Autonomous Vehicle Simulation (AVS) Laboratory

Basilisk Technical Memorandum

Document ID: Basilisk-rwMotorTorque

ALGORITHMS TO MAP DESIRED TORQUE VECTOR ONTO A SET OF REACTION WHEEL MOTOR TORQUES

Prepared by	H. Schaub

Status: Draft
Scope/Contents
This module take $\boldsymbol{L}_r$ as the input, and maps it onto a set of reaction wheel motor torques. The module allows for a 3-DOF torque vector to be controlled, or only a sub-set along 1 or 2 base vectors.

Rev:	**Change Description**	**By**
v0.1	Initial Draft of this documentation	H. Schaub

Contents

1 Introduction

This technical note describes a general algorithm that maps a desired ADCS external control torque $\boldsymbol{L}_r$ onto N Reaction Wheel (RW) motor torque commands $\boldsymbol{u}_s$. Let $\hat{\boldsymbol{c}}_j$ be the axis about which the thrusters are to produce the desired torque. The module can accept up to 3 orthogonal control axis $\hat{\boldsymbol{c}}_j$, where M is the number of axis to be controlled. The vectors are packing into an $M \times 3$ matrix

$$[C] = \begin{bmatrix} \hat{\boldsymbol{c}}_1^T \\ \vdots \end{bmatrix} \tag{1}$$

The reaction wheel spin axis are denoted through the unit direction vectors $\hat{\boldsymbol{g}}_{s_j}$. They are packed into a $3 \times N$ projection matrix $[G_s]$ through

$$[G_s] = \begin{bmatrix} \hat{\boldsymbol{g}}_{s_1} & \cdots & \hat{\boldsymbol{g}}_{s_N} \end{bmatrix} \tag{2}$$

2 Control Torque Mapping

The ADCS control condition is written such that

$$[G_s]\boldsymbol{u}_s = \boldsymbol{L}_r \tag{3}$$

where $\boldsymbol{L}_r$ is the 3-dimensional attitude control vector derived from a particular control solution. To limit the dimensionality of the control solution, and possibly only control a sub-set of axes with the RWs, the $[C]$ matrix is used:

$$[C][G_s]\boldsymbol{u}_s = [C]\boldsymbol{L}_r \tag{4}$$

A common solution to the RW motor torque mapping is to find the set of $\boldsymbol{u}_s$ such that they employ the smallest set of motor effort. This is achieved using a minimum norm inverse of Eq. (4).

$$\boldsymbol{u}_s = [G_s]^T[C]^T\left([C][G_s][G_s]^T[C]^T\right)^{-1}[C]\boldsymbol{L}_r \tag{5}$$

Note that this requires a matrix inverse of dimension M. If the RWs are used for full 3D attitude control, then this is a 3×3 matrix inverse. The numerical implementation of this is still pretty fast as an analytical solution to this inverse is feasible.

3 Module Parameters

3.1 $[C]$ matrix

The module requires control control axis matrix $[C]$ to be defined. Up to 3 orthogonal control axes can be selected. Not that in python the matrix is given in a 1D form by defining `controlAxes_B`. Thus, the $\hat{\boldsymbol{c}}_j$ axes are concatenated to produce the input matrix $[C]$.