*California Polytechnic State University, San Luis Obispo*

# Feasibility of CubeSat Formation Flight Using Rotation to Achieve Differential Drag

Skyler M. Shuford
Aerospace Engineering
June 2013

# Feasibility of CubeSat Formation Flight Using Rotation to Achieve Differential Drag

Skyler M. Shuford*
*California Polytechnic State University, San Luis Obispo, CA, 93401*

**This paper presents the results of a study conducted to understand the feasibility of CubeSat formation flight. The mechanism for separation and formation studied was differential drag, achieved by rotating the CubeSats to give them different cross-sectional areas. Intuitively, lower altitude orbits provide much higher separation effects. Although the most influential orbital effects occur with maximum and minimum cross-sectional areas, an attitude-controlled and a tumbling CubeSat may provide enough differential drag to meet separation requirements of a mission. Formation flight is possible, but due to the non-linearity of the system, gain scheduling may be the most effective method of long term formation control. Formation flight on missions with sun-tracking is also possible using the time in eclipse as the control time. Future studies will need to see how long formation can be maintained, as well as how significant altitude affects the total possible formation duration.**

## Nomenclature

| | | |
|---|---|---|
| $A$ | = | CubeSat cross-sectional area |
| $a$ | = | acceleration |
| $e$ | = | error |
| $h$ | = | angular momentum |
| $k$ | = | control gain |
| $m$ | = | CubeSat mass |
| $r$ | = | position |
| $v$ | = | velocity |

**Greek**

| | | |
|---|---|---|
| $\mu$ | = | gravitation parameter of Earth |
| $\rho$ | = | atmospheric density |
| $\Omega$ | = | rotation rate of the chief coordinate frame |

**Subscripts**

| | | |
|---|---|---|
| *chief* | = | chief spacecraft |
| *d* | = | derivative |
| *deputy* | = | deputy spacecraft |
| *desired* | = | the desired target value for the control scheme |
| *drag* | = | effect of drag |
| *ECI* | = | Earth-centered inertial |
| *grav* | = | effect of gravity |
| *I, J, K* | = | components of ECI coordinate frame |
| *LVLH* | = | local-vertical local-horizontal |
| *nom* | = | nominal |
| *p* | = | proportional |
| *x, y, z* | = | components of the LVLH coordinate frame |

**Symbols**

* Undergraduate Student, Aerospace Engineering Dept., San Luis Obispo, CA, AIAA Member Grade 0.

$C_d$ = coefficient of drag
$\bar{x}$ = state vector
$\delta A$ = differential cross-sectional area
$\delta r$ = relative postion
$\delta v$ = relative velocity

## I. Introduction

CUBESATS are spacecraft of the picosatellite class that follow the standard created by California Polytechnic State University (Cal Poly) and Stanford University. This standard limits 1 unit (1U) CubeSats to 1.5 kg and a 10 cm cube (fig. 1), and 3U CubeSats to 4 kg and a 10 cm x 10 cm x 30 cm envelope. This standard allows ease of integration into the Poly-Picosatellite Orbital Deployer (P-POD). The P-POD secures the CubeSats to the launch vehicle and ensures the protection of the primary payload from the secondary, CubeSat payload. Flying CubeSats as secondary payload gives companies and universities low-cost, reliable access to space.

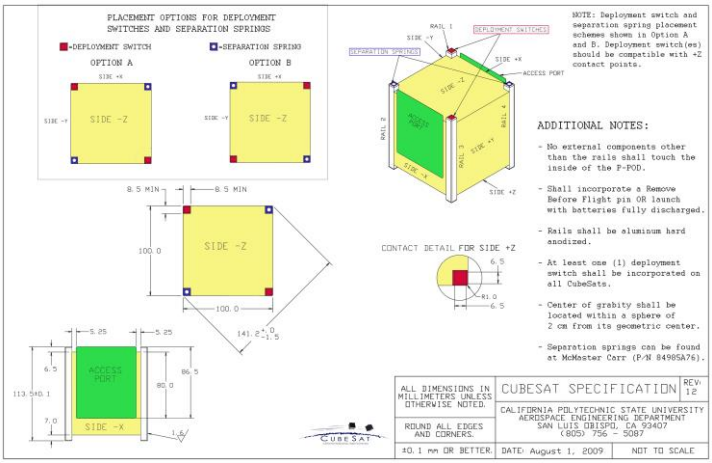


**Figure 1: Drawing of CubeSat 1U Standard** *(courtesy cubesat.org)*

The ever-increasing capabilities of CubeSats, along with relatively inexpensive development and flight costs, create the opportunity for multi-satellite formations and constellations. However, the CubeSat standard does not currently allow for propulsion pressurant tanks onboard the spacecraft due to the risk of rupture or misfire. Therefore, CubeSat developers must use means other than pressurized thrusters in order to change and control the orbit of the CubeSat. Electric propulsion, solar sails, the Lorentz force using Earth's magnetic field, and differential drag[1] are four non-pressurized options that would allow for orbit changes. The focus of this paper is to study the feasibility of simple means of attaining differential drag in order to separate and maintain formation.

Drag is a function of drag coefficient, atmospheric density, mass, cross-sectional area (CSA), and velocity. Assuming a constant drag coefficient and mass, the logical control surface for achieving differential drag is CSA. A difference in CSA creates differential accelerations on the CubeSats involved, which changes the orbits of both spacecraft. Consequently, the atmospheric density, which is a function of altitude in the simplified model, and velocity would also be changed.

CSA can be varied in a few different ways. The simplest is to rotate the CubeSat so that there is a different area in the ram direction. Consequently, a tumbling CubeSat has a different effective CSA than a controlled CubeSat with a constant face in the ram direction. Another method of changing CSA is to deploy solar panels or a sail. This allows for much larger differences in CSA, however the mechanisms for deployment add complexity and risk to the design.

Colony II is a 3U CubeSat bus created by Boeing. It has deployable solar panels that deploy to create a maximum CSA of 2100 cm$^2$. The spacecraft has 3-axis control in order to track the sun with the solar panels. Since rotating the CubeSat for drag-based formation flight is impossible while it tracks the sun, the only opportunity for orbit control is during eclipse.

## II. Procedure and Equations

A MATLAB simulation propagated the orbits of two CubeSats[2]. This was done using ode113 to integrate the derivative of the state vector in Earth Centered Inertial (ECI) coordinates.

$$\dot{\bar{x}} = \left[ v_I, v_J, v_K, a_I, a_J, a_K \right]^T \tag{1}$$

The velocity was updated from the state vector input to ode113. The accelerations included were from gravity and drag. The acceleration from gravity was in the reverse position vector and assumed the Earth to be a point mass. The acceleration from drag was in the reverse ram direction. The acceleration vectors are

$$\vec{a}_{grav} = -\frac{\mu}{r^2}\frac{\vec{r}}{\|\vec{r}\|} \tag{2}$$

$$\vec{a}_{drag} = -\frac{1}{2}\frac{C_d A \rho v^2}{m}\frac{\vec{v}}{\|\vec{v}\|} \tag{3}$$

where $\mu$ is the gravitational parameter of Earth, $r$ is the position, $C_d$ is the coefficient of drag, $A$ is the CSA, $\rho$ is the atmospheric density, $v$ is the velocity, and $m$ is the mass. The atmospheric density was estimated using the exponential model, which is a function of altitude.

The code converts the position and velocity vectors into the Local Vertical Local Horizontal (LVLH) coordinate frame in order to plot it in an easily analyzed format[3]. Since LVLH is a rotating coordinate frame, the relative velocity vector has an extra term to account for the rotation. The relative position and velocity vectors can be found using

$$\vec{h}_{chief} = \vec{r}_{chief} \times \vec{v}_{chief} \tag{4}$$

$$\vec{\Omega} = \frac{\vec{h}_{chief}}{\|\vec{r}_{chief}\|^2} \tag{5}$$

$$\delta\vec{r}_{ECI} = \vec{r}_{deputy} - \vec{r}_{chief} \tag{6}$$

$$\delta\vec{v}_{ECI} = \vec{v}_{deputy} - \vec{v}_{chief} - (\vec{\Omega} \times \delta\vec{r}_{ECI}) \tag{7}$$

where $\vec{r}_{chief}$ is the position vector of the chief in ECI, $\vec{v}_{chief}$ is the velocity vector, $\vec{h}_{chief}$ is the angular velocity vector, $\vec{\Omega}$ is the rate of the rotating relative coordinate frame, and $\delta\vec{r}_{ECI}$ and $\delta\vec{v}_{ECI}$ are the relative position and velocity vectors of the deputy in ECI, respectively. The relative position and velocity vectors are then converted from the ECI frame to the LVLH frame through a transformation matrix. The equations used are

$$\hat{i} = \frac{\vec{r}_{chief}}{\|\vec{r}_{chief}\|} \tag{8}$$

$$\hat{k} = \frac{\vec{h}_{chief}}{\|\vec{h}_{chief}\|} \tag{9}$$

$$\hat{j} = \hat{k} \times \hat{i} \tag{10}$$

$$Q_{ECI}^{LVLH} = \begin{bmatrix} \hat{i}_x & \hat{j}_x & \hat{k}_x \\ \hat{i}_y & \hat{j}_y & \hat{k}_y \\ \hat{i}_z & \hat{j}_z & \hat{k}_z \end{bmatrix} \tag{11}$$

$$\delta\vec{r}_{LVLH} = Q_{ECI}^{LVLH}\delta\vec{r}_{ECI} \tag{12}$$

$$\delta\vec{v}_{LVLH} = Q_{ECI}^{LVLH}\delta\vec{v}_{ECI} \tag{13}$$

where $\hat{i}$, $\hat{j}$, $\hat{k}$ are unit vectors that define the LVLH coordinate frame, $Q_{ECI}^{LVLH}$ is the transformation matrix, and $\delta\vec{r}_{LVLH}$ and $\delta\vec{v}_{LVLH}$ are the relative position and velocity vectors in LVLH.

The code also has an option to control the CSA in order to separate two CubeSats to a desired relative axial separation. The controller takes into account the error in relative position and velocity from desired to get an estimated differential area necessary. The equations are propagated for both spacecraft with the chief at a nominal CSA and the deputy having an area utilizing the differential area. This is limited to the maximum and minimum deputy areas using an if-statement. The error, differential area, and final acceleration vectors are

$$e = \delta r_{desired} - \delta r_y \tag{14}$$

$$\dot{e} = \delta v_{desired} - \delta v_y = -\delta v_y \tag{15}$$

$$\delta A = k_p e + k_d \dot{e} \tag{16}$$

$$\vec{a}_{deputy} = -\frac{1}{2}\frac{C_d A_{nom} \rho v^2}{m}\frac{\vec{v}}{\|\vec{v}\|} \tag{17}$$

$$\vec{a}_{deputy} = -\frac{1}{2}\frac{C_d (A_{nom} + \delta A) \rho v^2}{m}\frac{\vec{v}}{\|\vec{v}\|} \tag{18}$$

where $e$ and $\dot{e}$ is the relative axial position and relative axial velocity error, respectively, $\delta A$ is the differential area, $k_p$ and $k_d$ are the proportional and derivative gains, respectively, and $A_{nom}$ is the nominal CSA.

## III. Results and Discussion

### A. Separation and Approach

To study the possible separation and approach rates, the orbits of 1U chief and deputy CubeSats were propagated at starting altitudes of 200 km, 400 km, and 600 km using ode113. Since CubeSats are released from the P-POD in very close proximity, it was assumed that the chief and deputy had identical initial orbits. The chief was propagated with a constant minimum CSA, which was 100 cm$^2$. Three cases for the deputy were propagated: 3-axis-control maximum, yaw-control maximum, and tumble. The 3-axis-control maximum case is the absolute maximum differential area between the chief and the deputy. In this case, the deputy was rotated in the yaw and pitch axes so that the cross-section was a hexagon with CSA of 173.2 cm$^2$. The yaw-control maximum case rotated the deputy yaw angle to 45°. This put the deputy CSA at 141.2 cm$^2$. For the tumble case, it was assumed that every orientation of the deputy had an equal likelihood of occurring. The effective CSA was calculated by integrating the CSA for

every orientation in spherical coordinates and dividing by the range of limits of integration. This set the effective CSA of the tumbling deputy to 144.7 $cm^2$.

Intuitively, lower altitudes have much greater atmospheric densities, which allow for much faster relative motion between the two CubeSats. Initially, the differential area has a much greater effect since the density is higher. However, the effect is even more pronounced as the deputy continually lowers its orbit significantly more than the chief. Figures 2-4 show the separation distances and separation rates at 200 km, 400 km, and 600 km initial orbits, respectively. The periodic variation, which is most pronounced in the relative radial velocities, is a byproduct of the integration error in ode113. The amplitude of this error in the 400 km relative radial velocity is on the order of millimeters per second, so it can be assumed negligible. This error may be able to be avoided by propagating using a Variation of Parameters method.

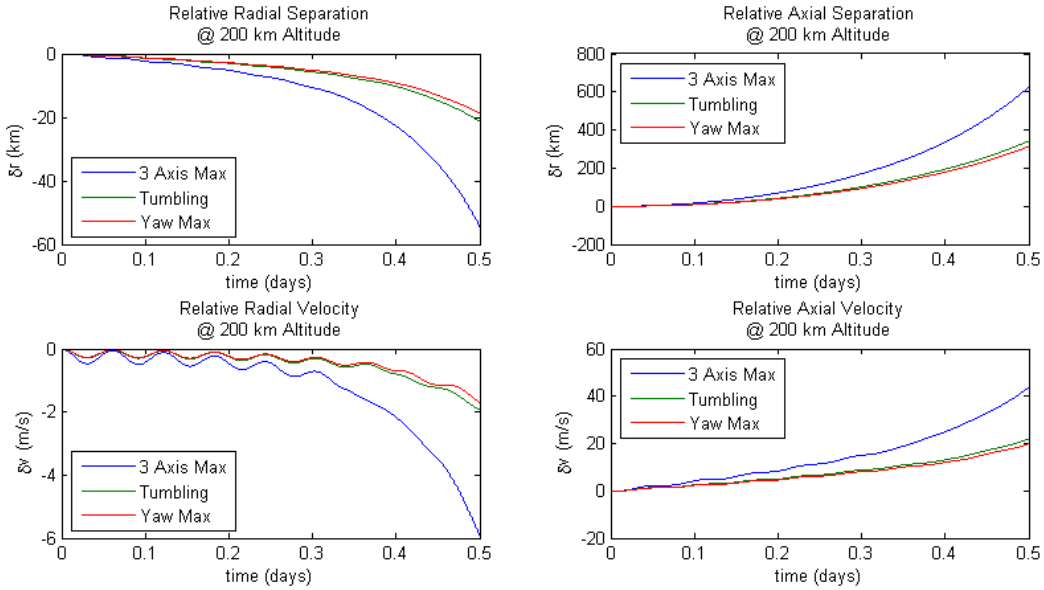


**Figure 2: Separation of 1U CubeSats starting at 200 km altitude**

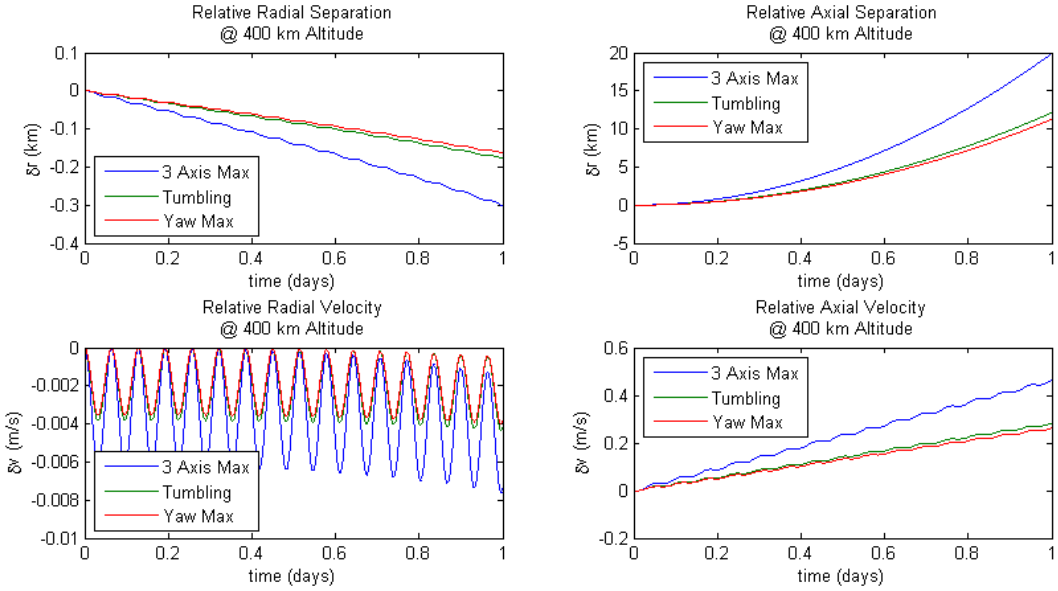


**Figure 3: Separation of 1U CubeSats starting at 400 km altitude**

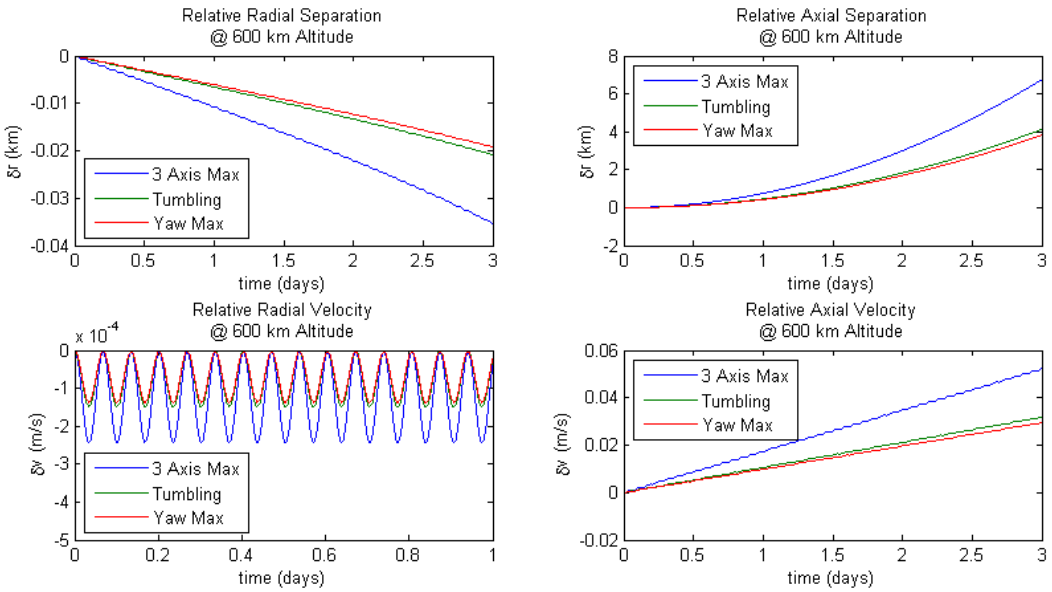


**Figure 4: Separation of 1U CubeSats starting at 600 km altitude**

## B. Control Scheme

The gains for the control scheme in eq. (16) were chosen using a guess-and-check method. The proportional and derivative gains chosen to produce the response in fig. 6 were 8 x $10^{-8}$ and 1.2 x $10^{-2}$, respectively. Figure 5 shows the necessary changes in deputy area to achieve control. The control scheme allowed the relative axial separation to converge near the desired separation within approximately three weeks. The seemingly unstable motion in figs. 5 and 6 is caused by the periodic error that is prevalent in the relative velocity. Since the unstable motion is caused by integration error, a 1000-value moving average[4] was used as a low-pass filter to remove the instability and better show the convergence (figs. 5 and 7). Figure 7 shows the limits of the CSA as well as the pseudo-convergent CSA for formation. The system eventually goes unstable, which is most likely caused by the continued buildup of integration error. Further investigation may prove that the system will remain stable indefinitely.

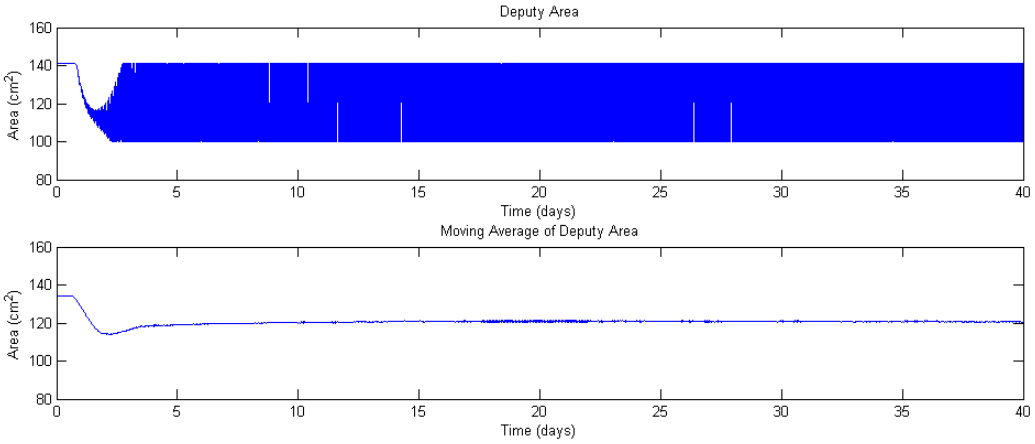


**Figure 5: Variation in Deputy Area over first 40 days**

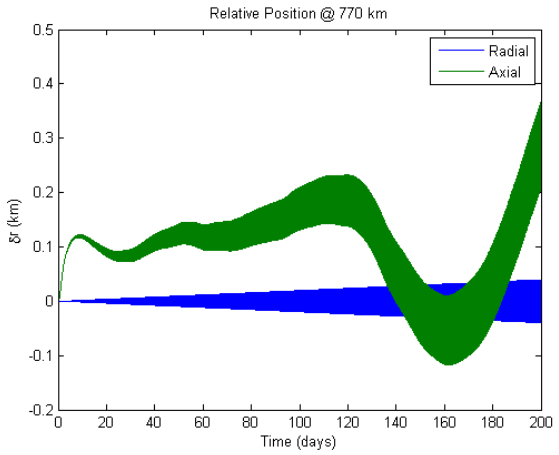


**Figure 6: Relative Position of CubeSats starting at 770 km altitude**

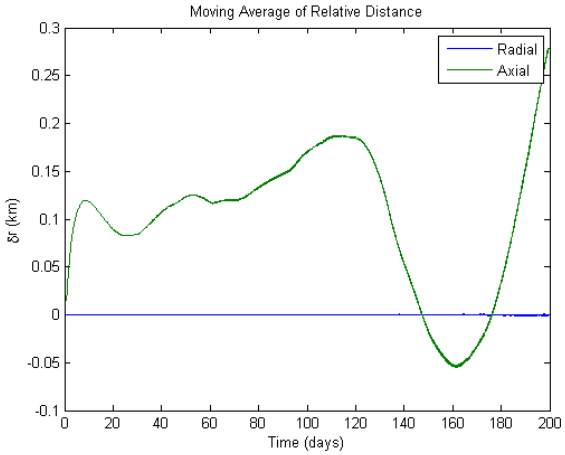


**Figure 7: Relative Position of CubeSats starting at 770 km altitude and including moving average for low-pass filter**

Since the system is non-linear, gain scheduling is a possible method for continued control of the separation. Figure 8 shows the averaged relative positions when a gain scheduling approach was used. In this case, time was used as the trigger for the change in gain. When the time reached 60 days, the code set the derivative gain to zero. With only proportional gain in the system, the relative position oscillates around the desired position. The gain scheduling can be further analyzed to minimize amplitude of oscillation; this analysis used a guess-and-check method to determine that gain scheduling is feasible. Figure 9 shows that the CubeSats remain at a relatively similar altitude throughout the simulation time. For longer duration flights and at lower altitudes, this may not remain true.

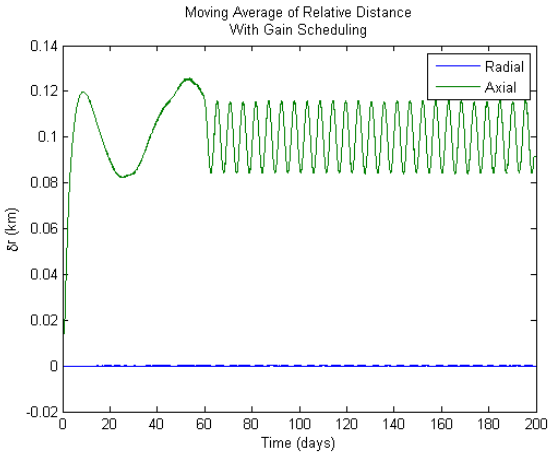


**Figure 8: Gain scheduled control of relative axial position. Desired distance of 0.1 km.**

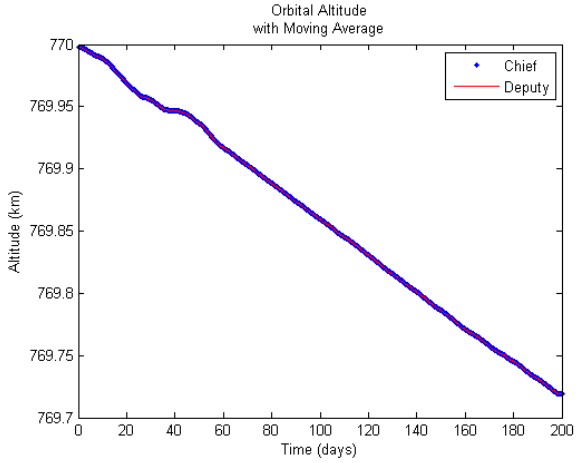


**Figure 9: CubeSat altitude throughout simulation**

### C. Control in Eclipse

The sun-tracking capability of the Colony II bus only allows for formation control in eclipse. To study the feasibility of eclipse-only control, the propagation code contains a check using the Vallado's "shadow" algorithm[5] along with the properties of the Colony II bus (max CSA of 2100 $cm^2$, 4 kg). If both the chief and deputy were in eclipse, then the previously mentioned control scheme was used. If one of the spacecraft was not in eclipse, then that spacecraft was set to an assumed nominal CSA of 1200 $cm^2$. The proportional and derivative gains for the control scheme were 8 x $10^{-7}$ and 1.2 x $10^{-2}$, respectively. The control scheduling time where the derivative gain was set to zero was 25 days.

Figure 10 shows the CSA of the deputy for the first 20 days of the simulation. This was done to show the periodic resetting of the CSA during the sunlit parts of the orbit, which may have not been visible if the entire duration was shown. The gains that were used showed that the system was slowly converging (fig. 11) on the desired distance of 0.1 km. These gains were also found using a guess-and-check method and could be much further refined. The moving average of the altitudes of both spacecraft stay very close together for the duration of the simulation (fig. 12), which allows for formation to be kept much more easily.

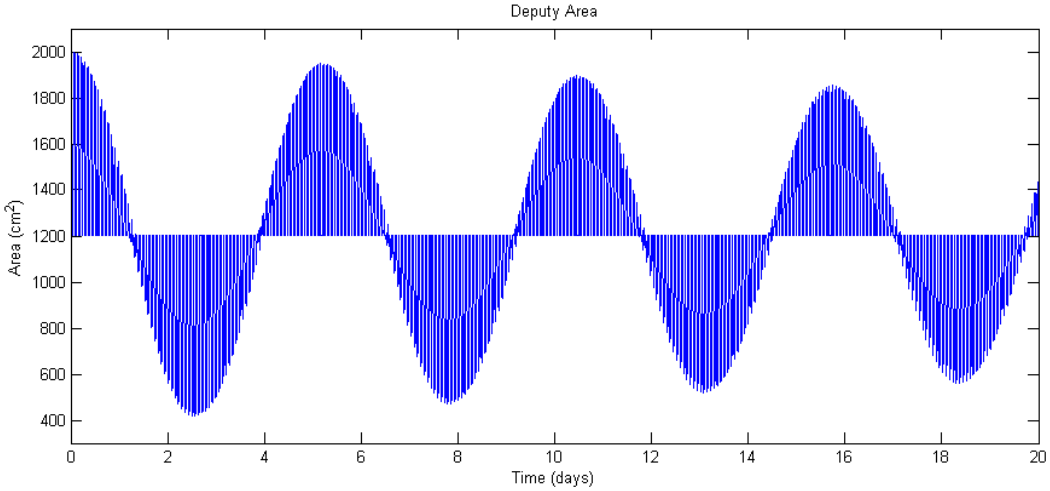


**Figure 10: Variation in Deputy Area over 20 days and showing nominal reset when not in eclipse**

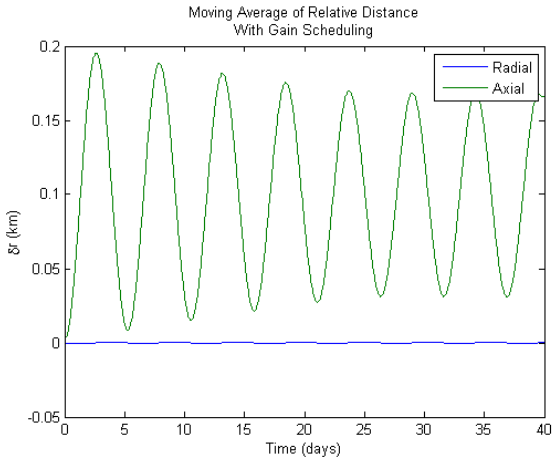


**Figure 11: Gain scheduled control of relative axial position. Desired distance of 0.1 km and only eclipse control**

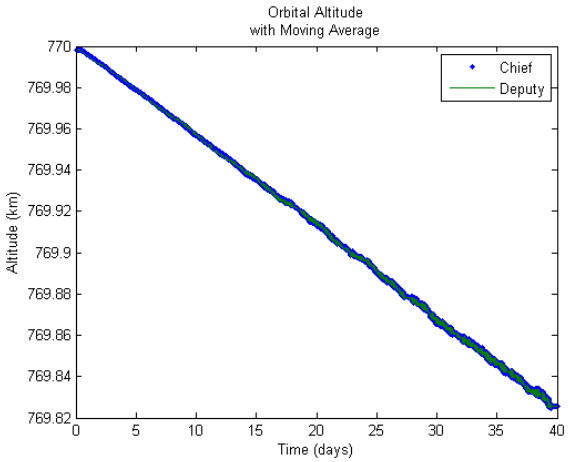


**Figure 12: CubeSat altitude throughout eclipse simulation**

## IV. Conclusion

This paper shows that significant separation and formation flight is possible using rotated control to achieve differential drag. The extent of the capabilities possible must be studied in more depth and for more specific mission requirements. For example, formations with more than two CubeSats add much more complexity and would need to be investigated. Also, altitude may have a significant effect on the duration of formation capable.

In terms of accuracy, future studies should verify that a Variation of Parameters method of orbit integration would eliminate the error. This may allow for a very useful control scheme to be implemented that would create a very stable formation. Other control schemes may be simpler, more effective, or more robust for the given system. Also, an industry standard propagator such as STK or an internal propagator can be used to verify accuracy or provide a higher level of precision.

## Appendix

```
%Skyler Shuford
%Feasibility of CubeSat Formation Flight Using Rotation to Achieve
Differential Drag

clear all;close all;clc

%Assumptions, CubeSat properties, Initial Conditions
date = 'Jan 1 2014';
mu = 398600;
Cd = 2.2;
m = 1;%kg
drdes = .1;%km
kp = 8e-8;
% ki = 1e-14;
ki = 0;
kd = 1.2e-2;
Rc = [0;770+6378;0];
Rd = Rc;%[0;-500-6378;0];
Vc = [-sqrt(mu/norm(Rc))/sqrt(2);0;sqrt(mu/norm(Rc))/sqrt(2)];
Vd = Vc;%[sqrt(mu/norm(Rc))/sqrt(2);0;-sqrt(mu/norm(Rc))/sqrt(2)];
er0 = drdes;
y0 = [Rc;Vc;Rd;Vd;er0];
tspan = [0,60*60*24*200];

opt = odeset('AbsTol',1e-8,'RelTol',1e-8);
[t,y] = ode113(@(t,y) ff_fun(t,y,mu,m,Cd,drdes,kp,ki,kd,date),tspan,y0,opt);
%
% save max600
for i = 1:length(t)

[~,Arc(:,i),Ard(:,i),rc(i),rd(i),dA(i),er(i),er_s(i),ers(i),rhod(i),Vd(:,i)]
= ff_fun(t(i),y(i,:)',mu,m,Cd,drdes,kp,ki,kd,date);
   [Rrel(:,i),Vrel(:,i)] =
relMot(y(i,1:3)',y(i,4:6)',y(i,7:9)',y(i,10:12)',mu);
   vrel(i) = norm(Vrel(:,i));
end

% figure(1)
% plot(t/60/60/24,dTA)
% title('Difference in True Anomaly')
% xlabel('Time (days)')
% ylabel('\delta\theta')
%
% figure(2)
% plot3(y(:,1),y(:,2),y(:,3))
% grid on
% title('Chief Orbit')
% xlabel('(km)')
% ylabel('(km)')
% axis square
% axis equal
% axis([-8000 8000 -8000 8000 -8000 8000])
%
figure(3)
```

```
plot(t/3600/24,rc-6378,t/3600/24,rd-6378)
title('Orbital Altitude')
xlabel('Time (days)')
ylabel('Altitude (km)')

figure(4)
plot(t/3600/24,Rrel(1:2,:))
title('Relative Position @ 770 km')
xlabel('Time (days)')
ylabel('\deltar (km)')
legend('Radial','Axial')
% hold on

figure(5)
plot(t/3600/24,Ard*1e10)
title('Deputy Area')
xlabel('Time (days)')
ylabel('Area (cm^2)')
axis([0 20 3e2 21e2])

% figure(6)
% plot(t/3600/24,dA)
% title('Differential Area')
% xlabel('Time (days)')
% ylabel('Area (km^2)')
% %
figure(7)
plot(t/3600/24,kd*ers,t/3600/24,kp*er,t/3600/24,ki*er_s)
title('Errors')
xlabel('Time (days)')
ylabel('Error Function')
legend('Derivative','Proportional')

% figure(8)
% plot(t/3600/24,Vrel)
% title('Relative Velocity @ 600 km')
% xlabel('Time (days)')
% ylabel('vRel (km/s)')
% legend('Radial','Axial','Out of Plane')

% figure(9)
% plot(t/3600/24,rhod)
% title('Atmospheric Density @ 600 km')
% xlabel('Time (days)')
% ylabel('\rho (kg/km^3)')
% axis([0 1 1.454e-4 1.4544e-4])
% %
% figure(10)
% plot(t/3600/24,eccd)
% title('Eccentricity @ 600 km')
% xlabel('Time (days)')
% ylabel('ecc')
%
% figure(11)
% plot3(Vd(1,:),Vd(2,:),Vd(3,:))
% title('Velocity direction @ 600 km')
```

```
% % xlabel('Time (days)')
% % ylabel('km/s')
%
% figure(12)
% plot(t/3600/24,vdirmag)
% title('Velocity magnitude @ 600 km')
% xlabel('Time (days)')


% vrel(1:8) = vrel(9);
% fit = polyfit(t,vrel',4);
%
% figure(13)
% plot(t/3600/24,fit(1)*t.^4+fit(2)*t.^3+fit(3)*t.^2+fit(4)*t+fit(5))
% hold all
% plot(t/3600/24,vrel)
% title('Magnitude of Relative Velocity')
% hold off

Rrelavgx = moving(Rrel(1,:),1001);
Rrelavgy = moving(Rrel(2,:),1001);
figure(14)
plot(t/3600/24,Rrelavgx,t/3600/24,Rrelavgy)
title('Moving Average of Relative Distance')
xlabel('Time (days)')
ylabel('\deltar (km)')
legend('Radial','Axial')

Ardavg = moving(Ard,1001);
figure(15)
plot(t/3600/24,Ardavg)
title('Moving Average of Deputy Area')
xlabel('Time (days)')
ylabel('Area (km^2)')
axis([0 20 .03e-6 .21e-6])

figure(16)
subplot(2,1,1)
plot(t/3600/24,Ard*1e10)
title('Deputy Area')
xlabel('Time (days)')
ylabel('Area (cm^2)')
axis([0 40 80 160])
subplot(2,1,2)
plot(t/3600/24,Ardavg*1e10)
title('Moving Average of Deputy Area')
xlabel('Time (days)')
ylabel('Area (cm^2)')
axis([0 40 80 160])

rcm = moving(rc,2001);
rdm = moving(rd,2001);
figure(17)
plot(t/3600/24,rcm-6378,'.',t/3600/24,rdm-6378,'r')
title({'Orbital Altitude','with Moving Average'})
```

```
xlabel('Time (days)')
ylabel('Altitude (km)')
legend('Chief','Deputy')



%Skyler Shuford

function [dy, Arc, Ard, rc, rd, dA, er, er_s, ers, rhod, Vd] =
ff_fun(t,y,mu,m,Cd,drdes,kp,ki,kd,date)

Rc = y(1:3);
rc = norm(Rc);
Vc = y(4:6);
vc = norm(Vc);

Rd = y(7:9);
rd = norm(Rd);
Vd = y(10:12);
vd = norm(Vd);

%shadow irrelevant
sc = 1;
sd = sc;

%Only Shadow control
% [sc] = shadow(Rc,Vc,date,3);
% [sd] = shadow(Rd,Vd,date,3);

Agravc = -mu/rc^3*Rc;
Agravd = -mu/rd^3*Rd;

%get rho
hElc = rc-6378;
hEld = rd-6378;

h0 = [0 25 30 40 50 60 70 80 90 100 110 ...
    120 130 140 150 180 200 250 300 350 ...
    400 450 500 600 700 800 900 1000];

rho0 = [1.225 3.899e-2 1.774e-2 3.972e-3 1.057e-3 3.206e-4 ...
    8.770e-5 1.905e-5 3.396e-6 5.297e-7 9.661e-8 2.438e-8 ...
    8.484e-9 3.845e-9 2.070e-9 5.464e-10 2.789e-10 7.248e-11 ...
    2.418e-11 9.518e-12 3.725e-12 1.585e-12 6.967e-13 1.454e-13 ...
    3.614e-14 1.170e-14 5.245e-15 3.019e-15];

H = [7.249 6.349 6.682 7.554 8.382 7.714 6.549 5.799 ...
    5.382 5.877 7.263 9.473 12.636 16.149 22.523 29.740 ...
    39.105 45.546 53.628 53.298 58.515 60.828 63.822 71.835 ...
    88.667 124.64 181.05 268];
k = 1;
if hElc>1000
    k = 28;
else
    while hElc>h0(k)
```

```
        k = k+1;
    end
end
l = 1;
if hEld>1000
    l = 28;
else
    while hEld>h0(l)
        l = l+1;
    end
end

rhoc = rho0(k)*1e9*exp((h0(k) - hElc)/H(k));%kg/km^3
rhod = rho0(l)*1e9*exp((h0(l) - hEld)/H(l));

%Get LVLH
[Rrel,Vrel] = relMot(Rc,Vc,Rd,Vd,mu);

%3U 50% Area
% ArMin = .01e-6;
% ArMax = .01*3e-6*sqrt(2);
% Ar1 = (ArMin+ArMax)/2;

%CubeSat 50% area
ArMin = .01e-6;
ArMax = .01e-6*sqrt(2);
Ar1 = (ArMin+ArMax)/2;

%Colony II 50% area
% ArMin = .01e-6*3;
% ArMax = .01e-6*21;
% Ar1 = (ArMin+ArMax)/2;

%Control
% if sc ~= 0 && sd ~= 0
    er = drdes-Rrel(2);
%     if t>3600*24*60
%         kd = 0;
%     end
%     er_s = 0;
    er_s = y(13);
    ers = -Vrel(2);%
    dA = kp*er-ki*er_s+kd*ers;
    if abs(dA) > Ar1-ArMin
        dA = sign(dA)*(Ar1-ArMin);
    end
    Arc = Ar1;
    Ard = Arc+dA;
% else
%     er_s = NaN;
%     ers = NaN;
%     Arc = Ar1;
%     Ard = Arc;
%     er = 0;
%     dA = 0;
```

```
% end



% %Max Separation or Approach
% Arc = .01e-6;
% Ard = .01e-6*1.7320;

% Separation of tumbling cube
% Arc = .01e-6;
% Ard = .01e-6*1.4472;

% Max sep with only yaw control
% Arc = .01e-6;
% Ard = .01e-6*sqrt(2);


Adragc = -1/2*Cd*Arc/m*rhoc*vc^2*Vc/vc;
Adragd = -1/2*Cd*Ard/m*rhod*vd^2*Vd/vd;

% %thrust test
% Adragc = .00005*Vc/vc;
% Adragd = .00005*Vd/vd;

Ac = Agravc+Adragc;
Ad = Agravd+Adragd;

dy = [Vc;Ac;Vd;Ad;er];
end


function [y]=moving(x,m,fun)
%MOVING will compute moving averages of order n (best taken as odd)
%
%Usage: y=moving(x,n[,fun])
%where x    is the input vector (or matrix) to be smoothed.
%      m    is number of points to average over (best odd, but even works)
%      y    is output vector of same length as x
%      fun  (optional) is a custom function rather than moving averages
%
% Note:if x is a matrix then the smoothing will be done 'vertically'.
%
%
% Example:
%
% x=randn(300,1);
% plot(x,'g.');
% hold on;
% plot(moving(x,7),'k');
% plot(moving(x,7,'median'),'r');
% plot(moving(x,7,@(x)max(x)),'b');
% legend('x','7pt moving mean','7pt moving median','7pt moving
max','location','best')
%
% optimized Aslak Grinsted jan2004
```

```
% enhanced Aslak Grinsted Apr2007


if m==1
    y=x;
    return
end
if size(x,1)==1
    x=x';
end

if nargin<3
    fun=[];
elseif ischar(fun)
    fun=eval(['@(x)' fun '(x)']);
end

if isempty(fun)

    f=zeros(m,1)+1/m;
    n=size(x,1);
    isodd=bitand(m,1);
    m2=floor(m/2);


    if (size(x,2)==1)
        y=filter(f,1,x);
        y=y([zeros(1,m2-1+isodd)+m,m:n,zeros(1,m2)+n]);
    else
        y=filter2(f,x);
        y(1:(m2-~isodd),:)=y(m2+isodd+zeros(m2-~isodd,1),:);
        y((n-m2+1):end,:)=y(n-m2+zeros(m2,1),:);
    end

else
    y=zeros(size(x));
    sx=size(x,2);
    x=[nan(floor(m*.5),sx);x;nan(floor(m*.5),sx)];
    m1=m-1;
    for ii=1:size(y,1);
        y(ii,:)=fun(x(ii+(0:m1),:));
    end

end

return
```

## References

[1]Kumar B. S., NG, A., Yoshihara, K., and De Ruiter, A., "Differential Drag as a Means of Spacecraft Formation Control," *IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS*, Vol. 47, No. 2, 2011, pp. 1125-1135.

[2]MATLAB, Software Package, Ver. 7.10.0 R2010b,The Mathworks Inc., Natick, MA, 2010.

[3]Curtis, H. D., *Orbital Mechanics for Engineering Students*, 2nd ed., Butterworth-Heinemann, Burlington, MA, 2010, Chap. 7.

[4]"moving", MATLAB function, A. Grinsted, 2007

[5]Vallado, D.A., *Fundamentals of Astrodynamics and Applications*. Microcosm, 2007. pp. 303.