AAS 18-420

REAL-TIME ANGULAR VELOCITY ESTIMATION OF NON-COOPERATIVE SPACE OBJECTS USING CAMERA MEASUREMENTS

Marcelino M. de Almeida*, Renato Zanetti†, Daniele Mortari‡, Maruthi Akella§

This paper presents an algorithm for angular velocity estimation of a non-cooperative space object using camera measurements. We consider that the non-cooperative space target is one whose inertia properties and actuation torques are not known. The relative pose of such space object with respect to the camera can be obtained using Simultaneous Localization and Mapping (SLAM) methods. In this paper, we specifically adopt the ORB-SLAM package, which has already been validated in prior research as a successful tool for SLAM applications in space missions. Using the relative pose between the target and the camera, the angular velocity can be obtained through attitude kinematics. However, the lack of a reliable propagation model for the angular velocity constrains the use of traditional Kalman Filter based methods, which typically require some knowledge of the inertia matrix and any perturbing torques governing the rotational dynamics of the non-cooperative space object. Instead, our work is based on the Discrete Adaptive Angular Velocity Estimator (DAAVE) algorithm to estimate for the target's spin axis, and use this as prior information for a modified version of the Multiplicative Extended Kalman Filter (MEKF) formulation. This work introduces both the DAAVE and the modified MEKF algorithms, and presents the performance of the angular velocity estimator using a camera-target simulator. In our simulator, we are able to use the 3D model of a target of interest, which can be configured to tumble with any desired angular rate, while being visually captured with a camera. The simulation results demonstrate that the algorithm pipeline engagind ORB-SLAM, DAAVE, and the modified MEKF, is successful in adequately tracking the angular velocity of targets in multiple tumbling configurations.

INTRODUCTION

This paper presents a solution to the problem of estimating the relative angular velocity (RAV) between a camera (onboard a chaser spacecraft) and an object in space (the target spacecraft or celestial object) using camera measurements only. Our approach assumes no prior knowledge of the inertial characteristics of the target space object such as shape, size, and mass distribution, making it seamlessly applicable to different applications. If we assume that the angular velocity of the chaser is known, then our approach provides the absolute angular velocity of the target object.

*PhD Student, Dept. of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin. email: marcelino.malmeidan@utexas.edu.

†Assistant Professor, Dept. of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, AAS Fellow. email: renato@utexas.edu.

‡Professor, Aerospace Engineering, Texas A&M University, AAS Fellow. e-mail: mortari@tamu.edu.

§E.P. Schoch Professor, Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, AAS Fellow. email: makella@mail.utexas.edu.

Using camera measurements, the relative pose between the chaser and the target can be estimated by tracking known features (assuming a known target) or through Simultaneous Localization and Mapping (SLAM) algorithms.[1] Previous works show that SLAM algorithms can be used for resolving the relative pose problem in space applications. More specifically, in Ref. 2, the authors use images obtained from NASA's STS-125 Service Mission 4* in tandem with the ORB-SLAM package,[3] demonstrating that it tracked closely the estimated relative pose during the mission.[4] In Ref. 5, the authors use data from the Rosetta mission† to feed an EKF-SLAM algorithm, which estimates Rosetta's spin state, mass, and moments, as well as the chaser's position and velocity.

The main issue with using EKF-based algorithms for estimating the RAV of a non-cooperative target is that the external torques upon the same might be unknown. In this case, any perturbing external torques have to be estimated by extending the states (assuming smooth torques with bounded derivatives) or by using a sufficiently large process noise in the angular velocity covariance propagation. The problem becomes even harder when the target's inertia matrix is unknown, since it is barely observable at long distances.[5]

The lack of precise knowledge of a system's inertia matrix and torque vector also poses a challenge to non Kalman-filtering techniques. Many of the existing angular velocity estimators[6–8] rely on the knowledge of the target's specific inertia and torque parameters. An exception can be made for the *derivative approach* described in Ref. 7, but as the author acknowledges, the angular velocity estimator can produce considerable error due to the presence of measurement noise. In Ref. 9, the authors present the Pseudolinear Kalman Filter (PSELIKA), which does not depend on knowledge of inertia matrix or input torques. However, PSELIKA is developed with the goal of "simplicity rather than accuracy",[9] serving as a crude angular velocity estimator for control loop damping purposes.

An alternative solution to the RAV problem is to use methods based on the Multiplicative Extended Kalman Filter (MEKF),[10–12] since these rely on kinematics only. Still, one needs to have tight bounds upon how fast the angular velocity of the target might be changing with time, and use the process noise covariance as a tuning parameter (i.e., a forgetting factor). If the target is being actuated or it is tumbling (e.g., the Toutatis asteroid‡), then the rate at which the target's angular velocity varies with time is not necessarily constant. In this scenario, properly tuning the forgetting factor becomes a formidable task, thereby providing a strong motivation for the need to resort to adaptive estimators.

In this context, the Discrete Adaptive Angular Velocity Estimator (DAAVE)[13] is an attractive option for real-time applications, since it adaptive, is based on kinematics, and is not computationally expensive. The DAAVE algorithm, as originally presented in Ref. 13, is divided in two parts: one that estimates the angular velocity direction (AVD), and another that estimates its magnitude. In order to calculate the AVD, the DAAVE algorithm uses a sliding window matrix of adaptive size stacked with quaternion measurements. It was shown in Ref. 13 that the singular values of this matrix can be investigated to determine whether the object is executing pure-spin around an inertially fixed axis or if the spin axis direction is changing. Using this crucial insight, the window size can adapt based on how close it captures pure spin with the given quaternion measurements.

In Ref. 13, the angular velocity's magnitude (AVM) is estimated by performing "*dirty*" deriva-

*Service Mission to the Hubble Space Telescope carried out in May-2009.

†https://www.aerosociety.com/news/lecture-report-rosetta-how-we-landed-on-a-comet/

‡https://science.nasa.gov/science-news/science-at-nasa/2012/12dec_toutatis/

tives on the most recently measured quaternions. In contrast, the work of 14 showed that one can obtain better results by pre-filtering the measured quaternions before employing the derivative. The results in Ref. 14 demonstrate the efficacy of the DAAVE algorithm for estimating the angular velocity of a gyroscope with unknown inertial properties, which is actuated by unknown external forces.

One important issue with the AVM solutions in Refs. 13 and 14 is that the AVM estimate can be biased in presence of noise. To address this limitation, the present work synthesizes the DAAVE algorithm together with a modified MEKF. In the formulation for this paper, the DAAVE algorithm is still employed to estimate the AVD, while a modified MEKF uses the knowledge of AVD to estimate the AVM only. Instead of three tuning parameters (as would be required in a pure MEKF approach[10–12]), only one parameter is needed in new this formulation, which encapsulates the rate of change of the AVM.

The work in this paper also departs from the results in Refs. 13 and 14 by utilizing a different method for adapting the size of the sliding window. Ref. 13 suggests to compare the lower singular values of the sliding window with a threshold, but the guidelines on how to establish this threshold for any specific application require further attention. In this paper, we propose the use of residual autocorrelation to determine whether the sliding window size can increase or not.[15–17]

In terms of the overall algorithm implementation, our approach uses camera images to feed into a SLAM algorithm, which is able to determine the relative pose between the target and the chaser. Towards this goal, we employ the ORB-SLAM algorithm that was also used earlier in Ref. 2. As already shown in Ref. 2, ORB-SLAM is capable of running in real time (no need for post-processing), and it has been documented to produce satisfactory results in numerous applications. In order to estimate the AVD, we use the DAAVE algorithm, and our Modified MEKF (MMEKF) is employed for estimating the AVM.

The remainder of this paper is organized as follows: first, we introduce the attitude kinematics and dynamics, and introduce notations and parametrizations. Then, we pose the problem of estimating the angular velocity of a target object using camera measurements, also introducing the assumed statistics of the measurement noise. The section that follows presents the DAAVE algorithm, followed by a section that introduces the MMEKF algorithm. Finally, we present simulation results, followed by conclusions for this work.

ATTITUDE KINEMATICS AND DYNAMICS

We adopt the notation $\boldsymbol{q}_A^B$ to represent the relative orientation quaternion between frames A and B. A quaternion is written in the form

$$\boldsymbol{q}_A^B = \begin{bmatrix} q_{As}^B \\ \boldsymbol{q}_{Av}^B \end{bmatrix}, \tag{1}$$

where $\boldsymbol{q}_{Av}^B$ and q_{As}^B are the vector and scalar components of the quaternion $\boldsymbol{q}_A^B$, respectively.

We denote the quaternion inverse rotation as $(\boldsymbol{q}_A^B)^{-1} = \boldsymbol{q}_B^A$, which is given by:

$$\boldsymbol{q}_B^A = \begin{bmatrix} q_{As}^B \\ -\boldsymbol{q}_{Av}^B \end{bmatrix}. \tag{2}$$

The quaternion composition rule is denoted as:

$$\boldsymbol{q}_A^C = \boldsymbol{q}_B^C \otimes \boldsymbol{q}_A^B, \tag{3}$$

in which:

$$\boldsymbol{q}_B^C \otimes = \begin{bmatrix} q_{Bs}^C & -(\boldsymbol{q}_{Bv}^C)^T \\ \boldsymbol{q}_{Bv}^C & q_{Bs}^C \boldsymbol{I} - [\boldsymbol{q}_{Bv\times}^C] \end{bmatrix}, \tag{4}$$

where $\boldsymbol{I}$ is the 3×3 identity matrix, and $[\boldsymbol{v}_\times]$ is the skew-symmetric matrix associated with a vector $\boldsymbol{v} \in \mathbb{R}^3$.

Given a vector $\boldsymbol{v} \in \mathbb{R}^3$, then we define $\boldsymbol{v} \otimes \in \mathbb{R}^{4\times4}$ as:

$$\boldsymbol{v} \otimes \triangleq \begin{bmatrix} 0 & -\boldsymbol{v}^T \\ \boldsymbol{v} & -[\boldsymbol{v}_\times] \end{bmatrix}. \tag{5}$$

With some slight abuse of notation, we define the composition of a quaternion $\boldsymbol{q} \in \mathbb{S}^3$ with a vector $\boldsymbol{v} \in \mathbb{R}^3$ as:

$$\boldsymbol{q} \otimes \boldsymbol{v} \triangleq \boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{v} \end{bmatrix}. \tag{6}$$

Given a vector $\boldsymbol{v}^A \in \mathbb{R}^3$ expressed in frame A, its representation in frame B can be obtained as:

$$\begin{bmatrix} 0 \\ \boldsymbol{v}^B \end{bmatrix} = \boldsymbol{q}_A^B \otimes \boldsymbol{v}^A \otimes (\boldsymbol{q}_A^B)^{-1}. \tag{7}$$

Alternatively, $\boldsymbol{v}^B$ can be calculated from $\boldsymbol{v}^A$ using the expression $\boldsymbol{v}^B = \boldsymbol{C}_A^B \boldsymbol{v}^A$, where $\boldsymbol{C}_A^B$ is the direction cosine matrix respective to $\boldsymbol{q}_A^B$:

$$\boldsymbol{C}_A^B = \boldsymbol{I} - 2q_{As}^B[\boldsymbol{q}_{Av\times}^B] + 2[\boldsymbol{q}_{Av\times}^B]^2. \tag{8}$$

An alternative attitude representation is the *Gibbs vector*,[18] which is utilized in the derivation of the modified MEKF. In this work, we use a scaled version of the traditional *Gibbs vector* $\boldsymbol{g}_A^B \in \mathbb{R}^3$, defined as*:

$$\boldsymbol{g}_A^B \triangleq 2\frac{\boldsymbol{q}_{Av}^B}{q_{As}^B}. \tag{9}$$

The transformation from *Gibbs vector* to quaternion is done as follows:

$$\begin{cases} q_{As}^B &= \sqrt{\frac{2}{2+\|\boldsymbol{g}_A^B\|^2}} \\ \boldsymbol{q}_{Av}^B &= \frac{1}{2} q_{As}^B \boldsymbol{g}_A^B \end{cases} \tag{10}$$

*The *Gibbs vector* is classically defined as $\boldsymbol{g}_A^B \triangleq \boldsymbol{q}_{Av}^B / q_{As}^B$. However, the scaled version of this attitude representation adopter here is convenient in the derivation of the MEKF, as shown in Ref. 12.

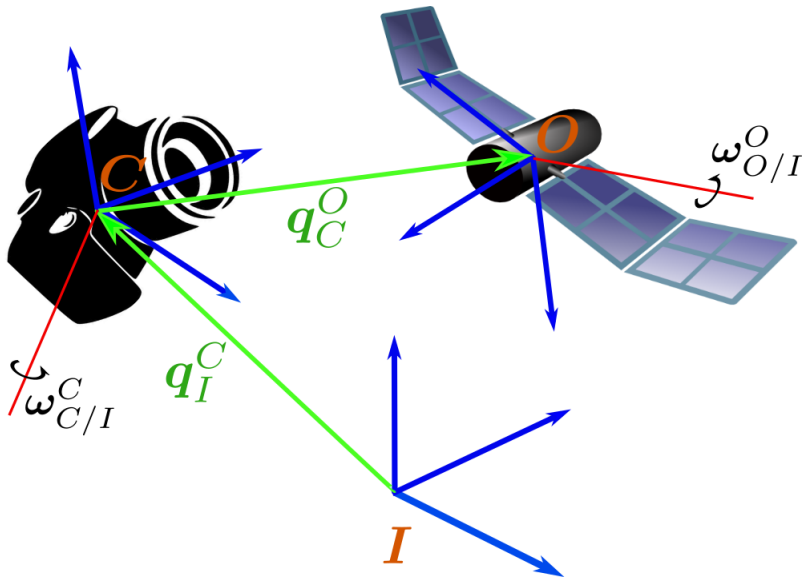


Figure 1: Reference frames and rotational transformations.

Denote $\boldsymbol{\omega}_{B/A}^C \in \mathbb{R}^3$ as the angular velocity of frame B w.r.t. frame A expressed in frame C. Then, the rotational kinematics for $\boldsymbol{q}_A^B$ is given by:

$$\dot{\boldsymbol{q}}_A^B = \frac{1}{2}\boldsymbol{\omega}_{B/A}^B \otimes \boldsymbol{q}_A^B. \tag{11}$$

Assume a frame A that is rotating w.r.t. an inertial frame I with angular velocity $\boldsymbol{\omega}_{A/I}^A$ and a frame B that is rotating w.r.t. the inertial frame with angular velocity $\boldsymbol{\omega}_{B/I}^B$. Then, the relative angular velocity between the two frames is given by:

$$\delta\boldsymbol{\omega}_{B/A}^B = \boldsymbol{\omega}_{B/I}^B - \boldsymbol{\omega}_{A/I}^B \tag{12}$$

$$= \boldsymbol{\omega}_{B/I}^B - \boldsymbol{C}_A^B\boldsymbol{\omega}_{A/I}^A \tag{13}$$

Denoting the matrix $\boldsymbol{J}^B \in \mathbb{R}^{3\times3} > 0$ as the inertia tensor of a rigid body expressed in frame B, the Euler rigid body attitude dynamics is given by:

$$\boldsymbol{J}^B\dot{\boldsymbol{\omega}}_{B/I}^B = -\boldsymbol{\omega}_{B/I}^B \times \boldsymbol{J}^B\boldsymbol{\omega}_{B/I}^B + \boldsymbol{\tau}^B, \tag{14}$$

where $\boldsymbol{\tau}^B \in \mathbb{R}^3$ is an external torque being actuated on the rigid body and I is an inertial frame.

PROBLEM FORMULATION

The various reference frames adopted for this problem is displayed in Fig. 1. We assume a chaser camera (frame C) with known orientation $\boldsymbol{q}_I^C$ w.r.t. a star tracker inertial frame of reference (frame I). We assume that the chaser angular velocity $\boldsymbol{\omega}_{C/I}^C$ is known. Also, we assume a target object (frame O) with unknown relative angular velocity $\boldsymbol{\omega}_{O/C}^O$, but within the field of view of the chaser's camera. In addition, we do not assume knowledge of the target's inertia matrix or actuation torques.

The objective of this work is to obtain the target's angular velocity $\boldsymbol{\omega}_{O/I}^{O}$ through visual inspection. We use ORB-SLAM[3] to measure the relative orientation between the chaser and the target. The measured relative orientation between C and O at time t_k is denoted as the quaternion $\boldsymbol{q}_{C}^{O}(t_k)$. The quaternion parameterizing the absolute pose of the target at time t_k is then calculated as:

$$\boldsymbol{q}_{I}^{O}(t_k) = \boldsymbol{q}_{C}^{O}(t_k) \otimes \boldsymbol{q}_{I}^{C}(t_k) \tag{15}$$

We use the target's pose measurements as inputs to the Discrete Adaptive Angular Velocity Estimator (DAAVE)[13] to estimate the target's axis of rotation, denoted as $\bar{\boldsymbol{\omega}}_{O/I}^{O}$. Finally, the angular velocity magnitude $\Omega_{O/I}^{O}$ (AVM) is estimated through the MMEKF. The target's estimated angular velocity is then:

$$\boldsymbol{\omega}_{O/I}^{O} = \Omega_{O/I}^{O} \bar{\boldsymbol{\omega}}_{O/I}^{O} \tag{16}$$

The target's kinematics can be described as:

$$\dot{\boldsymbol{q}}_{I}^{O} = \frac{1}{2} \boldsymbol{\omega}_{O/I}^{O} \otimes \boldsymbol{q}_{I}^{O}. \tag{17}$$

Assuming that the target's angular velocity $\boldsymbol{\omega}_{O/I}^{O}(t_k)$ is approximately constant throughout the period $t = [t_k, t_{k+1})$, then the solution to Eq. 17 is:

$$\boldsymbol{q}_{I}^{O}(t_{k+1}) = \left[\cos\tfrac{\Omega\delta_k}{2} \cdot \boldsymbol{I} + \sin\tfrac{\Omega\delta_k}{2} \cdot \bar{\boldsymbol{\omega}}_{O/I}^{O} \otimes\right] \boldsymbol{q}_{I}^{O}(t_k) \tag{18}$$

$$= \boldsymbol{A}_d \cdot \boldsymbol{q}_{I}^{O}(t_k), \tag{19}$$

where $\delta_k \triangleq t_{k+1} - t_k$ and $\boldsymbol{A}_d$ is the state transition matrix:

$$\boldsymbol{A}_d[k] \triangleq \cos\frac{\Omega\delta_k}{2} \cdot \boldsymbol{I} + \sin\frac{\Omega\delta_k}{2} \cdot \bar{\boldsymbol{\omega}}_{O/I}^{O} \otimes . \tag{20}$$

For simplicity of notation, the remainder of this paper will denote $\boldsymbol{q}_k \triangleq \boldsymbol{q}_{I}^{O}(t_k)$, $\boldsymbol{\omega} \triangleq \boldsymbol{\omega}_{O/I}^{O}$, $\bar{\boldsymbol{\omega}} \triangleq \bar{\boldsymbol{\omega}}_{O/I}^{O}$, and $\Omega \triangleq \Omega_{O/I}^{O}$. In the filtering section, we denote $\boldsymbol{X}_{k|k-1}$ as the propagated estimation of the quantity $\boldsymbol{X}$ at time k, while $\boldsymbol{X}_{k|k}$ denotes the estimate after the measurement update. We use the notation $\hat{\boldsymbol{X}}_k$ to denote a measurement of the variable $\boldsymbol{X}$ at instant k. To be specific, the quaternion measurement model is given by:

$$\hat{\boldsymbol{q}}_k = \boldsymbol{q}_{Nk} \otimes \boldsymbol{q}_k, \tag{21}$$

where $\boldsymbol{q}_N$ is the noise quaternion:

$$\boldsymbol{q}_{Nk} \triangleq \begin{bmatrix} \cos\frac{\theta_k}{2} \\ \boldsymbol{e}_{Nk} \sin\frac{\theta_k}{2} \end{bmatrix}, \tag{22}$$

in which θ_k and $\boldsymbol{e}_{Nk}$ are independent random variables. We assume that θ_k is Gaussian* such that $\theta_k \sim \mathcal{N}(0, \sigma_\theta^2)$, and $\boldsymbol{e}_{Nk} \in \mathbb{S}^2$ is a unit-norm random vector uniformly distributed in $\mathbb{S}^2 = \{\boldsymbol{x} \in \mathbb{R}^3 : ||\boldsymbol{x}|| = 1\}$ and has the characteristics $\mathbb{E}[\boldsymbol{e}_{Nk}] = \boldsymbol{0}$ and $\mathbb{E}[\boldsymbol{e}_{Nk}\boldsymbol{e}_{Nk}^T] = \frac{1}{3}\boldsymbol{I}$ (see Appendix A).

*Although it might be unrealistic to assume that angles are distributed as Gaussian, Ref. 19 has shown that this is a reasonable approximation for double-precision machines as long as $\sigma_\theta \leq 22\,\text{deg}$.

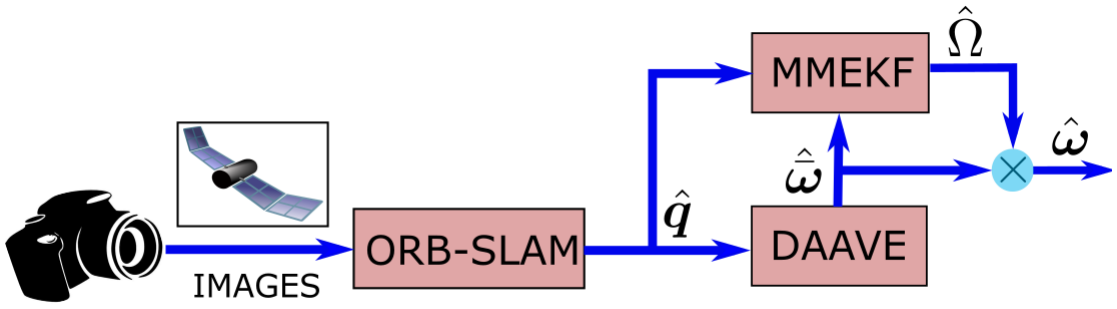


Figure 2: Proposed Algorithm Pipeline.

The Gibbs vector associated with the noise quaternion is given by:

$$\boldsymbol{g}_{Nk} = 2\boldsymbol{e}_{Nk} \tan \frac{\theta_k}{2}. \tag{23}$$

One should notice that $\mathbb{E}[\boldsymbol{g}_N] = 0$. In addition, assuming a small angle approximation, then $\tan^2 \frac{\theta_k}{2} \approx \frac{\theta_k^2}{4}$, leading to $\mathbb{E}[\boldsymbol{g}_N \boldsymbol{g}_N^T] = \frac{1}{3}\sigma_\theta^2 \boldsymbol{I}$.

Figure 2 depicts the suggested pipeline utilized in this work: images are fed to ORB-SLAM, which in turn produces a relative orientation. The relative orientation is used in DAAVE to estimate the AVD, which is then used in MMEKF to estimate for Ω.

THE DISCRETE ADAPTIVE ANGULAR VELOCITY ESTIMATOR

As previously mentioned, the DAAVE[13] algorithm is used to estimate the target's axis of rotation. DAAVE uses a sliding window of variable size, whose length is determined by whether the motion is in pure spin or not.

We denote the sliding window as $\hat{\boldsymbol{Q}} \in \mathbb{R}^{4\times L}$, which is filled with L previous quaternion measurements:

$$\hat{\boldsymbol{Q}} = \begin{bmatrix} \hat{\boldsymbol{q}}_{k-L+1} & \hat{\boldsymbol{q}}_{k-L+2} & \hat{\boldsymbol{q}}_{k-L+3} & \cdots & \hat{\boldsymbol{q}}_k \end{bmatrix}. \tag{24}$$

The length L of the sliding window $\hat{\boldsymbol{Q}}$ is adaptive and such that $3 \leq L \leq L_{max}$, where L_{max} is a user-specified upper bound on the window size, and $L \geq 3$ makes sure that at least three measurements are available to determine if the motion is in pure spin or not. A motion is considered to be in pure spin when $\operatorname{rank} \hat{\boldsymbol{Q}} = 2$, i.e., all quaternion measurements in the window $\hat{\boldsymbol{Q}}$ belong to the same plane of rotation. Since two quaternions always belong to the same plane of rotation, at least three measurements are needed in $\hat{\boldsymbol{Q}}$ to determine the existence of out-of-plane motion.

A brief summary of the DAAVE algorithm is now in order. The DAAVE algorithm tries to fit the quaternions in $\hat{\boldsymbol{Q}}$ into a 4-D plane spanned by vectors $\hat{\boldsymbol{u}}_1 \in \mathbb{S}^3$ and $\hat{\boldsymbol{u}}_2 \in \mathbb{S}^3$, with $\mathbb{S}^3 \triangleq \{\boldsymbol{x} \in \mathbb{R}^4 :$

$||\boldsymbol{x}|| = 1\}$. The cost function associated with this fitting problem is given by:

$$\begin{cases} \arg \max\limits_{\hat{\boldsymbol{u}}_1 \in \mathbb{S}^3, \hat{\boldsymbol{u}}_2 \in \mathbb{S}^3} \sum\limits_{i=k-L+1}^{L} \left[(\hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_1)^2 + (\hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_2)^2 \right] \\ \text{s.t. } \hat{\boldsymbol{u}}_1^T \hat{\boldsymbol{u}}_2 = 0 \end{cases} \tag{25}$$

Defining the cost function $J(\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2) \triangleq \sum_{i=k-L+1}^{L} \left[(\hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_1)^2 + (\hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_2)^2 \right]$, then $J(\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2)$ can be rewritten as:

$$J(\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2) = \sum_{i=k-L+1}^{L} \left[\hat{\boldsymbol{u}}_1^T \hat{\boldsymbol{q}}_i \hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_1 + \hat{\boldsymbol{u}}_2^T \hat{\boldsymbol{q}}_i \hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_2 \right] \tag{26}$$

$$= \hat{\boldsymbol{u}}_1^T \hat{\boldsymbol{Q}} \hat{\boldsymbol{Q}}^T \hat{\boldsymbol{u}}_1 + \hat{\boldsymbol{u}}_2^T \hat{\boldsymbol{Q}} \hat{\boldsymbol{Q}} \hat{\boldsymbol{u}}_2 \tag{27}$$

Then, the optimization problem in Eq. 25 can be restated as:

$$\begin{cases} \arg \max\limits_{\hat{\boldsymbol{u}}_1 \in \mathbb{S}^3, \hat{\boldsymbol{u}}_2 \in \mathbb{S}^3} (\hat{\boldsymbol{u}}_1^T + \hat{\boldsymbol{u}}_2^T) \hat{\boldsymbol{Q}} \hat{\boldsymbol{Q}}^T (\hat{\boldsymbol{u}}_1 + \hat{\boldsymbol{u}}_2) \\ \text{s.t. } \hat{\boldsymbol{u}}_1^T \hat{\boldsymbol{u}}_2 = 0 \end{cases} \tag{28}$$

The solution to the optimization problem in Eq. 25 can be obtained from the Singular Value Decomposition $\hat{\boldsymbol{Q}} = \hat{\boldsymbol{U}} \tilde{\boldsymbol{\Sigma}} \hat{\boldsymbol{V}}^T$, where $\hat{\boldsymbol{U}} \in \mathbb{R}^{4\times4} = \begin{bmatrix} \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{u}}_2 & \hat{\boldsymbol{u}}_3 & \hat{\boldsymbol{u}}_4 \end{bmatrix}$ contains the *left singular vectors of* $\hat{\boldsymbol{Q}}$, and $\hat{\boldsymbol{u}}_1$ and $\hat{\boldsymbol{u}}_2$ compose the solution to the optimization problem in Eq. 28. The matrix $\hat{\boldsymbol{V}} \in \mathbb{R}^{4\times L} = \begin{bmatrix} \hat{\boldsymbol{v}}_1 & \hat{\boldsymbol{v}}_2 & \cdots & \hat{\boldsymbol{v}}_L \end{bmatrix}$ contains the *right singular vectors of* $\hat{\boldsymbol{Q}}$, and $\tilde{\boldsymbol{\Sigma}} = \begin{bmatrix} \hat{\boldsymbol{\Sigma}}^{\frac{1}{2}} & \boldsymbol{0}_{4\times(L-4)} \end{bmatrix}$ contains the *singular values* of $\hat{\boldsymbol{Q}}$ within $\hat{\boldsymbol{\Sigma}}^{1/2} = \text{diag}(\hat{\sigma}_1^{1/2}, \hat{\sigma}_2^{1/2}, \hat{\sigma}_3^{1/2}, \hat{\sigma}_4^{1/2})$, wherein $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \hat{\sigma}_3 \geq \hat{\sigma}_4^{1/2} \geq 0$. The optimal cost is given by $J^*(\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2) = \hat{\sigma}_1 + \hat{\sigma}_2$.

For convenience, we introduce the matrix $\hat{\boldsymbol{Z}} \in \mathbb{R}^{4\times4} = \hat{\boldsymbol{Q}} \hat{\boldsymbol{Q}}^T$, which can be written as $\hat{\boldsymbol{Z}} = \hat{\boldsymbol{U}} \hat{\boldsymbol{\Sigma}} \hat{\boldsymbol{U}}^T$.

Without loss of generality, the matrix $\hat{\boldsymbol{U}}$ is assumed to be constructed in proper orthogonal form, i.e., $\det \hat{\boldsymbol{U}} = +1$. In addition, the directions of $\hat{\boldsymbol{u}}_1$ and $\hat{\boldsymbol{u}}_2$ must be selected consistent with the quaternions and their derivatives, respectively. Ref. 13 recommends to choose these vectors such that $\hat{\boldsymbol{u}}_1^T (\hat{\boldsymbol{q}}_k + \hat{\boldsymbol{q}}_{k-l+1}) > 0$ and $\hat{\boldsymbol{u}}_2^T (\hat{\boldsymbol{q}}_k - \hat{\boldsymbol{q}}_{k-l+1}) > 0$.

If the motion is in pure spin configuration (and in the absence of measurement noise), then $\hat{\sigma}_3 = \hat{\sigma}_4 = 0$, and the plane of rotation can be determined as $\boldsymbol{P}_1 \triangleq \begin{bmatrix} \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{u}}_2 \end{bmatrix}$, whereas the null space of the motion is given by $\boldsymbol{P}_2 \triangleq \begin{bmatrix} \hat{\boldsymbol{u}}_3 & \hat{\boldsymbol{u}}_4 \end{bmatrix}$. As shown in Ref. 13, the estimated angular velocity direction vector $\bar{\boldsymbol{\omega}}$ can be calculated as the unit vector:

$$\hat{\bar{\boldsymbol{\omega}}} = \boldsymbol{W}_1 (\boldsymbol{P}_1 \boldsymbol{J}_2 \boldsymbol{P}_1^T + \boldsymbol{P}_2 \boldsymbol{J}_2 \boldsymbol{P}_2^T) \boldsymbol{W}_2, \tag{29}$$

where $\boldsymbol{J}_2$, $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are given by:

$$\boldsymbol{J}_2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{W}_1 = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0}_{3\times1} \end{bmatrix}, \quad \boldsymbol{W}_2 = \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ 1 \end{bmatrix}, \tag{30}$$

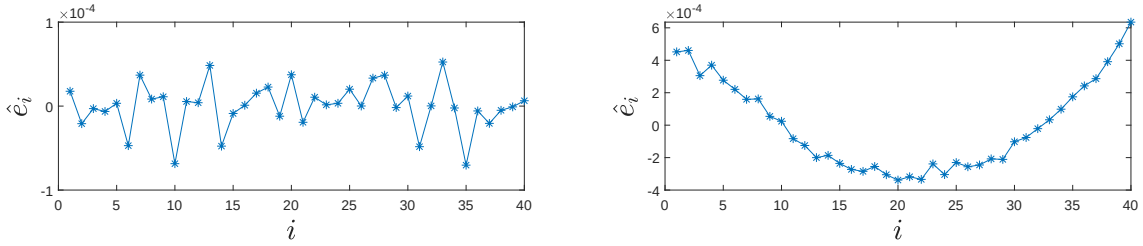


Figure 3: Residual plot for planar motion (left) and quaternion motion with out-of-plane component (right). The index i represents the subscript for $\hat{e}_i \triangleq \hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_3$.

and $\mathbf{0}_{3\times 1} \in \mathbb{R}^{3\times 1}$ is a vector filled with zeros.

In order to determine the size of the sliding window L, we need to identify whether the quaternion measurements in $\hat{\boldsymbol{Q}}$ seem to fit the model assumption of planar motion. A common procedure to verify whether a model fits the given data can be made by analyzing the regression residuals.[15] We define our vector of residuals as the projection of the quaternion measurements onto the third singular vector $\hat{\boldsymbol{u}}_3$:

$$\hat{e}_i \triangleq \hat{\boldsymbol{q}}_i^T \hat{\boldsymbol{u}}_3, \quad i \in \{k-L+1, \cdots k\} \quad \Longrightarrow \quad \hat{\boldsymbol{e}} \triangleq \begin{bmatrix} \hat{e}_{k-L+1} & \cdots & \hat{e}_k \end{bmatrix}^T = \hat{\boldsymbol{Q}}^T \hat{\boldsymbol{u}}_3. \tag{31}$$

The covariace of the residual sequence $\boldsymbol{P}_e \triangleq \mathbb{E}[\hat{\boldsymbol{e}}\hat{\boldsymbol{e}}^T]$ is typically a non-diagonal matrix, implying that the residuals within $\hat{\boldsymbol{e}}$ is an autocorrelated sequence.[15] However, this correlation is generally unimportant (weakly autocorrelated), as discussed in Ref. [16, p. 171]. Figure 3 depicts a typical simulated scenario displaying the residual sequence $\hat{e}_i,\ i \in \{1, \cdots, 40\}$ corresponding to the case when all quaternions within a window $\hat{\boldsymbol{Q}}$ of length $L = 40$ stem from planar motion measurements (left plot) and when they do not (right plot). Visually inspecting, the right-hand plot in Figure 3 is, qualitatively speaking, *more autocorrelated* than the plot on the left.

In order to quantify autocorrelation in the sequence $\hat{\boldsymbol{e}}$, we use the following one-lag autocorrelation formula [17, p. 31]:

$$r_1 = \frac{1}{L \cdot r_0} \sum_{i=k-L+1}^{k-1} (\hat{e}_i - \mu_e)(\hat{e}_{i+1} - \mu_e), \tag{32}$$

where μ_e and r_0 are, respectively, the mean and zero-lag autocorrelation of the sequence $\hat{\boldsymbol{e}}$:

$$\mu_e = \frac{1}{L} \sum_{i=k-L+1}^{k} \hat{e}_i, \quad r_0 = \frac{1}{L} \sum_{i=k-L+1}^{k} (\hat{e}_i - \mu_e)^2. \tag{33}$$

The one-lag autocorrelation signal as defined in Eq. 32 satisfies $0 \leq |r_1| \leq 1$, where the signal is one-lag perfectly correlated when $|r_1| \to 1$, and is one-lag uncorrelated when $|r_1| \to 0$. In addition, our experience suggest that one-lag autocorrelation of residuals are typically negative when the model fits the data (i.e., neighboring residuals tend to have opposite signs), while we expect positive autocorrelation when the model does not fit the data as in the right plot of Figure 3. For instance, the residuals in Figure 3 present one-lag autocorrelation of $r_1 = -0.15$ (left plot) and $r_1 = 0.8705$ (right plot).

In order to obtain confidence bounds on whether a sequence is autocorrelated, we need to estimate the autocorrelation covariance. To that end, we use the following expression [17, p. 188]:

$$\sigma_{r1}^2 \triangleq \mathrm{var}[r_1] = \frac{1}{N}\left(1 + 2r_1^2\right). \tag{34}$$

The decision of decreasing or increasing the window length $3 \leq L \leq L_{max}$ can be made by performing the comparison of r_1 with a tuning threshold r_1^*. Whenever the motion is close to pure spin, i.e. $r_1 < r_1^*$, the window length is allowed to increase ($L = L + 1$), otherwise whenever $r_1 \geq r_1^*$, it decreases ($L = L - 1$). Driven by extensive numerical simulations of this algorithm, we found that a reasonable choice for the threshold is $r_1^* = \sigma_{r1}$, where σ_{r1} is defined in Eq. 34.

The estimated direction of the angular velocity vector $\hat{\bar{\boldsymbol{\omega}}}$ from Eq. 29 will be used within the MMEKF for estimating the AVM $\hat{\Omega}$.

MODIFIED MEKF FOR ANGULAR VELOCITY MAGNITUDE ESTIMATION

The MMEKF presented herein assumes that the direction of the angular velocity vector is known to be $\bar{\boldsymbol{\omega}} = \hat{\bar{\boldsymbol{\omega}}}$ (Eq. 29) and that it is not a stochastic quantity.

Following the same derivations as in Ref. 12, we define the reference trajectory kinematics:

$$\dot{\boldsymbol{q}}_R = \frac{1}{2}\boldsymbol{\omega}_R \otimes \boldsymbol{q}_R, \tag{35}$$

where $\boldsymbol{q}_R \triangleq \begin{bmatrix} q_{Rs} & \delta\boldsymbol{q}_{Rv}^T \end{bmatrix}^T$ is the reference quaternion and $\boldsymbol{\omega}_R = \Omega_R \cdot \bar{\boldsymbol{\omega}}$ is the reference angular velocity of the reference attitude, with magnitude Ω_R and axis of rotation aligned with the true angular velocity $\boldsymbol{\omega} = \Omega \cdot \bar{\boldsymbol{\omega}}$.

The true attitude $\boldsymbol{q}$ can be represented as:

$$\boldsymbol{q} = \delta\boldsymbol{q} \otimes \boldsymbol{q}_R, \tag{36}$$

where $\delta\boldsymbol{q} \triangleq \begin{bmatrix} \delta q_s & \delta\boldsymbol{q}_v^T \end{bmatrix}^T$ represents the rotation from $\boldsymbol{q}_R$ to the true rotation.

Differentiating Eq. 36, we get:

$$\dot{\boldsymbol{q}} = \delta\dot{\boldsymbol{q}} \otimes \boldsymbol{q}_R + \delta\boldsymbol{q} \otimes \dot{\boldsymbol{q}}_R \tag{37}$$

$$\frac{1}{2}\boldsymbol{\omega} \otimes \boldsymbol{q} = \delta\dot{\boldsymbol{q}} \otimes \boldsymbol{q}_R + \frac{1}{2}\delta\boldsymbol{q} \otimes \boldsymbol{\omega}_R \otimes \boldsymbol{q}_R. \tag{38}$$

Post-multiplying Eq. 38 by $\boldsymbol{q}_R^{-1}$ and isolating $\delta\dot{\boldsymbol{q}}$, we get:

$$\delta\dot{\boldsymbol{q}} = \frac{1}{2}\left(\boldsymbol{\omega} \otimes \boldsymbol{q} \otimes \boldsymbol{q}_R^{-1} - \delta\boldsymbol{q} \otimes \boldsymbol{\omega}_R\right) \tag{39}$$

$$= \frac{1}{2}\left(\boldsymbol{\omega} \otimes \delta\boldsymbol{q} - \delta\boldsymbol{q} \otimes \boldsymbol{\omega}_R\right) \tag{40}$$

$$= \frac{1}{2}\left(\begin{bmatrix} 0 & -\boldsymbol{\omega} \\ \boldsymbol{\omega} & -[\boldsymbol{\omega}_\times] \end{bmatrix}\begin{bmatrix} \delta q_s \\ \delta\boldsymbol{q}_v \end{bmatrix} - \begin{bmatrix} \delta q_s & -\delta\boldsymbol{q}_v^T \\ \delta\boldsymbol{q}_v & \delta q_s\boldsymbol{I} - [\delta\boldsymbol{q}_{v\times}] \end{bmatrix}\begin{bmatrix} 0 \\ \boldsymbol{\omega}_R \end{bmatrix}\right) \tag{41}$$

After some algebraic manipulations, we get that:

$$\delta\dot{\boldsymbol{q}}_s = \Big(\boldsymbol{\omega}_R - \boldsymbol{\omega}\Big)^T \delta\boldsymbol{q}_v \tag{42}$$

$$\delta\dot{\boldsymbol{q}}_v = \Big(\boldsymbol{\omega} - \boldsymbol{\omega}_R\Big)\,\delta\boldsymbol{q}_s - \Big(\boldsymbol{\omega} + \boldsymbol{\omega}_R\Big) \times \delta\boldsymbol{q}_v. \tag{43}$$

Defining the attitude error Gibbs vector:

$$\boldsymbol{g} \triangleq 2\frac{\delta\boldsymbol{q}_v}{\delta\boldsymbol{q}_s}, \tag{44}$$

then the Gibbs error kinematics is described as:

$$\dot{\boldsymbol{g}} = 2\frac{\delta\dot{\boldsymbol{q}}_v}{\delta\boldsymbol{q}_s} - 2\frac{\delta\boldsymbol{q}_v}{\delta\boldsymbol{q}_s}\frac{\delta\dot{\boldsymbol{q}}_s}{\delta\boldsymbol{q}_s} \tag{45}$$

$$= \Big[\boldsymbol{I} + \tfrac{1}{4}\boldsymbol{g}\boldsymbol{g}^T\Big]\Big(\boldsymbol{\omega} - \boldsymbol{\omega}_R\Big) - \frac{1}{2}\Big(\boldsymbol{\omega} + \boldsymbol{\omega}_R\Big) \times \boldsymbol{g}. \tag{46}$$

Since $\boldsymbol{\omega}_R$ is assumed to be aligned with $\boldsymbol{\omega}$, then $\delta\boldsymbol{\omega} = \boldsymbol{\omega} - \boldsymbol{\omega}_R = (\Omega - \Omega_R)\bar{\omega}$. In addition, we assume the first order approximations $\boldsymbol{g}\boldsymbol{g}^T \approx 0$, and $\delta\boldsymbol{\omega} \times \boldsymbol{g} \approx 0$, leading to:

$$\dot{\boldsymbol{g}} \approx \delta\boldsymbol{\omega} - \boldsymbol{\omega}_R \times \boldsymbol{g}. \tag{47}$$

Since we have no model for the dynamics of Ω, we assume it to be a random walk process:

$$\dot{\Omega} = \eta_\Omega, \tag{48}$$

where $\eta_\Omega \sim \mathcal{N}(0, Q_\Omega)$ and Q_Ω is a tuning knob that accommodates the expected rate at which Ω changes with time. Alternatively, if the angular velocity magnitude Ω is also known to be bounded with known upper bounds, then the dynamics for Ω could also be modeled as a first order Gauss-Markov process of the type:[11]

$$\dot{\Omega} = -\frac{1}{\tau}\Omega + \eta_\Omega, \tag{49}$$

where τ is a time constant. The model of Eq. 49 implies that, in steady state, Ω is zero-mean and has bounded covariance, i.e., it is likely to be in the range $\Omega_{min} \leq \Omega \leq \Omega_{max}$ with probability p_Ω, where Ω_{min} and Ω_{max} are function of Q_Ω, τ and the likelihood parameter $p_\Omega \in [0, 1]$. On the other hand, the model in Eq. 48 has no steady state value for the covariance, implying that Ω has no range of higher likelihood.

Defining the state vector $\boldsymbol{X} \triangleq \begin{bmatrix}\boldsymbol{g}^T & \Omega\end{bmatrix}^T$ and the dynamics of Eqs. 47 and 48, then the linearized state dynamics is given by:

$$\dot{\boldsymbol{X}} = \underbrace{\begin{bmatrix}-\Omega_R[\bar{\boldsymbol{\omega}}_\times] & \bar{\boldsymbol{\omega}} \\ \boldsymbol{0} & 0\end{bmatrix}}_{\triangleq \boldsymbol{F}} \boldsymbol{X} + \underbrace{\begin{bmatrix}\boldsymbol{0} \\ 1\end{bmatrix}}_{\triangleq \boldsymbol{G}} \eta. \tag{50}$$

We define the state transition matrix:

$$\boldsymbol{F}_d[k] \triangleq e^{F\delta_k}, \quad \delta_k \triangleq t_{k+1} - t_k. \tag{51}$$

In the propagation step, the following equations are used:

$$\boldsymbol{q}_{k+1|k} = \boldsymbol{A}_d[k]\boldsymbol{q}_{k|k}, \tag{52}$$

$$\Omega_{k+1|k} = \Omega_{k|k}, \tag{53}$$

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{F}_d[k]\boldsymbol{P}_{k|k}\boldsymbol{F}_d^T[k] + \boldsymbol{Q}_d[k] \tag{54}$$

where $\boldsymbol{P}_{k|k} \triangleq \mathbb{E}[\boldsymbol{X}_{k|k}\boldsymbol{X}_{k|k}^T]$, and $\boldsymbol{P}_{k+1|k} \triangleq \mathbb{E}[\boldsymbol{X}_{k+1|k}\boldsymbol{X}_{k+1|k}^T]$, $\boldsymbol{A}_d[k]$ is defined in Eq. 20, and:

$$\boldsymbol{Q}_d[k] = Q_\Omega \boldsymbol{G}_d[k]\boldsymbol{G}_d^T[k], \tag{55}$$

where:

$$\boldsymbol{G}_d[k] = \int_{t_k}^{t_{k+1}} e^{F(t_{k+1}-\sigma)}\boldsymbol{G}\, d\sigma = \begin{bmatrix} \frac{\delta_k^2}{2}\bar{\boldsymbol{\omega}} \\ \delta_k \end{bmatrix}. \tag{56}$$

As for the measurement model, only quaternion measurements are available. The innovation term is given by:

$$\boldsymbol{\nu}_k = 2\frac{\tilde{\boldsymbol{q}}_v[k]}{\tilde{q}_s[k]}, \tag{57}$$

where $\tilde{q}_s[k]$ and $\tilde{\boldsymbol{q}}_v[k]$ are, respectively, the scalar and vector parts of $\tilde{\boldsymbol{q}}_k$, defined as:

$$\tilde{\boldsymbol{q}}_k \triangleq \begin{bmatrix} \tilde{q}_s[k] \\ \tilde{\boldsymbol{q}}_v[k] \end{bmatrix} \triangleq \hat{\boldsymbol{q}}_k \otimes \boldsymbol{q}_{k|k-1}^{-1}. \tag{58}$$

Assuming the measurement noise defined in Eq. 23, the measurement covariance is given by $\boldsymbol{R}_k \triangleq \mathbb{E}[\boldsymbol{g}_N\boldsymbol{g}_N^T] = \frac{1}{3}\sigma_\theta^2\boldsymbol{I}$.

The measurement update step uses the following expressions:

$$\boldsymbol{H}_k = \begin{bmatrix} \boldsymbol{I}_3 & \boldsymbol{0}_{3\times 1} \end{bmatrix}, \tag{59}$$

$$\boldsymbol{S}_k = \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k, \tag{60}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T\boldsymbol{S}_k^{-1}, \tag{61}$$

$$\Delta\boldsymbol{x}_{k|k} = \boldsymbol{K}_k\boldsymbol{\nu}_k, \tag{62}$$

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_{k|k-1}(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k)^T + \boldsymbol{K}_k\boldsymbol{R}_k\boldsymbol{K}_k^T, \tag{63}$$

where $\Delta\boldsymbol{x}_{k|k} \triangleq \begin{bmatrix} \Delta\boldsymbol{x}_1 & \Delta\boldsymbol{x}_2 & \Delta\boldsymbol{x}_3 & \Delta\boldsymbol{x}_4 \end{bmatrix}^T$ is the incremental state update typical for standard EKF formulations.

The updated state $\boldsymbol{g}_{k|k}$ can be obtained from $\Delta\boldsymbol{x}_{k|k}$ as $\boldsymbol{g}_{k|k} = \begin{bmatrix} \Delta\boldsymbol{x}_1 & \Delta\boldsymbol{x}_2 & \Delta\boldsymbol{x}_3 \end{bmatrix}^T$. Bearing in mind that $\boldsymbol{g}_{k|k}$ represents the attitude error respective to $\delta\boldsymbol{q}_k$ (see Eqs. 36 and 44), then $\delta\boldsymbol{q}_{k|k}$ can be obtained from $\boldsymbol{g}_{k|k}$ using the transformation in Eq. 10. Finally, the updated states are given by:

$$\boldsymbol{q}_{k|k} = \delta\boldsymbol{q}_{k|k} \otimes \boldsymbol{q}_{k|k-1}, \tag{64}$$

$$\Omega_{k|k} = \Omega_{k|k-1} + \Delta\boldsymbol{x}_4. \tag{65}$$

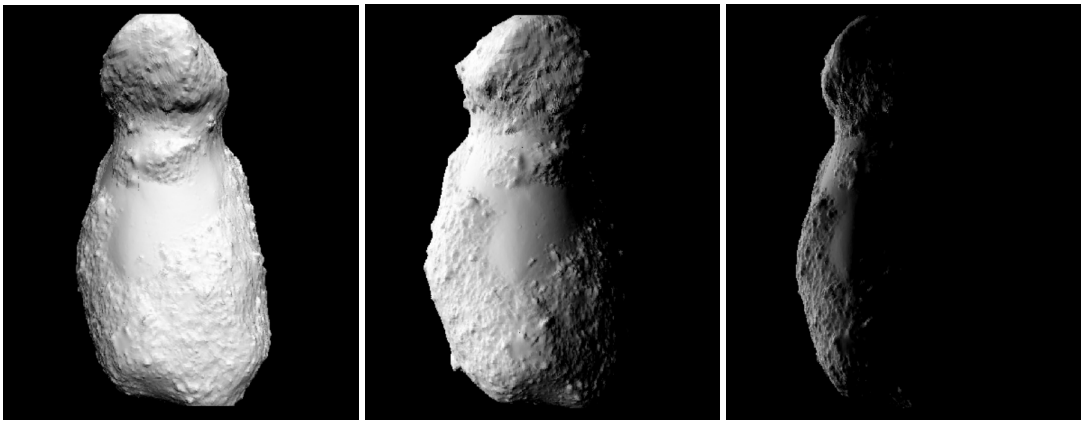


Figure 4: Itokawa rendering with different light sources.

SIMULATION RESULTS

In order to numerically test our proposed algorithm pipeline shown in Figure 2, we have developed a simulator that can obtain visual feed of a tumbling object*. The simulator is able to obtain rendered images in either monocular or stereo modes from 3D CAD models. The simulator is able to display the 3D model in any pose, as well as set the camera at any pose as well, allowing us to have a truth baseline. In addition, one can prescribe any desired values for the camera's resolution, focal lengths, and stereo baseline. Figure 4 shows some examples of renderings that were obtained with the simulator using a 3D model† for the Itokawa asteroid,[20] assuming a camera with resolution of 720p. The images in Figure 4 (from left to right) display the asteroid with frontal light source, lateral light source, and a fading lateral light source (near eclipse).

In order to test the proposed DAAVE-MMEKF algorithm outlined in this paper, we set the Itokawa asteroid to tumble according with unperturbed attitude dynamics, assuming a normalized inertia matrix (inertia matrix divided by the asteroid's mass) $\boldsymbol{J}_I$ given by Ref. 20:

$$\boldsymbol{J}_I = \begin{bmatrix} 0.00673 & 0 & 0 \\ 0 & 0.02122 & 0 \\ 0 & 0 & 0.02235 \end{bmatrix} \text{km}^2. \tag{66}$$

We have simulated Itokawa's attitude dynamics in different hypothetical tumbling and lighting conditions. Each experiment is recorded for 20 minutes and the camera pose is assumed stationary, without loss of generality. For each scenario, we run ORB-SLAM to determine the relative pose of the camera with respect to the asteroid. An example of the camera's relative trajectory w.r.t. the tumbling asteroid is shown in Figure 5-(left), while 5-(right) displays a sample of tracked Orb features in one frame. The ORB-SLAM algorithm is able to produce a sequence of relative poses at a rate of approximately 10Hz‡, hence $\delta_k \approx 0.1$s. According with the data we've obtained, ORB-SLAM is able to produce orientation measurements with an accuracy of $\sigma_\theta \approx 0.002rad= 412.5$arcsec. These orientation measurements are fed incrementally to the DAAVE-MEKF algorithm to estimate the

*The simulator is open source and can be downloaded from `https://github.com/marcelinomalmeidan/view_asteroid`.

†`https://nasa3d.arc.nasa.gov/detail/itokawa`

‡These results were obtained in a computer with an Intel Core i5-4690K CPU (Quad Core 3.50GHz).

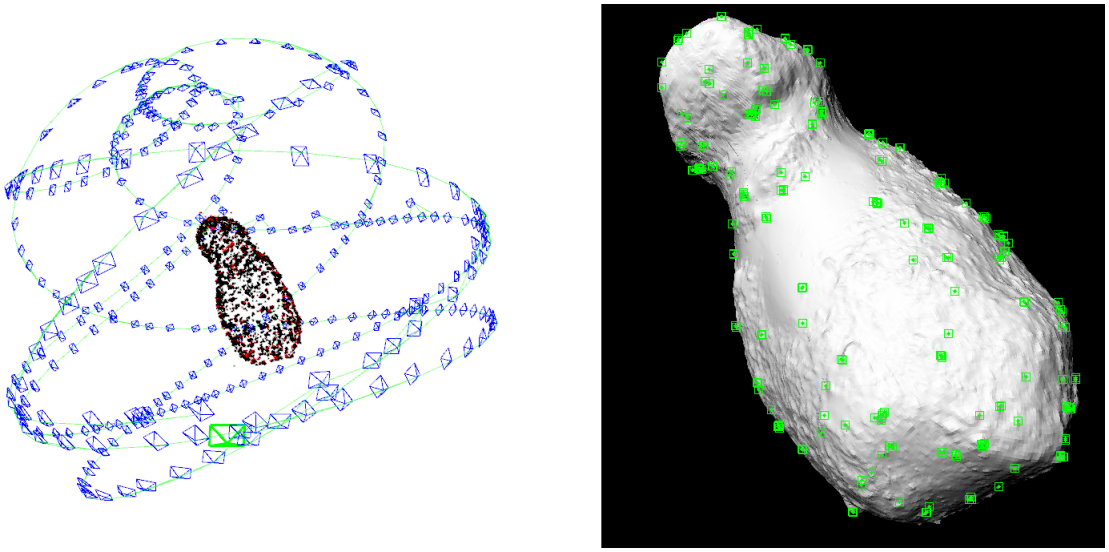


Figure 5: Left: History of the camera's pose with respect to the asteroid's fixed frame determined from running the ORB-SLAM algorithm. The red and black dots are features on the asteroid surface. Right: Example of features taken from one frame in the image plane.

target's RAV. The algorithm parameters for all simulations were chosen as $L_{max} = 200$, $r_1^* = \sigma_{r1}$ (as defined in Eq. 34), and $Q_\Omega = 10^{-5}$.

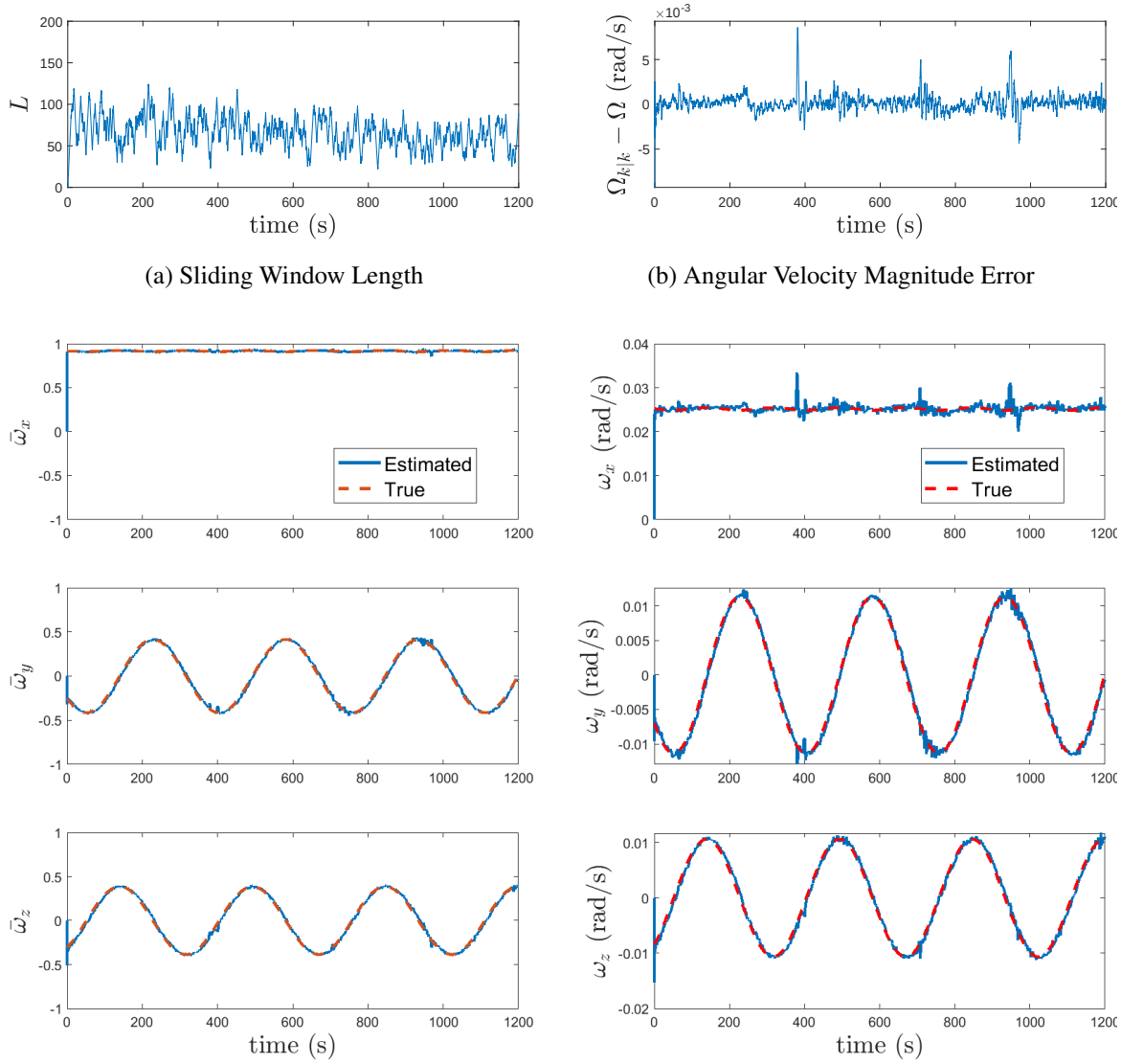
Figure 6 shows the results for a simulation in which Itokawa's initial angular velocity is given by $\boldsymbol{\omega}(0) = \begin{bmatrix} 0.025, & 0.01, & 0.005 \end{bmatrix}^T$. Figure 6(a) shows the sliding window length for $\hat{\boldsymbol{Q}}$, Figure 6(b) shows the angular velocity magnitude error, Figure 6(c) superimposes the true axis of rotation with the estimated one, and Figure 6(d) superimposes the true angular velocity with the estimated one. Figure 7 shows the results for a simulation with initial angular velocity $\boldsymbol{\omega}(0) = \begin{bmatrix} 0.01, & 0.02, & -0.005 \end{bmatrix}^T$ (higher angular velocity in the unstable axis of rotation), but with fading lateral light source (near eclipse - see Figure 4). We do not observe any algorithm performance degradation on these results when compared to the previous one, which had better light conditions.

We have also executed some simulations using a 3D model* of the Cassini spacecraft (see Figure 8), assuming the inertia tensor:[21]

$$\boldsymbol{J}_c = \begin{bmatrix} 8810 & -136.8 & 115.3 \\ -136.8 & 7922.7 & 192.1 \\ 115.3 & 192.1 & 4586.2 \end{bmatrix} \tag{67}$$

Figure 9 shows the results for a tumbling motion of Cassini with initial angular velocity $\boldsymbol{\omega}_2 \triangleq \boldsymbol{\omega}(0) = \begin{bmatrix} 0.01, & 0.02, & 0.005 \end{bmatrix}^T$ (again, principal motion is around the unstable axis of rotation). Similarly, Figure 10 shows the results for a perturbed tumbling motion of Cassini, with perturbation

*https://nasa3d.arc.nasa.gov/detail/jpl-vtad-cassini



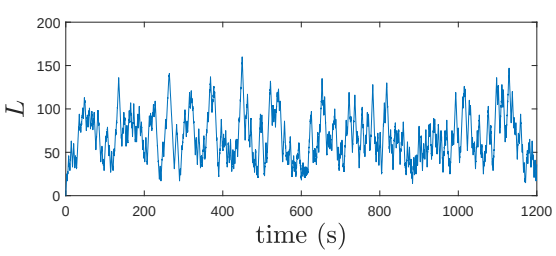
(a) Sliding Window Length

(b) Angular Velocity Magnitude Error

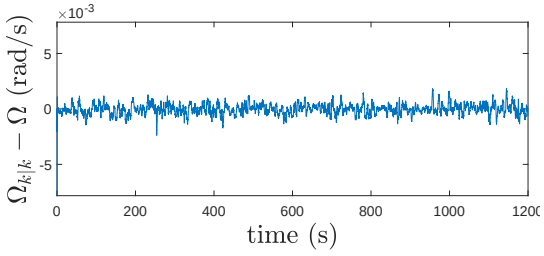
(c) Axis of Rotation

(d) Angular Velocity

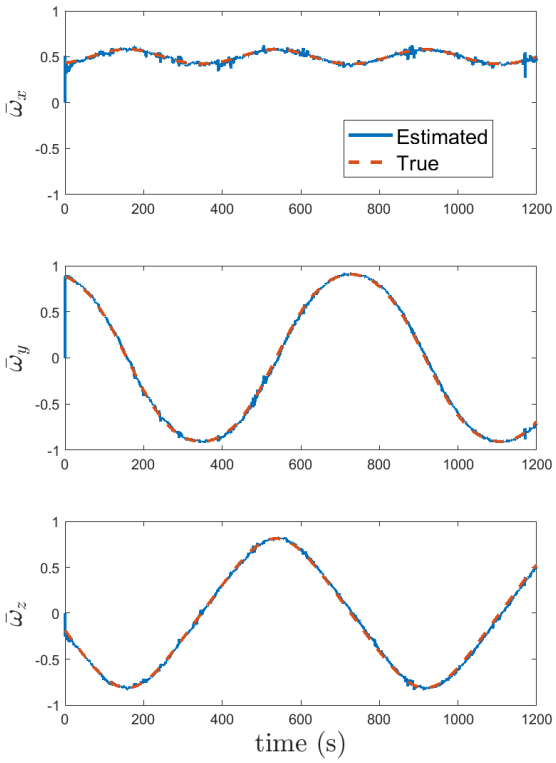
Figure 6: Simulation results for Itokawa's tumbling motion assuming initial angular velocity of $\boldsymbol{\omega}(0) = [0.025,\ 0.01,\ 0.005]^T$



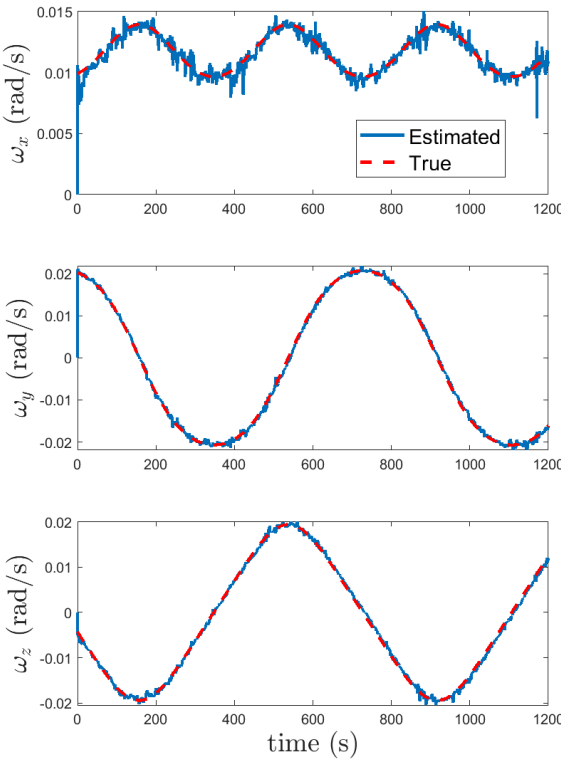
(a) Sliding Window Length



(b) Angular Velocity Magnitude Error



(c) Axis of Rotation



(d) Angular Velocity

Figure 7: Simulation results for Itokawa's tumbling motion with poor lighting conditions assuming initial angular velocity of $\boldsymbol{\omega}(0) = [0.01,\ 0.02,\ -0.005]^T$

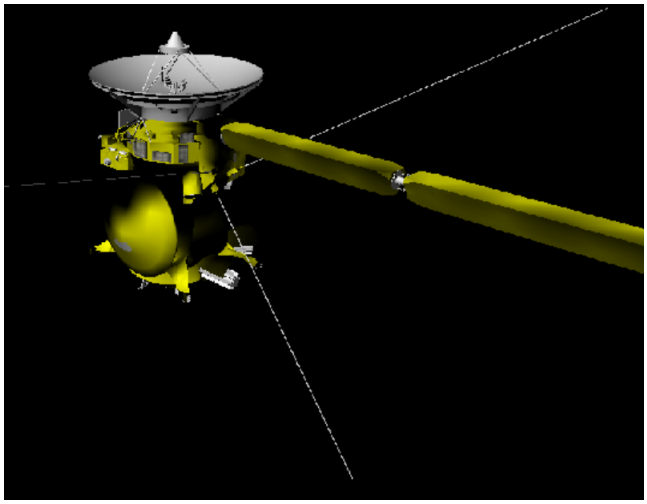


Figure 8: Simulated view of the Cassini spacecraft.

given by:

$$\boldsymbol{\tau}^B(t) = 10 \cdot \begin{bmatrix} \sin(0.01t) \\ \sin(0.01t + \frac{2\pi}{3}) \\ \sin(0.01t + \frac{4\pi}{3}) \end{bmatrix} \tag{68}$$

Metrics for Analysis of Simulation Results

The simulation results in Figures 6-10 show that the DAAVE-MMEKF algorithm is able to track closely the true angular velocities.

We define $\bar{\omega}_{ek} \triangleq \bar{\boldsymbol{\omega}}_k^T \hat{\bar{\boldsymbol{\omega}}}_k$ as the axis estimated pointing error for the angular velocity vector, and $\bar{e}_{\Omega k} \triangleq \Omega_k - \Omega_{k|k}$ is the AVM estimation error. The error metrics are defined as:

$$\bar{e}_{\bar{\omega}} = \frac{1}{N} \sum \bar{\omega}_{ek}, \tag{69}$$

$$\sigma_{\bar{\omega}} = \frac{1}{N-1} \sum \left(\bar{\omega}_{ek} - \bar{e}_{\bar{\omega}} \right)^2, \tag{70}$$

$$\bar{e}_{\Omega} \triangleq \frac{1}{N} \sum \bar{e}_{\Omega k}, \tag{71}$$

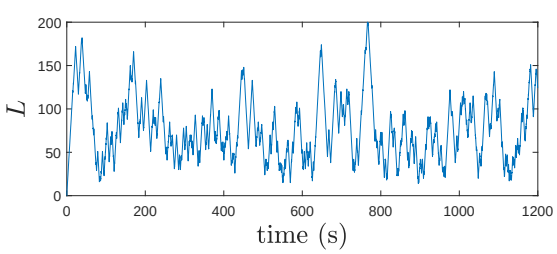
$$\sigma_{\Omega} \triangleq \frac{1}{N-1} \sum \left(\bar{e}_{\Omega k} - \bar{e}_{\Omega} \right)^2. \tag{72}$$

Additionally, we define the mean window length as:

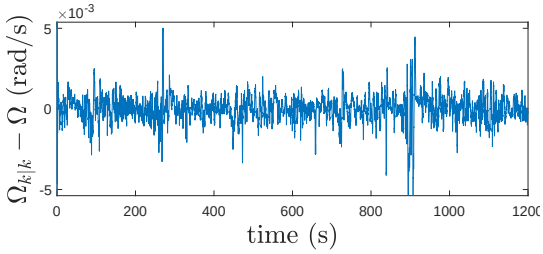
$$\mu_L \triangleq \frac{1}{N} \sum L_k, \tag{73}$$

where L_k is the window length of $\hat{\boldsymbol{Q}}$ at the $k-th$ iteration of the algorithm.

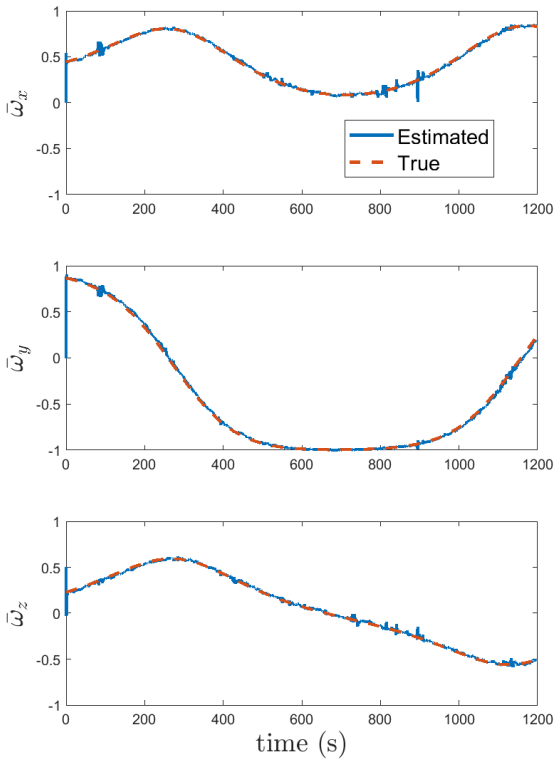
Using the definitions above, Table 1 presents a performance comparison among the various simulation results. All simulation results indicate nearly identical performance, except for the actuated



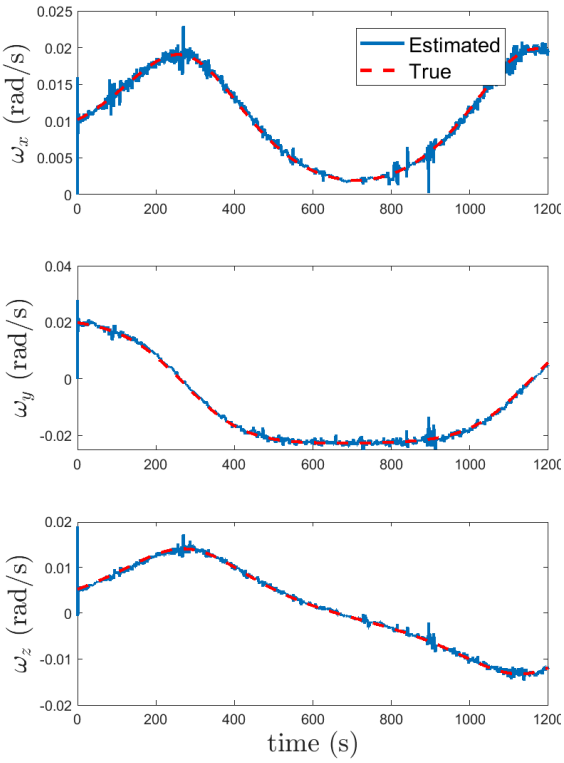
(a) Sliding Window Length



(b) Angular Velocity Magnitude Error

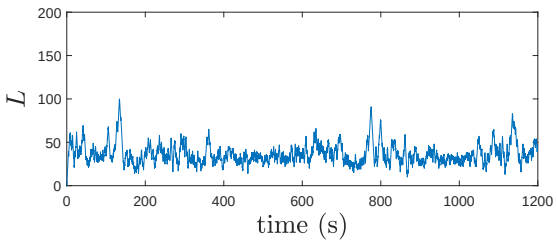


(c) Axis of Rotation

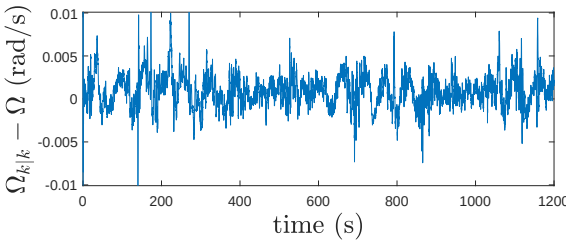


(d) Angular Velocity

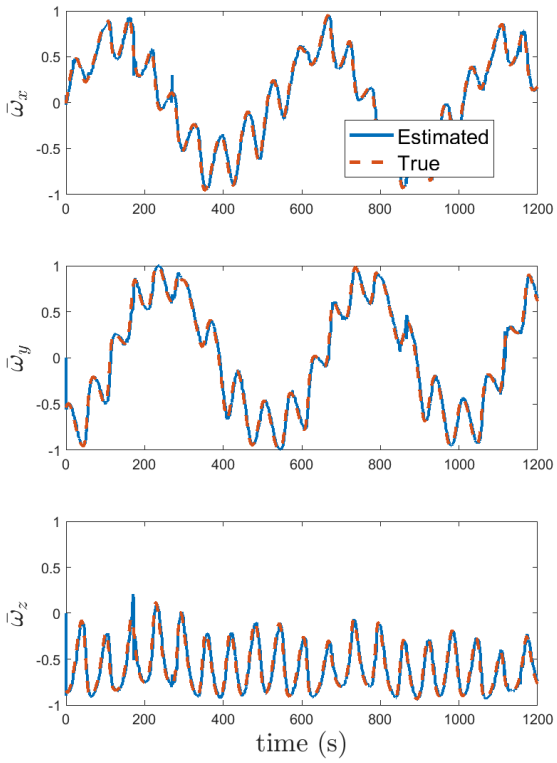
Figure 9: Simulation results for Cassini's tumbling motion assuming initial angular velocity of $\boldsymbol{\omega}(0) = [0.01,\ 0.02,\ 0.005]^T$



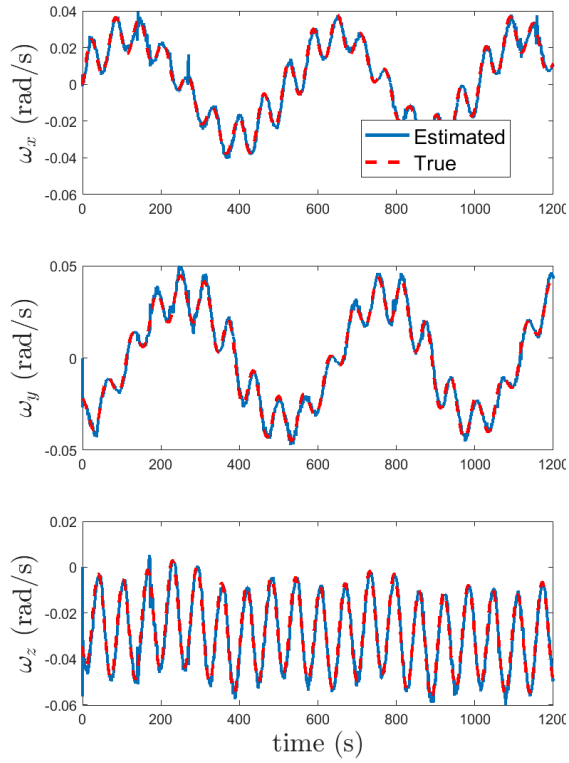
(a) Sliding Window Length



(b) Angular Velocity Magnitude Error



(c) Axis of Rotation



(d) Angular Velocity

Figure 10: Simulation results for Cassini's tumbling perturbed motion assuming initial angular velocity of $\boldsymbol{\omega}(0) = [0.0,\ -0.02,\ -0.035]^T$

	Itokawa	Itokawa Dark	Cassini	Cassini Actuated
μ_L	66.74	67.74	76.16	36.59
$\bar{e}_{\bar{\omega}}$ (deg)	1.68	1.78	1.43	4.50
$\sigma_{\bar{\omega}}$ (deg)	1.64	1.53	1.81	2.86
$\bar{e}_{\Omega}$ (rad/s)	$1.92 \cdot 10^{-4}$	$-2.33 \cdot 10^{-5}$	$2.49 \cdot 10^{-5}$	$9.17 \cdot 10^{-4}$
σ_{Ω} (rad/s)	$1.3 \cdot 10^{-3}$	$8.08 \cdot 10^{-4}$	$1.4 \cdot 10^{-3}$	$2.1 \cdot 10^{-3}$

Table 1: Performance comparison for the multiple simulations.

Cassini case, which performed worse. This is expected, since all other simulations present only tumbling motions, while the last one had the spacecraft being actuated. This led to a quickly changing motion (see Fig. 10), which substantially reduced the average window length L. An immediate consequence of a reduced window length L is higher variance in the axis estimation error $\bar{\omega}_{ek}$. Since the angular velocity axis estimation performs worse, then it is logical that the MMEKF also performs worse in the estimation of the AVM $\Omega_{k|k}$.

It is important to point out that even though the Itokawa simulation with poor light conditions performed nearly at par with the simulation that used fair light conditions, one should not jump to conclusions that light source quality does not play an important role. Whereas the performance deterioration has not been captured in the simulated environment presented in this paper, one would need to further validate these results with carefully conducted experiments using a real camera in a real space mission. An interesting avenue for further work would be to improve the camera model of the simulator to to make it more realistic (i.e., add measurement noise, image blur, radiation noise).

The algorithm presented in this work has two tuning parameters: r_1^* (autocorrelation threshold) and Q_Ω (random walk covariance for the AVM). All our simulations were executed with $r_1^* = \sigma_{r1}$ and $Q_\Omega = 10^{-5}$. Some comments follow on algorithm tuning:

- Regarding the choice for r_1^*, we are satisfied with the given choice, and we believe that this is appropriate for the problem at hand. However, there could be other setting wherein, one could desire to be less conservative by choosing $r_1^* = 2 \cdot \sigma_{r1}$ or $r_1^* = 3 \cdot \sigma_{r1}$. This would imply that that the window length would only decrease when there is more evidence that the motion is not in pure spin. This leads to a higher average window size L, and consequently adds more lag to the estimation of $\bar{\boldsymbol{\omega}}$ (not to mention having larger requirements for the memory buffer). Instead, we prefer to choose $r_1^* = \sigma_{r1}$ because this is a conservative choice, preventing the window from growing too much.

- When choosing Q_Ω, it is important to evaluate how much this parameter affects the performance of the algorithm. Motivated by this consideration, we did not want this parameter to have a different value for each simulation scenario. Rather, we like the choice of the Q_Ω parameter to be as much *hands-off* as possible. In a real situation, we might not know a priori how the target's angular velocity magnitude changes with time. Hence, it is quite satisfying that a single value of Q_Ω delivered satisfactory performance in all the various simulation examples presented in this paper. Still, if one expects the AVM to change more aggressively with, then a higher value for Q_Ω may be indicated for certain applications. Future work might explore adaptiveness of Q_Ω based on whiteness tests of the innovation sequence $\boldsymbol{\nu}_k$ (Eq. 57).

When formulating the MMEKF, we have assumed that the instantaneous spin axis $\bar{\boldsymbol{\omega}}$ is known

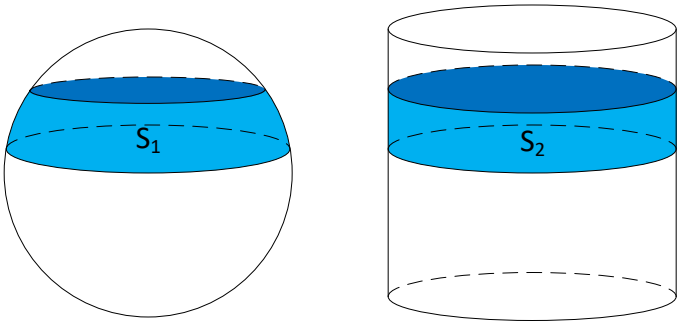


Figure 11: Illustration of Archimedes' Hat-Box Theorem.

and that it is a deterministic quantity. However, the assumption is not true, since $\hat{\tilde{\boldsymbol{\omega}}}$ is estimated from stochastic measurements. An important track for future work would be the statistical analysis of the covariance of $\hat{\tilde{\boldsymbol{\omega}}}$ given by the DAAVE algorithm.

CONCLUSIONS

In this paper, we have introduced and analyzed the performance of the DAAVE+MMEKF algorithm for the angular velocity of a non-cooperative target through visual inspection. The relative pose between the chaser and the target is estimated using ORB-SLAM, and this information is used to get the relative spin axis (through the DAAVE algorithm) and the relative angular velocity magnitude (through the MMEKF algorithm).

Simulation results demonstrate that the algorithm is successful in tracking the true angular velocity of the target without much need for tuning. The same tuning parameters were used throughout all the simulations, showing robustness of the algorithm to different scenarios. Still, tuning of a single parameter Q_Ω might be necessary if the target's angular velocity is expected to change a lot faster than what has been analyzed in our simulation section. A suggestion of future work would be to analyze whiteness properties of the innovation sequence $\boldsymbol{\nu}_k$ to test consistency of MMEKF, allowing to have an adaptive Q_Ω.

A surprising result that we had was that the algorithm did not perform much worse when lighting conditions were not favorable. However, we believe that we need to improve our camera models to make it more realistic in order to have a deeper analysis of the algorithm deterioration in face of poor lighting conditions.

APPENDIX A

In this section we prove that if $\boldsymbol{e} \in \mathbb{S}^2$ is a unit vector uniformly distributed in the 3-D unit sphere, then: $\mathbb{E}[\boldsymbol{e}] = \mathbf{0}$ and $\mathbb{E}[\boldsymbol{e}\boldsymbol{e}^T] = \frac{1}{3}\boldsymbol{I}$.

Assume a unit radius sphere and a cylinder of radius $r = 1$ and height $h = 2$. According with Archimedes' Hat-Box Theorem,[22] if we slice both the cylinder and the sphere at the same height as shown on Fig. 11, then the lateral surface area of the spherical segment (S_1) is equal to the lateral surface area of the cylindrical segment (S_2).

To be specific, the surface area S of the cylinder parametrized with radius $r = 1$ and height $h = 2$ is the same as the unit-radius sphere, i.e, $S = 4\pi$. A commonly used method[23] to generate uniformly distributed samples on a sphere $\boldsymbol{e} \in \mathbb{S}^2$ is to uniformly sample a point in the cylinder

through a height value $z \sim \mathcal{U}[-1,\, 1]$, and an angle value $\phi \sim \mathcal{U}[-\pi,\, \pi]$, and then map it to the sphere through the transformation:

$$\boldsymbol{e} = \begin{bmatrix} \sqrt{1-z^2}\cos(\phi) \\ \sqrt{1-z^2}\sin(\phi) \\ z \end{bmatrix}. \tag{74}$$

The transformation of Eq. 74 guarantees that areas in the cylinder are preserved in the sphere after the projection. Therefore, if a random variable is uniformly distributed in the prior space (cylindrical space), then it should still be uniformly distributed in the posterior space (spherical space).

Denoting $P_z(x)$ and $P_\phi(x)$ as the probability distributions of the scalar variables z and ϕ respectively, then:

$$\mathbb{E}[z] = \int_{-1}^{1} x P_z(x)\, dx = \frac{1}{2}\int_{-1}^{1} x\, dx = \frac{1}{4}x^2\Big|_{-1}^{1} = 0 \tag{75}$$

$$\mathbb{E}[z^2] = \int_{-1}^{1} x^2 P_z(x)\, dx = \frac{1}{2}\int_{-1}^{1} x^2\, dx = \frac{1}{6}x^3\Big|_{-1}^{1} = \frac{1}{3} \tag{76}$$

$$\mathbb{E}[1-z^2] = 1 - \frac{1}{3} = \frac{2}{3} \tag{77}$$

$$\mathbb{E}[\cos\phi] = \int_{-\pi}^{\pi} \cos x P_\phi(x)\, dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} \cos x\, dx = \frac{1}{2\pi}\sin x\Big|_{-\pi}^{\pi} = 0 \tag{78}$$

$$\mathbb{E}[\sin\phi] = \int_{-\pi}^{\pi} \sin x P_\phi(x)\, dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sin x\, dx = -\frac{1}{2\pi}\cos x\Big|_{-\pi}^{\pi} = 0 \tag{79}$$

$$\mathbb{E}[\cos\phi\sin\phi] = \int_{-\pi}^{\pi} \cos x \sin x P_\phi(x)\, dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} \cos x \sin x\, dx = -\frac{1}{4\pi}cos^2 x\Big|_{-\pi}^{\pi} = 0 \tag{80}$$

$$\mathbb{E}[\cos^2\phi] = \int_{-\pi}^{\pi} \cos^2 x P_\phi(x)\, dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} \cos^2 x\, dx = \frac{1}{8\pi}(2x + \sin 2x)\Big|_{-\pi}^{\pi} = \frac{1}{2} \tag{81}$$

$$\mathbb{E}[\sin^2\phi] = \int_{-\pi}^{\pi} \sin^2 x P_\phi(x)\, dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sin^2 x\, dx = \frac{1}{8\pi}(2x - \sin 2x)\Big|_{-\pi}^{\pi} = \frac{1}{2} \tag{82}$$

Therefore, given that z and ϕ are independently distributed, we have that:

$$\mathbb{E}[\boldsymbol{e}] = \begin{bmatrix} \mathbb{E}[\sqrt{1-z^2}\cos(\phi)] \\ \mathbb{E}[\sqrt{1-z^2}\sin(\phi)] \\ \mathbb{E}[z] \end{bmatrix} = \begin{bmatrix} \mathbb{E}[\sqrt{1-z^2}]\mathbb{E}[\cos(\phi)] \\ \mathbb{E}[\sqrt{1-z^2}]\mathbb{E}[\sin(\phi)] \\ \mathbb{E}[z] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{83}$$

Also, we have that:

$$\mathbb{E}[\boldsymbol{e}\boldsymbol{e}^T] = \mathbb{E}\left[\begin{bmatrix} (1-z^2)\cos^2\phi & (1-z^2)\cos\phi\sin\phi & (1-z^2)z\cos\phi \\ (1-z^2)\cos\phi\sin\phi & (1-z^2)\sin^2\phi & (1-z^2)z\sin\phi \\ (1-z^2)z\cos\phi & (1-z^2)z\sin\phi & z^2 \end{bmatrix}\right] \tag{84}$$

$$= \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix} = \frac{1}{3}\boldsymbol{I}. \tag{85}$$

REFERENCES

[1] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[2] M. Dor and P. Tsiotras, "ORB-SLAM Applied to Spacecraft Non-Cooperative Rendezvous," *2018 Space Flight Mechanics Meeting*, 2018, p. 1963.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, Vol. 31, No. 5, 2015, pp. 1147–1163.

[4] B. N. Van Ecpoel and S. Queen, "Flight Results from the HST SM4 Relative Navigation Sensor System," *33rd Annual AAS Guidance And Control Conference, Guidance and Control 2010, Advances in the Astronautical Sciences*, 2010.

[5] C. Olson, R. P. Russell, and S. Bhaskaran, "Spin State Estimation of Tumbling Small Bodies," *The Journal of the Astronautical Sciences*, Vol. 63, No. 2, 2016, pp. 124–157.

[6] S. Salcudean, "A globally convergent angular velocity observer for rigid body motion," *IEEE transactions on Automatic Control*, Vol. 36, No. 12, 1991, pp. 1493–1497.

[7] I. Y. Bar-Itzhack, "Classification of algorithms for angular velocity estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, 2001, pp. 214–218.

[8] Y. Oshman, Franc-para, and o. Dellus, "Spacecraft angular velocity estimation using sequential observations of a single directional vector," *Journal of Spacecraft and Rockets*, Vol. 40, No. 2, 2003, pp. 237–247.

[9] I. Y. Bar-Itzhack, R. R. Harman, and J. K. Thienel, "Rigid body rate inference from attitude variation," *Journal of guidance, control, and dynamics*, Vol. 30, No. 1, 2007, pp. 275–281.

[10] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 5, No. 5, 1982, pp. 417–429.

[11] E. Gai, K. Daly, J. Harrison, and L. Lemos, "Star-sensor-based satellite attitude/attitude rate estimator," *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 5, 1985, pp. 560–565.

[12] F. L. Markley, "Attitude error representations for Kalman filtering," *Journal of guidance, control, and dynamics*, Vol. 26, No. 2, 2003, pp. 311–317.

[13] D. Mortari and M. Akella, "Discrete and Continuous Time Adaptive Angular Velocity Estimators," *Proceedings of the AAS/AIAA Space Flight Mechanics Conference, Williamsburg, VA*, 2015.

[14] M. Almeida and M. Akella, "Discrete Adaptive Angular Velocity Estimation - An Experimental Analysis," *2016 Space Flight Mechanics Meeting*, 2016.

[15] R. D. Cook and S. Weisberg, *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[16] S. Weisberg, *Applied linear regression*, Vol. 528. John Wiley & Sons, 2005.

[17] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[18] M. D. Shuster, "A survey of attitude representations," *Navigation*, Vol. 8, No. 9, 1993, pp. 439–517.

[19] D. Mortari and M. Majji, "Multiplicative measurement model," *The Journal of the Astronautical Sciences*, Vol. 57, No. 1-2, 2009, pp. 47–60.

[20] D. Scheeres, R. Gaskell, S. Abe, O. Barnouin-Jha, T. Hashimoto, J. Kawaguchi, T. Kubota, J. Saito, M. Yoshikawa, N. Hirata, *et al.*, "The actual dynamical environment about Itokawa," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2006, p. 6661.

[21] A. Y. Lee and J. A. Wertz, "In-flight estimation of the Cassini spacecraft's inertia tensor," *Journal of spacecraft and rockets*, Vol. 39, No. 1, 2002, pp. 153–155.

[22] H. Cundy and A. Rollett, "Sphere and cylinderArchimedes theorem," *Mathematical Models*, 1989, pp. 172–173.

[23] E. W. Weisstein, "Sphere Point Picking. From MathWorld–A Wolfram Web Resource.," `http://mathworld.wolfram.com/SpherePointPicking.html`. Accessed: 2018-08-08.