



**Autonomous Vehicle Simulation (AVS) Laboratory,  
University of Colorado**

**Basilisk Technical Memorandum**  
Document ID: Basilisk-thrMomentumDumping

**THRUSTING FIRING MODULE TO CYCLE THROUGH THE RW MOMENTUM  
DUMPING**

Prepared by	H. Schaub
-------------	-----------

<b>Status:</b> initial document draft
<b>Scope/Contents</b>
<p>This module reads in the desired impulse that each thruster must produce to create inertial momentum change to despin the RWs. The output of the module is a setup of thruster firing times. Each thruster can only fire for a maximum time that matches a single control period. After this the thrusters are off for an integer number of control periods to let the RW re-stabilize the attitude about an inertial pointing scenario.</p>

Rev:	Change Description	By
Draft	Initial document creation	H. Schaub
1.0	Document update after code review	H. Schaub

## Contents

<b>1</b>	<b>Module Description</b>	<b>1</b>
1.1	Overall RW Momentum Management Outline . . . . .	1
1.2	Momentum Management Algorithm . . . . .	2
1.3	Module Messages . . . . .	3
1.4	Reset () Method . . . . .	3
<b>2</b>	<b>Module Functions</b>	<b>3</b>
<b>3</b>	<b>Module Assumptions and Limitations</b>	<b>3</b>
<b>4</b>	<b>Test Description and Success Criteria</b>	<b>4</b>
<b>5</b>	<b>Test Parameters</b>	<b>4</b>
<b>6</b>	<b>Test Results</b>	<b>4</b>
<b>7</b>	<b>User Guide</b>	<b>4</b>

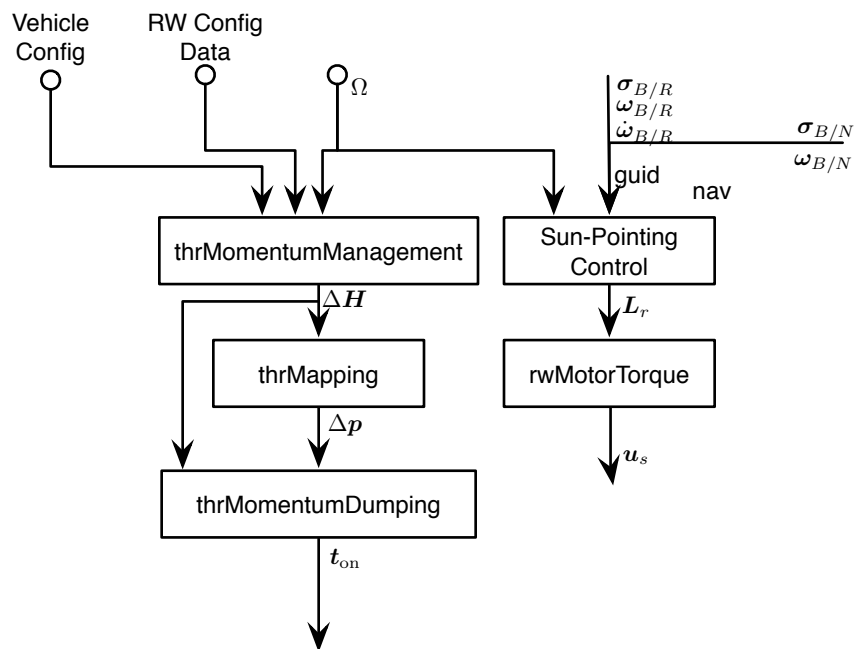
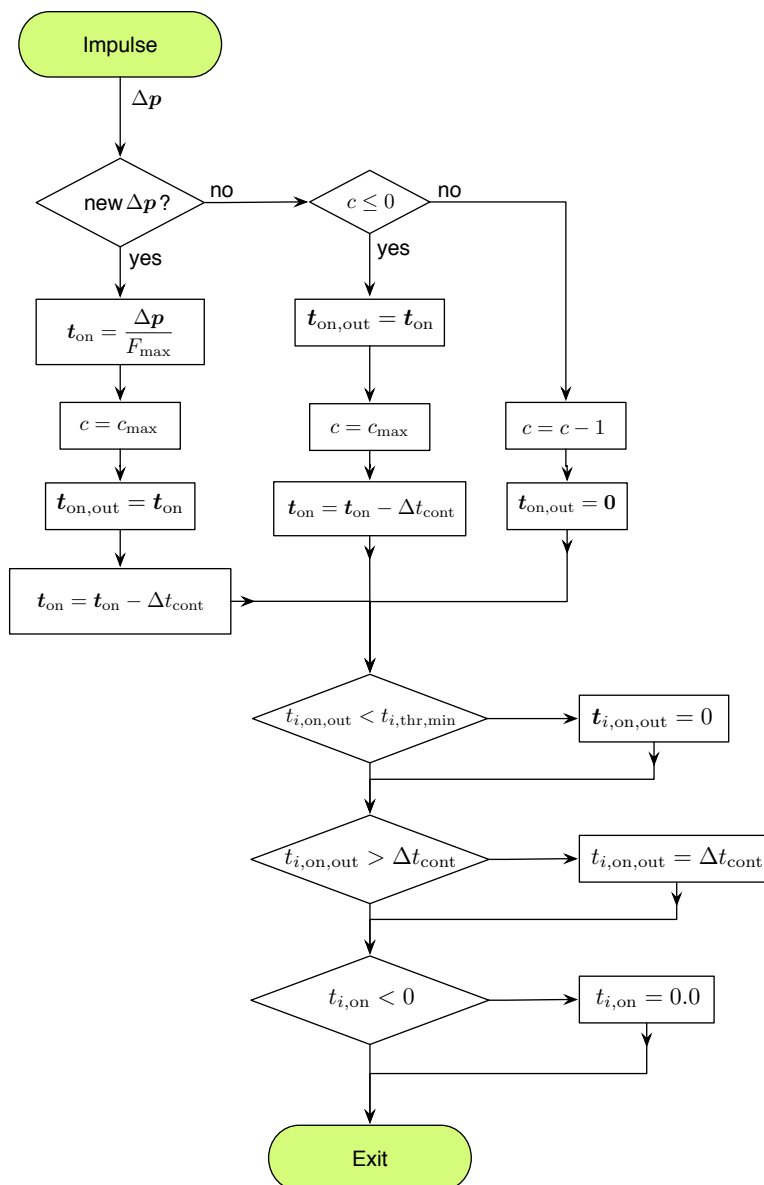


Fig. 1: Overview of the Modules Used to Perform Reaction Wheel Angular Momentum Dumping.

## 1 Module Description

### 1.1 Overall RW Momentum Management Outline

To manage the Reaction Wheel (RW) angular momentum build-up over time, a thruster-based momentum dumping strategy is used. Assume the spacecraft contains  $N_{RW}$  RWs, and  $M_{thr}$  thrusters.



**Fig. 2:** Overview of the Reaction Wheel Angular Momentum Dumping Module Logic.

Figure 1 illustrates how the momentum dumping will occur simultaneously with an inertial pointing control solution. The output of `thrMomentumManagement` module is a  $\Delta \mathbf{H}$  vector. This output is then mapped into a thruster impulse request using the `thrForceMapping` module. Note that this latter module is designed to map a control torque vector into thruster forces. If the input torque and output force sets are multiplied by time, the same module also functions to map a desired angular momentum changes vector  $\Delta \mathbf{H}$  into a set of thruster impulse requests. The final module `thrMomentumDumping` in the series takes the thruster impulse requests and determines a thruster firing sequence to achieve this desired momentum change. The spacecraft attitude is held constant by simultaneously having a RW control module holding an inertial attitude. The process of holding the desired attitude leads to the RWs despinning to the desired level due the external thruster disturbances.

## 1.2 Momentum Management Algorithm

The following module algorithm logic is presented as a flow-diagram in Figure 2. This modules receives the vector of thruster impulses  $\Delta \mathbf{p}$  from a thruster mapping module. It first checks to see if this  $\Delta \mathbf{p}$  is

new by comparing the time tag of the input message with the time tag of the last used thruster impulse message. If yes, the overall thruster on-time values are computed using

$$t_{i,\text{on}} = \frac{\Delta p}{F_{i,\text{max}}} \quad (1)$$

where  $F_{i,\text{max}}$  is the maximum thrust force achievable with the on-off type thruster. These on-times are copied to the output thruster on-times, and the thruster cycling counter is reset.

If the  $\Delta p$  message is not new, then the counter flag is checked. If the counter is strictly positive, the thrusters should be off to allow for the RW to stabilize the orientation. In this case the output thruster times are set to zero, and the counter is decremented, and the remaining required thruster on-times are decremented by the control time period.

If the counter flag reaches zero it is time to fire the required thrusters again to implement the next chunk of the required thruster impulse. The output thrust on times are set to the remaining thruster on times, the counter is reset, and the remaining thruster on times is decremented again by the control period.

Finally, the thruster on times are checked to be greater than the thruster minimum firing time, or the output time is set to zero. Similarly, if the required thruster on time is larger than a control period, the thruster on time is set equal to the control period. This ensures that the thrusters will fire for a single control period duration only before returning to an off-mode. The final check is to see if the thruster on-time remainders are negative, in which case they are reset to zero.

### 1.3 Module Messages

The module has a single output message of type `THRArrayOnTimeCmdMsgPayload` which contains the desired thruster burn times for the next control cycle.

There are 3 required input messages. The message of type `THRArrayConfigMsgPayload` is read in during the `Reset()` method and provides the thruster configuration states. The message of type `THRArrayCmdForceMsgPayload` provides the currently requested thruster impulse values and is read in during the `Update()` method. The message of type `CmdTorqueBodyMsgPayload` is used to pull the time stamp of the currently requested  $\Delta H$  command. It is read in during both reset and update methods.

### 1.4 Reset() Method

The `Reset()` method reads in the thruster message, and then resets dumping counter flag. The stored thruster on time remainders and the request thruster impulse arrays are reset to zero. Further, the  $\Delta H$  request input message is read to see if an old message exists. If yes, then the module variable tracking the time tag of the last impulse request is reset to this current message time tag. This avoids an old message being used after a reset. If the input message has never been written, then the prior message time tag is set to zero.

## 2 Module Functions

- **Keep checking for new thruster impulse requests:** If a new thruster impulse message is read in, then the momentum dumping cycle resumes again.
- **Create an output message with the desired thruster on-times:** The output is used to control the thruster burn duration.

## 3 Module Assumptions and Limitations

The module assumes the spacecraft is holding a steady inertial orientation during the momentum dumping maneuver.

## 4 Test Description and Success Criteria

A module unit test is created which creates the required input messages and runs the module for a single iteration. Two cases are considered where the minimum RWA momentum threshold is exceed or not.

## 5 Test Parameters

The simulation is setup with 8 thrusters with identical  $F_{\max}$  values. The unit test verifies that the module output message vector matches expected values.

**Table 2:** Error tolerance for each test.

Output Value Tested	Tolerated Error
OnTimeRequest	1e-12

## 6 Test Results

All permutations of the test passed:

**Table 3:** Test results

resetCheck	largeMinFireTime	Pass/Fail
False	False	PASSED
False	True	PASSED
True	False	PASSED

## 7 User Guide

The module configurable parameters include:

- `maxCounterValue` Parameter: This parameter dictates the number of control periods where the thrusters cycle off to allow the RWs to re-stabilize the orientation. It must be set to a strictly positive value.
- `thrMinFireTime` Parameter: This parameter determines what the minimum thruster firing time is, provided in units of seconds.