Coordinate Transformations for Unsteady Frames

Johnathan Clouse*
University of Colorado, Boulder, CO, 80309

A Frenet frame is a useful coordinate system to describe motion with respect to the motion of a particle. A system utilizing a ground station, mothership, and Micro Aerial Vehicle (MAV) is explored. MAV motion is determined by coordinate transformation to provide position, velocity, and acceleration with respect to the ground station. Error between direct numerical differentiation of the position and rotational kinematics is investigated.

Nomenclature

$\hat{\boldsymbol{t}}$	=	tangent vector of Frenet frame
$\hat{\boldsymbol{b}}$	=	binormal vector of Frenet frame
$\hat{\boldsymbol{n}}$	=	normal vector of Frenet frame
$\boldsymbol{r}$	=	position vector of Micro Aerial Vehicle (MAV) expressed in the ground station (GS) Cartesian frame
$\boldsymbol{r}', \boldsymbol{v}$	=	velocity vector of MAV expressed in the GS Cartesian frame
$\boldsymbol{r}'', \boldsymbol{a}$	=	acceleration vector of MAV expressed in the GS Cartesian frame
t	=	time of measurement
G	=	fundamental metric tensor between two frames
$\boldsymbol{\omega}$	=	rotation rate between two frames
$\boldsymbol{\alpha}$	=	angular acceleration between two frames
$\overline{\boldsymbol{r}}, \overline{\boldsymbol{v}}, \overline{\boldsymbol{a}}$	=	MAV quantities expressed in the mothership (MS) Frenet frame
$\boldsymbol{R}, \boldsymbol{V}, \boldsymbol{A}$	=	MS quantities expressed in the GS Cartesian frame

I. Introduction

THE Frenet frame is a useful set of coordinates centered on a particle. The basis vectors describe the motion of the particle, showing the instantaneous direction of motion and the instantaneous radius of curvature of the path[1]. This project focuses on the use of the Frenet frame on the paths of an unmanned aerial vehicle (UAV) mothership (MS) and Micro Aerial Vehicles (MAVs) deployed by the MS to act as sensory equipment[1]. The MAV motion with respect to the MS are a known quantity. Sensory data is then relayed from the MS to a ground station (GS), so coordinate transformations must be applied to obtain the MAV motion with respect to a Cartesian frame centered on the GS.

The Frenet frame for a particle on a path is obtained by defining the tangent vector $\hat{\boldsymbol{t}}$ as

$$\hat{\boldsymbol{t}} = \frac{\boldsymbol{r}'}{\|\boldsymbol{r}'\|} \tag{1}$$

One can see that the tangent vector is solely in the direction of motion. The binormal vector $\hat{\boldsymbol{b}}$ is defined as

$$\hat{\boldsymbol{b}} = \frac{\boldsymbol{r}' \times \boldsymbol{r}''}{\|\boldsymbol{r}' \times \boldsymbol{r}''\|} \tag{2}$$

Thus, the binormal vector is normal to both the direction of motion and the particle acceleration. It is important to note that this vector cannot be defined when the velocity and acceleration are collinear. Finally, the normal vector $\hat{\boldsymbol{n}}$ is found to complete the three-dimensional coordinate frame by

$$\hat{\boldsymbol{n}} = \hat{\boldsymbol{b}} \times \hat{\boldsymbol{t}} \tag{3}$$

This normal vector is normal to the curve of the path, and so it is useful to find the normal component of a particle's acceleration.

* Graduate Student, Aerospace Engineering Sciences, 1111 Engineering Drive, Boulder, CO, 80309-0429

To find the Frenet basis vectors for a given instant, the first and second derivatives of the position vector must be known. An analytic expression for the position vector will easily yield an analytic expression for these derivatives, but it is not the case when only time-stamped position data are provided. In the latter case, one can forward-differentiate the *n*th data point by

$$x'_n = \frac{x_{n+1} - x_n}{t_{n+1} - t_n} \tag{4}$$

This method will reduce the amount of data points by one for each derivative taken. For example, given five position vectors in time, one will end up with four velocity vectors and three acceleration vectors.

II. Simulation of Known Analytical Path

Consider the path of the MS, with respect to the GS and expressed in a Cartesian frame[1]:

$$\boldsymbol{R}(t) = t\cos t\,\hat{\boldsymbol{\imath}} + t\sin 2t\,\hat{\boldsymbol{\jmath}} + t\hat{\boldsymbol{k}},\ \ t \in [0,2\pi] \tag{5}$$

And the path of an MAV in the MS Frenet frame[1]:

$$\bar{\boldsymbol{r}}(t) = \cos t\,\hat{\boldsymbol{t}} + \sin 2t\,\hat{\boldsymbol{n}} + \cos 2t\,\hat{\boldsymbol{b}},\ \ t \in [0,2\pi] \tag{6}$$

Units are not given, so generic distance units (DU) and time units (TU) are used in the plots.

The derivatives required to obtain the Frenet bases are obtained by numerical forward differentiation, rather than analytically, to allow the script to handle discrete data from another source. For this analytical case, 1000 time-points are used. Figure 1 and Figure 2 show the positions of the mothership and MAV, with their Frenet basis vectors.

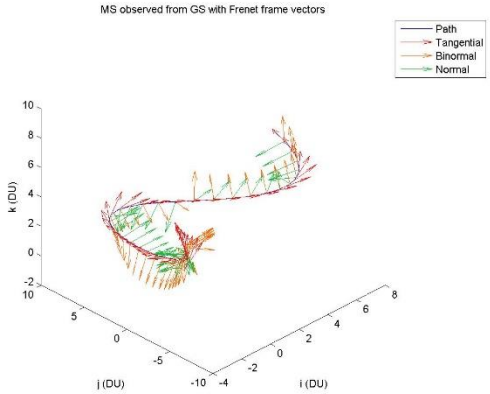

Figure 1:Mothership position wrt GS Cartesian, with its Frenet basis vectors

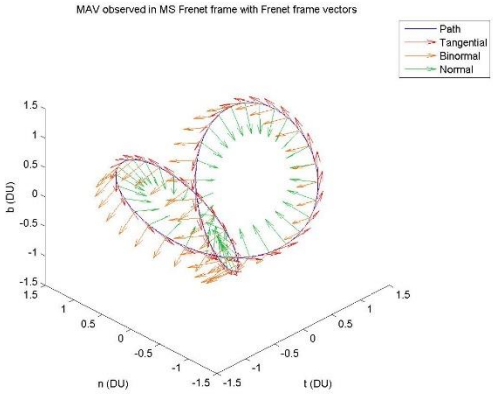

Figure 2: MAV position wrt MS Frenet frame, with its Frenet basis vectors

The mothership's motion experiences an inflection in its path curvature, causing the normal and binormal Frenet vectors to change substantially. The MAV experiences a path with no inflections, so the change in the Frenet vectors is continuous.

The mothership's speed and accelerations with respect to the GS Cartesian frame is shown in Figure 3 and Figure 4 below:

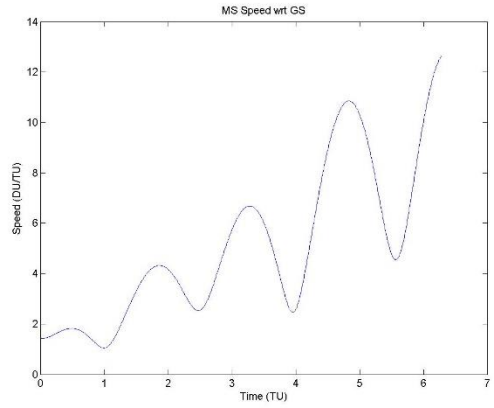

Figure 3: MS speed wrt GS

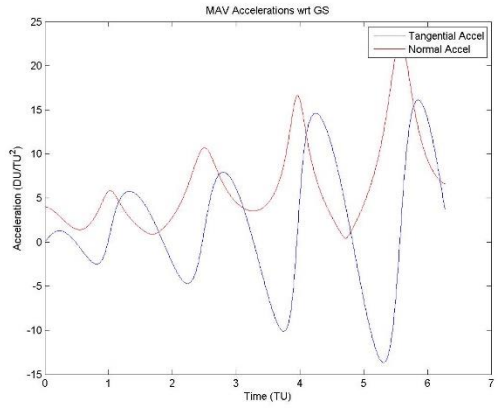

Figure 4: MS normal and tangential accelerations wrt GS

The MAV speed and accelerations with respect to the MS Frenet frame is shown in Figure 5 and Figure 6.

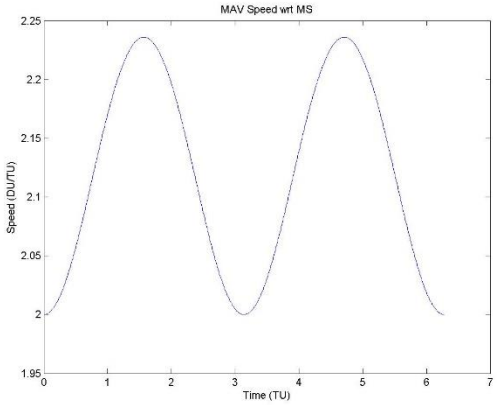

Figure 5: MAV speed wrt MS

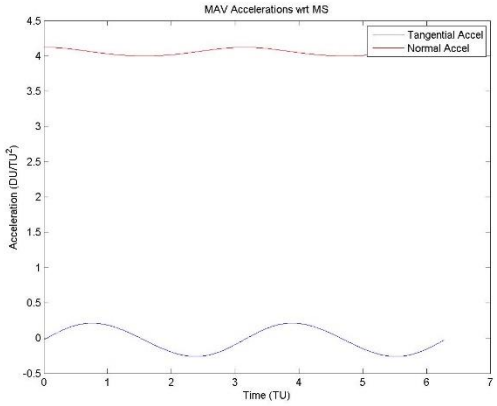

Figure 6: MAV accelerations wrt MS

The fundamental metric tensor between the GS Cartesian and MS Frenet frames is

$$G = \left[g_{ij}\right] = \left[\overline{\boldsymbol{e}}_i \cdot \hat{\boldsymbol{\imath}}_j\right] = [\hat{\boldsymbol{t}}\,\hat{\boldsymbol{n}}\,\hat{\boldsymbol{b}}]^T \qquad (7)$$

where the barred vectors are $\hat{\boldsymbol{t}}, \hat{\boldsymbol{n}}, \hat{\boldsymbol{b}}$ for i=1,2,3 respectively and the Cartesian vectors are $\hat{\boldsymbol{\imath}}, \hat{\boldsymbol{\jmath}}, \hat{\boldsymbol{k}}$ for j=1,2,3 respectively. The position of the MAV in the GS coordinate frame is then represented by

$$\boldsymbol{r} = \boldsymbol{R} + G\overline{\boldsymbol{r}} \qquad (8)$$

The resultant MAV positions are shown in Figure 7. There is a large change in MAV position in the vicinity of the MS inflection point.

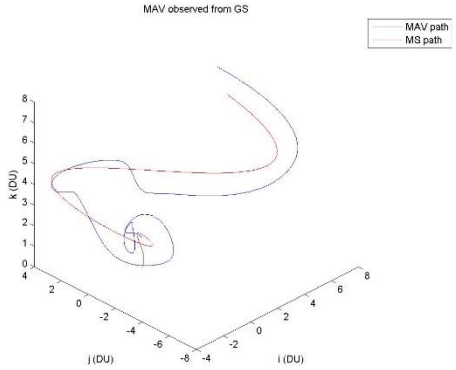

Figure 7: MAV position in GS Cartesian frame

The MAV speed and acceleration with respect to the ground station frame are found by finding the rotational rate of the MS Frenet frame as seen by the GS and using relative motion equations. The rotational rate of one frame with respect to another can be found by[2]

$$\boldsymbol{\omega} = \left(\dot{\psi}\sin\theta\sin\phi + \dot{\theta}\cos\phi\right)\hat{\boldsymbol{\imath}} + \left(-\dot{\psi}\sin\theta\cos\phi + \dot{\theta}\sin\phi\right)\hat{\boldsymbol{\jmath}} + \left(\dot{\psi}\cos\theta + \dot{\phi}\right)\hat{\boldsymbol{k}} \qquad (9)$$

Between the GS Cartesian and MS Frenet frames, and the Euler angles are found by using the fundamental metric tensor[2]

$$\phi = \tan^{-1}\frac{g_{13}}{-g_{23}}, \qquad \theta = \tan^{-1}\frac{\sqrt{1-g_{33}^2}}{g_{33}}, \qquad \psi = \tan^{-1}\frac{g_{31}}{-g_{32}} \qquad (10)$$

The velocity in the GS Cartesian frame is calculated by

$$\boldsymbol{v} = \dot{\boldsymbol{R}} + G\overline{\boldsymbol{v}} + \boldsymbol{\omega}\times G\overline{\boldsymbol{r}} \qquad (11)$$

The acceleration in the GS Cartesian frame is

$$\boldsymbol{a} = \ddot{\boldsymbol{R}} + G\overline{\boldsymbol{a}} + \boldsymbol{\alpha}\times G\overline{\boldsymbol{r}} + 2\boldsymbol{\omega}\times G\overline{\boldsymbol{v}} + \boldsymbol{\omega}\times(\boldsymbol{\omega}\times G\overline{\boldsymbol{r}}) \qquad (12)$$

The angular velocity $\boldsymbol{\alpha}$ is the time-derivative of $\boldsymbol{\omega}$, which is

$$\begin{aligned}\boldsymbol{\alpha} = {} & \left(\ddot{\varphi}\sin\theta\sin\phi + \dot{\varphi}\dot{\theta}\cos\theta\sin\phi + \dot{\varphi}\dot{\phi}\sin\theta\cos\phi + \ddot{\theta}\cos\phi - \dot{\theta}\dot{\phi}\sin\phi\right)\hat{\boldsymbol{\imath}} \\ & + \left(-\ddot{\varphi}\sin\theta\cos\phi - \dot{\varphi}\dot{\theta}\cos\theta\cos\phi + \dot{\varphi}\dot{\phi}\sin\theta\sin\phi + \ddot{\theta}\sin\phi + \dot{\theta}\dot{\phi}\cos\phi\right)\hat{\boldsymbol{\jmath}} \\ & + \left(\ddot{\varphi}\cos\theta - \dot{\varphi}\dot{\theta}\sin\theta + \ddot{\phi}\right)\hat{\boldsymbol{k}}\end{aligned} \qquad (13)$$

The application of this kinematic method can be seen in Figure 8 and Figure 9 for the speed calculation in both the rotating coordinate method and direct numerical differentiation of $\boldsymbol{r}$. Figure 10 shows the normal and tangential accelerations of the MAV in the GS Cartesian frame as calculated from the above kinematic equations; and Figure 11 shows the error between the kinematic solution and direct numerical differentiation of $\boldsymbol{r}$.

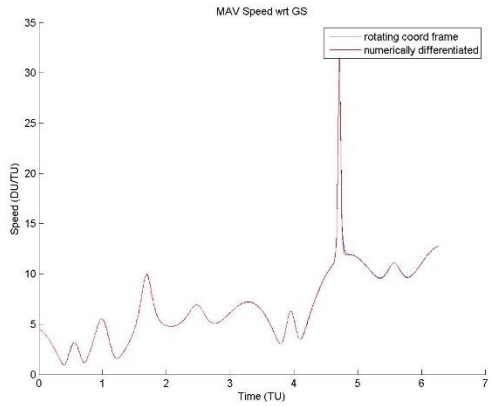

Figure 8:MAV speed in GS Cartesian frame

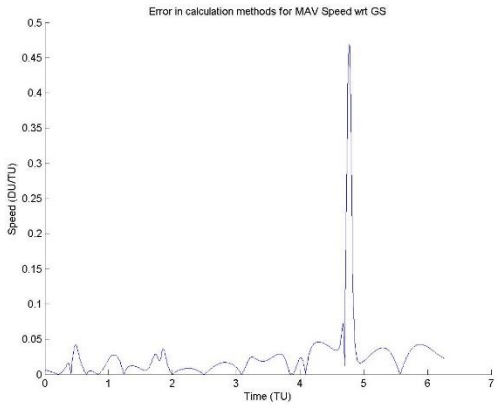

Figure 9: Error between methods for MAV speed wrt GS

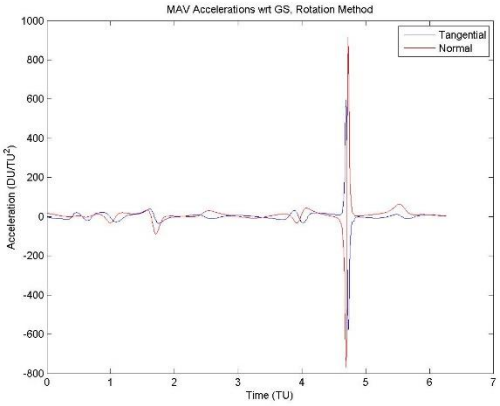

Figure 10:MAV accelerations in the GS Cartesian frame

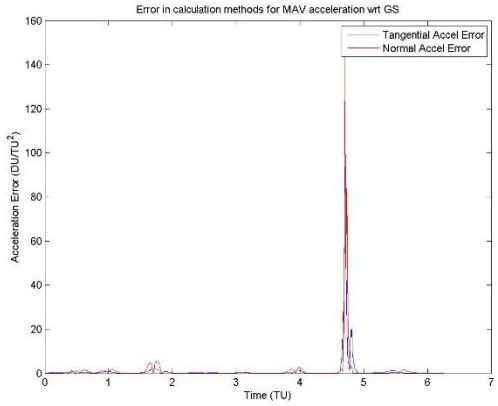

Figure 11: Error between methods for MAV accelerations.

From the above acceleration plots, it is clear that the inflection point of the MS affects the acceleration of the MAV viewed by the GS. The fast change in the mothership's normal and binormal Frenet vectors require such a change in the MAV's position as well, causing the spike in acceleration. The error between the two methods in acceleration calculation are also affected by this discontinuity.

III. Discrete Data for MAV Position

Now, a data file with MAV positions in the MS Frenet frame is used to provide MAV position data; MS position is the same as the previous section. The data file lists 100 evenly-spaced data points for the same timescale as the last section. Figure 12 shows the MAV position in the MS Frenet frame, with its own Frenet vectors. Figure 13 shows the MAV position in the GS Cartesian frame. The speed and accelerations of the MAV can be seen in Figure 14 and Figure 15.

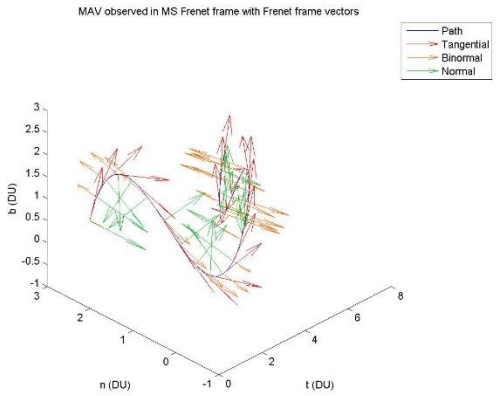

Figure 12: MAV position in MS Frenet frame, with Frenet vectors

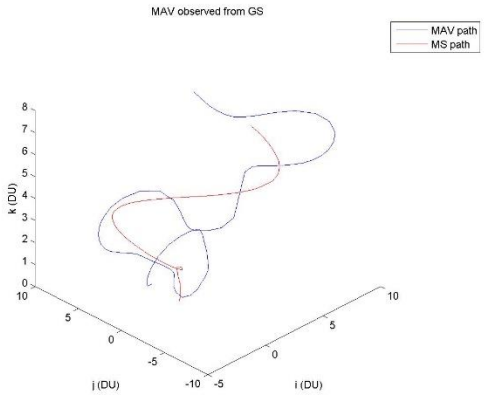

Figure 13: MAV position in GS Cartesian frame

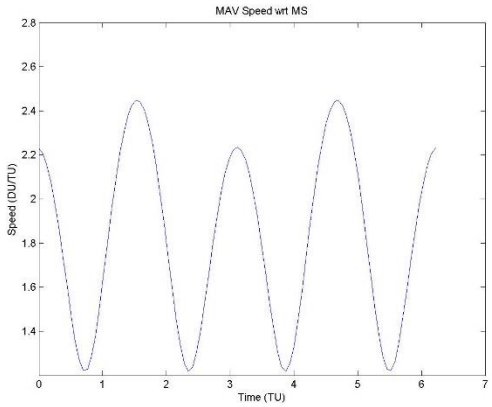

Figure 14: MAV speed wrt MS

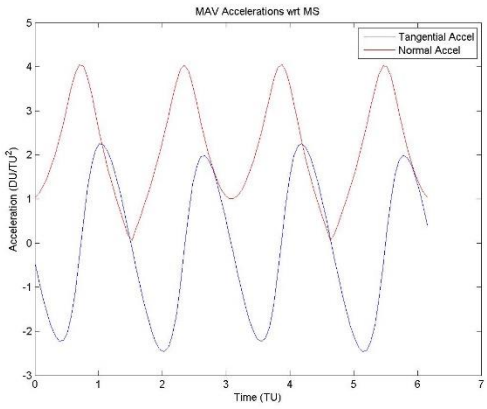

Figure 15: MAV accelerations wrt MS

The curvature of the MAV's path in the MS Frenet frame experiences inflections in its own path this time. The MAV position in the GS Cartesian frame again experiences a large position change in the vicinity of the MS inflection.

The speed of the MAV with respect to the GS is found kinematically and numerically, and the results are displayed in Figure 16. The error between the two methods' results are shown in Figure 17. The kinematically-computed accelerations of the MAV with respect to the GS are seen in Figure 18, and the numerically-differentiated result is shown in Figure 19. Figure 20 shows the error between the acceleration computations.

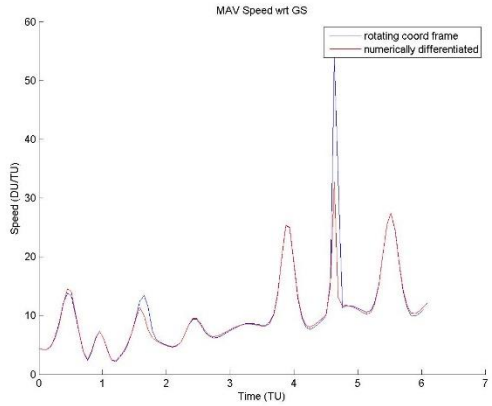

Figure 16: MAV speed wrt GS

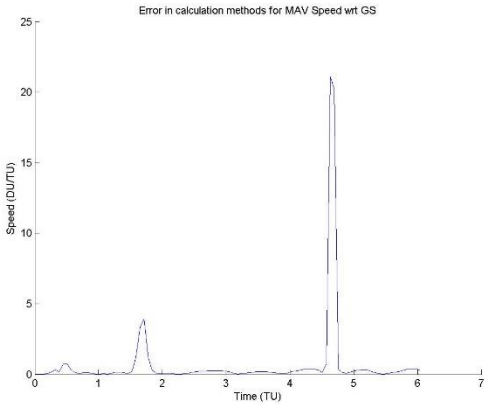

Figure 17: Error between methods for MAV speed wrt GS

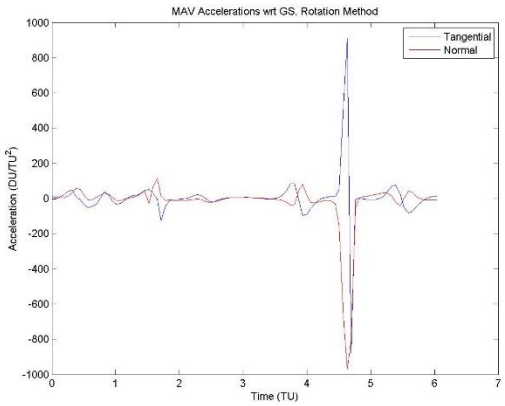

Figure 18: MAV accelerations wrt GS Cartesian frame, rotation method

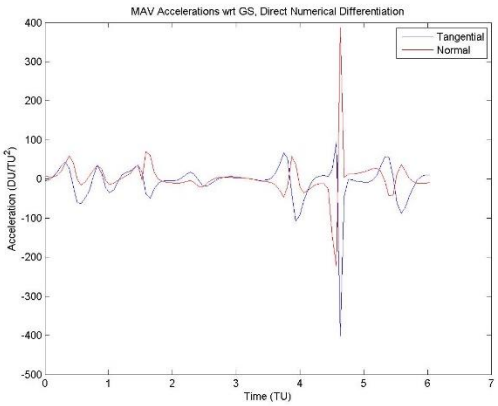

Figure 19: MAV accelerations wrt GS Cartesian frame, direct numerical differentiation

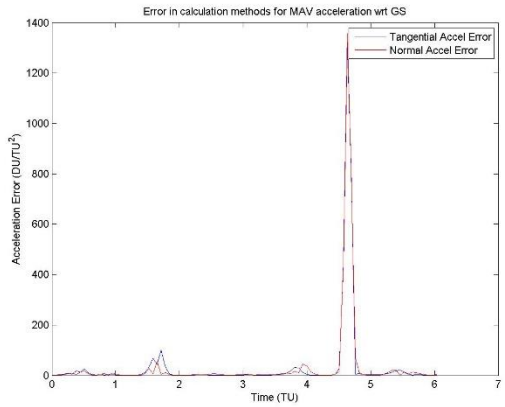

Figure 20: Error between methods for MAV accelerations.

The error between the kinematic and numerical differentiation methods does not track as well as before. This is due in part to the larger time step between the provided positions of the MAV, giving the numerical differentiation less granularity. The MS inflection point exacerbates the spike in the accelerations and the error calculation in the region. In fact, the accelerations are shown to increase in opposite directions between the two methods.

IV. Conclusion

The Frenet frame is a useful way to describe things with respect to a particle in motions, but there are caveats to its usage. First, the frame is undefined if the velocity and acceleration are collinear. One may work around this by fixing the basis vectors when the velocity and acceleration are nearly collinear, but care should be take that this does not cause different problems. Second, motion of a vehicle maintaining some position in the Frenet of another particle is at the mercy of the second particle's motion. Inflections in the path could cause unsafe accelerations in the vehicle, or cause it to diverge from the desired path as it corrects its motion.

While a MAV may experience the aforementioned issues, an orbital mothership may be a better-suited use for the Frenet frame. Since two-body motion results in a continuously curved path, no inflections would be experienced and a secondary vehicle's motion could easily be described.

Direct numerical differentiation of the MAV with respect to the ground station tracked well with rotational kinematics solution, but not when there were large changes in acceleration. Directly sensing the Euler angles would improve the kinematic solution's performance. A lower timestep between the data points would also help drive the error down. However, the kinematic solution was computationally costly to solve, compared to the direct differentiation.

References

[1]Hussein, M., "Project: Coordinate Transformations for Unsteady Frames," Fall 2016.
[2]Hussein, M., "Project Supplement: Rotating Coordinate Frames: Euler Angles," Fall 2016.

Appendix

MATLAB code for the project follows. A significant amount of space was taken by the plot generation and saving.

```
%% ASEN 5227 Project
% John Clouse
%% Initialize
clear all
close all

hw_pub.figWidth = 1120; % pixels
hw_pub.figHeight = 840; % pixels
hw_pub.figPosn = [0, 0, hw_pub.figWidth, hw_pub.figHeight];

scenarios = {'Part 1', 'Part 2'};

% Some color definitions
orange = [1.0,0.4,0.0];
lgreen = [20 187 51] ./ 255;

%% Loop through the analytical and discreet scenarios
% The mothership calculations are only done on the first part
% The big thing to remember here is that only forward differentiation is
% used, so each differentiation has one fewer element than the diff'ed
% quantity.
for scenario = scenarios

% get time for the analysis.
if strcmp(scenario, 'Part 1')
    num_pts = 1000;
    time = linspace(0,2*pi,num_pts);
else
    mav_data = dlmread('mav_data_txt.txt');
    time = mav_data(:,4)';
    num_pts = length(time);
end
t_diffs = (time(2:end) - time(1:end-1));
R = [time.*cos(time); time.*sin(2*time); time];
V = [(cos(time)-time.*sin(time)); ...
    (sin(2*time)+2*time.*cos(2*time));...
    1*ones(1,num_pts)];
A = [(-2*sin(time)-time.*cos(time));...
    (4*cos(2*time)-4*time.*sin(2*time));...
    zeros(1,num_pts)];

if strcmp(scenario, 'Part 1')
    r_bar = [cos(time); sin(2*time); cos(2*time)]; %Frenet
    v_bar = [-sin(time); 2*cos(2*time); -2*sin(2*time)];
    a_bar = [-cos(time); -4*sin(2*time); -4*cos(2*time)];
else
    r_bar = mav_data(:,1:3)';
end

% Forward differentiation
% Mothership velocity
V_forward = forward_diff(R,time);
figure('Position', hw_pub.figPosn);
hold on
plot(abs(V(1,1:end-1) - V_forward(1,:)));
plot(abs(V(2,1:end-1) - V_forward(2,:)),'r');
plot(abs(V(3,1:end-1) - V_forward(3,:)),'k');
title('MS Velocity: Error Between Analytical and Numerical Solution')

% Mothership acceleration
A_forward = forward_diff(V_forward,time);
figure('Position', hw_pub.figPosn);
hold on
plot(abs(A(1,1:end-2) - A_forward(1,:)));
plot(abs(A(2,1:end-2) - A_forward(2,:)),'r');
plot(abs(A(3,1:end-2) - A_forward(3,:)),'k');
title('MS Acceleration: Error Between Analytical and Numerical Solution')

v_bar_forward = forward_diff(r_bar,time);
a_bar_forward = forward_diff(v_bar_forward,time);
if strcmp(scenario, 'Part 1')
    % MAV velocity
    figure('Position', hw_pub.figPosn);
    hold on
    plot(abs(v_bar(1,1:end-1) - v_bar_forward(1,:)));
    plot(abs(v_bar(2,1:end-1) - v_bar_forward(2,:)),'r');
    plot(abs(v_bar(3,1:end-1) - v_bar_forward(3,:)),'k');
    title('MAV Velocity: Error Between Analytical and Numerical Solution')

    % MAV acceleration
    figure('Position', hw_pub.figPosn);
    hold on
    plot(abs(a_bar(1,1:end-2) - a_bar_forward(1,:)));
    plot(abs(a_bar(2,1:end-2) - a_bar_forward(2,:)),'r');
    plot(abs(a_bar(3,1:end-2) - a_bar_forward(3,:)),'k');
    title(['MAV Acceleration:'...
```

```
        'Error Between Analytical and Numerical Solution'])
end

% Mothership's local Frenet frame
t = zeros(3,num_pts);
b = zeros(3,num_pts);
n = zeros(3,num_pts);
for ii = 1:num_pts-2
    % Analytically differentiated
%     t(:,ii) = V(:,ii)/norm(V(:,ii));
%     b(:,ii) = cross(V(:,ii),A(:,ii))/norm(cross(V(:,ii),A(:,ii)));
%     n(:,ii) = cross(b(:,ii),t(:,ii));
    % Numerically differentiated
    t(:,ii) = V_forward(:,ii)/norm(V_forward(:,ii));
    b(:,ii) = cross(V_forward(:,ii),A_forward(:,ii))...
        /norm(cross(V_forward(:,ii),A_forward(:,ii)));
    n(:,ii) = cross(b(:,ii),t(:,ii));
end

% plot the MS path
if strcmp(scenario, 'Part 1')
    figure('Position', hw_pub.figPosn);
    plot3(R(1,:),R(2,:),R(3,:));
    xlabel('i (DU)'); ylabel('j (DU)'); zlabel('k (DU)')
    hold on
    plot_idx = [1:20:num_pts num_pts];
    quiver3(R(1,plot_idx),R(2,plot_idx),R(3,plot_idx),...
        t(1,plot_idx),t(2,plot_idx),t(3,plot_idx),'r');
    quiver3(R(1,plot_idx),R(2,plot_idx),R(3,plot_idx),...
        b(1,plot_idx),b(2,plot_idx),b(3,plot_idx),'color',orange);
    quiver3(R(1,plot_idx),R(2,plot_idx),R(3,plot_idx),...
        n(1,plot_idx),n(2,plot_idx),n(3,plot_idx),'color',lgreen);
    xlabel('i (DU)'); ylabel('j (DU)'); zlabel('k (DU)')
    title('MS observed from GS with Frenet frame vectors')
    view([1,0,0])
    saveas(gcf, ['Figures\' 'MS_JK'],'jpg')
    view([0,0,1])
    saveas(gcf, ['Figures\' 'MS_IJ'],'jpg')
    view([-1,-1,1])
    legend({'Path','Tangential','Binormal','Normal'});
    saveas(gcf, ['Figures\' 'MS_ISO'],'jpg')
end

% MAV's Frenet frame expressed in the MS Frenet frame
t_MAV = zeros(3,num_pts);
b_MAV = zeros(3,num_pts);
n_MAV = zeros(3,num_pts);
for ii = 1:num_pts-2 % missing 2 points at end from 2 forward diffs
    t_MAV(:,ii) = v_bar_forward(:,ii)/norm(v_bar_forward(:,ii));
    b_MAV(:,ii) = cross(v_bar_forward(:,ii),a_bar_forward(:,ii))...
        /norm(cross(v_bar_forward(:,ii),a_bar_forward(:,ii)));
    n_MAV(:,ii) = cross(b_MAV(:,ii),t_MAV(:,ii));
end

% Plot MAV path
if strcmp(scenario, 'Part 1')
    plot_idx = [1:20:num_pts num_pts];
else
    plot_idx=[1:4:num_pts num_pts];
end
figure('Position', hw_pub.figPosn);
plot3(r_bar(1,:),r_bar(2,:),r_bar(3,:));
xlabel('i (DU)'); ylabel('j (DU)'); zlabel('k (DU)')
hold on
quiver3(r_bar(1,plot_idx),r_bar(2,plot_idx),r_bar(3,plot_idx),...
    t_MAV(1,plot_idx),t_MAV(2,plot_idx),t_MAV(3,plot_idx),'r');
quiver3(r_bar(1,plot_idx),r_bar(2,plot_idx),r_bar(3,plot_idx),...
    b_MAV(1,plot_idx),b_MAV(2,plot_idx),b_MAV(3,plot_idx),'color',orange);
quiver3(r_bar(1,plot_idx),r_bar(2,plot_idx),r_bar(3,plot_idx),...
    n_MAV(1,plot_idx),n_MAV(2,plot_idx),n_MAV(3,plot_idx),'color',lgreen);
xlabel('t (DU)'); ylabel('n (DU)'); zlabel('b (DU)')
title('MAV observed in MS Frenet frame with Frenet frame vectors')
view([1,0,0])
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_NB'],'jpg')
view([0,0,1])
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_TN'],'jpg')
view([-1,-1,1])
legend({'Path','Tangential','Binormal','Normal'});
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_ISO'],'jpg')

if strcmp(scenario, 'Part 1')
    % Mothership normal, tangent accels wrt GS
    MS_accel_tangent = zeros(1,length(A_forward));
    MS_accel_normal = zeros(1,length(A_forward));
    MS_accel_bi = zeros(1,length(A_forward));
    for ii = 1:length(A_forward)
        MS_accel_tangent(ii) = dot(t(:,ii),A_forward(:,ii));
        MS_accel_normal(ii) = dot(n(:,ii),A_forward(:,ii));
        MS_accel_bi(ii) = dot(b(:,ii),A_forward(:,ii));
    end

%     figure('Position', hw_pub.figPosn);
%     plot(MS_accel_tangent)
```

```
%     title('MS Tangential Acceleration wrt GS')
%     figure('Position', hw_pub.figPosn);
%     plot(MS_accel_normal)
%     title('MS Normal Acceleration wrt GS')
%     figure('Position', hw_pub.figPosn);
%     plot(MS_accel_bi)
%     title('MS Binormal Acceleration wrt GS')

    % Plot the accelerations, speed of MS
    figure('Position', hw_pub.figPosn);
    plot(time(1:end-2),MS_accel_tangent)
    hold on
    plot(time(1:end-2),MS_accel_normal,'r')
    title('MAV Accelerations wrt GS')
    xlabel('Time (TU)'); ylabel('Acceleration (DU/TU^2)')
    legend('Tangential Accel', 'Normal Accel')
    saveas(gcf, ['Figures\' 'MS_Accels'],'jpg')

    figure('Position', hw_pub.figPosn);
    plot(time(1:end-1),sqrt(sum(V_forward.^2,1)))
    title('MS Speed wrt GS')
    xlabel('Time (TU)'); ylabel('Speed (DU/TU)')
    saveas(gcf, ['Figures\' 'MS_Speed'],'jpg')

end

% MAV normal, tangent accels wrt MS
MAV_accel_tangent = zeros(1,length(a_bar_forward));
MAV_accel_normal = zeros(1,length(a_bar_forward));
MAV_accel_bi = zeros(1,length(a_bar_forward));
for ii = 1:length(a_bar_forward)
    MAV_accel_tangent(ii) = dot(t_MAV(:,ii),a_bar_forward(:,ii));
    MAV_accel_normal(ii) = dot(n_MAV(:,ii),a_bar_forward(:,ii));
    MAV_accel_bi(ii) = dot(b_MAV(:,ii),a_bar_forward(:,ii));
end

% figure('Position', hw_pub.figPosn);
% plot(MAV_accel_tangent)
% title('MAV Tangential Acceleration wrt MS')
% figure('Position', hw_pub.figPosn);
% plot(MAV_accel_normal)
% title('MAV Normal Acceleration wrt MS')
% figure('Position', hw_pub.figPosn);
% plot(MAV_accel_bi)
% title('MAV Binormal Acceleration wrt MS')

% Plot the accelerations, speed of MAV
figure('Position', hw_pub.figPosn);
plot(time(1:end-2),MAV_accel_tangent)
hold on
plot(time(1:end-2),MAV_accel_normal,'r')
title('MAV Accelerations wrt MS')
xlabel('Time (TU)'); ylabel('Acceleration (DU/TU^2)')
legend('Tangential Accel', 'Normal Accel')
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_Accels'],'jpg')

figure('Position', hw_pub.figPosn);
plot(time(1:end-1),sqrt(sum(v_bar_forward.^2,1)))
title('MAV Speed wrt MS')
xlabel('Time (TU)'); ylabel('Speed (DU/TU)')
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_Speed'],'jpg')

% MAV wrt GS
% The euler angles are found from G
% These are plugged into the w/alpha calcs
% nans are used to init so that the zeros don't drag down the plots
r = nan(3,length(t));
v = nan(3,length(t));
a = nan(3,length(t));
phi = zeros(1,length(t));
theta = zeros(1,length(t));
psi = zeros(1,length(t));
phi_dot = zeros(1,length(t));
theta_dot = zeros(1,length(t));
psi_dot = zeros(1,length(t));
phi_dotdot = zeros(1,length(t));
theta_dotdot = zeros(1,length(t));
psi_dotdot = zeros(1,length(t));
w_MS_Frenet_wrt_GS = zeros(3,length(t));
w_MS_Frenet_wrt_body = zeros(3,length(t));
alpha_MS_Frenet_wrt_body = zeros(3,length(t));
alpha_MS_Frenet_wrt_GS = zeros(3,length(t));
% MAV's Frenet frame in GS cartesian
t_MAV_GS = zeros(3,num_pts);
b_MAV_GS = zeros(3,num_pts);
n_MAV_GS = zeros(3,num_pts);
for ii = 1:length(t) - 2 % due to forward diff
    rot = [t(:,ii)';n(:,ii)';b(:,ii)'];
    % Orthogonal transformation, so the inverse is the transpose.
    G_MS2GS = rot';
    r(:,ii) = R(:,ii)+G_MS2GS*r_bar(:,ii);
    phi(ii) = atan2(G_MS2GS(1,3),-G_MS2GS(2,3));
```

```
    theta(ii) = atan2(sqrt(1-G_MS2GS(3,3)^2),G_MS2GS(3,3));
    psi(ii) = atan2(G_MS2GS(3,1),G_MS2GS(3,2));

    % Unroll the angles so we have continuous derivatives
    if phi(ii) < 0
        phi(ii) = phi(ii) + 2*pi;
    end
    if theta(ii) < 0
        theta(ii) = theta(ii) + 2*pi;
    end
    if psi(ii) < 0
        psi(ii) = psi(ii) + 2*pi;
    end
    if ii >= 2 && abs(phi(ii)-phi(ii-1)) > pi
        phi(ii) = phi(ii) + 2*pi;
    end
    if ii >= 2 && abs(theta(ii)-theta(ii-1)) > pi
        theta(ii) = theta(ii) + 2*pi;
    end
    if ii >= 2 && abs(psi(ii)-psi(ii-1)) > pi
        psi(ii) = psi(ii) + 2*pi;
    end
end
% Euler angle derivs
for ii = 1:length(t) - 1 % due to forward diff
        phi_dot(ii) = (phi(ii+1)-phi(ii))/t_diffs(ii);
        theta_dot(ii) = (theta(ii+1)-theta(ii))/t_diffs(ii);
        psi_dot(ii) = (psi(ii+1)-psi(ii))/t_diffs(ii);
end
% Euler double-derivs + body rates/accel
for ii = 1:length(t) - 4 % due to forward diff
    rot = [t(:,ii)';n(:,ii)';b(:,ii)'];
    % Orthogonal transformation, so the inverse is the transpose.
    G_MS2GS = rot';

    % Euler angle double-derivs
    phi_dotdot(ii) = (phi_dot(ii+1)-phi_dot(ii))/t_diffs(ii);
    theta_dotdot(ii) = (theta_dot(ii+1)-theta_dot(ii))/t_diffs(ii);
    psi_dotdot(ii) = (psi_dot(ii+1)-psi_dot(ii))/t_diffs(ii);

    % MS frame rot rate in a couple systems
    w_MS_Frenet_wrt_GS(1,ii) = psi_dot(ii)*sin(theta(ii))*sin(phi(ii)) ...
        + theta_dot(ii)*cos(phi(ii));
    w_MS_Frenet_wrt_GS(2,ii) = -psi_dot(ii)*sin(theta(ii))*cos(phi(ii)) ...
        + theta_dot(ii)*sin(phi(ii));
    w_MS_Frenet_wrt_GS(3,ii) = psi_dot(ii)*cos(theta(ii))+phi_dot(ii);
    w_MS_Frenet_wrt_body(1,ii) = phi_dot(ii)*sin(theta(ii))*sin(psi(ii))...
        +theta_dot(ii)*cos(psi(ii));
    w_MS_Frenet_wrt_body(2,ii) = phi_dot(ii)*sin(theta(ii))*cos(psi(ii))...
        -theta_dot(ii)*sin(psi(ii));
    w_MS_Frenet_wrt_body(3,ii) = phi_dot(ii)*cos(theta(ii))+psi_dot(ii);
    % The MAV velocity in GS calculation
    v(:,ii) = V(:,ii) + G_MS2GS*(v_bar_forward(:,ii) ...
        + cross(w_MS_Frenet_wrt_body(:,ii),r_bar(:,ii)));

    % MS frame rot rate-rate in a couple systems
    alpha_MS_Frenet_wrt_body(1,ii) = ...
        phi_dotdot(ii)*sin(theta(ii))*sin(psi(ii)) ...
        + phi_dot(ii)*theta_dot(ii)*cos(theta(ii))*sin(psi(ii))...
        + phi_dot(ii)*psi_dot(ii)*sin(theta(ii))*cos(psi(ii))...
        + theta_dotdot(ii)*cos(psi(ii)) ...
        - theta_dot(ii)*psi_dot(ii)*sin(psi(ii));
    alpha_MS_Frenet_wrt_body(2,ii) = ...
        phi_dotdot(ii)*sin(theta(ii))*cos(psi(ii)) ...
        + phi_dot(ii)*theta_dot(ii)*cos(theta(ii))*cos(psi(ii))...
        - phi_dot(ii)*psi_dot(ii)*sin(theta(ii))*sin(psi(ii))...
        - theta_dotdot(ii)*sin(psi(ii)) ...
        - theta_dot(ii)*psi_dot(ii)*cos(psi(ii));
    alpha_MS_Frenet_wrt_body(3,ii) = ...
        phi_dotdot(ii)*cos(theta(ii)) ...
        - phi_dot(ii)*theta_dot(ii)*sin(theta(ii))...
        + psi_dotdot(ii);

    alpha_MS_Frenet_wrt_GS(1,ii) = ...
        psi_dotdot(ii)*sin(theta(ii))*sin(phi(ii)) ...
        + psi_dot(ii)*theta_dot(ii)*cos(theta(ii))*sin(phi(ii))...
        + psi_dot(ii)*phi_dot(ii)*sin(theta(ii))*cos(phi(ii))...
        + theta_dotdot(ii)*cos(phi(ii)) ...
        - theta_dot(ii)*phi_dot(ii)*sin(phi(ii));
    alpha_MS_Frenet_wrt_GS(2,ii) = ...
        -psi_dotdot(ii)*sin(theta(ii))*cos(phi(ii)) ...
        - psi_dot(ii)*theta_dot(ii)*cos(theta(ii))*cos(phi(ii))...
        + psi_dot(ii)*phi_dot(ii)*sin(theta(ii))*sin(phi(ii))...
        + theta_dotdot(ii)*sin(phi(ii)) ...
        + theta_dot(ii)*phi_dot(ii)*cos(phi(ii));
    alpha_MS_Frenet_wrt_GS(3,ii) = ...
        psi_dotdot(ii)*cos(theta(ii)) ...
        - psi_dot(ii)*theta_dot(ii)*sin(theta(ii))...
        + phi_dotdot(ii);

%     a(:,ii) = A(:,ii) + G_MS2GS*(a_bar_forward(:,ii) ...
%         + cross(alpha_MS_Frenet_wrt_body(:,ii),r_bar(:,ii))...
%         + 2*cross(w_MS_Frenet_wrt_body(:,ii),v_bar_forward(:,ii))...
```

```
%         + cross(w_MS_Frenet_wrt_body(:,ii),...
%                 cross(w_MS_Frenet_wrt_body(:,ii),r_bar(:,ii))));
    % The MAV accel in GS
    a(:,ii) = A(:,ii) + G_MS2GS*a_bar_forward(:,ii) ...
        + cross(alpha_MS_Frenet_wrt_GS(:,ii),G_MS2GS*r_bar(:,ii))...
        + 2*cross(w_MS_Frenet_wrt_GS(:,ii),G_MS2GS*v_bar_forward(:,ii))...
        + cross(w_MS_Frenet_wrt_GS(:,ii),...
              cross(w_MS_Frenet_wrt_GS(:,ii),G_MS2GS*r_bar(:,ii)));

    % MAV's Frenet frame in GS cartesian
    t_MAV_GS(:,ii) = v(:,ii)/norm(v(:,ii));
    b_MAV_GS(:,ii) = cross(v(:,ii),a(:,ii))...
        /norm(cross(v(:,ii),a(:,ii)));
    n_MAV_GS(:,ii) = cross(b_MAV(:,ii),t_MAV(:,ii));

end
% Plot the MAV position in GS Cartesian
figure('Position', hw_pub.figPosn)
plot3(r(1,:),r(2,:),r(3,:));
hold on
plot3(R(1,:),R(2,:),R(3,:),'r');
xlabel('i (DU)'); ylabel('j (DU)'); zlabel('k (DU)')
title('MAV observed from GS')
view([1,0,0])
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_JK'],'jpg')
view([0,0,1])
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_')) '_MAV_IJ'],'jpg')
view([-1,-1,1])
legend({'MAV path','MS path'});
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_GS_ISO'],'jpg')

% Plot the velocity component err
v_forward = forward_diff(r,time);
figure('Position', hw_pub.figPosn)
hold on
plot(time(1:end-1),abs(v(1,1:end-1) - v_forward(1,:)));
plot(time(1:end-1),abs(v(2,1:end-1) - v_forward(2,:)),'r');
plot(time(1:end-1),abs(v(3,1:end-1) - v_forward(3,:)),'k');
title(['MAV Velocity wrt GS: Error Between Euler-Angle-Rotation and '...
    'Numerical Differentiation Solution'])
legend('v_x error', 'v_y error', 'v_z error');
xlabel('Time'); ylabel('Velocity Error');
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_vel_err'],'jpg')

% Plot the MAV speed wrt GS
figure('Position', hw_pub.figPosn)
hold on
rotated_speed = sqrt(sum(v.*v,1));
nd_speed = sqrt(sum(v_forward.*v_forward,1));
plot(time(1:length(rotated_speed)),rotated_speed);
plot(time(1:length(v_forward)),nd_speed,'r');
title('MAV Speed wrt GS')
xlabel('Time (TU)'); ylabel('Speed (DU/TU)');
legend('rotating coord frame', 'numerically differentiated')
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_GS_speed'],'jpg')

% Plot the MAV speed err between the two methods
figure('Position', hw_pub.figPosn)
hold on
plot(time(1:length(nd_speed)),...
    abs(nd_speed-rotated_speed(1:length(nd_speed))));
% plot(time(1:length(v_forward)),sqrt(sum(v_forward.*v_forward,1)),'r');
title('Error in calculation methods for MAV Speed wrt GS')
xlabel('Time (TU)'); ylabel('Speed (DU/TU)');
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_GS_speed_err'],'jpg')

% Compute the MAV Frenet frame in GS Cartesian
a_forward = forward_diff(v_forward,time(1:end-1));
t_MAV_GS_ND = zeros(3,length(a_forward));
b_MAV_GS_ND = zeros(3,length(a_forward));
n_MAV_GS_ND = zeros(3,length(a_forward));
for ii = 1:length(a_forward)
    t_MAV_GS_ND(:,ii) = v_forward(:,ii)/norm(v_forward(:,ii));
    b_MAV_GS_ND(:,ii) = cross(v_forward(:,ii),a_forward(:,ii))...
        /norm(cross(v_forward(:,ii),a_forward(:,ii)));
    n_MAV_GS_ND(:,ii) = cross(b_MAV(:,ii),t_MAV(:,ii));

end
% Plot the kinematic-method results
figure('Position', hw_pub.figPosn)
plot(time,sum(t_MAV_GS.*a,1))
hold on
plot(time,sum(n_MAV_GS.*a,1),'r')
% plot(time,sum(b_MAV_GS.*a,1))
legend('Tangential','Normal')
xlabel('Time (TU)');ylabel('Acceleration (DU/TU^2)');
title('MAV Accelerations wrt GS, Rotation Method')
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_rot_accel'],'jpg')
```

```
% Plot the ND results
figure('Position', hw_pub.figPosn)
plot(time(1:end-2),sum(t_MAV_GS_ND.*a_forward,1))
hold on
plot(time(1:end-2),sum(n_MAV_GS_ND.*a_forward,1),'r')
legend('Tangential','Normal')
xlabel('Time (TU)');ylabel('Acceleration (DU/TU^2)');
title('MAV Accelerations wrt GS, Direct Numerical Differentiation')
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_nd_accel'],'jpg')

% Plot the error between the two methods
figure('Position', hw_pub.figPosn)
tan_diff = sum(t_MAV_GS.*a,1);
tan_diff = abs(tan_diff(1:end-2)-sum(t_MAV_GS_ND.*a_forward,1));
normal_diff = sum(n_MAV_GS.*a,1);
normal_diff = abs(normal_diff(1:end-2)-sum(n_MAV_GS_ND.*a_forward,1));
plot(time(1:length(tan_diff)),tan_diff)
hold on
plot(time(1:length(normal_diff)),normal_diff,'r')
title('Error in calculation methods for MAV acceleration wrt GS')
xlabel('Time (TU)');ylabel('Acceleration Error (DU/TU^2)');
legend('Tangential Accel Error', 'Normal Accel Error');
saveas(gcf, ['Figures\' char(strrep(scenario,' ','_'))...
    '_MAV_rot_nd_accel_err'],'jpg')
end
```