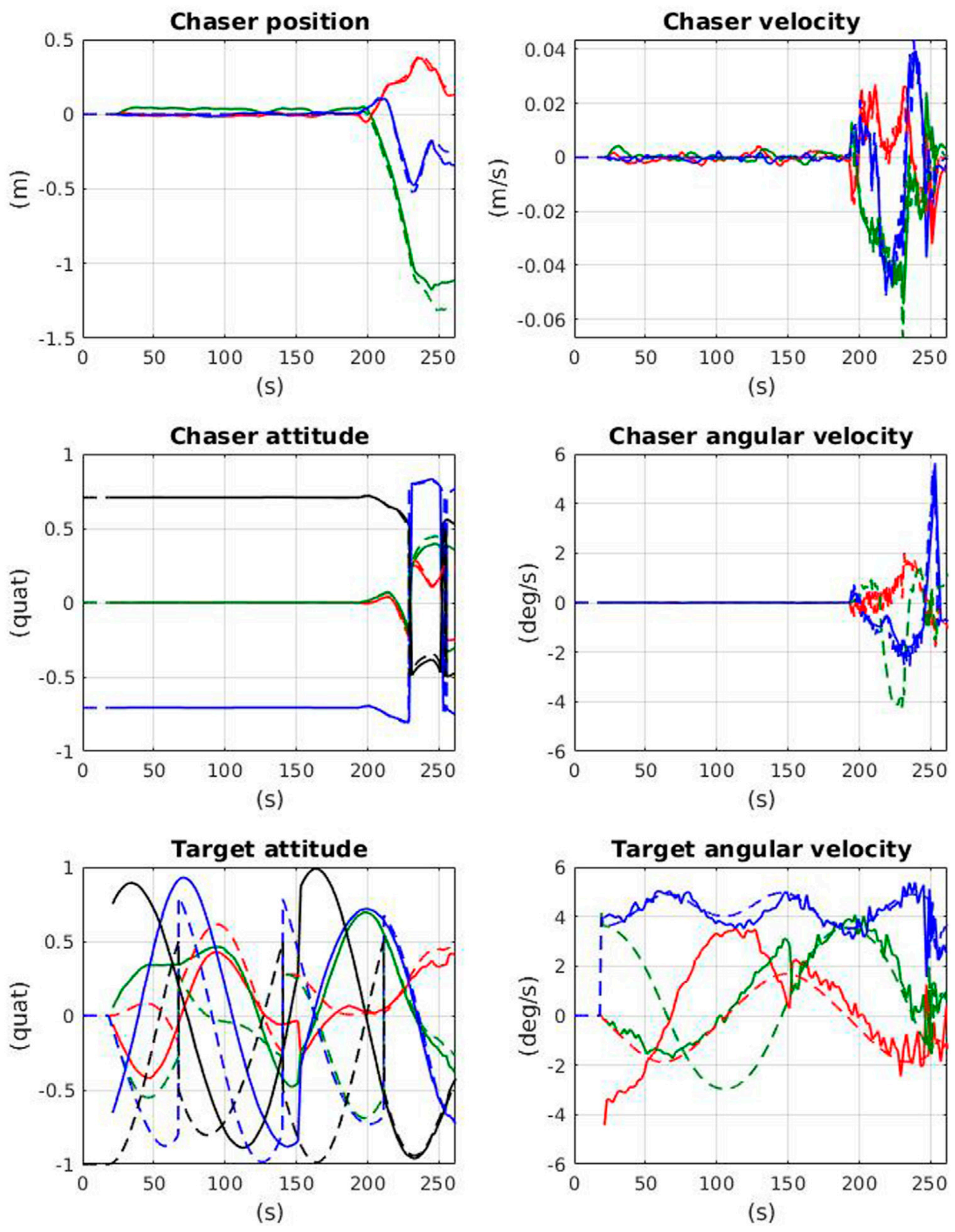


FIGURE 10 | Chaser and Target state estimates over the time-history of the full pipeline demonstration. Solid lines indicate estimated values, dashed lines indicate simulator truth. Notice the "snap" that occurs in the Target estimates when the principal axes are determined (~150 s).

(MLP and LLP). These processors run the general-purpose operating system Ubuntu 16.04 and ROS Kinetic with a sensor suite including time of flight cameras, IMUs, and more as previously detailed.

Simulation results for this scenario were obtained using NASA's ROS/Gazebo-based Astrobee simulation environment. The simulation environment includes extensive modeling of Astrobee including its impeller propulsion system, onboard visual navigation, environmental disturbances, and many more true-to-life models Flückiger et al. (2018). The full rendezvous pipeline was implemented as an additional set of Python and C++ ROS nodes and nodelets that run alongside the default Astrobee flight software. Significant development effort was dedicated to architecting software that would be usable for both simulation and hardware testing Albee et al. (2020). It may be helpful to the reader to follow **Figure 4**, which fully delineates the flow of information in the rendezvous pipeline.

4.4.1 The Scenario

For this scenario it is assumed that the participant Astrobee robots have been placed at their respective starting positions

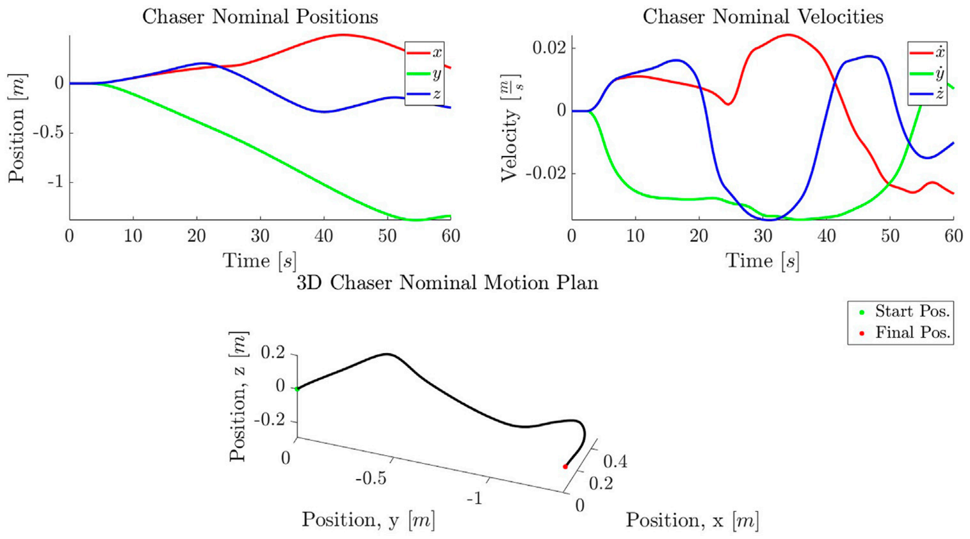


FIGURE 11 | A planned trajectory in position **(A)** and velocity **(B)** for a tri-axial tumble query with inputs supplied by the Target estimation, with final Chaser position and velocity equal to that of the MP. The 3D representation of the trajectory is given **(C)**. The trajectory is presented in the inertial frame.

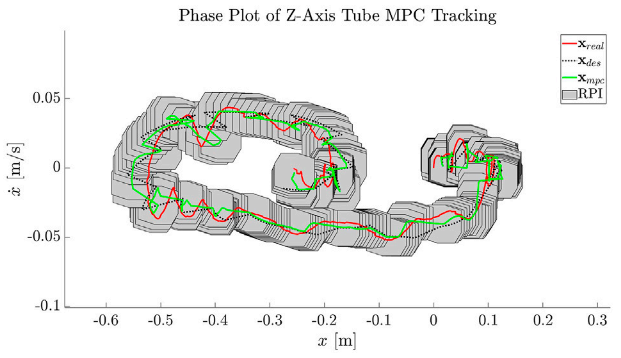


FIGURE 12 | An example of the robust tube MPC maintaining the real **x** (red) within an invariant set (gray polytope) centered on the nominally planned MPC solution (green). Here, the RPI is shown for the first timestep of each receding horizon MPC solution.

within the Japanese Experiment Module (JEM) and are initially station-keeping. The relative distance between the two robots is 1.5 [m] in the y-axis of the ISS world frame, $\boldsymbol{W}$. Both robots begin at rest, with the Target facing forward (positive x-direction) and the Chaser facing port (negative y-direction).

The Target performs a tri-axial tumble mimicking Envisat's inertia $\mathbf{I}_{ES}$ with initial angular velocity $\boldsymbol{\omega}_T(0) = [0, 3.53, 3.53]$ [deg/s] and maintains its translational position in the ISS inertial frame. The Chaser conducts Target observation at its initial position and maintains this position while the Target state and principal axes are estimated, the motion plan for a rendezvous maneuver with a duration of 60 s is generated, and the controller parameters determined. When the starting time for the approach provided by the motion planner is reached, the Chaser begins its approach to the MP.

4.4.2 State and Principal Axes Estimation

The Chaser successfully observed the Target's tumbling motion and provided accurate state estimates for both spacecraft throughout the entire maneuver. The observation period lasted 120 s to accumulate enough Target angular velocity estimates before determining the orientation of the principal axes. **Figure 10** shows the time history of state estimates throughout the maneuver. The "snap" in Target's attitude and angular velocity estimates is clearly evident when the principal axes are determined and proper alignment occurs. The average time taken for a factor graph update was 0.08 s on the same machine as **Section 3.5**. These values were then handed off to the Chaser motion planner.

4.4.3 Chaser Motion Planning

The estimated Target parameters and the measured Chaser and estimated Target positions in the inertial frame are of interest for the motion planner. The motion planner uses the LUT designed for a given set of Target parameters as indicated in **Section 3.3** to determine the warm-start parameters for the given Target parameter query and from which a motion plan is developed. Included in this plan are Chaser states at each via point and a start time for the approach maneuver. A planned trajectory for this scenario is provided in **Figure 11**, using the outputs of the visual estimation procedure.

For the tumble described in this case study, the motion planner produces plans with 100% success for random initial conditions. The mean time for plan generation for this scenario is 1.12 s and the 3σ is 3.12 s.

4.4.4 Uncertainty Bound and Robust Tube MPC

The resulting motion plan, ${}^{B}\mathbf{x}_{ref}(t_{0:N})$ (defined in the body frame) and $\mathbf{u}_{ref}(t_{0:N})$, is handed off to the robust controller as in

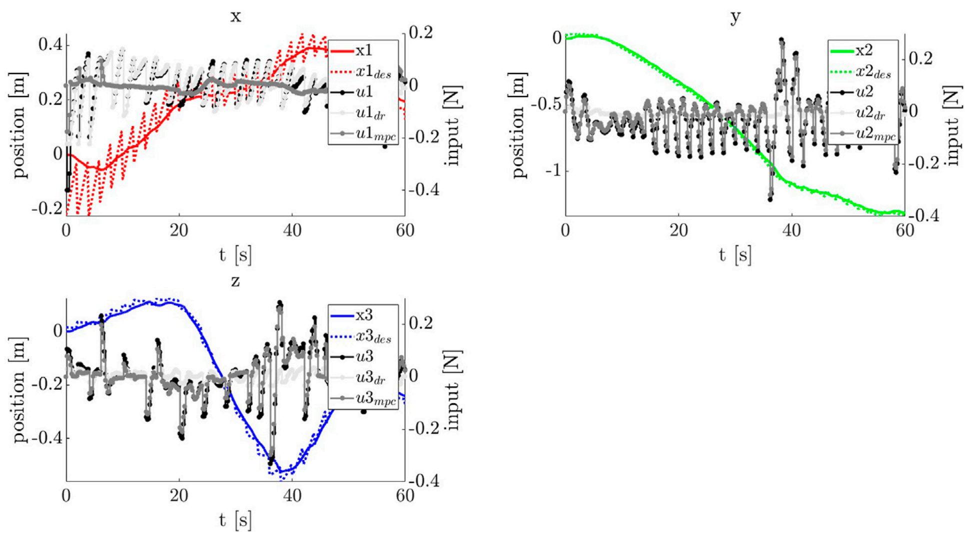


FIGURE 13 | The input and state histories for the robust controller tracking the *online-updated* reference trajectory (dashed). It is important to remember that the reference trajectory in this scheme actually shifts in real-time as new SLAM estimates of the true Target attitude, $\hat{q}_T$ become available. It is precisely this stepwise shifting for which the tube robustness guarantee is produced. u_{dr} indicates the disturbance rejection component, while u_{mpc} indicates the nominal MPC component.

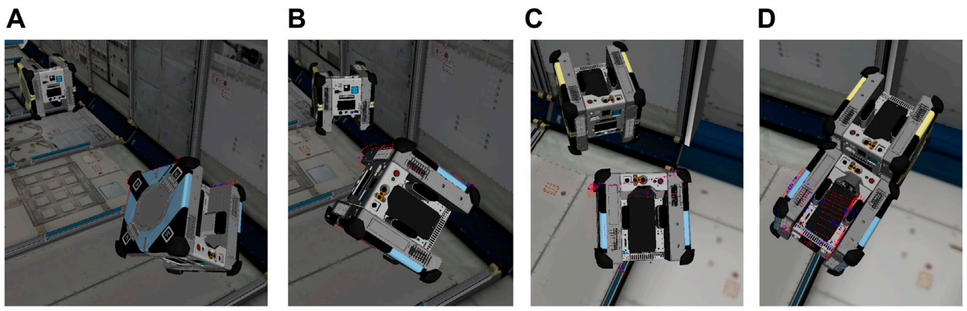


FIGURE 14 | A timelapse of the rendezvous portion of the pipeline for an arbitrary tri-axial tumble. **(A–D)**, the Chaser begins tracking the motion plan with its robust controller while periodically updating the inertial frame reference trajectory based on SLAM estimates of $\hat{q}_T$.

Figure 4. The primary ingredients remaining before robust tube MPC can begin tracking are an approximation of the robust invariant set (RPI), $\mathbb{Z}$ and accompanying tightened input and state constraints, and real-time updates of the estimated Target orientation $\hat{\mathbf{q}}_T(i)$.

The first step in calculating these values is to determine the uncertainty bound, $\mathbf{w}$. The motion plan is handed off to the uncertainty bound approximation module along with the vital full statistics of estimated values from the visual estimation: $\mathcal{N}(\hat{\mathbf{q}}_T(0), \Sigma_\mathbf{q})$, $\mathcal{N}(\hat{\boldsymbol{\omega}}_T(0), \Sigma_{\boldsymbol{\omega}})$. Using the algorithm of **Figure 6**, the uncertainty bound can be approximated. In this case, the stepwise uncertainty due to Target estimation inaccuracy is $\mathbf{w} = [0.0145, 0.0065, 0.0152, 0.0024, 0.0012, 0.0027]$, representing stepwise error in reference trajectory positions and velocities.

Now, $\mathbf{w}$ is handed off to the robust set approximation method detailed in Section 3.5 along with other vital information including the disturbance rejection gain $\mathbf{K}_{anc}$, system dynamics $f(\mathbf{x}, \mathbf{u})$, and constraints as detailed in **Figure 4**. The goal is to produce the robust set approximation $\mathbb{Z}$, with feasible state and input constraints. In this particular scenario, the uncertainty levels are too great for the available thruster authority from $\mathbb{U}$; this is a useful feature, indicating that an absolute robustness guarantee for this particular level of uncertainty is not possible, and prompting two options. The agent could, in theory, choose to reobserve and replan for new Target statistics, thus increasing confidence in the Target motion. On the other hand, it is possible to obtain a safety guarantee for a reduced uncertainty bound, $\mathbf{w}_s$ for a less conservative σ of the uncertainty approximation. This can still provide a robustness certification for a likely level of uncertainty.

In this case, a shrunken $\mathbf{w}_s$ is used resulting in an approximation of the RPI visible in **Figure 12**, along with tightened input and state constraints. Now, the robust tube MPC is operated, constantly adjusting ${}^B\mathbf{x}_{ref}(t_{0:N})$ to inertial frame coordinates based on an updated $\hat{\mathbf{q}}_T(i)$. The resulting track, showing the online-updated reference trajectory in dashed lines, is shown in **Figure 13**. The shifting of the reference trajectory is accounted for in the robust tube MPC setup; this is visible in **Figure 12**, where the RPI centered on the nominal MPC plan always contains the real trajectory. The MPC runs at 5 Hz over a 5 s horizon with a computation time of $\mathcal{N}(7.2, 1.6)$ ms on the machine used in **Section 3.5**. Even assuming a generous ten-fold computational slowdown on the Astrobee processors (a value that has held in practice) this timing is well within the required 200 ms desired. The result is a successful track to the MP, as shown in the simulation progression of **Figure 14**.

5 CONCLUSION

The framework and algorithms proposed in this study are a significant step toward autonomous rendezvous with tumbling Targets, uniting multiple key algorithmic components of the autonomy pipeline. Furthermore, key uncertainty sources are considered throughout and robustness and constraint satisfaction despite these uncertainties are incorporated into the pipeline logic. The treatment of uncertainty due to imperfect Target estimation combined with online-updating is considered, along with the implications of choosing overly conservative uncertainty bounds. Planned ISS tests in mid-2021 on the Astrobee platform will provide extensive experimental validation of this study, which has been shown algorithmically defined here and demonstrated in a detailed simulation environment that directly transfers to the Astrobee hardware.

Some major lessons learned from the development of this framework include the need for early standardization and the practical difficulties of moving to hardware implementation. Hardware implementation is vital, giving a direct look at the actual sensors, noise, computational power, and environments that will be seen by the algorithms developed for autonomous systems. However, hardware implementation leads to many complications, particularly in moving from desktop-based computational tools to embedded programming that may be lacking important libraries or computational power for example. Additionally, because of the number of algorithmic components, standardization and message-passing procedures must be settled early in development; luckily, ROS takes care of some of this complexity on the Astrobee platform. Some practical lessons learned are further documented in (Albee et al., 2020).

Algorithmically, the estimation, motion planning, uncertainty propagation, and robust control components have been outlined in detail and their application to a relevant satellite system explained. Individual performance metrics have been analyzed, showing the effectiveness of each of these portions independently. Additionally, the full pipeline presented in **Figure 4** has been shown, demonstrating success of the autonomous rendezvous procedure in a detailed simulation environment, in real-time. A significant next step is ISS demonstration of the proposed pipeline on Astrobee hardware.

An interesting future direction for this study from a motion planning and controls perspective is the determination of system unknowns during execution. Prior work in this area for robotic free-flyers has recently been proposed and is applicable here in the case of online motion planning that accounts for the learning of Target inertial properties and permitting online recomputation, for instance (Albee et al., 2019). As it stands, the robust rendezvous framework documented here creates a full framework of algorithms needed to perform autonomous rendezvous with uncertain tumbling targets and adds robustness and systems integration considerations to an important open problem in microgravity close proximity operations, and demonstrates its effectiveness in a detailed simulation environment. Future study will document the framework's performance as it moves toward on-orbit hardware demonstration.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

Development of the robust tube MPC method for tumbling reference tracking: KA, CO, and CS; uncertainty bound formulation for robust tube MPC: CO and KA; vision-based state and parameter estimation: CO, JT, and TT; NLP-based motion planning for tumbling docking: RLa and CS; systems integration and CONOPS: KA, CO, and IH; editing and oversight: RLi and RLa.

FUNDING

A portion of this work was sponsored by the NASA Space Technology Mission Directorate through a NASA Space Technology Research Fellowship under grant 80NSSC17K0077. The Draper Fellow Program also sponsored a portion of this work. Support for this work was also provided under DLR research agreement 67272241.

ACKNOWLEDGMENTS

The authors would like to thank the NASA Ames Astrobee operations and development teams, as well as others at NASA Ames' Intelligent Robotics Group (IRG) for their support, particularly Brian Coltin, Jose Benavides, Ruben Garcia Ruiz, and In Won Park. The authors also thank Alvar Saenz-Otero and David W. Miller for providing their insights. Research reported in this study was supported by the International Space Station U.S. National Laboratory under agreement number UA-2019-969. The authors gratefully acknowledge the support that enabled this research.

REFERENCES

Aghili, F. (2012). A Prediction and Motion-Planning Scheme for Visually Guided Robotic Capturing of Free-Floating Tumbling Objects with Uncertain Dynamics. *IEEE Trans. Robot.* 28, 634–649. doi:10.1109/TRO.2011.2179581

Aghili, F. (2008). Optimal Control for Robotic Capturing and Passivation of a Tumbling Satellite with Unknown Dynamics. *AIAA Guidance, Navigation Control. Conf. Exhibit*. doi:10.2514/6.2008-7274

Aghili, F. (2020). Optimal Trajectories and Robot Control for Detumbling a Non-cooperative Satellite. *J. Guidance, Control Dyn.* 43, 981–988. doi:10.2514/1.g004758

Albee, K., Ekal, M., and Oestreich, C. (2020). *A Brief Guide to Astrobee's Flight Software*.

Albee, K., Ekal, M., Ventura, R., and Linares, R. (2019). "Combining Parameter Identification and Trajectory Optimization: Real-Time Planning for Information Gain," in *ESA Advanced Space Technologies for Robotics and Automation (ASTRA)* (Noordwijk, Netherlands: ESA).

Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). CasADi: a Software Framework for Nonlinear Optimization and Optimal Control. *Math. Prog. Comp.* 11, 1–36. doi:10.1007/s12532-018-0139-4

Buckner, C., and Lampariello, R. (2018). "Tube-Based Model Predictive Control for the Approach Maneuver of a Spacecraft to a Free-Tumbling Target Satellite," in Proceedings of the American Control Conference, 5690–5697. doi:10.23919/ACC.2018.8431558

Carlone, L., Kira, Z., Beall, C., Indelman, V., and Dellaert, F. (2014). "Eliminating Conditionally Independent Sets in Factor Graphs: A Unifying Perspective Based on Smart Factors," in IEEE Intl. Conf. On Robotics and Automation (ICRA) (IEEE), 4290–4297. doi:10.1109/icra.2014.6907483

ESA (2015). *Phase B1 of an Active Debris Removal mission (E. DEORBIT Mission)*. Noordwijk, Netherlands: Tech. rep., European Space Agency.

Fehse, W. (2003). *Automated Rendezvous and Docking of Spacecraft*. doi:10.1177/095441000722100603

Fejzi, A., and Miller, D. W. (2008). *Development of Control and Autonomy Algorithms for Autonomous Docking to Complex Tumbling Satellites*. Cambridge, MA: Massachusetts Institute of Technology. Ph.D. thesis.

Flores-Abad, A., Ma, O., Pham, K., and Ulrich, S. (2014). A Review of Space Robotics Technologies for On-Orbit Servicing. *Prog. Aerospace Sci.* 68, 1–26. doi:10.1016/j.paerosci.2014.03.002

Flückiger, L., Browne, K., Coltin, B., Fusco, J., Morse, T., and Symington, A. (2018). *Astrobee Robot Software: Enabling Mobile Autonomy on the ISS*. Madrid, Spain: Tech. rep., National Aeronautics and Space Administration.

Forster, C., Carlone, L., Dellaert, F., and Scaramuzza, D. (2016). On-manifold Preintegration for Real-Time Visual–Inertial Odometry. *IEEE Trans. Robotics* 33, 1–21. doi:10.1109/TRO.2016.2597321

Fourie, D., Terán Espinoza, A., Kaess, M., and Leonard, J. J. (2020). "Characterizing Marginalization and Incremental Operations on the Bayes Tree," in *Intl. Workshop on the Algorithmic Foundations of Robotics (WAFR)* (Finland: Springer), 1–16.

Fourie, D., Tweddle, B. E., Ulrich, S., and Saenz-Otero, A. (2014). Flight Results of Vision-Based Navigation for Autonomous Spacecraft Inspection of Unknown Objects. *J. Spacecraft Rockets* 51, 2016–2026. doi:10.2514/1.a32813

Harris, A. W. (1994). Tumbling Asteroids. *Icarus* 107, 209–211. doi:10.1006/icar.1994.1017

Hillenbrand, U., and Lampariello, R. (2005). *Motion and Parameter Estimation of a Free-Floating Space Object from Range Data for Motion Prediction*. Munich, Germany: European Space Agency, (Special Publication) ESA SP, 461–470.

Hurtado, J. E., and Satak, N. (2011). "Poinsot Motion Attitude," in International Conference on Computational & Experimental Engineering and Sciences. Vol. 16.

Johnson, S. (2017). The Nlopt Nonlinear-Optimization Package. Available at: http://github.com/stevengj/nlopt.version 2.4.2.

Kaess, M., Ila, V., Roberts, R., and Dellaert, F. (2010). "The Bayes Tree: An Algorithmic Foundation for Probabilistic Robot Mapping," in *Algorithmic Foundations of Robotics IX. Springer Tracts in Advanced Robotics*. Editors D. Hsu, V. Isler, J. C. Latombe, and M. C. Lin (Berlin: Springer) Vol. 68, 1–17. doi:10.1007/978-3-642-17452-0_10

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2012). iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree. *Intl. J. Robotics Res. (Ijrr)* 31, 217–236. doi:10.1177/0278364911430419

Kaess, M., Ranganathan, A., and Dellaert, F. (2008). iSAM: Incremental Smoothing and Mapping. *IEEE Trans. Robot.* 24, 1365–1378. doi:10.1109/tro.2008.2006706

Lampariello, R., Mishra, H., Oumer, N., and Peters, J. (2021). Robust Motion Prediction of a Free-Tumbling Satellite with On-Ground Experimental Validation (Accepted). *J. Guidance, Control. Dyn.* 1–17. doi:10.2514/1.G005745

Lampariello, R. (2010). Motion Planning for the On-Orbit Grasping of a Non-cooperative Target Satellite with Collision Avoidance. *Int. Symp. Artif. Intelligence, Robotics Automation Space* 1, 636–643.

Limon, D., Alvarado, I., Alamo, T., and Camacho, E. F. (2008). On the Design of Robust Tube-Based MPC for Tracking. *IFAC Proc. Volumes* 41, 15333–15338. doi:10.3182/20080706-5-kr-1001.02593

Lupton, T., and Sukkarieh, S. (2011). Visual-inertial-aided Navigation for High-Dynamic Motion in Built Environments without Initial Conditions. *IEEE Trans. Robotics* 28, 61–76. doi:10.1109/TRO.2011.2170332

Majumdar, A., and Tedrake, R. (2017). Funnel Libraries for Real-Time Robust Feedback Motion Planning. *Int. J. Robotics Res.* 36, 947–982. doi:10.1177/0278364917712421

Mammarella, M., Capello, E., Park, H., Guglieri, G., and Romano, M. (2018). Tube-based Robust Model Predictive Control for Spacecraft Proximity Operations in the Presence of Persistent Disturbance. *Aerospace Sci. Tech.* 77, 585–594. doi:10.1016/j.ast.2018.04.009

Mascellani, G. (2019). Boost. Available at: https://www.boost.org/.version 1.71.0

Mayne, D. Q., Kerrigan, E. C., and Falugi, P. (2011). Robust Model Predictive Control: Advantages and Disadvantages of Tube-Based Methods. *IFAC Proc. Volumes* 44, 191–196. doi:10.3182/20110828-6-IT-1002.01893

NASA (2015). *Reference Guide to the International Space Station*. Washington, DC: Tech. rep., National Aeronautics and Space Administration.

Park, H., Zappulla, R., Zagaris, C., Virgili-Llop, J., and Romano, M. (2017). "Nonlinear Model Predictive Control for Spacecraft Rendezvous and Docking with a Rotating Target," in 27th AAS/AIAA Space Flight Mechanics Meeting (San Antonio, TX, USA: AIAA), 1–14.

Rusu, R. B., Blodow, N., and Beetz, M. (2009). "Fast point Feature Histograms (Fpfh) for 3d Registration," in 2009 IEEE International Conference on Robotics and Automation (Kobe) (IEEE), 3212–3217. doi:10.1109/robot.2009.5152473

Saenz-Otero, A., and Miller, D. W. (2005). *Design Principles for the Development of Space Technology Maturation Laboratories Aboard the International Space Station*. Cambridge, MA: Massachusetts Institute of Technology. Ph.D. thesis.

Setterfield, T. P., Miller, D. W., Leonard, J. J., and Saenz-Otero, A. (2018a). Mapping and Determining the center of Mass of a Rotating Object Using a Moving Observer. *Int. J. Robotics Res.* 37, 83–103. doi:10.1177/0278364917749024

Setterfield, T. P., and Miller, D. W. (2017). *On-Orbit Inspection of a Rotating Object Using a Moving Observer*. Cambridge, MA: Massachusetts Institute of Technology. Ph.D. thesis.

Setterfield, T. P., Miller, D. W., Saenz-Otero, A., Frazzoli, E., and Leonard, J. J. (2018b). Inertial Properties Estimation of a Passive On-Orbit Object Using Polhode Analysis. *J. Guidance, Control Dyn.* 41, 2214–2231. doi:10.2514/1.G003394

Smith, R. (2009). Open Dynamics Engine. Available at: http://www.ode.org/. version 0.11.

Sternberg, D. C., and Miller, D. (2018). Parameterization of Fuel-Optimal Synchronous Approach Trajectories to Tumbling Targets. *Front. Robot. AI* 5, 1–11. doi:10.3389/frobt.2018.00033

Stoneman, S., and Lampariello, R. (2016). "A Nonlinear Optimization Method to Provide Real-Time Feasible Reference Trajectories to Approach a Tumbling Target Satellite," in 2016 International Symposium on Artificial Intelligence, Robotics and Automation in Space (Beijing, P.R. China: AIRAS), 1–10.

Sullivan, B. R., Kelm, B., Roesler, G., and Henshaw, C. G. (2015). "Darpa Robotic Space Servicer: On-Demand Gapabilities in Geo," in AIAA SPACE 2015 Conference and Exposition (Pasadena, CA: AIAA).

Teran, A. (2021). *Versatile Inference Algorithms Using the Bayes Tree for Robot Navigation*.

Terán Espinoza, A., Hettrick, H., Albee, K., Hernandez, A. C., and Linares, R. (2019). "End-to-End Framework for Close Proximity In-Space Robotic Missions," in *International Astronautical Congress (IAC)* (Washington, D.C.: IAC), 1–13.

Terán Espinoza, A. (2020). *Versatile Inference Algorithms Using the Bayes Tree for Robot Navigation*. Cambridge, MA: Massachusetts Institute of Technology. Ph.D. thesis.

Tweddle, B. E., Saenz-Otero, A., Leonard, J. J., and Miller, D. W. (2015). Factor Graph Modeling of Rigid-Body Dynamics for Localization, Mapping, and Parameter Estimation of a Spinning Object in Space. *J. Field Robotics* 32, 897–933. doi:10.1002/rob.21548

Tweddle, B. E., Setterfield, T. P., Saenz-Otero, A., and Miller, D. W. (2016). An Open Research Facility for Vision-Based Navigation Onboard the International Space Station. *J. Field Robotics* 33, 157–186. doi:10.1002/rob.21622

Ventura, J., Romano, M., and Walter, U. (2015). Performance Evaluation of the Inverse Dynamics Method for Optimal Spacecraft Reorientation. *Acta Astronautica* 110, 266–278. doi:10.1016/j.actaastro.2014.11.041

Virgili-Llop, J., Zagaris, C., Zappulla, R., Bradstreet, A., and Romano, M. (2019). A Convex-Programming-Based Guidance Algorithm to Capture a Tumbling Object on Orbit Using a Spacecraft Equipped with a Robotic Manipulator. *Int. J. Robotics Res.* 38, 40–72. doi:10.1177/0278364918804660

Yang, H., Shi, J., and Carlone, L. (2020). *Teaser: Fast and Certifiable point Cloud Registration*. IEEE Trans. Robotics.

Conflict of Interest: CO was employed by the company The Charles Stark Draper Laboratory, Inc.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.