Autonomous Vehicle Simulation (AVS) Laboratory,
University of Colorado

Basilisk Technical Memorandum

Document ID: Basilisk-attTrackingError

ATTITUDE TRACKING ERROR

Prepared by	H. Schaub

Status: Reviewed

Scope/Contents

This module is intended to be the last module in the guidance module chain. It's input is the reference motion message generated by a prior module. It's output is at the guidance attitude tracking errors relative to a moving reference frame. This module applies the body to corrected body attitude correction.

Rev	Change Description	By	Date
1.0	First Verision	H. Schaub	2016-01-15
1.1	Revision and reformating	T. Teil	2019-02-20

Contents

---

1 Model Description

1.1 Introduction

This technical note outlines how the attitude tracking errors are evaluated relative to a given reference frame. The reference frame from the chain of guidance modules is called $\mathcal{R}_0$, while the body corrected reference frame orientation is $\mathcal{R}$.

1.2 Reference Frame Definitions

Let the primary body-fixed coordinate frame be $\mathcal{B} : \{\hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$. However, instead of aligning this frame with a reference, a corrected body frame $\mathcal{B}_c$ is to be aligned with a reference frame. Let the uncorrected reference orientation be given by $\mathcal{R}_0$. Thus, the guidance goal is to drive $\mathcal{B}_c \to \mathcal{R}_0$, which yields

$$[R_0N] = [B_cB][BN] \tag{1}$$

where $\mathcal{N}$ is an inertial reference frame. Rearranging this relationship, with perfect attitude tracking the inertial body frame orientation should be

$$[BN] = [B_cB]^T[R_0N] = [RN] \tag{2}$$

where $\mathcal{R}$ is a corrected reference frame. Note that $[B_cB] = [R_0R]$. Thus, the corrected reference orientation is computed using

$$[RN] = [R_0R]^T[R_0N] \tag{3}$$

where the body-frame correction is subtracted from the original reference orientation.

The benefit of of driving $\mathcal{B} \to \mathcal{R}$ instead of $\mathcal{B}_c \to \mathcal{R}_0$ is that the body frame, along with the many device position and orientation vectors expressed in body-frame components, don't have to be rotated for each control evaluation. In simple terms, if the corrected body frame is a 60°rotation from the body frame, then the 60°is subtracted from the original reference orientation. This allows all body inertia tensor and reaction wheel heading vector descriptions to remain in the primary body frame $\mathcal{B}$.

Assume the initial uncorrected reference frame $\mathcal{R}_0$ is given through the MRP set $\boldsymbol{\sigma}_{R_0/N}$

$$[R_0 N(\boldsymbol{\sigma}_{R_0/N})] \tag{4}$$

The relative orientation of the corrected body frame relative to the primary body frame is a constant MRP set

$$[B_c B(\boldsymbol{\sigma}_{B_c/B})] = [R_0 R(\boldsymbol{\sigma}_{R_0/R})] \tag{5}$$

To apply this correction to the original reference frame, using the Direction Cosine Matrix (DCM) description, this is determined through

$$[RN(\boldsymbol{\sigma}_{R/N})] = [R_0 R(\boldsymbol{\sigma}_{R_0/R})]^T [R_0 N(\boldsymbol{\sigma}_{R_0/N})] = [R_0 R(-\boldsymbol{\sigma}_{R_0/R})][R_0 N(\boldsymbol{\sigma}_{R_0/N})] \tag{6}$$

where the convenient MRP identity

$$[R_0 R(\boldsymbol{\sigma}_{R_0/R})]^T = [R_0 R(-\boldsymbol{\sigma}_{R_0/R})] \tag{7}$$

Note the following MRP addition property developed in Reference **?**. If

$$[BN(\boldsymbol{\sigma})] = [FB(\boldsymbol{\sigma}'')][BN(\boldsymbol{\sigma}')] \tag{8}$$

then

$$\boldsymbol{\sigma} = \frac{(1 - |\boldsymbol{\sigma}'|^2)\boldsymbol{\sigma}'' + (1 - |\boldsymbol{\sigma}''|^2)\boldsymbol{\sigma}' - 2\boldsymbol{\sigma}'' \times \boldsymbol{\sigma}'}{1 + |\boldsymbol{\sigma}'|^2 |\boldsymbol{\sigma}''|^2 - 2\boldsymbol{\sigma}' \cdot \boldsymbol{\sigma}''} \tag{9}$$

In the RigidBodyKinematics software library of Reference **?**, this MRP evaluation is achieved with

$$\boldsymbol{\sigma} = \texttt{addMRP}(\boldsymbol{\sigma}', \boldsymbol{\sigma}'')$$

Thus, to properly apply the body frame orientation correction to the original reference frame, this function should be used with

$$\boldsymbol{\sigma}_{R/N} = \texttt{addMRP}(\boldsymbol{\sigma}_{R_0/N}, -\boldsymbol{\sigma}_{R_0/R})$$

The attitude tracking error of $\mathcal{B}$ relative to $\mathcal{R}$ is

$$\boldsymbol{\sigma}_{B/R} = \texttt{subMRP}(\boldsymbol{\sigma}_{B/N}, -\boldsymbol{\sigma}_{R/N})$$

1.3 Reference Frame Angular Velocity Vector

The angular velocity of the original reference frame $\mathcal{R}_0$ is

$$\boldsymbol{\omega}_{R_0/N} \tag{10}$$

The angular velocity tracking error is defined as

$$\delta\boldsymbol{\omega} = \boldsymbol{\omega}_{B/N} - \boldsymbol{\omega}_{R/N} \tag{11}$$

The correct reference frame angular velocity is

$$\boldsymbol{\omega}_{R/N} = \boldsymbol{\omega}_{R/R_0} + \boldsymbol{\omega}_{R_0/N} = \boldsymbol{\omega}_{R_0/N} \tag{12}$$

because the body frame correction $[B_c B] = [R_0 R]$ is a constant angular offset.

The required inertial reference frame rate vector, in body frame components, is then given by

$${}^{\mathcal{B}}\boldsymbol{\omega}_{R/N} = [BN]^{\mathcal{N}}\boldsymbol{\omega}_{R/N} \tag{13}$$

1.4 Reference Frame Angular Acceleration Vector

With $\dot{\boldsymbol{\omega}}_{R/N}$ given in the inertial frame, in the body frame this vector is expressed as

$$^{\mathcal{B}}\dot{\boldsymbol{\omega}}_{R/N} = [BN]^{\mathcal{N}}\dot{\boldsymbol{\omega}}_{R/N} \tag{14}$$

1.5 Angular Velocity Tracking Error

Finally, the angular velocity tracking error is expressed in body frame components as

$$^{\mathcal{B}}\delta\boldsymbol{\omega} = {}^{\mathcal{B}}\boldsymbol{\omega}_{B/R} = {}^{\mathcal{B}}\boldsymbol{\omega}_{B/N} - {}^{\mathcal{B}}\boldsymbol{\omega}_{R/N} \tag{15}$$

2 Module Functions

The only specific function to this module is the error computation function. It's only goal is to remove the computation from the update function itself.

- **computeAttitudeError**: This function calculates computes the attitude error between the reference and the spacecraft attitude.

3 Module Assumptions and Limitations

No assumptions or limitations are made specifically in this module. It simply uses the input messages to output an error between two different attitudes and rates.

4 Test Description and Success Criteria

The unit test instantiates the module and writes out a reference attitude message as well as a spacecraft navigation message. It then compares the module outputs to expected results.

5 Test Parameters

In order to test the proper implementation of this module, the unit test verify that the module output guidance message vectors match expected values.

Table 2: Error tolerance for each test.

Output Value Tested	Tolerated Error
σ_{BR}	10^{-12}
$^{\mathcal{B}}\omega_{BR}$	10^{-12}
$^{\mathcal{B}}\omega_{RN}$	10^{-12}
$^{\mathcal{B}}\delta\omega_{RN}$	10^{-12}

The error tolerances are given in Table 2, while initial conditions used in this test are as follows:

Table 3: Initial conditions

Navigation Information	Value
σ_{BN}	$[0.25 \quad -0.45 \quad 0.75]^T$
$^{\mathcal{B}}\omega_{BN}$	$[0.25 \quad -0.45 \quad 0.75]^T$
Reference Information	**Value**
σ_{RN}	$[0.35 \quad -0.25 \quad 0.15]^T$
$^{\mathcal{N}}\omega_{RN}$	$[0.018 \quad -0.032 \quad 0.015]^T$
$\delta^{\mathcal{N}}\omega_{RN}$	$[0.048 \quad -0.022 \quad 0.025]^T$

Finally the precomputed expected values for the test are give in the following table:

Table 4: Precomputed Expected values

Output Value Tested	Tolerated Error
σ_{BR}	$[0.1836841481753408 \quad -0.0974447769418166 \quad -0.09896069560518146]^T$
${}^{\mathcal{B}}\omega_{BR}$	$[-0.01181207648013235 \quad -0.008916032420030655 \quad -0.0344122606253076]^T$
${}^{\mathcal{B}}\omega_{RN}$	$[-0.003187923519867655 \quad -0.003083967579969345 \quad 0.0394122606253076]^T$
${}^{\mathcal{B}}\delta\omega_{RN}$	$[-0.02388623421245188 \quad -0.02835600277714878 \quad 0.04514847640452802]^T$

6 Test Results

All of the tests passed:

Table 5: Test results

Check	Pass/Fail
1	PASSED
2	PASSED
3	PASSED
4	PASSED

7 User Guide

The user only needs to setup the module and link the proper message names:

- Build the C-struct:
  `moduleConfig = attTrackingError.attTrackingErrorConfig()`
- Wrap the module:
  ```
  moduleWrap = alg_contain.AlgContain(moduleConfig,
  attTrackingError.Update_attTrackingError,
  attTrackingError.SelfInit_attTrackingError,
  attTrackingError.CrossInit_attTrackingError)
  moduleWrap.ModelTag = "attTrackingError"
  ```
- Add the module to the task:
  `unitTestSim.AddModelToTask(unitTaskName, moduleWrap, moduleConfig)`
- Set the R0R vector:
  `moduleConfig.sigma_R0R = [0.01, 0.05, -0.55]`