Table of Contents

ICGE 2

Liam Hood, Michael Watkins, and Michael Randolph

```
clear ; close all; clc;

L = 4 ; % Cylinder length in meters
RR = 0.25 ; % Rocket radius in meters
hn = 1 ; % nose height in meters
Mb = 600 ; % Mass of body in kg
Mn = 50 ; % Mass of nose in kg
rW = 0.2 ; % radius of wheel in meters
tW = 0.04 ; % thickness of wheel in meters
mW = 10 ; % mass of wheel in kg

% inertia matrix of the rocket body
Ib=[ (1/12)*Mb*(3*RR^2+L^2) , 0 , 0 ; ...
    0 , (1/12)*Mb*(3*RR^2+L^2) , 0 ; ...
    0 , 0 , 0.5*Mb*RR^2 ] ;

% inertia matrix of the rocket nose
In = [ ((1/10)*Mn*hn^2)+((3/20)*Mn*RR^2) , 0 , 0 ; ...
    0 , ((1/10)*Mn*hn^2)+((3/20)*Mn*RR^2) , 0 ; ...
    0 , 0 , (3/10)*Mn*RR^2 ] ;

% inertia matrix of the wheel
Iw=[ (1/12)*mW*(3*rW^2+tW^2) , 0 , 0 ; ...
    0 , (1/12)*mW*(3*rW^2+tW^2) , 0 ; ...
    0 , 0 , 0.5*mW*rW^2 ] ;

% Centers of mass from bottom of rocket in meters
CoM_rocket = [ 0 ; 0 ; (2*600+4.25*50)/650 ] ;
CoM_cone = [ 0 ; 0 ; 4.25 ] ;
CoM_body = [ 0 ; 0 ; 2 ] ;

% distance of center of mass of piece from system center of mass
rwc = [ 0 ; 0 ; 0 ] ;
rbc = CoM_rocket - CoM_body ;
rnc = CoM_rocket - CoM_cone ;

% inertia matrix about rockets center of mass
Jb = Ib - Mb*crossmatrix(rbc)*crossmatrix(rbc) ;
Jn = In - Mn*crossmatrix(rnc)*crossmatrix(rnc) ;
```

```
Ir = Jb + Jn ;

% Set up
opts = odeset( 'AbsTol' , 10^-8 , 'RelTol' , 10^-8 ) ;
tspan = [ 0 1000 ] ; % Time span to integrate over
dwrel = [ 0 ; 0 ; 0.05 ] ; % angular acceleration
```

3

```
Td = [ 0 ; 0 ; 0 ] ;
state = [ 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ] ;
RocketAndWheelPlots( tspan , state , opts , dwrel , Td , Iw , Ir , '3
 (no torque)' ) ;
```

4

```
Td = [ 0.1 ; 0 ; 0 ] ;
%4a
statea = [ 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ] ;
RocketAndWheelPlots( tspan , state , opts , dwrel , Td , Iw ,
 Ir , '4a' ) ;

%4b
stateb = [ 0 ; 0 ; 0 ; 0 ; 0 ; 0.1 ; 0 ; 0 ; 0 ] ;
RocketAndWheelPlots( tspan , stateb , opts , dwrel , Td , Iw ,
 Ir , '4b' ) ;

%4c
statec = [ 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 100 ] ;
RocketAndWheelPlots( tspan , statec , opts , dwrel , Td , Iw ,
 Ir , '4c' ) ;

%4d
stated = [ 0 ; 0 ; 0 ; 0 ; 0 ; 0.1 ; 0 ; 0 ; 100 ] ;
RocketAndWheelPlots( tspan , stated , opts , dwrel , Td , Iw ,
 Ir , '4d' ) ;
```

Functions

```
function [ across ] = crossmatrix( a )
    across = [ 0 -a(3) a(2) ; ...
               a(3) 0 -a(1) ; ...
              -a(2) a(3) 0 ] ;
end

function RocketAndWheelPlots( tspan , state , opts , dwrel , Td , Iw ,
 Ir , name )

    [t,state]=ode45( @RocketAndWheel , tspan , state , opts , dwrel ,
 Td , Iw , Ir ) ;
```

```
    figure( 'Name' , name , 'NumberTitle' , 'off' , 'Position' , [ 100
 50 1000 650 ] ) ;

    subplot(2,3,1)
    plot( t , state(:,1) )
    title( 'Angular Displacemet X vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Displacement X (rad/s)' )

    subplot(2,3,2)
    plot( t , state(:,2) )
    title( 'Angular Displacemet Y vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Displacement Y (rad/s)' )

    subplot(2,3,3)
    plot( t , state(:,3) )
    title( 'Angular Displacemet Z vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Displacement Z (rad/s)' )

    subplot(2,3,4)
    plot(t,state(:,4))
    title( 'Angular Velocity X vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Velocity X (rad/s)' )

    subplot(2,3,5)
    plot(t,state(:,5))
    title( 'Angular Velocity Y vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Velocity Y (rad/s)' )

    subplot(2,3,6)
    plot(t,state(:,6))
    title( 'Angular Velocity Z vs Time' )
    xlabel( 'Time(s)' )
    ylabel( 'Angular Velocity Z (rad/s)' )

end

function [ dstate ] = RocketAndWheel( t , state , dwrel , Td , Iw ,
 Ir )

    thetar = state(1:3) ;
    wr = state(4:6) ;
    wrel = state(7:9) ;
    dstate = zeros( 9 , 1 ) ;

    dwr = -inv( Ir+Iw ) * ( Iw*dwrel + crossmatrix(wr+wrel)*Iw*(wr
+wrel) + crossmatrix(wr)*Ir*wr - Td ) ;
    dstate(1:3) = wr ;
    dstate(4:6) = dwr ;
    dstate(7:9) = dwrel ;
```

```
end
```

Published with MATLAB® R2018b