

Jeff's Laboratory

## NM10 - Thrust Vector Assembly Databook

Comments	Revision	Date	Author
Initial Release	A	January 16, 2025	J. Mays

**1. References**

1. <https://hitecrd.com/products/servos/analog/micromini/hs-65mg/product>
2. NM09 – HS-65MG Electro-Mechanical Servo Actuator Databook

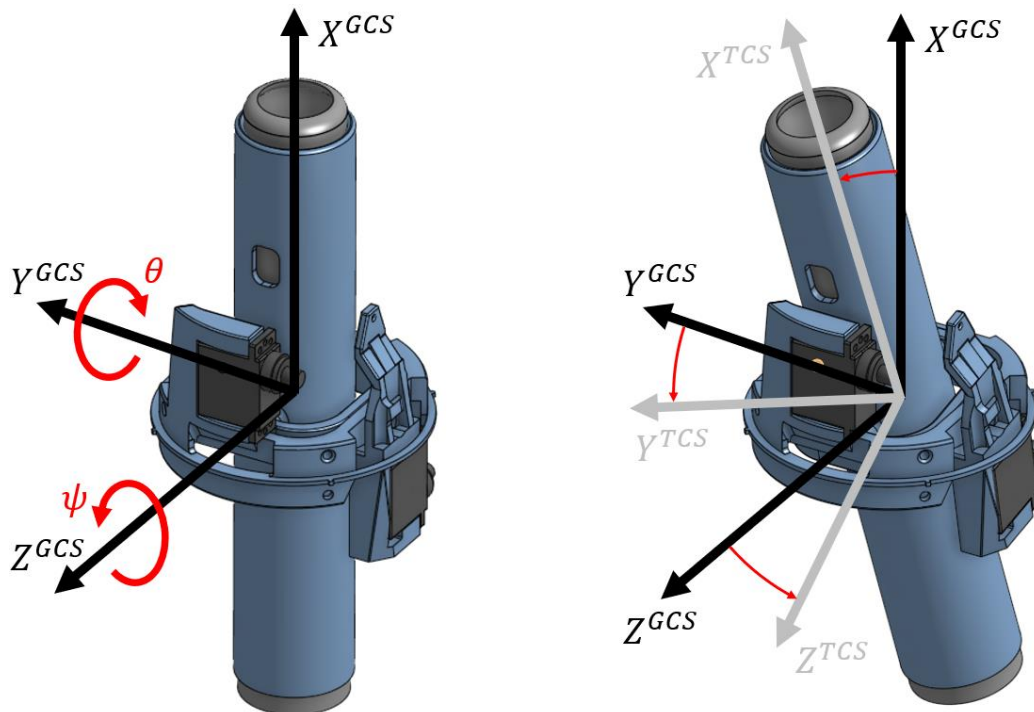
**2. Purpose**

This document describes the thrust vector assembly (TVA) at a high level as it pertains to the New Mays Model Rocket. While it does cover valuable information that may be of interest to the reader, this document is not intended to discuss every aspect of the model. The model was intentionally designed to be simple, only containing enough fidelity for the purposes of Monte-Carlo (MC) simulations of the New Mays vehicle to test and validate the flight software algorithms and verify requirements. This document discusses the respective simulation model written in Matlab/Simulink.

**3. Hardware Context**

**3.1. Thrust Vector Assembly**

Figure 1 illustrates a computer aided design (CAD) rendering of the TVA. The TCA is a 3D printed assembly of three parts: outer gimbal, inner gimbal, and solid rocket motor (SRM) sleeve. Included in the figure is the gimbal coordinate system (GCS) and TVA coordinate system (TCS). The GCS is attached to the outer gimbal and conveniently aligned with the vehicle fabrication frame (FAB). The TCS moves with the SRM sleeve but is nominally aligned with the GCS. The outer gimbal contains the fixed attachment points that fix the assembly into the vehicle. The outer gimbal contains an electro-mechanical servo actuator (servo) to rotate the inner gimbal along the  $Y^{GCS}$  axis of rotation, or  $\theta$ . The inner gimbal contains a separate servo that can rotate the SRM sleeve along the  $Z^{TCA}$  axis of rotation, or  $\psi$ . When there is thrust produced in the  $X^{TCS}$  direction, these two sequential rotations can be used to generate a set of desired body pitch and yaw torques.



**Figure 1: CAD TVA**

Figure 2 is the system physically built with mounted actuator connections. This picture was taken before installation into the receiving aft-vehicle attachment points.



**Figure 2: Physically Built TCA**

Note that the gimbals are all mechanically separated, meaning they are independent coplanar dual axes. The state of one gimbal does not impose alterations on the other. This means the rotations can be treated individually without knowing the state of the other gimbal. This drastically simplifies the modeling required.

**3.2. HS-65MG Electro-Mechanical Servo Hardware**

The HiTec HS-65MG Electro-Mechanical Servo actuator (servo) is a micro class servo used for various hobbyist and robotic applications. This servo is a pulse width modulated (PWM) closed loop control system. Providing the servo with a particular pulse width will internally correspond to a setpoint for the servo arm (otherwise known as the servo-horn) to track. Therefore, we can generate a PWM given what angle we desire the servo to be at. By having a map of these commands, we can effectively generate a PWM-to-Servo Angle and then a Servo Angle-to-TVA Angle map lookup table. The servo model is documented in [2].

**3.3. Servo to TCA Mapping**

For the Servo to TCA mapping, each axis is treated as an independent axis due to the nature of the gimbal axis design as previously noted. Therefore, we can decouple the rotations into their own reference frames in 2D and later use them sequentially to determine the final TCS state for the 3D rotation.

The outer-gimbal servo is configured such that a positive servo rotation generates a positive  $Y^{GCS}$  rotation (assuming no  $Z^{GCS}$  rotation). Similarly, the inner-gimbal servo is configured such that a positive servo rotation generates a positive  $Z^{GCS}$  rotation (assuming no  $Y^{GCS}$  rotation). In terms of the axes stack, Table 1 defines the nominal generated rotations and realized torques.

**Table 1: TVA Mapping Rotations**

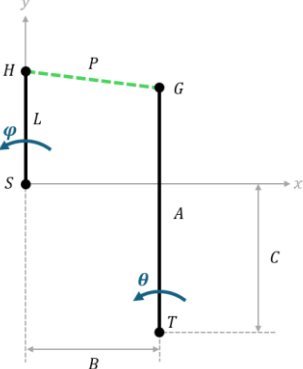
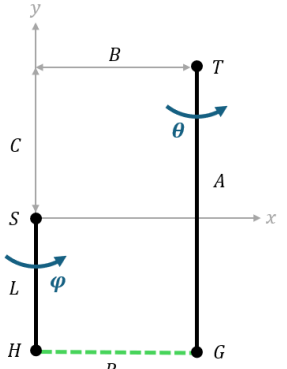
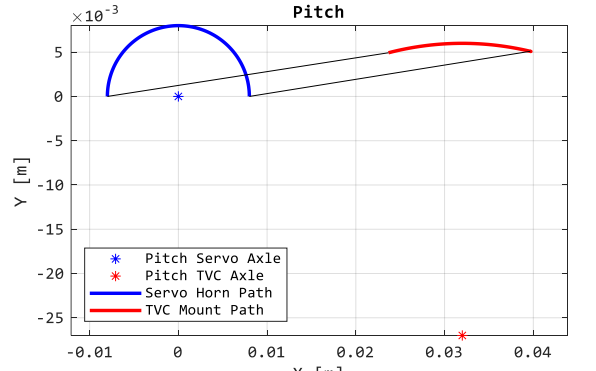
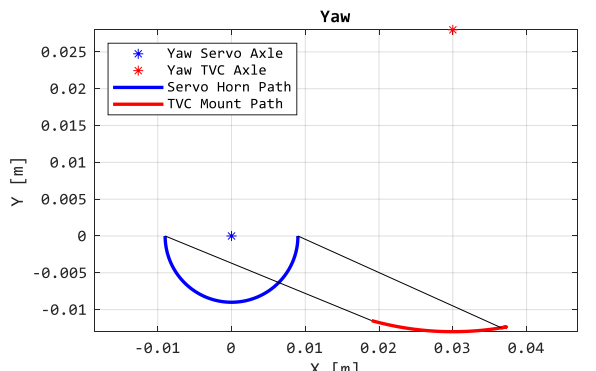
Servo	TVA GCS	FAB Force	FAB Moment
SV-01	+Y	$-F_z$	$-M_y$ (pitch down)
SV-02	+Z	$+F_y$	$-M_z$ (yaw left)

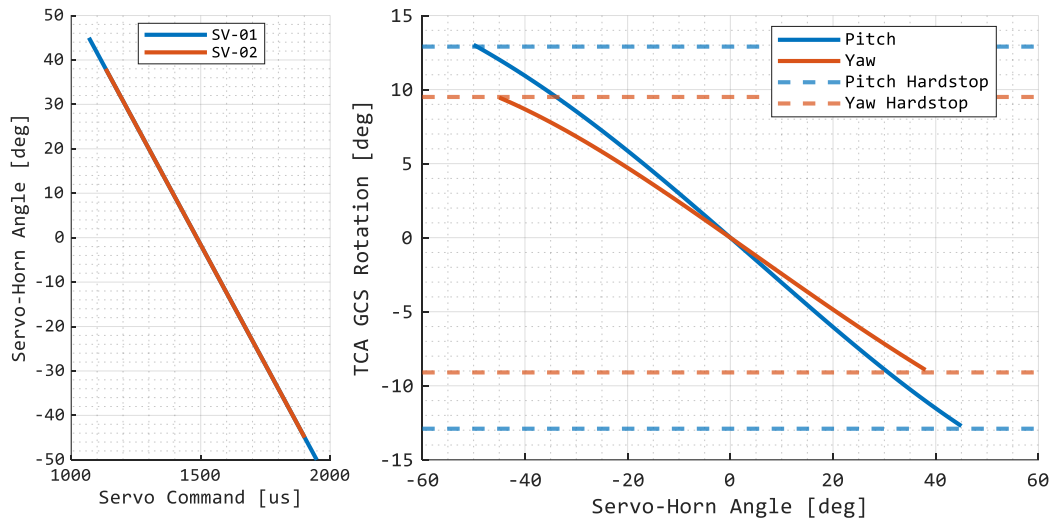
The TVA also contains hardstop limits due to the physical design. We must define these limits since the servo actuators are able to command beyond these hardstop limits, causing possible damage. The hardstop limits were found using the CAD model of the TVA. These limits are defined in the plant model of the TVA, and the

flight software is constrained to never command beyond these limits via requirement FLT-100. Note that these hardstops are not symmetric.

**Error! Reference source not found.** and Figure 3 illustrate the mapping lookup table between servo angles and resultant TVC angle in the pitch and yaw planes. We can set up a simple set of algebraic equations that map the relationship between the servo and TVA angle based on a free-body diagram. Since the linkages are all rigid but allowed to swivel amongst each other on a 2D plane, we can solve these set of equations at various servo angles to get a TVA angle. This is performed on both the pitch and yaw planes, generating a nonlinear lookup table to use in the model. Figure 3 also shows the hardstops as noted in CAD.

**Table 2: TVC Servo to angle mapping free body diagrams and mathematical formulations**

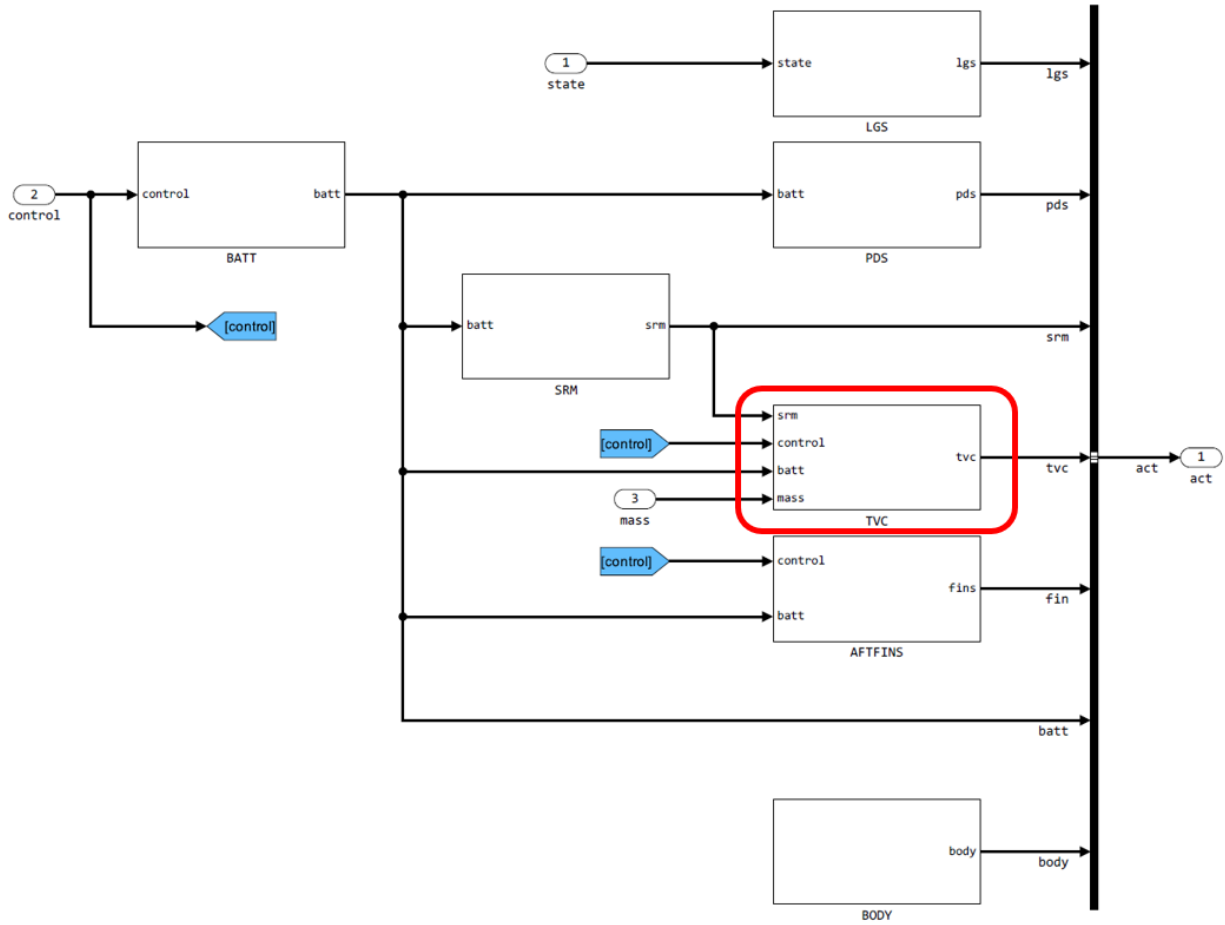
Pitch Plane	Yaw Plane
 $H_x = -L \sin \varphi$ $H_y = L \cos \varphi$ $G_x = -A \sin \theta + B$ $G_y = A \cos \theta - C$ $P = \sqrt{(H_x - G_x)^2 + (H_y - G_y)^2}$	 $H_x = L \sin \varphi$ $H_y = -L \cos \varphi$ $G_x = A \sin \theta + B$ $G_y = -A \cos \theta + C$ $P = \sqrt{(H_x - G_x)^2 + (H_y - G_y)^2}$
	



**Figure 3: TCA Mapping**

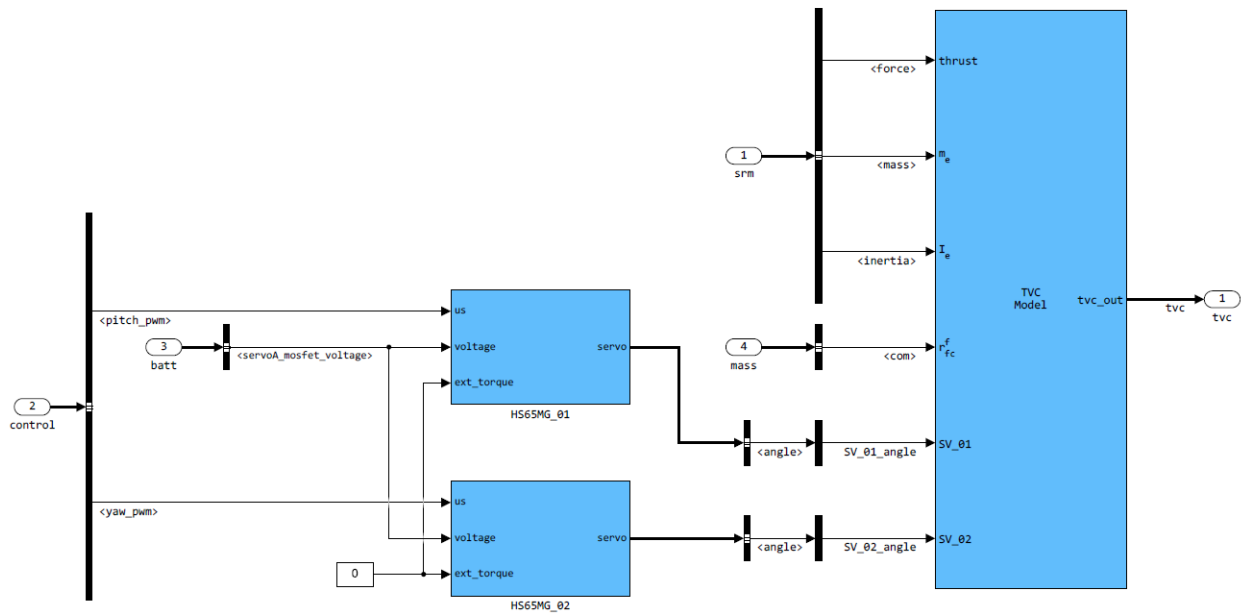
**4. Modeling & Simulation**

The TVA simulation model is implemented in Matlab/Simulink and highlighted in Figure 4 as the TVC model, or “thrust vector control” model. The Simulink diagram shown in Figure 4 is the ACTUATORS block, and it contains all plant actuator models used to simulate New Estes, including other models such as the battery (BATT), solid rocket motor (SRM), and aft fins (AFTFINS).



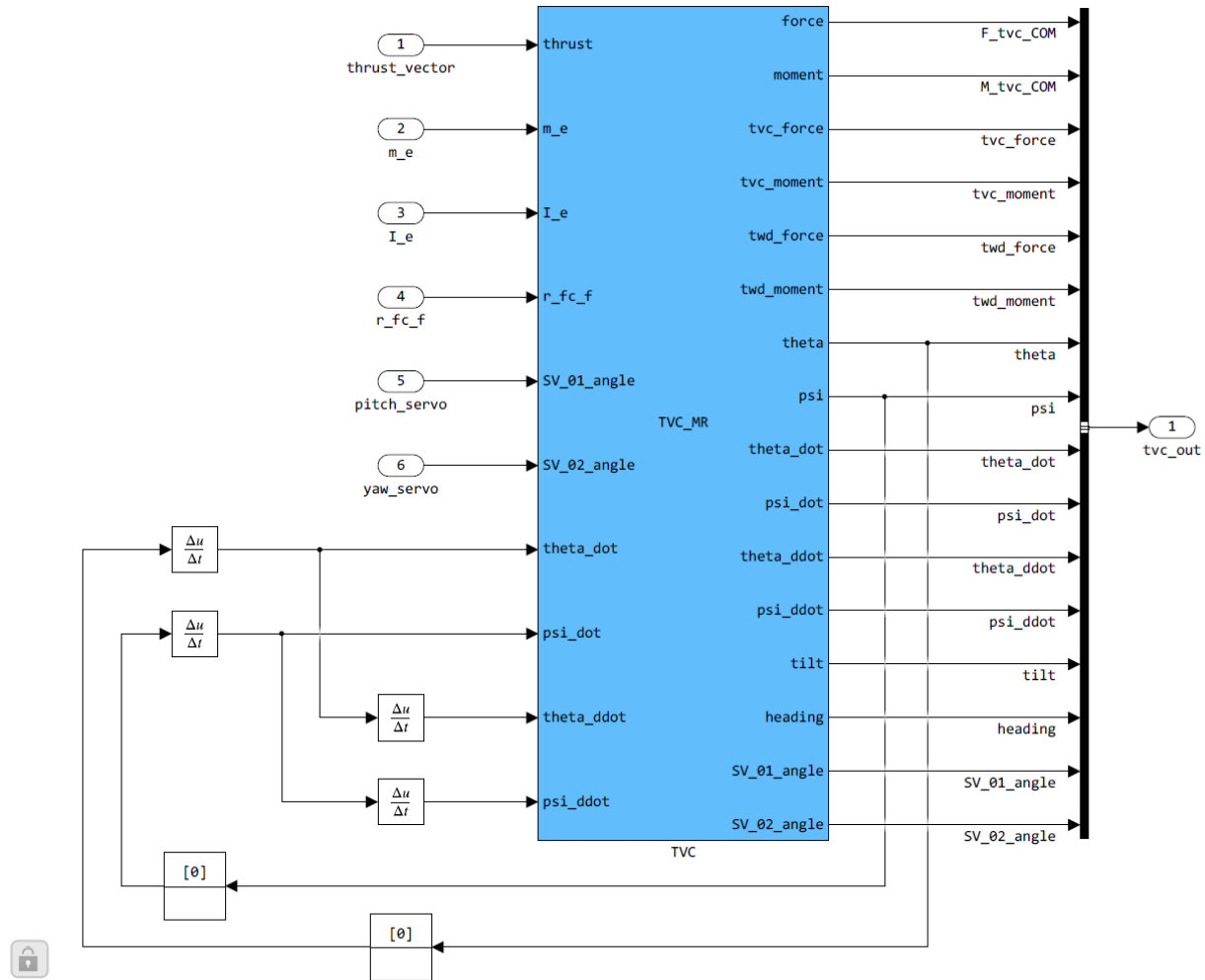
**Figure 4: nm\_sim/ROCKET/MODELS/ACTUATORS TVA Model Reference**

Moving into the TVC Simulink model, we find custom library references for both servo computational models and TVA gimbal mapping, shown in Figure 5. These library references are configured to accept MC tunable and constant parameters used for MC analysis.



**Figure 5: nm\_sim/ROCKET/MODELS/ACTUATORS/TVC**

Moving into the TVC library block, we see the TVC model reference (MR) block, shown in Figure 6. The TVC\_MR block is a `@matlab.system`, which is an object in Simulink that allows Matlab based algorithms to be implemented. This enables the TVC model to be implemented as a Matlab script-based class rather than a collection of many Simulink blocks. This design choice was made since the Home License only allows a certain number of Simulink blocks to be used at any one time. Regardless, constant and tunable MC parameters from the model workspace can be uploaded to this model during the simulation initialization procedure, which alters the parameterized performance based on the MC index randomizer seed.



**Figure 6: TVC Model Reference Library block**

#### 4.1. Gimbal Model

The TVC is an independent coplanar dual axial gimbal system. We can model the final rotation of the thrust vector given  $\theta - \psi$  sequential rotations as a coordinate transformation matrix (CTM). The FAB ( $f$ ) to TCS ( $t$ ) CTM, or  $C_f^t$ , can be described as

$$C_f^t(\theta, \psi) = R_z(\psi)R_y(\theta) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad (1)$$

where  $\theta$  is the local TCS frame pitching rotation and  $\psi$  is the TCS frame yawing rotation.

Given a thrust vector,  $\mathbf{T}^t$ , with its  $x$  component of thrust acting in the direction of the SRM (attached to the TCS frame), we can then use the transpose of this CTM to find the vehicle FAB frame components of thrust.

$$\mathbf{T}^f = C_f^t(\theta, \psi)^T * \mathbf{T}^t \quad (2)$$



Assuming we have positional references for the GCS frame in FAB,  $\mathbf{r}_{fg}^f$ , and the center of mass in FAB,  $\mathbf{r}_{fc}^f$ , we can find the moment arm from the GCS frame gimbal center to the center of mass. Performing the cross product against the thrust in the FAB frame provides us with the moment produced at the vehicle's center of mass.

$$\mathbf{M}^c = (\mathbf{r}_{fg}^f - \mathbf{r}_{fc}^f) \times \mathbf{T}^f \quad (3)$$

This generates the following outcome, which aligns with the servo to TVC mapping as described in Table 1.

**Table 3: TVA Model Resultant Forces and Moments**

Rotation	Body Force Direction	Body Moment
$+\delta_{pitch}$	-Z	-Y (negative pitching)
$+\delta_{yaw}$	+Y	-Z (negative yawing)

#### 4.2. Tail Wags Dog

Tail Wags Dog (TWD) is a conservation of angular momentum problem that is present when an off-axis arm rotates while attached to a freely rotating body. Even though the effects of TWD are negligible for New Estes, we can still derive its dynamics and include them in the model for the sake of enjoyment.

Since the TCS frame is attached to the SRM sleeve, the dynamics can be simply expressed within the TCS frame as the following, with  $e$  representing the center of mass of the engine/SRM,  $g$  representing the center of the GCS frame,  $f$  representing the FAB frame, and  $t$  representing the TCS frame.

$$\boldsymbol{\omega}_{ge}^t = \begin{bmatrix} 0 \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}, \quad \boldsymbol{\alpha}_{ge}^t = \begin{bmatrix} 0 \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} \quad (4)$$

We can move those to the body frame by

$$\boldsymbol{\omega}_{ge}^f = \mathbf{C}_t^f(\theta, \psi) \boldsymbol{\omega}_{ge}^t, \quad \boldsymbol{\alpha}_{ge}^f = \mathbf{C}_t^f(\theta, \psi) \boldsymbol{\alpha}_{ge}^t \quad (5)$$

Angular momentum can be expressed as

$$\mathbf{h} = J\boldsymbol{\omega} + \mathbf{h}_e$$

Taking the derivative in the rotating center of mass, we get

$$\sum \mathbf{M} = \dot{\mathbf{h}} = J\dot{\boldsymbol{\omega}} + J\boldsymbol{\omega} + \dot{\mathbf{h}}_e + \boldsymbol{\omega} \times (J\boldsymbol{\omega} + \mathbf{h}_e) \quad (6)$$

Assuming rigid body,  $\dot{J} = 0$  and  $\dot{\mathbf{h}}_e = 0$ . Since we are accelerating the engine from rest, we have the rotational dynamics

$$\sum \mathbf{M} = J\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times J\boldsymbol{\omega} \quad (7)$$

$$\dot{\boldsymbol{\omega}} = J^{-1}(\sum \mathbf{M} - \boldsymbol{\omega} \times J\boldsymbol{\omega}) \quad (8)$$

Assuming no external torques for the purposes of the TWD dynamics along with a non-accelerating vehicle (trimmed), this indicates that the total torque on the vehicle is induced by the conservation/exchange of momentum.

$$-\dot{\mathbf{h}}_e = J_e \dot{\boldsymbol{\omega}}_e \quad (9)$$

The force and moment equations for TWD as they are applied to the vehicle are then as follows

$$F^g = -1 * [m_e (\omega_{ge}^g \times (\omega_{ge}^g \times r_{ge}^g)) + m_e (\dot{\omega}_{ge}^g \times r_{ge}^g)] \tag{10}$$

$$M^c = (r_{fg}^f - r_{fc}^f) \times F^g + M^g \tag{11}$$

$$M^g = -1 * [J_e \dot{\omega}_{ge}^g] \tag{12}$$

### 4.3. Monte-Carlo

Using a custom @MonteCarlo class, we can disperse parameters given a pre-defined dispersion and seed. For example, the TVC servo to gimbal mapping is only a best approximation. We can disperse the mapping to account for estimation errors of this mapping. Below is a code snip of the loadTVCPParameters.m script that disperses these TVC parameters.

```
function [tvc_constant, tvc_tunable] = loadTVCPParameters(mc, config)
% Load Thrust Vector Control (TVC) parameters
%
% Author: Jeff Mays

tvc_constant = struct();
tvc_tunable = struct();

% Load in the TVC servo state to gimbal map
tvc_map = readtable("tvc_map.xlsx");

%% Hardstop limits
% Must state the gimbal physical hardstop limits, we will ensure the SW
% never commands beyond these limits to prevent physical damage
tvc_constant.outergimbal.limits = [-12.9 12.9] * C_DEG;
tvc_constant.innergimbal.limits = [-9.1 9.5] * C_DEG;

max_theta = max(tvc_constant.outergimbal.limits);
min_theta = min(tvc_constant.outergimbal.limits);
max_psi = max(tvc_constant.innergimbal.limits);
min_psi = min(tvc_constant.innergimbal.limits);

% Ensure unique lookup table for correct lookups in plant and SW. We must
% have unique values, and we cannot let multiple servo angles equate to the
% same TVA angle.
valid_tvc_pitch_ii = find(tvc_map.PITCH > min_theta/C_DEG,1,'first') : find(tvc_map.PITCH >
max_theta/C_DEG,1,'first');
valid_tvc_yaw_ii = find(tvc_map.YAW > min_psi/C_DEG,1,'first') : find(tvc_map.YAW > max_psi/C_DEG,1,'first')-1;

%% Outer gimbal map (pitch plane)

tvc_tunable.outergimbal.tvc = tvc_map.PITCH(valid_tvc_pitch_ii) * C_DEG;
tvc_tunable.outergimbal.servo = tvc_map.SERVO(valid_tvc_pitch_ii) * C_DEG;
tvc_tunable.outergimbal.pwm = tvc_map.PWM(valid_tvc_pitch_ii);
% figure; plot(tvc_tunable.outergimbal.servo, tvc_tunable.outergimbal.tvc/C_DEG)
% figure; plot(tvc_tunable.outergimbal.pwm, tvc_tunable.outergimbal.tvc/C_DEG)

% Scale
outergimbal_scale = mc.disperse('outergimbal_scale', ...
'Dispersion', 'Normal', ...
'Nominal', 1.0, ...
'StdDev', 0.02/3);

% Bias
outergimbal_bias = mc.disperse('outergimbal_augment_bias', ...
'Dispersion', 'Normal', ...
'Nominal', 0.0, ...
'StdDev', 0.02 * C_DEG);

tvc_tunable.outergimbal.tvc = tvc_tunable.outergimbal.tvc * outergimbal_scale + outergimbal_bias;

%% Inner gimbal map (yaw plane)
tvc_tunable.innergimbal.tvc = tvc_map.YAW(valid_tvc_yaw_ii) * C_DEG;
tvc_tunable.innergimbal.servo = tvc_map.SERVO(valid_tvc_yaw_ii) * C_DEG;
tvc_tunable.innergimbal.pwm = tvc_map.PWM(valid_tvc_yaw_ii);
% figure; plot(tvc_tunable.innergimbal.servo, tvc_tunable.innergimbal.tvc/C_DEG)
% figure; plot(tvc_tunable.outergimbal.pwm, tvc_tunable.innergimbal.tvc/C_DEG)

% Scale
innergimbal_scale = mc.disperse('innergimbal_scale', ...
```

```

'Dispersion', 'Normal', ...
'Nominal',    1.0, ...
'StdDev',     0.02/3);

% Bias
innergimbal_bias = mc.disperse('innergimbal_augment_bias', ...
'Dispersion', 'Normal', ...
'Nominal',    0.0, ...
'StdDev',     0.02 * C_DEG);
tvc_tunable.innergimbal.tvc = tvc_tunable.innergimbal.tvc * innergimbal_scale + innergimbal_bias;

%% Position of the co-planer independent dual axle (g)
tvc_tunable.r_fg_f = [0.25; 0.0; 0.0] * C_M; % Position from FAB (f) to gimbal (g)

%% Position of the engine COM in FAB
r_fe_f = [0.20; 0.0; 0.0] * C_M; % Position from FAB (f) to engine COM (e)
tvc_tunable.r_ge_g = eye(3) * r_fe_f - tvc_tunable.r_fg_f; % No rotation when defined
    
```

### 4.4. Model Validation

The model needs to verify the gimbal model directions produce the expected forces and moments and that the model moves in the correct directions based on the servo angle as well as validate the correct TWD forces and moments are applied. The TVC model validation test harness is shown below.

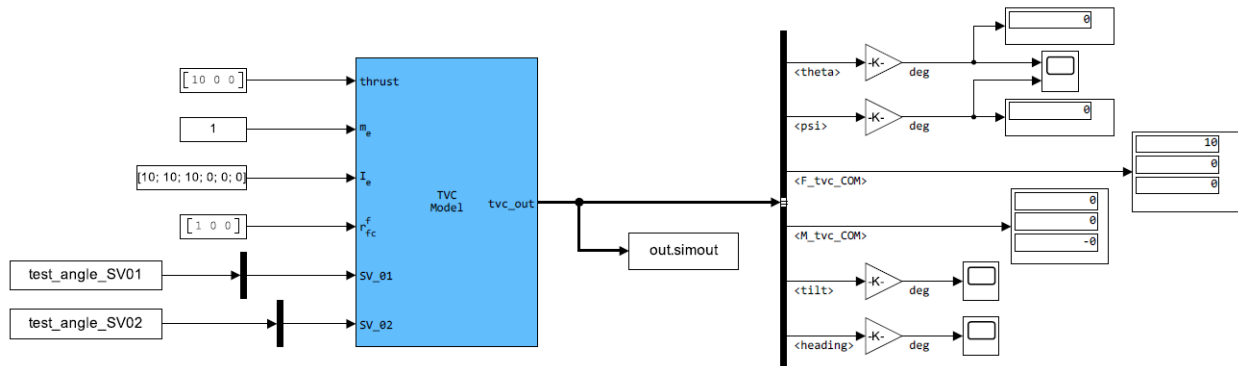


Figure 7: TVC Test Harness

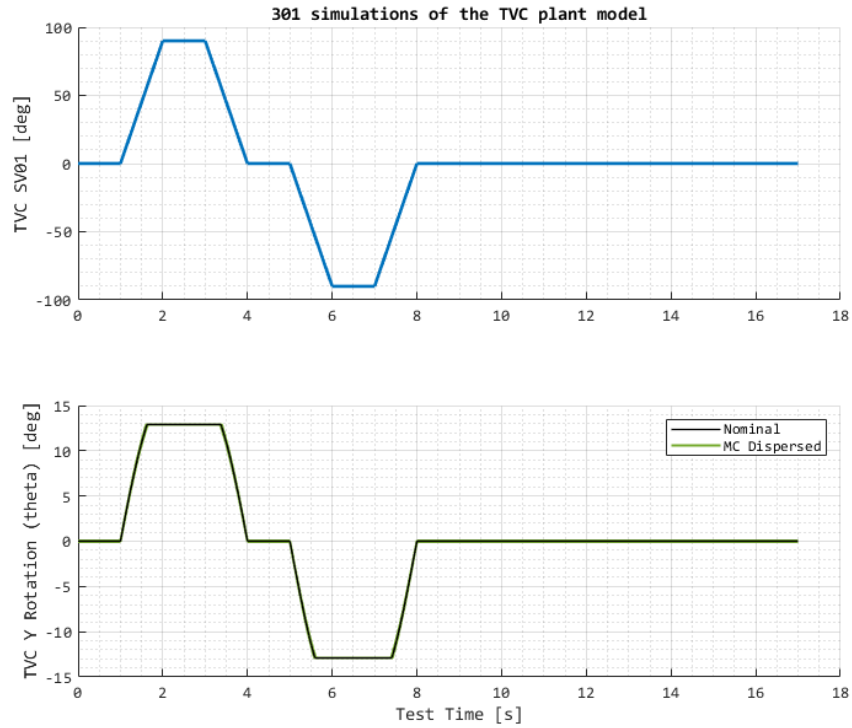
For the sake of not diving into a lengthy validation effort, two tests were decided, providing the developer with sufficient information to indicate correct implementation. Note that a more exhaustive routine would normally be performed on a system such as this. Other tests have been performed during development, it's just that they are not formally documented.

Table 4: Validation Tests

Test ID	Test Description
Limit Test	The limit test is designed to slew each servo and ensure that the TVA does not achieve an angle that is beyond its hard-stops. Post simulation processing will be able to flag when this occurs for requirement FM-100.
Gimbal Test	Servo 1 and 2 input profiles are pushed through the TVC model over time. The model is valid should the output profile and resulting thrust vector be as expected as defined in the previous mapping. This should be true for a given set of MC runs
TWD Test	Servo 1 and 2 input profiles of specific rates and accelerations are pushed through the TVC model over time. The model is valid should the output dynamics be as expected given the TVC motion.

### 4.4.1. Limit Test

The limit test ensures that the TVA limits cannot be exceeded. The following plots illustrate the test and give confidence that the TVA model is working as intended.



**Figure 8: Limit Test - TVC Z Rotation**

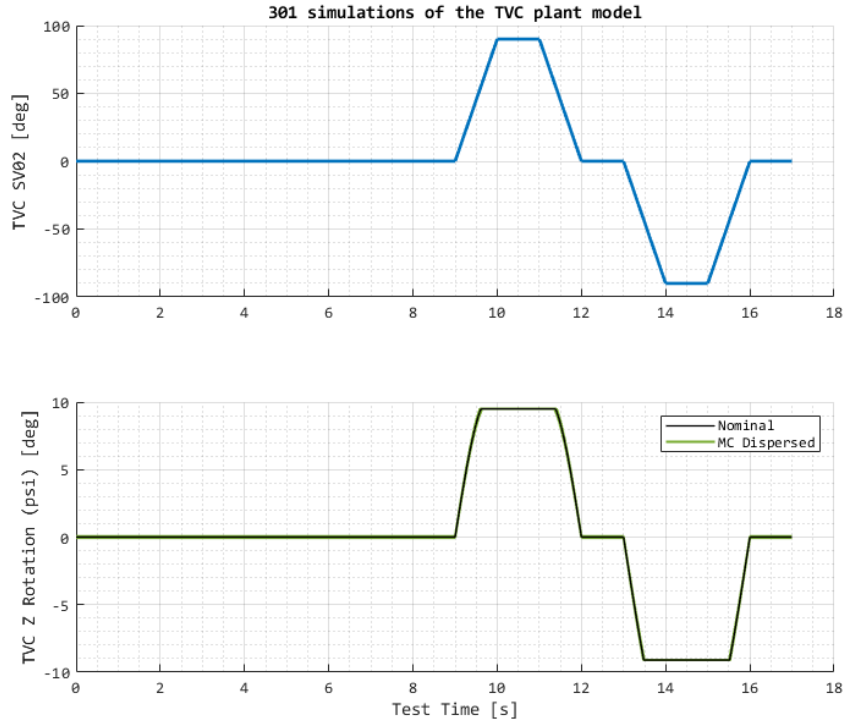


Figure 9: Limit Test - TVC Z Rotation

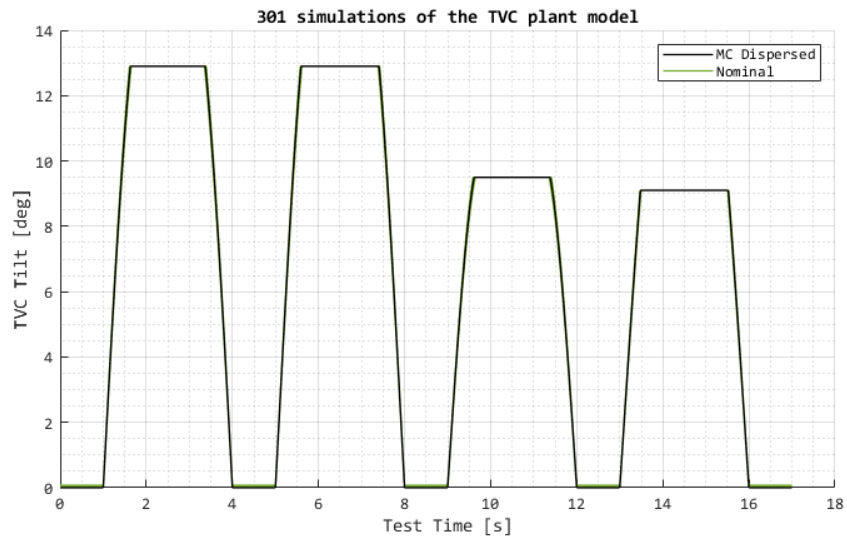
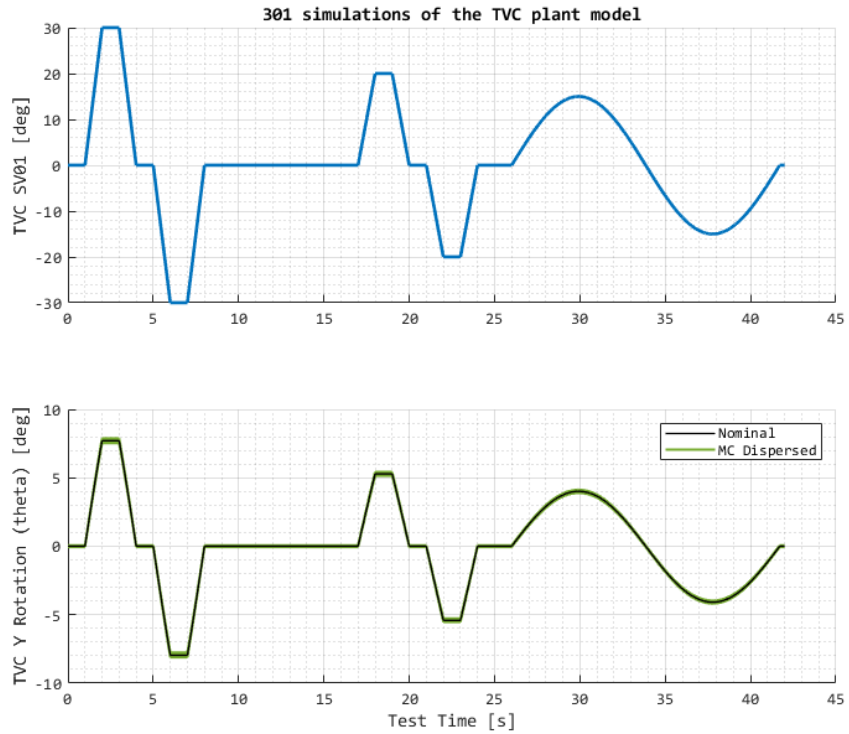


Figure 10: Limit Test - TVC Tilt

### 4.4.2. Gimbal Test

The gimbal test starts with an isolated pivot of the SV01 servo, followed by an isolated pilot of the SV02 servo, and then finishing with a wavelength cycle sinusoidal of each servo at the same time. The following plots illustrate the test and give confidence that the TVA model is working as intended.



**Figure 11: Gimbal Test - TVC Y Rotation**

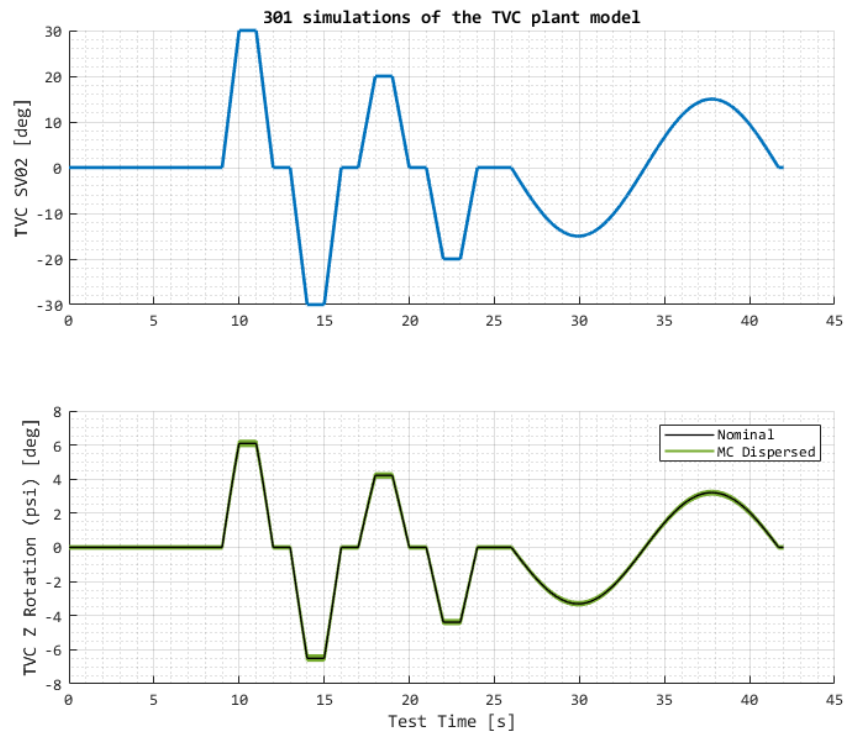


Figure 12: Gimbal Test - TVC Z Rotation

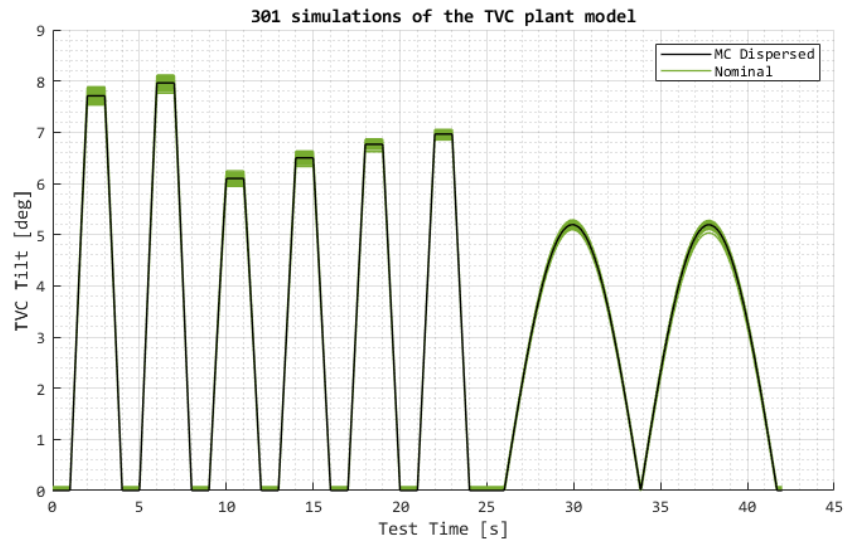


Figure 13: Gimbal Test - TVC Tilt

#### **4.4.3. TWD Test**

Validation not yet completed.