
16.851 - SATELLITE ENGINEERING MEMORANDUM

TO: 16.851 FACULTY
FROM: STUDENTS
SUBJECT: PROBLEM SET #3 SOLUTION
DATE: 10/15/2003

Subject: Space Environment and Attitude Control Systems

Motivation:

The near-Earth space and atmospheric environments strongly influence the performance and lifetime of operational space systems. A study of the interaction between environmental disturbances, mission objectives, and selection and sizing of ACS actuators will help us understand some of the considerations needed when designing spacecraft systems for harsh environments. Our goal is to design a tool that helps size ACS actuators for a satellite mission given specific mission objectives and environmental disturbances. Such a tool will be valuable when sizing ACS components next semester.

Problem Statement:

How does the space environment affect the kind and sizes of ACS actuators? How does the size of the actuators vary given the different kinds of environmental disturbances?

Approach:

We will write a Matlab program to find the kind and sizes of ACS actuators needed for a satellite mission given environmental disturbances and mission objectives. A literature search will allow us to improve on or justify the approximations given in SMAD. The program user will input the values for the mission objectives and environmental disturbances. The program will then calculate the kinds and sizes of ACS actuators needed for the space mission. The results of the type and size of ACS actuators needed will be outputted.

1. Determine type and range of environmental disturbances to include
2. Determine type and range of mission objectives
3. Determine relevant spacecraft characteristics such as mass and moments of inertia
4. Map mission objectives to types of actuators

5. Use values of environmental disturbances and values of mission objectives to determine size and weight of actuators.

Solution:

A. Derivations

There are many factors that must be analyzed when sizing an attitude control system. External torques due to solar radiation, interactions with the Earth's magnetic field, atmospheric drag, and the gravity gradient must be computed. Internal torques or center of gravity changes can be caused by fuel slosh, deployable appendages, gimballed instruments or antennae, and even human movement and cargo relocation in manned space flight. All of these things have the potential to change the attitude and must be accounted for by the attitude control system, since most spacecraft have a desired orientation during at least some part of their mission. For most practical, unmanned, Earth-orbiting spacecraft, the primary perturbations to the vehicle's attitude will be from the external (environmental) effects: gravity gradient, solar radiation pressure, magnetic field interactions, and atmospheric drag.

The first step in sizing an attitude control system is to determine the worst-case disturbance torques that the spacecraft can expect to see, based on the known orbital parameters. This section presents the equations that are used in this analysis to estimate those worst-case torques.

Gravity Gradient

Because of the inverse-square nature of force of gravity, any object with nonzero dimensions orbiting a massive body will be subjected to a "gravity-gradient" torque. In short, the portions of the satellite that are closer to the central body will experience a slightly larger force than those parts farther away. This creates a force imbalance that has a tendency to orient the satellite with its long axis pointing towards the central body. Author C. B. Spence, Jr. derives the gravity gradient torqueⁱ on a body to be:

$$T_g = \frac{3\mu}{2R_s^3} \left[\hat{R}_s \times (I \cdot \hat{R}_s) \right] \quad (\text{Eq. 1})$$

Here R_s is the vector from the center of the Earth to the spacecraft, and is assumed to be much, much larger than the distance from the center of mass of the spacecraft to any other point on the spacecraft. The quantity μ is the gravitational parameter of the Earth, and I is the moment of inertia tensor for the spacecraft. When the body frame of the spacecraft is defined to be along the principle axes set, this equation expands toⁱⁱ:

$$\begin{aligned}
M_1 &= -\frac{3\mu \sin \theta_3 \sin \theta_2 \cos \theta_2}{R_s^3} (I_{33} - I_{22}) \\
M_2 &= \frac{3\mu \cos \theta_3 \cos \theta_2 \sin \theta_2}{R_s^3} (I_{11} - I_{33}) \\
M_3 &= -\frac{3\mu \cos \theta_3 \sin \theta_3 \cos^2 \theta_2}{R_s^3} (I_{22} - I_{11})
\end{aligned} \tag{Eq. 2}$$

Here M_1 , M_2 , and M_3 are the torques about each principle axis, I_{11} , I_{22} , and I_{33} are the principle moments of inertia, and θ_1 , θ_2 , and θ_3 , are the Euler angles that describe the orientation of the body frame with respect to an equilibrium position, in the notation of the authors. This is further simplified into a single axis equation by John S. Eterno in SMADⁱⁱⁱ to:

$$T_g = \frac{3\mu}{2R^3} |I_z - I_y| \sin(2\theta) \tag{Eq. 3}$$

Where T_g is the maximum torque, I_x is the largest principle moment of inertia and I_y is the smallest, and θ is the maximum deviation of the z-axis from the local vertical. While perhaps a more detailed analysis could be achieved by using Equation 1 or Equation 2 in this analysis, the simplicity of Equation 3 and the result of a maximum gravity gradient disturbance torque make it the preferred equation for this report.

Solar Radiation

Solar radiation pressure is a result of the transfer of momentum from photons of light to the surface of the spacecraft. The result of this pressure across the spacecraft surface is a force that acts through the center of pressure, ϕ , of the spacecraft. In most cases, the center of pressure will not be on the line from the center of mass of the spacecraft to the Sun (particularly for nonspherical spacecraft that change their orientation with respect to the Sun), and thus a torque will be generated around the center of mass, cm . For Earth-orbiting spacecraft, where the distance from the spacecraft to the Earth is small compared to the Earth-Sun distance, the mean solar flux acting on the spacecraft is considered a constant (regardless of orbital radius or position). In general, the solar radiation torque is found to be much smaller than the other environmental disturbance torques, and is frequently neglected.

Spenceⁱ derives the force due to the solar radiation pressure to be:

$$\vec{F} = -\frac{F_e}{c} A \cos \theta \left[(1 - C_s) \hat{s} + 2(C_s \cos \theta + \frac{1}{3} C_d) \hat{n} \right] \tag{Eq. 4}$$

Here F_e is the solar constantⁱ ($\sim 1358 \text{ W/m}^2$), c is the speed of light, A is the planar area of the spacecraft, θ is the angle between the vector to the Sun and the normal vector to the surface, C_s is the coefficient of specular reflection, C_d is the coefficient of diffuse reflection, \hat{s} is the unit vector to the sun, and \hat{n} is the unit vector normal to the planar area. When C_a is the absorption coefficient, then $C_s + C_d + C_a = 1$. In the worst case scenario, the planar area is normal to the sun vector, and the cosine is 1 and the terms in brackets

combine. Since C_s is usually large (around 0.6), one-third of C_d will be very small. Neglecting this term, the resulting equation is:

$$\vec{F} = -\frac{F_s}{c} A [1 + C_s] \hat{n} \quad (\text{Eq. 5})$$

Compare this to the approximation presented in SMADⁱⁱⁱ:

$$F = \frac{F_s}{c} A_s (1 + q) \cos(i) \quad (\text{Eq. 6})$$

Where, in Eterno's notation, F_s is the solar constant, A_s is the planar area, q is the reflectance factor (C_s), and i is the angle of incidence of the sunlight. Using a worst-case angle of incidence of 90 degrees, Equation 6 matches Equation 5 exactly. With the confidence of Spence's derivation, the worst case Equations 5 and 6 can be used with confidence in this analysis.

$$T_{sp} = F(c_{ps} - cg) \quad (\text{Eq. 7})$$

The torque due to the solar radiation pressure can be obtained by multiplying the force by its moment arm, the distance from the center of pressure to the center of gravity.

Magnetic Field

Spacecraft in orbit around the Earth experience a torque caused by the interaction of their residual magnetic fields with the Earth's magnetic field. The strength of the Earth's dipole field can be approximated with a 90 percent accuracyⁱⁱ as:

$$B = \frac{\mu_e}{r^3} [1 + 3(\sin \theta_m)^2]^{1/2} \quad (\text{Eq. 8})$$

Here r is the radius of the spacecraft orbit, μ_e is the magnitude of the Earth's magnetic moment vector along the dipole axis, and θ_m is the magnetic latitude (note the magnetic north pole is inclined 11.5 degrees from the Earth's spin axis). The magnetic torque is then:

$$T_m = MB \sin \eta \quad (\text{Eq. 9})$$

Here M is the spacecraft's magnetic moment and η is the angle between the magnetic moment and the local field lines of the Earth's magnetic field. In a worst case scenario, when the spacecraft is at the magnetic pole ($\sin \theta_m=1$) and the spacecraft magnetic moment is normal to the local field lines ($\sin \eta =1$), the magnetic torque is calculated as:

$$T_m = M \frac{2\mu_e}{r^3} \quad (\text{Eq. 10})$$

Compare this formulation to that presented in SMADⁱⁱⁱ:

$$T_m = DB \quad (\text{Eq. 11})$$

$$B = \frac{2M}{R^3} \quad (\text{Eq. 12})$$

Here D is the residual dipole of the vehicle, B is the Earth's magnetic field strength, M is the magnetic moment of the Earth, and R is the radius to the spacecraft. Thus, once again, the approximation from SMAD is justified for use in this analysis.

Aerodynamic Drag

For spacecraft orbiting in low Earth orbit (LEO), particularly those below approximately 350 km, the force of atmospheric drag is a serious consideration. The drag force robs a spacecraft of orbital energy, forcing it to reboost periodically, and can also create a significant torque if it does not act through the center of mass. The force of drag is universally defined as [i, ii, iii]:

$$F = \frac{1}{2} \rho C_d A V^2 \quad (\text{Eq. 13})$$

Where ρ is the atmospheric density, C_d is the spacecraft coefficient of drag (normally between 2 and 4), A is the drag-exposed area of the spacecraft, and V is the spacecraft velocity. The torque on the spacecraft is thus:

$$T_a = F(c_{pa} - cg) \quad (\text{Eq. 14})$$

The atmospheric density can be looked up in a table or approximated using an exponential model of the atmosphere, such as the one presented by Davisⁱⁱ:

$$\rho = \rho_o e^{(-h/H_o)} \quad (\text{Eq. 15})$$

Here ρ_o is the density at some reference altitude, h is the spacecraft altitude, and H_o is the atmospheric scale height at the reference altitude. For this analysis, worst-case atmospheric density (solar max data) has been tabulated from SMAD and interpolation is performed for intermediate data points.

B. Requirements Specification

The *ACSSize* program shall allow the user to:

1. Choose their desired combination of mission objectives and spacecraft characteristics.
2. Enter all parameters needed through a user interface function *ACSSize*.

3. Display the results of the calculations numerically.
4. Easily compare and contrast the results obtained from environmental disturbances and mission objectives.

1. *Choose their desired combination of power requirements and orbital elements.*

To achieve this requirement, the user must input the following information:

a [m]: semimajor axis of the orbit

e : eccentricity of the orbit

MU [m^3/s^2]: gravitational parameter for the central body

Cd : the coefficient of drag on the spacecraft

A [m^2]: the “shadow area” of the spacecraft (worst-case cross-sectional area)

rt [m]: the moment arm for the drag force (center of pressure – center of mass)

D [amp*m*s]: residual dipole of the vehicle

I_{max}, I_{min} [$kg*m^2$]: maximum and minimum moments of inertia of the spacecraft

$pointing_accu$ [deg]: pointing accuracy required for mission

$pointing_req$ []: where you want your spacecraft pointing (ep == earth-pointing, ip == inertial pointing)

$slew_dist$ [deg] - amount needed to slew

$slew_time$ [s] - amount of time needed to slew

$moment_arm = [m]$ - moment arm from thruster to cg

The applicable formulas can be found in chapter 11.1 of SMADⁱⁱⁱ or as presented above. These calculations are also documented in the comments of the code.

2. *Enter all parameters needed through a user interface function ACSsize*

The input and output values of the ACSsize program are illustrated in the blackbox behavior diagram of Figure 1. The parameters are entered into Matlab through the function as indicated above. The program outputs are displayed in the Matlab command window.

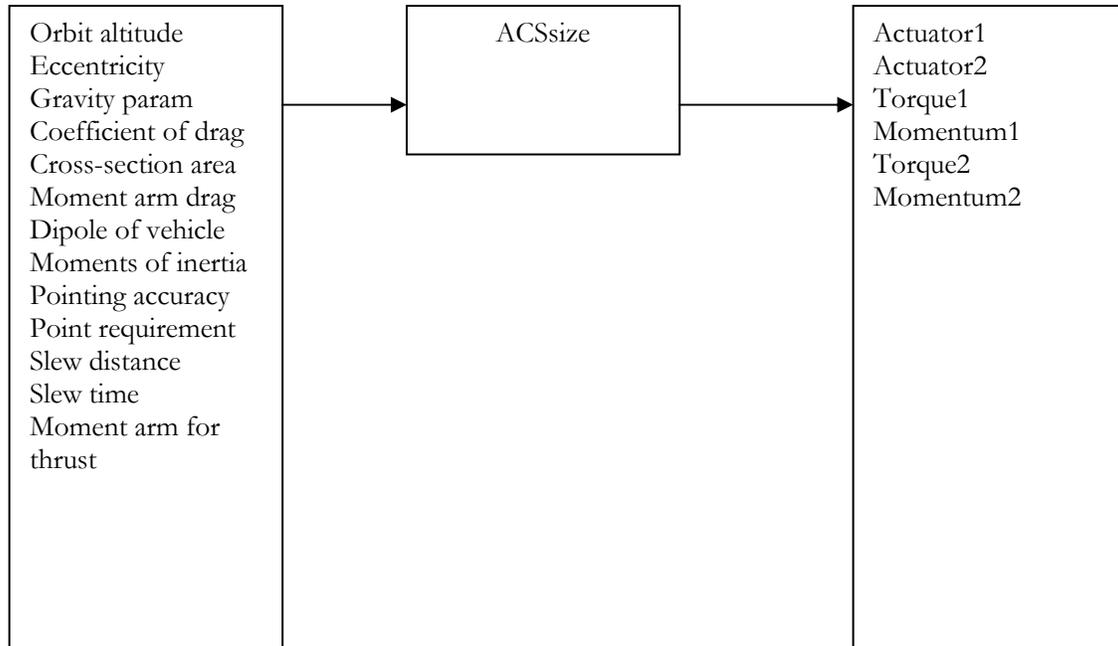


Figure 1 – Blackbox behavior diagram

3. *Display the results of the calculations numerically.*

The numerical output is also displayed in Matlab after the *ACSsize* program calculates the types and sizes of the actuators needed.

4. *Easily compare and contrast the results obtained from mission objectives and mission parameters.*

The numerical outputs of the *ACSsize* can be compiled in tabular form (as shown in Table 2) in order to perform trade-off analyses between the various materials.

C. Assumptions

- a. Only two actuators are handled by the *ACSsize* program.
- b. If sizing is done for a Magnetic torquer, the dipole value is placed in torque placeholder. The B field is assumed to be the worst case Earth field for calculating the dipole for the magnetic torquer.
- c. Other minor assumptions as listed in the comments in the Matlab code.

D. Sample Test Run

One example test run was executed using typical values for a simplified satellite. Table 1 contains the information entered into the program through the function for test run.

Parameter	Value
a	700 km
e	0
MU	$3.986 \times 10^{14} \text{ m}^3/\text{s}^2$
Cd	2
A	3 m^2
rt	0.2 m
Moment_arm	0.5 m
D	1
Imax	90
Imin	60
Pointing_accu	2 degs
Pointing_req	E_p
Slew_dist	30
Slew_time	600 s

Table 1. Inputs for Test Run

The outputs here can be then be compared with outputs generated from different mission objectives to determine which types and sizes of actuators best suits the needs of the project. The results of the example test run are listed in Table 2. The results include the names of the actuators and the parameters that are needed to define the size. As seen in Table 2, this spacecraft would have a reaction wheel and thrusters on it. The torque is outputted in Newtons and the momentum is outputted in N.m.s. As one can see the thrusters do not store momentum so here the momentum is zero.

Actuator1	Reaction wheel
Actuator2	Thruster
Torque1	0.5229 N
Momentum1	1.7036 N.m.s
Torque2	0.3 N
Momentum2	0

Table 2. Output from ACSsize program

E. User's Guide

a. Setup: To run the ACSsize program, first open Matlab. Note that STK Version 5 must be used to run the program. Before running the program, make sure that the following files are all in the Matlab working directory:

ACSsize.m
ae2rv.m
magtorsizing.m
momwheelsizing.m

obj2actuator.m
reacwbeel_cmgsizex.m
t_aero.m
t_bfield.m
t_gg.m
t_solar.m
thrustersizing.m

b. Running the Program: The *ACSsize* program is run from the Matlab command prompt by entering the command *ACSsize*.

c. Entering Parameters: Start by entering all the mission objectives and spacecraft requirements for your power system and your desired orbit. The required data for the calculations is listed in the requirements specification.

d. Calculating Results: When all the necessary parameters have been entered, hit enter to calculate the types and sizes of actuators needed for the spacecraft. The program will perform the necessary calculations to calculate the specific types and sizes of actuators needed for your spacecraft.

e. Displaying Results: The output parameters are displayed in the output window in Matlab. For sample output, see Table 2.

F. Code

```

function [actuator1, actuator2, torque1, momentum1, torque2, momentum2] = ACSsize(a, e,
MU, Cd, A, rt, moment_arm, D, Imax, Imin, pointing_accu, pointing_req, slew_dist,
slew_time)

% function ACSsize.m calculates the size of ACS actuators given mission
% objectives and spacecraft characteristics
% a [ m ]: semimajor axis of the orbit
% e [ ]: eccentricity of the orbit
% MU [m^3/s^2]: gravitational parameter for the central body
% Cd [ ]: the coefficient of drag on the spacecraft
% A [m^2]: the "shadow area" of the spacecraft (worst-case cross-sectional area)
% rt [ m ]: the moment arm for the drag force (center of pressure - center
% of mass)
% moment_arm = [ m ] - moment arm from thruster to cg
% D [amp*m^s]: residual dipole of the vehicle
% Imax, Imin [kg*m^2]: maximum and minimum moments of inertia of the spacecraft
% pointing_accu [degrees]: pointing accuracy required for mission
% pointing_req [ ]: where you want your spacecraft pointing (ep == earth-pointing,
% ip == inertial pointing)
% slew_dist [degrees] - amount needed to slew
% slew_time [seconds] - amount of time needed to slew

[r,v] = ae2rv(a, e, MU);

%find disturbance torques

tq_aero = t_aero(r, Cd, A, v, rt);

tq_bfield = t_bfield(r, D);

```

```

tq_g = t_gg(r, I_max, I_min, MU);

tq_solar = t_solar(A, rt);

% map mission objectives to type of actuators
[actuator1, actuator2] = obj2actuator(pointing_accu, pointing_req, slew_dist/slew_time);

%determine sizes

%assume a circular orbit for calculating periods, and I_max for use in
%craft inertia
P = (2*pi*r)/(60*v);
torque1 = 0;
torque2 = 0;
momentum1 = 0;
momentum2 = 0;
if(strcmp(actuator1, 'rw') == 1 || strcmp(actuator1, 'cmg') == 1)
    [torque1, momentum1] = reacwheel_cmgsizing(tq_aero, tq_bfield, tq_g, tq_solar,
slew_dist, slew_time, I_max, P);
elseif(strcmp(actuator1, 'mw') == 1)
    momentum1 = momwheelsizing(tq_aero, tq_bfield, tq_g, tq_solar, pointing_accu, P);
elseif(strcmp(actuator1, 'thruster') == 1)
    torque1 = thrustersizing(tq_aero, tq_bfield, tq_g, tq_solar, slew_dist, slew_time,
moment_arm, I_max);
else
    torque1 = magtorsizing(tq_aero,tq_bfield,tq_g,tq_solar);
end

if(strcmp(actuator2, 'rw') == 1 || strcmp(actuator2, 'cmg') == 1)
    [torque2, momentum2] = reacwheel_cmgsizing(tq_aero, tq_bfield, tq_g, tq_solar,
slew_dist, slew_time, I_max, P);
elseif(strcmp(actuator2, 'mw') == 1)
    momentum2 = momwheelsizing(tq_aero, tq_bfield, tq_g, tq_solar, pointing_accu, P);
elseif(strcmp(actuator2, 'thruster') == 1)
    torque2 = thrustersizing(tq_aero, tq_bfield, tq_g, tq_solar, slew_dist, slew_time,
moment_arm, I_max);
else
    torque2 = magtorsizing(tq_aero,tq_bfield,tq_g,tq_solar);
end

function [r, v] = ae2rv(a, e, MU)

% function ae2rv calculates the scalar minimum radius (rp) and maximum
% velocity for a spacecraft in a circular or elliptical orbit

% a [ m ]: semimajor axis of the orbit
% e [ ]: eccentricity of the orbit
% MU [m^3/s^2]: gravitational parameter for the central body

r = a*(1 - e); % radius of perigee

v = sqrt(2*(MU/r - MU/(2*a))); % velocity at perigee

function dipole = magtorsizing(t_aero,t_bfield,t_gg,t_solar)

% function magtorsizing.m sizes reaction wheels and cmgs:

% t_aero,t_bfield,t_gg,t_solar [N m]- disturbance torques from the environment

if(t_aero > t_bfield && t_aero > t_gg && t_aero > t_solar)
    t_d = t_aero;
elseif(t_bfield > t_aero && t_bfield > t_gg && t_bfield > t_solar)
    t_d = t_bfield;
elseif(t_gg > t_aero && t_gg > t_bfield && t_gg > t_solar)
    t_d = t_gg;
else
    t_d = t_solar;
end

B = 4.5e-15; % worst case Earth field

```

```

D = t_d/B;

function momentum = momwheelsizing(t_aero,t_bfield,t_gg,t_solar,accu, orbit_period)

% function momwheelsizing.m sizes momentum wheels:

% t_aero,t_bfield,t_gg,t_solar [N m]- disturbance torques from the environment
% accu [degrees] - accuracy needed
% orbit_period [min] - orbit period

if(t_aero > t_bfield && t_aero > t_gg && t_aero > t_solar)
    t_d = t_aero;
elseif(t_bfield > t_aero && t_bfield > t_gg && t_bfield > t_solar)
    t_d = t_bfield;
elseif(t_gg > t_aero && t_gg > t_bfield && t_gg > t_solar)
    t_d = t_gg;
else
    t_d = t_solar;
end

momentum = (t_d*orbit_period*60)/(4*0.01745*accu);

function [actuator1, actuator2] = obj2actuator(pointing_accu, pointing_req, slew_rate)

% function obj2actuator.m maps mission requirements to acuator choices:

% pointing_accu [degrees]: pointing accuracy required for mission
% pointing_req [ ]: where you want your spacecraft pointing (ep == earth-pointing,
% ip == inertial pointing)
% slew rate [deg/s]: maximum slew rate required for spacecraft

if(pointing_accu > 5)
    control_type = 'gg';
elseif(pointing_accu < 5 && pointing_accu > 1)
    control_type = 'ss';
else
    control_type = '3ax';
end

if(strcmp(pointing_req, 'ep') == 1)
    if(strcmp(control_type, 'gg') == 1)
        actuator1 = 'mw';
        actuator2 = 'mw';
    elseif(strcmp(control_type, '3ax') == 1)
        actuator1 = 'rw';
        actuator2 = 'cmg';
    else
        actuator1 = 'rw';
        actuator2 = 'thruster';
    end
else
    actuator1 = 'rw';
    actuator2 = 'thruster';
end

if(slew_rate < 0.05 && strcmp(actuator2, 'thruster') == 1)
    actuator2 = 'magtor';
end
if((slew_rate > 0.05 || slew_rate == 0.05) && slew_rate < 0.5 && strcmp(actuator2,
'thruster') ~= 1)
    actuator2 = 'thruster';
end
if (slew_rate > 0.5 || slew_rate == 0.5)
    actuator2 = 'cmg';
end

function [torque, momentum] =
reacwheel_cmgsize(t_aero,t_bfield,t_gg,t_solar,slew_dist,slew_time, craft_inertia,
orbit_period)

```

```

% function reacwheel&cmgsize.m sizes reaction wheels and cmgs:

% t_aero,t_bfield,t_gg,t_solar [N m]- disturbance torques from the environment
% slew_dist [degrees] - amount needed to slew
% slew_time [seconds] - amount of time needed to slew
% craft_inertia [kg.m^2] - spacecraft moment of inertia
% orbit_period [min] - orbit period
% torque [N.m] - torque required
% momentum = [N.m.s] - angular momentum required

if(t_aero > t_bfield && t_aero > t_gg && t_aero > t_solar)
    t_d = t_aero;
elseif(t_bfield > t_aero && t_bfield > t_gg && t_bfield > t_solar)
    t_d = t_bfield;
elseif(t_gg > t_aero && t_gg > t_bfield && t_gg > t_solar)
    t_d = t_gg;
else
    t_d = t_solar;
end

margin = 10; % margin factor for disturbance rejection

torque_d = 10*t_d;

torque_slew = (4*slew_dist*0.01745*craft_inertia)/(slew_time)^2;

if(torque_slew > torque_d)
    torque = torque_slew;
else
    torque = torque_d;
end

% 0.707 is the rms average of the sinusoidal function
momentum = (t_d*orbit_period*60*0.707)/4;

function [torque, momentum] =
reacwheel_cmgsize(t_aero,t_bfield,t_gg,t_solar,slew_dist,slew_time, craft_inertia,
orbit_period)

% function reacwheel&cmgsize.m sizes reaction wheels and cmgs:

% t_aero,t_bfield,t_gg,t_solar [N m]- disturbance torques from the environment
% slew_dist [degrees] - amount needed to slew
% slew_time [seconds] - amount of time needed to slew
% craft_inertia [kg.m^2] - spacecraft moment of inertia
% orbit_period [min] - orbit period
% torque [N.m] - torque required
% momentum = [N.m.s] - angular momentum required

if(t_aero > t_bfield && t_aero > t_gg && t_aero > t_solar)
    t_d = t_aero;
elseif(t_bfield > t_aero && t_bfield > t_gg && t_bfield > t_solar)
    t_d = t_bfield;
elseif(t_gg > t_aero && t_gg > t_bfield && t_gg > t_solar)
    t_d = t_gg;
else
    t_d = t_solar;
end

margin = 10; % margin factor for disturbance rejection

torque_d = 10*t_d;

torque_slew = (4*slew_dist*0.01745*craft_inertia)/(slew_time)^2;

if(torque_slew > torque_d)
    torque = torque_slew;
else
    torque = torque_d;
end

```

```

% 0.707 is the rms average of the sinusoidal function
momentum = (t_d*orbit_period*60*0.707)/4;

function torque = t_aero(h, Cd, A, v, rt)

% function t_aero.m calculates the aerodynamic torque on a vehicle given
% the following inputs:

% h [ m ]: minimum altitude of spacecraft (rp - re)
% Cd [ ]: the coefficient of drag on the spacecraft
% A [m^2]: the "shadow area" of the spacecraft (worst-case cross-sectional area)
% v [m/s]: the velocity of the spacecraft (velocity at perigee)
% rt [ m ]: the moment arm for the drag force (center of pressure - center of mass)

rho_max = [5.10e-7; 2.04e-9; 3.52e-10; 1.06e-10; 3.96e-11; 1.66e-11; 7.55e-12; 3.61e-12;
...
1.80e-12; 9.25e-13; 4.89e-13; 2.64e-13; 1.47e-13; 8.37e-14; 4.39e-14; 3.00e-
14; 1.91e-14; ...
1.27e-14; 8.84e-15; 2.59e-15; 1.22e-15]; % densities during solar max
[kg/m^3]
rho_alt = [100; 150; 200; 250; 300; 350; 400; 450; 500; 550; 600; 650; ...
700; 750; 800; 850; 900; 950; 1000; 1250; 1500]*1000; % corresponding
altitudes in [m]

rho = spline(rho_alt, rho_max, h); % interpolate the density [kg/m^3]

F_aero = (1/2)*rho*Cd*A*v*v; % aerodynamic drag force [N]

torque = F_aero*rt; % aerodynamic torque [N*m]

function torque = t_bfield(r, D)

% function t_bfield.m calculates the gravity gradient torque on a vehicle given
% the following inputs:

% r [m]: minimum radius from center of massive body to spacecraft
% D [amp*m^s]: residual dipole of the vehicle

M = 7.96e15; % [tesla*m^3], magnetic moment of the Earth
B = (2*M)/r^3; % [tesla], earth's magnetic field

torque = D*B;

function torque = t_gg(r, Imax, Imin, MU)

% function t_gg.m calculates the gravity gradient torque on a vehicle given
% the following inputs:

% r [m] : minimum radius from center of massive body to spacecraft
% Imax, Imin [kg*m^2]: maximum and minimum moments of inertia of the spacecraft
% MU [m^3/s^2]: gravitational parameter for the central body

torque = (3*MU/(2*r^3))*(Imax - Imin);

function torque = t_solar(A, rt)

% function t_solar.m calculates the solar radiation torque on a vehicle given
% the following inputs:

% A [m^2]: the "shadow area" of the spacecraft (worst-case cross-sectional area)
% rt [ m ]: the moment arm for the drag force (center of pressure - center of mass)

Fs = 1367; % [W/m^2], solar constant at 1AU
c = 3e8; % [m/s], speed of light
q = 0.6; % [ ], reflectance factor

torque = (Fs/c)*A*(1+q)*rt;

```

```

function force =
thrustersizing(t_aero,t_bfield,t_gg,t_solar,slew_dist,slew_time,moment_arm,craft_inertia)

% function thrustersizing.m sizes thrusters and cmgs:

% t_aero,t_bfield,t_gg,t_solar [N m]- disturbance torques from the environment
% slew_dist [degrees] - amount needed to slew
% slew_time [seconds] - amount of time needed to slew
% moment_arm = [ m ] - moment arm
% craft_inertia [kg.m^2] - spacecraft moment of inertia

if(t_aero > t_bfield && t_aero > t_gg && t_aero > t_solar)
    t_d = t_aero;
elseif(t_bfield > t_aero && t_bfield > t_gg && t_bfield > t_solar)
    t_d = t_bfield;
elseif(t_gg > t_aero && t_gg > t_bfield && t_gg > t_solar)
    t_d = t_gg;
else
    t_d = t_solar;
end

%force level sizing
force1 = t_d/moment_arm;

%sizing force to meet slew rate accelerating for 5% of the time, coasting
%for 90% and decelerating for 5%.

slew_rate = slew_dist/slew_time;

accel_time = 0.05*slew_time;

accel = slew_rate/accel_time;
force2 = (craft_inertia*accel)/moment_arm;

if(force1 > force2)
    force = force1;
else
    force = force2;
end

```

References

- ⁱ Wertz, James R. (Editor), *Spacecraft Attitude Determination and Control*, D. Reidel Publishing Company, Boston, MA. 1986.
- ⁱⁱ Davis, D., Agnes, G., *Environmental Disturbance Modeling for Inflatable Space Structures*, 43rd AIAA/ASME Structures, Structural Dynamics, and Materials Conference, 22-25 April 2002. Paper AIAA 2002-1266.
- ⁱⁱⁱ Wertz, James R. (Editor), *Space Mission Analysis and Design*, Microcosm Press, El Segundo, CA, 1999.