

# Aerodynamics & Flight Mechanics Research Group

**The Simulation of Aircraft  
Motion by using MATLAB**

S. J. Newman

Technical Report AFM-11/27

January 2011

UNIVERSITY OF SOUTHAMPTON

SCHOOL OF ENGINEERING SCIENCES

AERODYNAMICS AND FLIGHT MECHANICS RESEARCH GROUP

**The Simulation of Aircraft Motion by using MATLAB**

by

**S. J. Newman**

AFM Report No. AFM 11/27

January 2011

© School of Engineering Sciences, Aerodynamics and Flight Mechanics Research Group



## COPYRIGHT NOTICE

(c) SES University of Southampton All rights reserved.

SES authorises you to view and download this document for your personal, non-commercial use. This authorization is not a transfer of title in the document and copies of the document and is subject to the following restrictions: 1) you must retain, on all copies of the document downloaded, all copyright and other proprietary notices contained in the Materials; 2) you may not modify the document in any way or reproduce or publicly display, perform, or distribute or otherwise use it for any public or commercial purpose; and 3) you must not transfer the document to any other person unless you give them notice of, and they agree to accept, the obligations arising under these terms and conditions of use. This document, is protected by worldwide copyright laws and treaty provisions.



# Introduction

---

The following analysis is to provide a simulation of aircraft motion using MATLAB to create 3D animations.

In essence, the method moves an aircraft representation under specification of the position of a reference point together with three Euler rotations. These are specified by the user and then a sequence of views ids generated by MATLAB and converted to an AVI file. These can then be assembled into a working presentation video using Windows Movie Maker or other video editing software.

# Coordinate System

---

The coordinated system is shown in Figures 1 & 2.

Figure 1 shows the axes orientation and the variables for translation.

The X axis points in the direction of flight, the Y axis points to port and the Z axis is vertically upwards. This is not the usual convention, in that Y is usually to starboard and Z vertically downwards. The adoption of this non-standard convention does not affect the Euler rotations but requires a sign reversal for the Y and Z coordinates. It is used in order to allow MATLAB to portray the aircraft properly. Turning the view upside down causes some difficulty in the order in which components of the aircraft are painted. This creates an optical illusion.



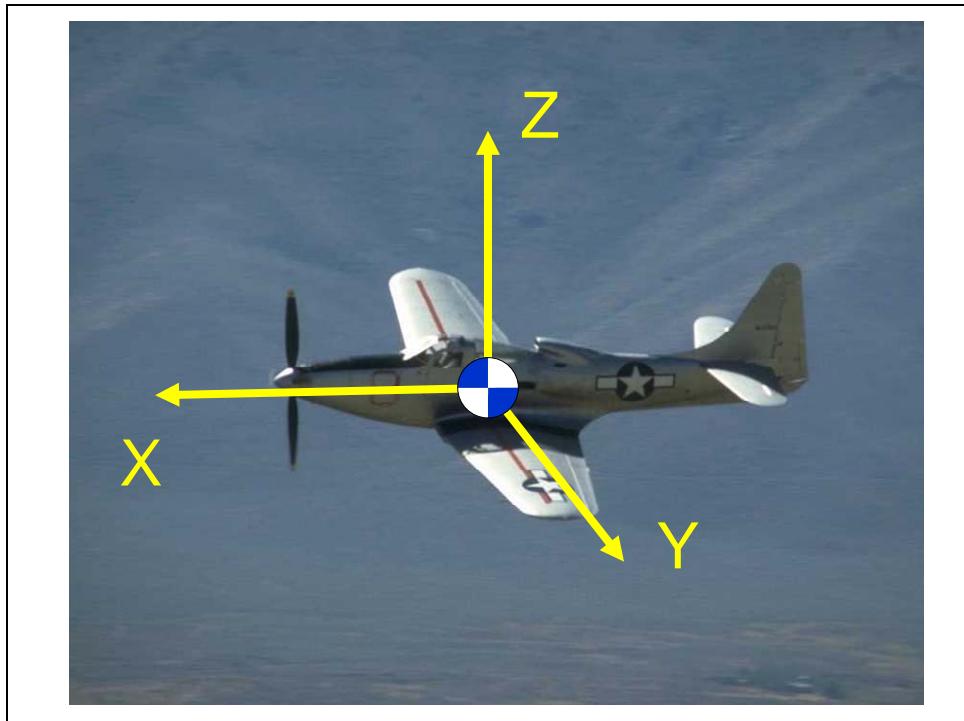


Figure 1 - Translation Axes

If  $(x,y,z)$  are coordinates in the airframe then the translation is specified by equation 1.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \end{bmatrix} + \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (1.)$$

Figure 2 shows the Euler rotations. These are detailed in Figures 3 - 5 and equations 2 – 4, respectively.



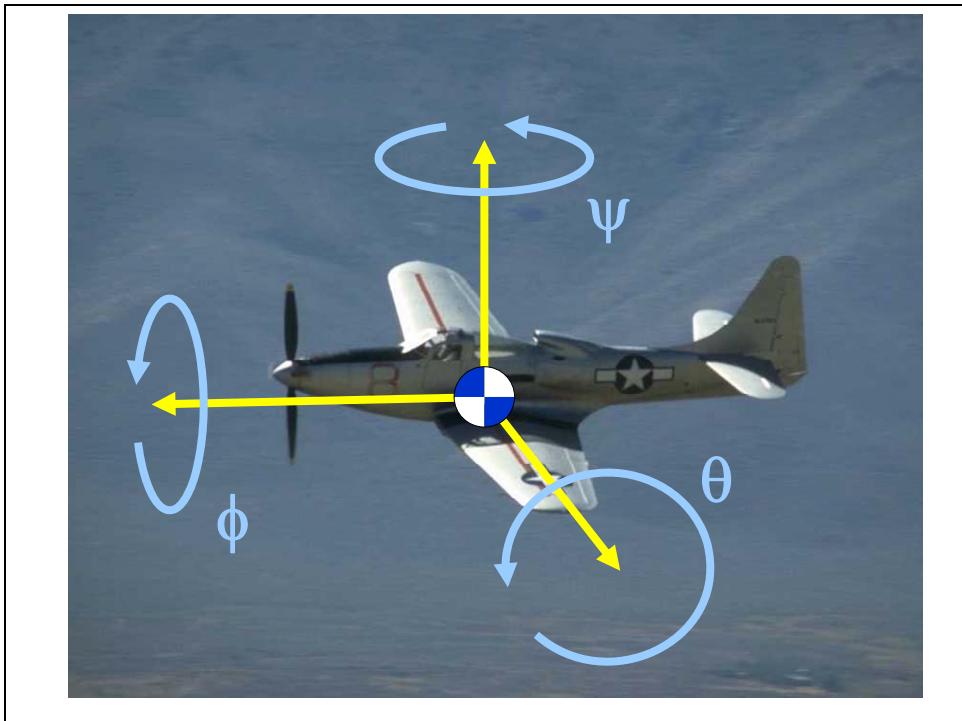


Figure 2 – Rotation Axes



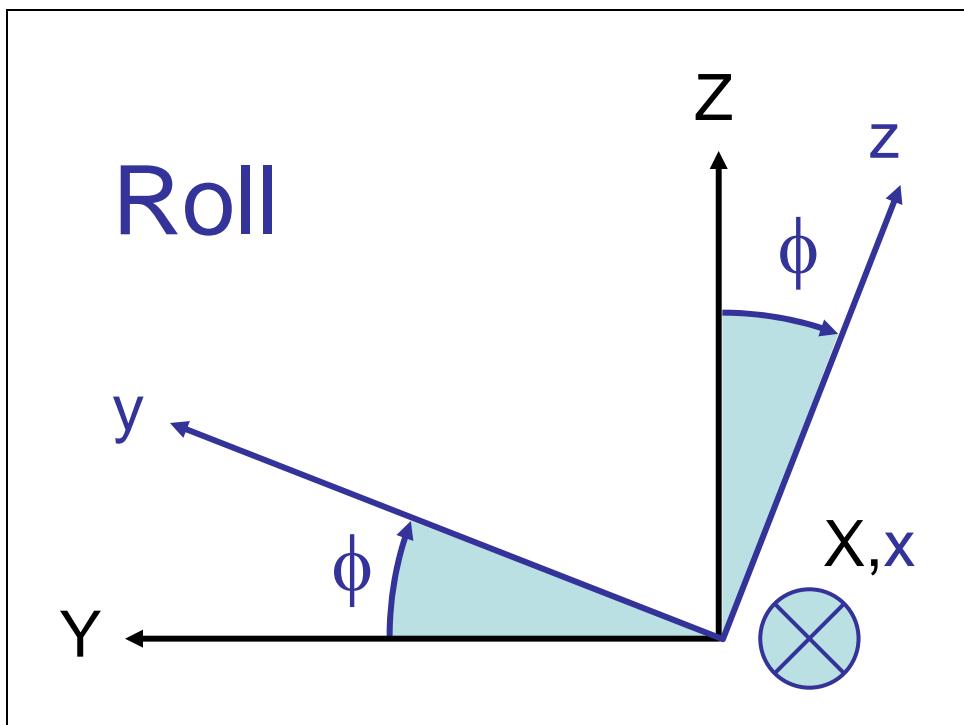
*Roll*

Figure 3 – Roll Rotation Transformation

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (2.)$$



## Pitch

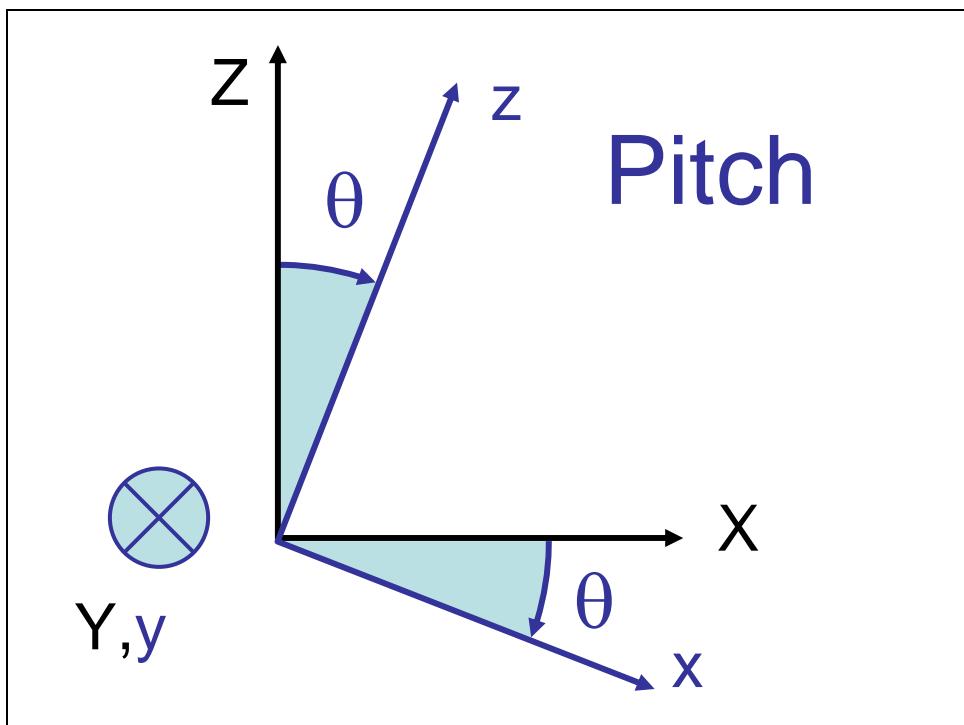


Figure 4 - Pitch Rotation Transformation

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (3.)$$



## Yaw

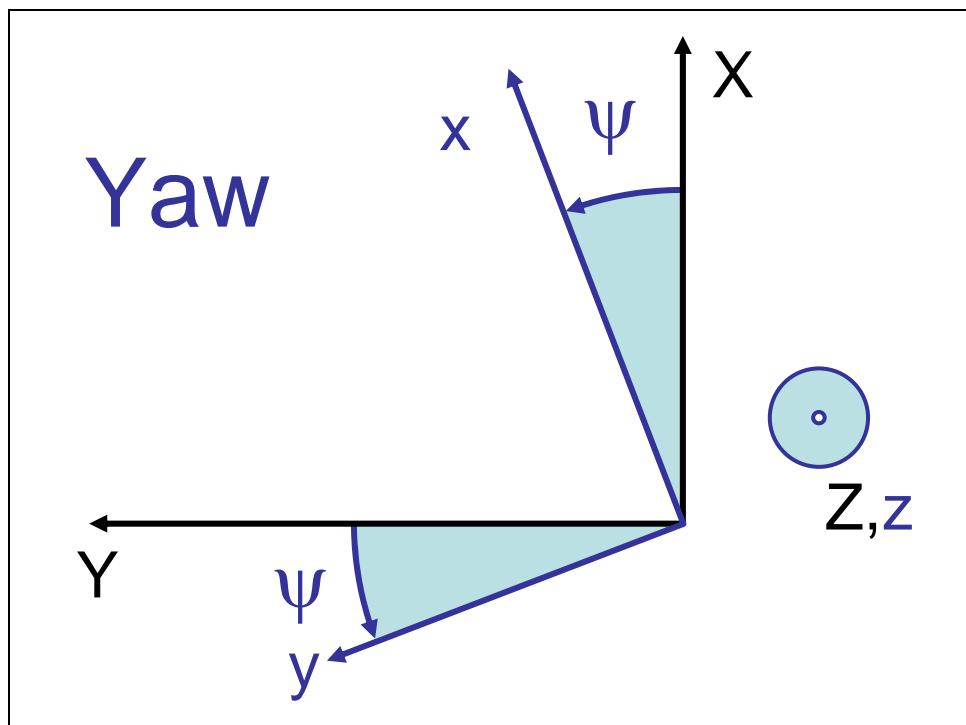


Figure 5 - Yaw Rotation Transformation

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (4.)$$



## Rotation Matrices

---

These three rotations – see equation 5 - are combined to form the final rotation matrix.

$$\begin{aligned}
 M_{ROLL} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix} \\
 M_{PITCH} &= \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \\
 M_{YAW} &= \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}
 \end{aligned} \tag{5.}$$

Examples of this method applied to fixed and rotary wing aircraft are now outlined.

They are:

- ➡ Fixed Wing - Phugoid
- ➡ Fixed Wing - Dutch Roll
- ➡ Fixed Wing - Short Period Oscillation
- ➡ Fixed Wing - Spin
- ➡ Rotary Wing – Spot Turn
- ➡ Rotary Wing – Ship Approach, Transit & Deck Landing



# Fixed Wing

---

The Phugoid is the long period longitudinal motion. The rotation matrix order is shown in equation 6 & the details of the synthesis in equation 7.

## Phugoid

---

$$M = M_{YAW} \cdot M_{PITCH} \cdot M_{ROLL} \quad (6.)$$

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} -l_{PHUG} \cdot \sin \omega t \\ 0 \\ h_{PHUG} \cdot \cos \omega t \\ 0 \\ \Theta_{PHUG} \cdot \sin \omega t \\ 0 \end{bmatrix} \quad (7.)$$

## Dutch Roll

---

The Dutch Roll is the long period lateral motion. The rotation matrix order is shown in equation 8 & the details of the synthesis in equation 9.

$$M = M_{YAW} \cdot M_{PITCH} \cdot M_{ROLL} \quad (8.)$$



$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ Y_{DR} \cdot \sin \phi \\ 0 \\ \Phi_{DR} \cdot \sin \omega t \\ \Theta_{DR} \\ -\Psi_{DR} \cdot \cos \omega t \end{bmatrix} \quad (9.)$$

## ***Short Period Oscillation (SPO)***

---

The SPO is the short period longitudinal motion. The rotation matrix order is shown in equation 10 & the details of the synthesis in equation 11.

$$M = M_{YAW} \cdot M_{PITCH} \cdot M_{ROLL} \quad (10.)$$

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \Theta_{SPO} \cdot \sin \omega t \\ 0 \end{bmatrix} \quad (11.)$$

The Spin is an unusual motion



on and in order to synthesise it, the rotation order is changed. The rotation matrix order is shown in equation 12 & the details of the synthesis in equation 13.

$$M = M_{ROLL} \cdot M_{YAW} \cdot M_{PITCH} \quad (12.)$$

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ r_{SPIN} \cdot \sin \omega t \\ -r_{SPIN} \cdot \cos \omega t \\ \omega t \\ \Theta_{SPIN} \\ \Psi_{SPIN} \end{bmatrix} \quad (13.)$$



# Rotary Wing

---

## Spot Turn

---

The Spot Turn is, probably, the simplest type of rotational motion. The rotation matrix order is shown in equation 14 & the details of the synthesis in equation 15.

$$M = M_{YAW} \cdot M_{PITCH} \cdot M_{ROLL} \quad (14.)$$

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \omega t \end{bmatrix} \quad (15.)$$

# Deck Landing

---

The deck landing manoeuvre consists of three separate translations. Rotation can be included but in this document, only translations are considered. For reference, the rotation matrix order is shown in equation 16 & the details of the synthesis in equations 17 - 19.



$$M = M_{YAW} \cdot M_{PITCH} \cdot M_{ROLL} \quad (16.)$$

## Approach

---

This is where the helicopter flies in along the port side of the ship, at the hangar roof height, until it is positioned at the buttock line of the deck.

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} -2 * r_{Main} \rightarrow 0 \\ r_{Main} + \frac{W_{Deck}}{2} \\ y_{Hangar} - h_{Main} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (17.)$$

*Tra  
nsit*  
 It  
the  
n  
mov  
es  
late

rally at constant height until it is positioned at the centreline of the ship.



$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ \left( r_{Main} + \frac{W_{Deck}}{2} \right) \rightarrow 0 \\ y_{Hangar} - h_{Main} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (18.)$$

## Touchdown

---

The final phase is where the helicopter descends to the deck surface.

$$\begin{bmatrix} X_{CG} \\ Y_{CG} \\ Z_{CG} \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ (y_{Hangar} - h_{Main}) \rightarrow 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (19.)$$



# Specifying the Aircraft

---

The shapes of the aircraft, fixed wing and helicopter are now detailed.

The fixed wing consists of a solid of revolution fuselage, and flat plates for the wing, tail plane and fin. A single propeller is placed at the nose of the fuselage. The CG is placed at the quarter length point of the fuselage together with the wing quarter chord point. The tailplane & fin trailing edges are placed at the rear point of the fuselage. There is an option whereby the tip vortices are drawn. They are assumed to be parallel with the X axis.

The helicopter consists of two ellipsoids portraying the main body and tail boom. The CG is placed at the centre point of the main body. The main and tail rotors are circular discs appropriately placed. A tricycle undercarriage is assumed.



# Aircraft Specification – Fixed Wing

The aircraft specification is detailed in Figures 6 – 8.

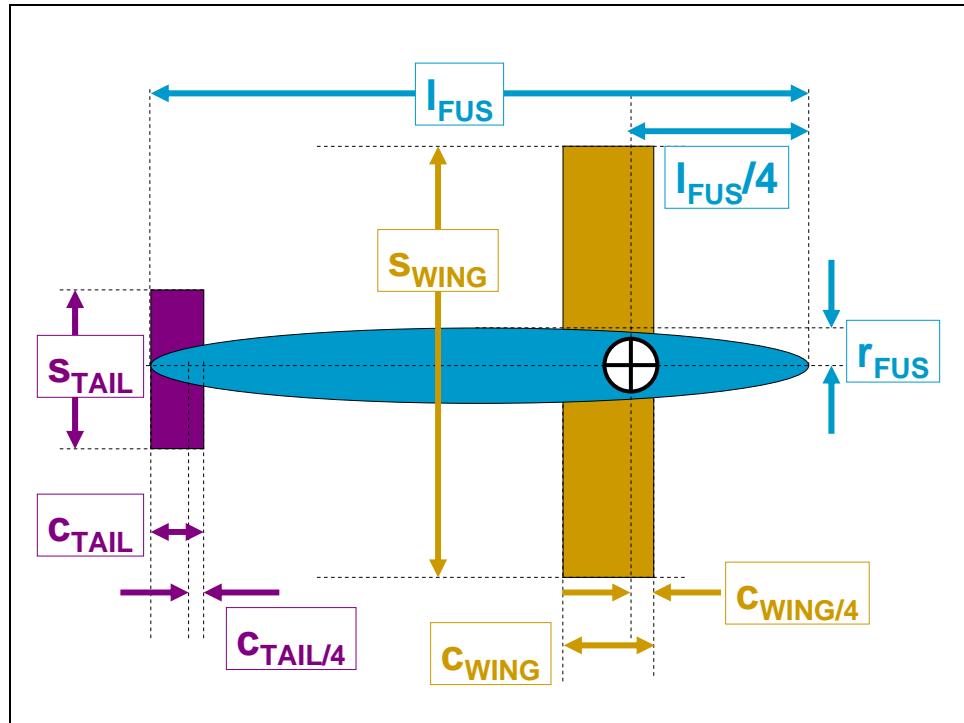


Figure 6 - Horizontal Dimensions

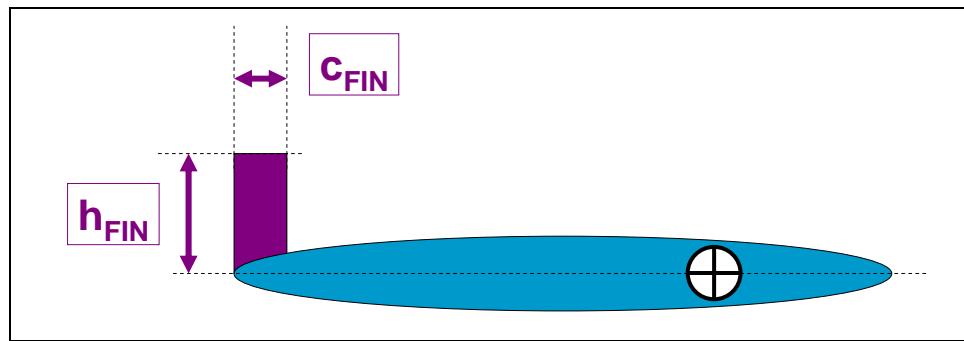


Figure 7 - Vertical Dimensions



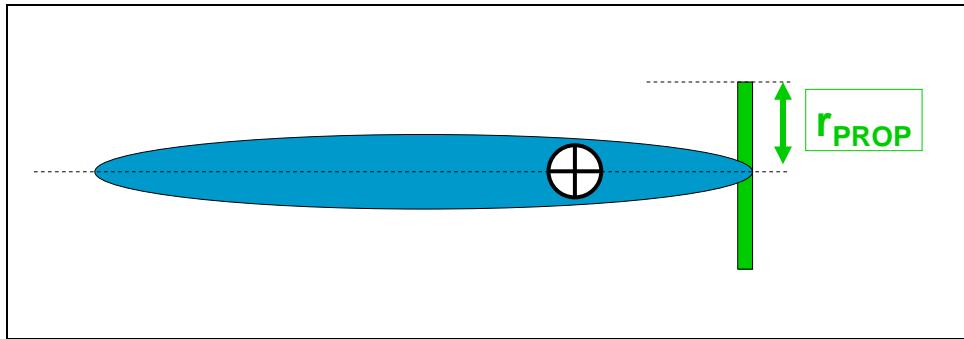


Figure 8 - Propeller Dimensions



# Aircraft Specification – Rotary Wing

The aircraft specification is detailed in Figures 9 – 12.

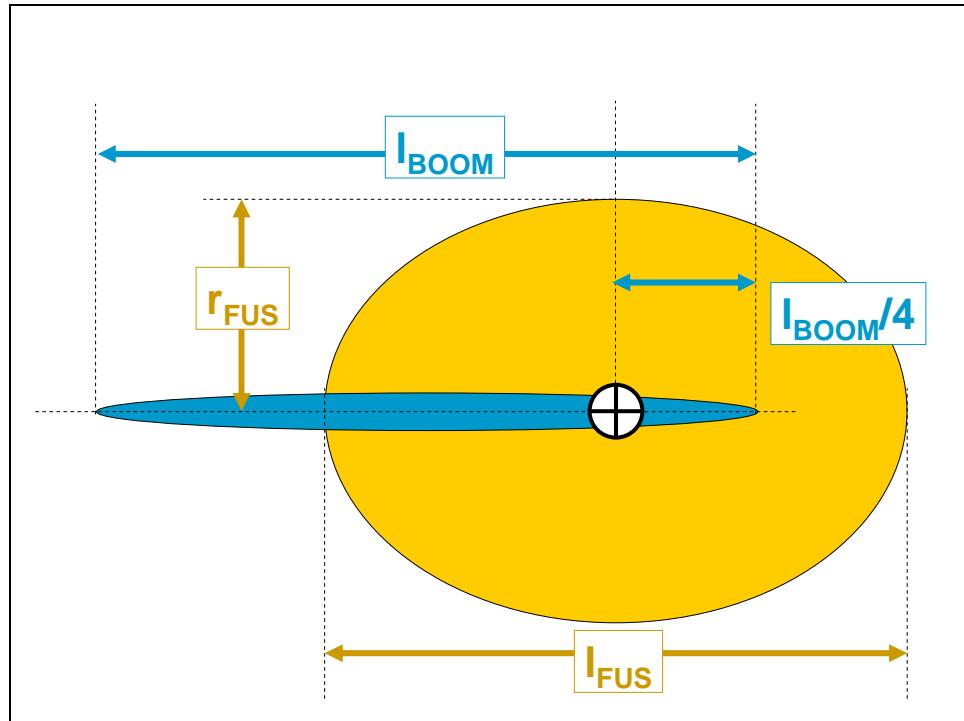


Figure 9 - Horizontal Dimensions - Fuselage & Tail Boom

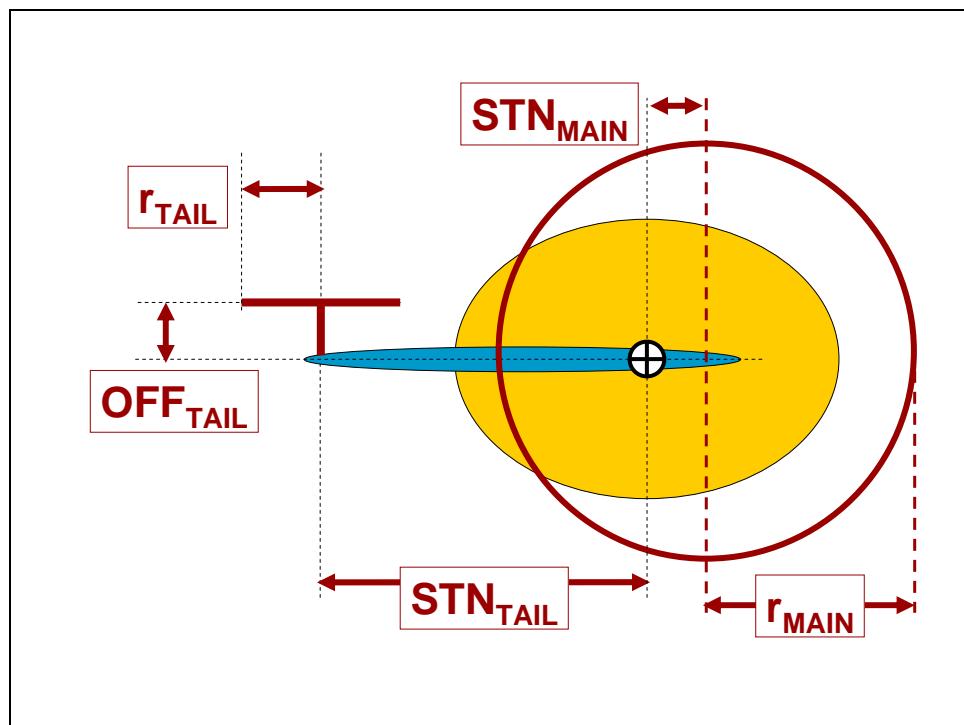


Figure 10 - Horizontal Dimensions – Rotors



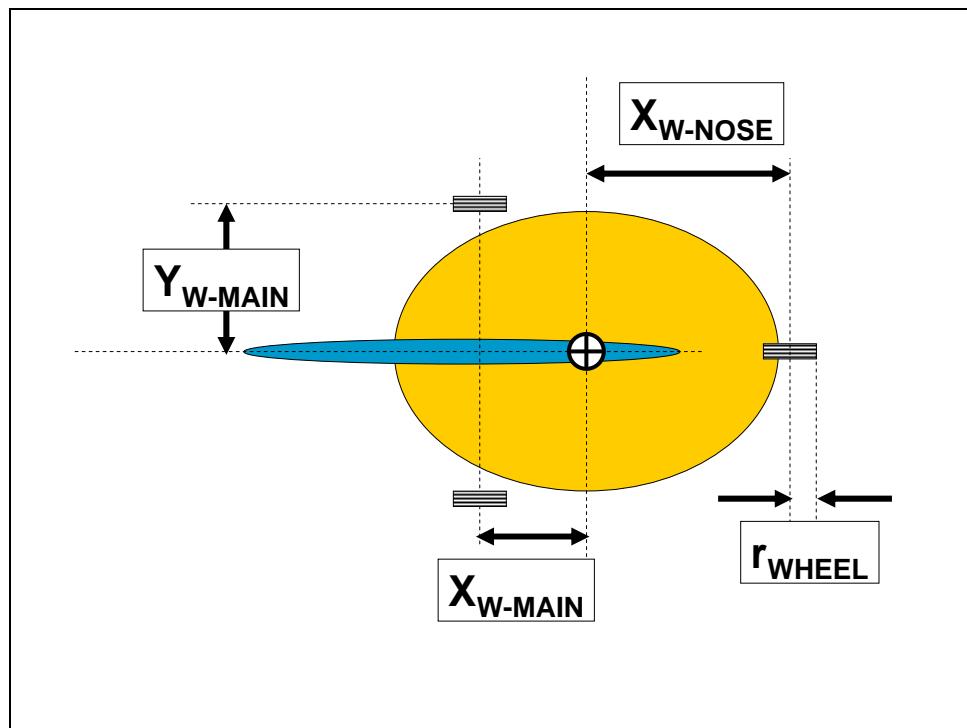


Figure 11 - Horizontal Dimensions – Undercarriage

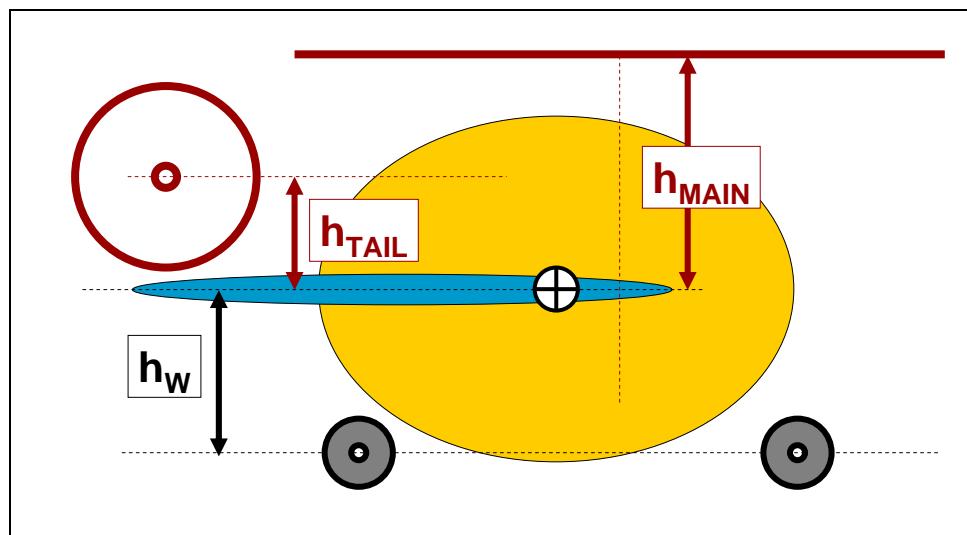


Figure 12 - Vertical Dimensions



# Ship Specification – Rotary Wing

The ship flight deck specification is detailed in Figure 13. There is an option whereby three lights are placed around the deck as shown in the Figure.

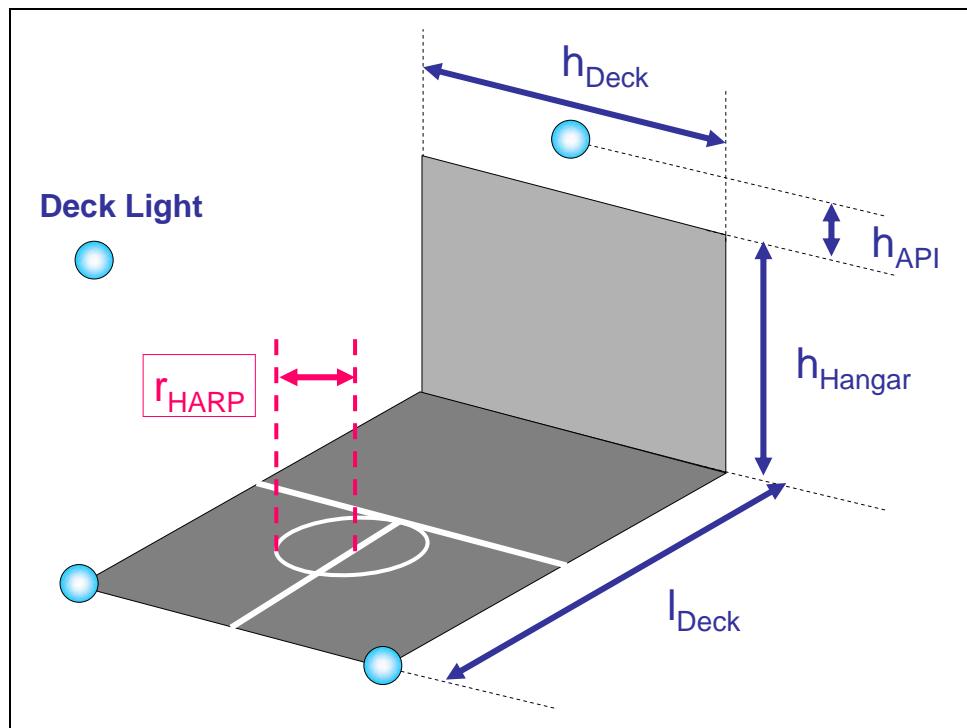


Figure 13 - Ship Deck Measurements & Lighting



# MATLAB M File Variables & Default Values

---

The M files variable names are presented in the following tables together with the numerical values used in the included M file listings.

## *Fixed Wing Aircraft*

---

Parameter	Program Variable Name	Default Value
$l_{FUS}$	fusl	4.0
$r_{FUS}$	fusr	0.25
$s_{WING}$	wingspan	5.0
$c_{WING}$	wingchord	1.0
$s_{TAIL}$	tailspan	1.0
$c_{TAIL}$	tailchord	0.4
$h_{FIN}$	finh	0.5
$r_{PROP}$	propr	0.5

---



## Rotary Wing Aircraft (Helicopter)

---

Parameter	Program Variable Name	Default Value
$l_{FUS}$	fusl	3.0
$r_{FUS}$	fusr	1.0
$l_{BOOM}$	booml	4.0
$r_{BOOM}$	boomr	0.2
$r_{MAIN}$	mainr	2.5
$STN_{MAIN}$	mainstn	0
$h_{MAIN}$	mainh	1.0
$r_{TAIL}$	tailr	0.6
$STN_{TAIL}$	tailstn	-3.0
$OFF_{TAIL}$	tailoffst	0.2
$h_{TAIL}$	tailh	0.5
$r_{WHEEL}$	wheelr	0.25
$h_{WHEEL}$	wheelh	-1.0
$X_{W-NOSE}$	nosewhlx	1.0
$X_{W-MAIN}$	mainwhlx	-0.6
$Y_{W-MAIN}$	mainwhly	0.7



## Phugoid

---

Parameter	Program Variable Name	Default Value
$l_{PHUG}$	phuht	4.0
$h_{PHUG}$	phusrg	2.0
$\Theta_{PHUG}$	phuthetad	15°
$\omega t$	angmovd	0 - 360°

## Dutch Roll

---

Parameter	Program Variable Name	Default Value
$Y_{DR}$	drsway	3
$\Phi_{DR}$	drrollang	20
$\Psi_{DR}$	dryawang	20
$\omega t$	angmovd	0 - 360°

## Short Period Oscillation

---

Parameter	Program Variable Name	Default Value
$\Theta_{SPO}$	sopoamp	40
$\omega t$	angmovd	0 - 360°

## Spin

---

Parameter	Program Variable Name	Default Value
$r_{SPIN}$	spinr	0.5
$\Theta_{SPIN}$	spinthetad	-20°
$\Psi_{SPIN}$	spinsid	20°
$\omega t$	angmovd	0 - 360°





## Spot Turn

---

Parameter	Program Variable Name	Default Value
$\omega_t$	angmovd	0 - 360°

## Deck Landing

---

Parameter	Program Variable Name	Default Value
$l_{DECK}$	deckl	8.0
$w_{DECK}$	deckw	6.0
$h_{HANGAR}$	hangary	2.0
$h_{API}$	hapi	1.0
$r_{HARP}$	harpoonr	1.5



# MATLAB Files

---

The MATLAB M files for the above aircraft motions are presented below:

## Fixed Wing

---

### *Phugoid*

---

```
%  
%     Flight Mechanics Simulation - Fixed Wing - Phugoid  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = -40;  
el = 40;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
nframe=181;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
propx=zeros(73,1);  
propy=propx;  
propz=propx;  
xcg=zeros(nframe,1);  
ycg=zeros(nframe,1);  
zcg=zeros(nframe,1);  
phid=zeros(nframe,1);  
thetad=zeros(nframe,1);  
psid=zeros(nframe,1);  
  
%=====Specify Aircraft Components  
% Fuselage Specification  
fusl=4;  
fusr=.25;  
% Wing Specification  
wingspan=5;
```



```

wingchord=1;

    s=wingspan/2;
    qchd=wingchord/4;
    wingx=[-3*qchd -3*qchd qchd qchd];
    wingy=[-s s s -s];
    wingz=[0 0 0 0];

% Tailplane Specification
tailspan=1;
tailchord=.4;

    qtchd=tailchord/4;
    ts=tailspan/2;

% Fin Specification
finh=.5;
% Propeller Specification
propR=0.5;
% Tip Vortices ? (=0 NO)
tipvortex=0;
%=====
%?????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
%=====Set Up Circle Angles
angmovid=linspace(0,360,nframe);% 1 Cycle in Degrees
angmovr=angmovid*pi/180;% 1 Cycle in Radians

phuht=4; % Phugoid Height
phusrg=4; % Phugoid Surge
phuthetad=15 % Phugoid Pitch

for iframe=1:nframe
%=====Specify Euler Angles in Degrees
thetad(iframe)=phuthetad*sin(angmovr(iframe));
%=====Specify - CG Position
xcg(iframe)=-phusrg*sin(angmovr(iframe));
ycg(iframe)=0;
zcg(iframe)=phuht*cos(angmovr(iframe));
%=====
end
%=====
%?????????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
phir=phid(iframe)*pi/180;
thetar=thetad(iframe)*pi/180;
psir=psid(iframe)*pi/180;
%=====
cphi=cos(phir);
sphi=sin(phir);
ctheta=cos(theta);
stheta=sin(theta);
cpsi=cos(psir);
spsi=sin(psir);

```



```

%***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,sttheta;0,1,0;-sttheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
%*****
%=====Invoke View
%
% Calculate Wing
%
wing=[wingx; wingy; wingz];
dwing=D*wing;
dwingx=dwing(1,:)+xcg(iframe);
dwingy=dwing(2,:)+ycg(iframe);
dwingz=dwing(3,:)+zcg(iframe);

fill3(dwingx,dwingy,dwingz,'c'); % Draw Wing

view(az, el);
hold on
%
% Calculate Tailplane
%
tailx=[-3*qtchd -3*qtchd qtchd qtchd]-(.75*fusl-3*qtchd)*[1 1 1
1];

taily=[-ts ts ts -ts];
tailz=[0 0 0 0];
tail=[tailx; taily; tailz];
dtail=D*tail;
dtailx=dtail(1,:)+xcg(iframe);
dtaily=dtail(2,:)+ycg(iframe);
dtailz=dtail(3,:)+zcg(iframe);
fill3(dtailx,dtaily,dtailz,'c'); % Draw Tailplane

%
% Calculate Fuselage
%
t=linspace(0,pi,11);
r=sin(t).^.5;
[fusz,fusy,fusx]=cylinder(r);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-.75*ones(size(fusx)));


[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz); % Draw Fuselage

```



```

%
% Calculate Fin
%
tailr=.75*fusl;
tailf=tailr-tailchord;
finx=[-tailr -tailr -tailf -tailf];
finy=[0 0 0 0];
finz=[0 finh finh 0];
fin=[finx; finy; finz];
dfin=D*fin;
dfinx=dfin(1,:)+xcg(iframe);
dfiny=dfin(2,:)+ycg(iframe);
dfinz=dfin(3,:)+zcg(iframe);
fill3(dfinx,dfiny,dfinz,'c'); % Draw Fin
%
% Calculate Prop
%
for iprop=1:73
    propaz=2*pi*(iprop-1)/72;
    propx(iprop)=.25*fusl;
    propy(iprop)=propr*cos(propaz);
    propz(iprop)=propr*sin(propaz);
    prop=[propx'; propy'; propz'];
    dprop=D*prop;
    dpropx=dprop(1,:)+xcg(iframe);
    dpropy=dprop(2,:)+ycg(iframe);
    dpropz=dprop(3,:)+zcg(iframe);
end
fill3(dpropx,dpropy,dpropz,'w'); % Draw Prop
%
% Calculate Wing Tip Vortices - If Required
%
if tipvortex~=0
    dvortlx(1)=dwingx(1);
    dvortrx(1)=dwingx(2);
    dvortly(1)=dwingy(1);
    dvortry(1)=dwingy(2);
    dvortlz(1)=dwingz(1);
    dvortrz(1)=dwingz(2);

    dvortlx(2)=xmin;
    dvortrx(2)=xmin;
    dvortly(2)=dwingy(1);
    dvortry(2)=dwingy(2);
    dvortlz(2)=dwingz(1);
    dvortrz(2)=dwingz(2);
    %
    % Plot Wing Tip Vortices
    %

    plot3(dvortlx,dvortly,dvortlz,'linewidth',3,'Color','b');
    plot3(dvortrx,dvortry,dvortrz,'linewidth',3,'Color','b');
end

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

```



```
heading=['X = ',num2str(xcg(iframe)), ' Y =  
' ,num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll =  
' ,num2str(phid(iframe)), '\circ', ' Pitch =  
' ,num2str(thetaid(iframe)), '\circ', ' Yaw =  
' ,num2str(psid(iframe)), '\circ'];  
title(heading,'Background','b','Margin',5,'Edgecolor','y');  
grid on  
m(iframe)=getframe(gcf);  
  
%axis equal  
end  
  
%movie2avi(m,'jim','Compression','None');
```



## Dutch Roll

---

```
%  
%     Flight Mechanics Simulation - Fixed Wing  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = -40;  
el = 40;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
nframe=181;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
propx=zeros(73,1);  
propy=propx;  
propz=propx;  
xcg=zeros(nframe,1);  
ycg=zeros(nframe,1);  
zcg=zeros(nframe,1);  
phid=zeros(nframe,1);  
thetad=zeros(nframe,1);  
psid=zeros(nframe,1);  
  
%=====Specify Aircraft Components  
% Fuselage Specification  
fusl=4;  
fusr=.25;  
% Wing Specification  
wingspan=5;  
wingchord=1;  
  
s=wingspan/2;  
qchd=wingchord/4;  
wingx=[-3*qchd -3*qchd qchd qchd];  
wingy=[-s s s -s];  
wingz=[0 0 0 0];  
  
% Tailplane Specification  
tailspan=1;  
tailchord=.4;  
  
qtchd=tailchord/4;
```



```

ts=tailspan/2;

% Fin Specification
finh=.5;
% Propeller Specification
propr=0.5;
% Tip Vortices ? (=0 NO)
tipvortex=0;
%=====
%?????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
%=====Set Up Circle Angles
angmovd=linspace(0,360,nframe);% 1 Cycle in Degrees
angmovr=angmovd*pi/180;% 1 Cycle in Radians
drsway=3; % Dutch Roll Sway
drrollang=20; %Dutch Roll - Roll Angle Amplitude
dryawang=20; %Dutch Roll - Yaw Angle Amplitude
for iframe=1:nframe
%=====Specify Euler Angles in Degrees
phid(iframe)=drrollang*sin(angmovr(iframe));
psid(iframe)=-dryawang*cos(angmovr(iframe));
%thetad(iframe)=-15;
%=====Specify - CG Position
xcg(iframe)=0;
ycg(iframe)=drsway*sin(phid(iframe)*pi/180);
zcg(iframe)=0;
%=====
end
%=====

%?????????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
phir=phid(iframe)*pi/180;
thetar=thetad(iframe)*pi/180;
psir=psid(iframe)*pi/180;
%=====
cphi=cos(phir);
sphi=sin(phir);
ctheta=cos(thetar);
stheta=sin(thetar);
cpsi=cos(psir);
spsi=sin(psir);

%***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,stheta;0,1,0;-stheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
%*****
%=====Invoke View
%
% Calculate Wing
%
wing=[wingx; wingy; wingz];
dwing=D*wing;

```



```

dwingx=dwing(1,:)+xcg(iframe);
dwingy=dwing(2,:)+ycg(iframe);
dwingz=dwing(3,:)+zcg(iframe);

fill3(dwingx,dwingy,dwingz,'c'); % Draw Wing

view(az, el);
hold on
%
% Calculate Tailplane
%
tailx=[-3*qtchd -3*qtchd qtchd qtchd]-(.75*fusl-3*qtchd)*[1 1 1
1];

taily=[-ts ts ts -ts];
tailz=[0 0 0 0];
tail=[tailx; taily; tailz];
dtail=D*tail;
dtailx=dtail(1,:)+xcg(iframe);
dtaily=dtail(2,:)+ycg(iframe);
dtailz=dtail(3,:)+zcg(iframe);
fill3(dtailx,dtaily,dtailz,'c'); % Draw Tailplane

%
% Calculate Fuselage
%
t=linspace(0,pi,11);
r=sin(t).^.5;
[fusz,fusy,fusx]=cylinder(r);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-.75*ones(size(fusx))));

[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz); % Draw Fuselage
%
% Calculate Fin
%
tailr=.75*fusl;
tailf=tailr-tailchord;
finx=[-tailr -tailr -tailf -tailf];
finy=[0 0 0 0];
finz=[0 finh finh 0];
fin=[finx; finy; finz];
dfin=D*fin;
dfinx=dfin(1,:)+xcg(iframe);
dfiny=dfin(2,:)+ycg(iframe);
dfinz=dfin(3,:)+zcg(iframe);

```



```

fill3(dfinx,dfiny,dfinz,'c'); % Draw Fin
%
% Calculate Prop
%
for iprop=1:73
    propaz=2*pi*(iprop-1)/72;
    propx(iprop)=.25*fusl;
    propy(iprop)=propr*cos(propaz);
    propz(iprop)=propr*sin(propaz);
    prop=[propx'; propy'; propz'];

    dprop=D*prop;
    dpropx=dprop(1,:)+xcg(iframe);
    dpropy=dprop(2,:)+ycg(iframe);
    dpropz=dprop(3,:)+zcg(iframe);
end
fill3(dpropx,dpropy,dpropz,'w'); % Draw Prop
%
% Calculate Wing Tip Vortices - If Required
%
if tipvortex~=0

dvortlx(1)=dwingx(1);
dvortrx(1)=dwingx(2);
dvortly(1)=dwingy(1);
dvortry(1)=dwingy(2);
dvortlz(1)=dwingz(1);
dvortrz(1)=dwingz(2);

dvortlx(2)=xmin;
dvortrx(2)=xmin;
dvortly(2)=dwingy(1);
dvortry(2)=dwingy(2);
dvortlz(2)=dwingz(1);
dvortrz(2)=dwingz(2);
%
% Plot Wing Tip Vortices
%

plot3(dvortlx,dvortly,dvortlz,'linewidth',3,'Color','b');
plot3(dvortrx,dvortry,dvortrz,'linewidth',3,'Color','b');
end

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y =
',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll =
',num2str(phid(iframe)), '\circ', ' Pitch =
',num2str(thetaid(iframe)), '\circ', ' Yaw =
',num2str(psid(iframe)), '\circ'];
title(heading,'Background','b','Margin',5,'Edgecolor','y');
grid on
m(iframe)=getframe(gcf);

%
%axis equal
end

```



```
%movie2avi(m,'jim','Compression','None');
```

## Short Period Oscillation

---

```

%
%     Flight Mechanics Simulation - Fixed Wing - SPO
%
%=====Set Up Colours & Perspective View
clear
colordef black
camproj('perspective');
%=====Set Up View Angles
az = -40;
el = 40;
%=====
xmin=-5;
xmax=5;
ymin=-5;
ymax=5;
zmin=-5;
zmax=5;
%=====Define Number of Time Steps
nframe=181;
%=====Set Up Holding Arrays

dpt=zeros(3,1);

propx=zeros(73,1);
propy=propx;
propz=propx;
xcg=zeros(nframe,1);
ycg=zeros(nframe,1);
zcg=zeros(nframe,1);
phid=zeros(nframe,1);
thetad=zeros(nframe,1);
psid=zeros(nframe,1);

%=====Specify Aircraft Components
% Fuselage Specification
fusl=4;
fusr=.25;
% Wing Specification
wingspan=5;
wingchord=1;

s=wingspan/2;
qchd=wingchord/4;
wingx=[-3*qchd -3*qchd qchd qchd];
wingy=[-s s s -s];
wingz=[0 0 0 0];

% Tailplane Specification
tailspan=1;
tailchord=.4;
```



```

qtchd=tailchord/4;
ts=tailspan/2;

% Fin Specification
finh=.5;
% Propeller Specification
propr=0.5;
% Tip Vortices ? (=0 NO)
tipvortex=0;
%=====
%?????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
%=====Set Up Circle Angles
angmovid=linspace(0,360,nframe);% 1 Cycle in Degrees
angmovr=angmovid*pi/180;% 1 Cycle in Radians

spoamp=40; % SPO Pitch Amplitude

for iframe=1:nframe
%=====Specify Euler Angles in Degrees
thetad(iframe)=spoamp*sin(angmovr(iframe));
%=====Specify - CG Position
xcg(iframe)=0;
ycg(iframe)=0.;
zcg(iframe)=0;
%=====
end
%=====

%?????????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
phir=phid(iframe)*pi/180;
thetar=thetad(iframe)*pi/180;
psir=psid(iframe)*pi/180;
%=====
cphi=cos(phir);
sphi=sin(phir);
ctheta=cos(theta);
stheta=sin(theta);
cpsi=cos(psir);
spsi=sin(psir);

%***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;sphi,cphi];
mpitch=[ctheta,0,stheta;0,1,0;-stheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
%*****
%=====Invoke View
%
% Calculate Wing
%
wing=[wingx; wingy; wingz];
dwing=D*wing;

```



```

dwingx=dwing(1,:)+xcg(iframe);
dwingy=dwing(2,:)+ycg(iframe);
dwingz=dwing(3,:)+zcg(iframe);

fill3(dwingx,dwingy,dwingz,'c'); % Draw Wing

view(az, el);
hold on
%
% Calculate Tailplane
%
tailx=[-3*qtchd -3*qtchd qtchd qtchd]-(.75*fusl-3*qtchd)*[1 1 1
1];

taily=[-ts ts ts -ts];
tailz=[0 0 0 0];
tail=[tailx; taily; tailz];
dtail=D*tail;
dtailx=dtail(1,:)+xcg(iframe);
dtaily=dtail(2,:)+ycg(iframe);
dtailz=dtail(3,:)+zcg(iframe);
fill3(dtailx,dtaily,dtailz,'c'); % Draw Tailplane

%
% Calculate Fuselage
%
t=linspace(0,pi,11);
r=sin(t).^.5;
[fusz,fusy,fusx]=cylinder(r);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-.75*ones(size(fusx))));

[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz); % Draw Fuselage
%
% Calculate Fin
%
tailr=.75*fusl;
tailf=tailr-tailchord;
finx=[-tailr -tailr -tailf -tailf];
finy=[0 0 0 0];
finz=[0 finh finh 0];
fin=[finx; finy; finz];
dfin=D*fin;
dfinx=dfin(1,:)+xcg(iframe);
dfiny=dfin(2,:)+ycg(iframe);
dfinz=dfin(3,:)+zcg(iframe);

```



```

fill3(dfinx,dfiny,dfinz,'c'); % Draw Fin
%
% Calculate Prop
%
for iprop=1:73
    propaz=2*pi*(iprop-1)/72;
    propx(iprop)=.25*fusl;
    propy(iprop)=propr*cos(propaz);
    propz(iprop)=propr*sin(propaz);
    prop=[propx'; propy'; propz'];

    dprop=D*prop;
    dpropx=dprop(1,:)+xcg(iframe);
    dpropy=dprop(2,:)+ycg(iframe);
    dpropz=dprop(3,:)+zcg(iframe);
end
fill3(dpropx,dpropy,dpropz,'w'); % Draw Prop
%
% Calculate Wing Tip Vortices - If Required
%
if tipvortex~=0

dvortlx(1)=dwingx(1);
dvortrx(1)=dwingx(2);
dvortly(1)=dwingy(1);
dvortry(1)=dwingy(2);
dvortlz(1)=dwingz(1);
dvortrz(1)=dwingz(2);

dvortlx(2)=xmin;
dvortrx(2)=xmin;
dvortly(2)=dwingy(1);
dvortry(2)=dwingy(2);
dvortlz(2)=dwingz(1);
dvortrz(2)=dwingz(2);
%
% Plot Wing Tip Vortices
%
plot3(dvortlx,dvortly,dvortlz,'linewidth',3,'Color','b');
plot3(dvortrx,dvortry,dvortrz,'linewidth',3,'Color','b');
end

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y =
',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll =
',num2str(phid(iframe)), '\circ', ' Pitch =
',num2str(thetaid(iframe)), '\circ', ' Yaw =
',num2str(psid(iframe)), '\circ'];
title(heading,'Background','b','Margin',5,'Edgecolor','y');
grid on
m(iframe)=getframe(gcf);

%axis equal
end

```



```
%movie2avi(m,'jim','Compression','None');
```



## Spin

---

```
%  
%     Flight Mechanics Simulation - Fixed Wing - Spin  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = -90;  
el = 0;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
nframe=181;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
propx=zeros(73,1);  
propy=propx;  
propz=propx;  
xcg=zeros(nframe,1);  
ycg=zeros(nframe,1);  
zcg=zeros(nframe,1);  
phid=zeros(nframe,1);  
thetad=zeros(nframe,1);  
psid=zeros(nframe,1);  
  
%=====Specify Aircraft Components  
% Fuselage Specification  
fusl=4;  
fusr=.25;  
% Wing Specification  
wingspan=5;  
wingchord=1;  
  
s=wingspan/2;  
qchd=wingchord/4;  
wingx=[-3*qchd -3*qchd qchd qchd];  
wingy=[-s s s -s];  
wingz=[0 0 0 0];  
  
% Tailplane Specification  
tailspan=1;  
tailchord=.4;  
  
qtchd=tailchord/4;
```



```

ts=tailspan/2;

% Fin Specification
finh=.5;
% Propeller Specification
propr=0.5;
% Tip Vortices ? (=0 NO)
tipvortex=0;
%=====
%?????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
%=====Set Up Circle Angles
angmovid=linspace(0,360,nframe);% 1 Cycle in Degrees
angmovr=angmovid*pi/180;% 1 Cycle in Radians

spinthetaad=-20; % Spin Pitch Attitude
spinspid=20; %Spin Yaw Attitude
spinr=0.5; % Spin Radius of CG Motion

for iframe=1:nframe
%=====Specify Euler Angles in Degrees
phid(iframe)=angmovid(iframe);
psid(iframe)=0;
thetad(iframe)=spinthetaad;
%=====Specify - CG Position
xcg(iframe)=0;
ycg(iframe)=spinr*sin(angmovr(iframe));
zcg(iframe)=-spinr*cos(angmovr(iframe));
%=====
end
%=====
%?????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
phir=phid(iframe)*pi/180;
thetar=thetad(iframe)*pi/180;
psir=psid(iframe)*pi/180;
%=====
cphi=cos(phir);
sphi=sin(phir);
ctheta=cos(theta);
stheta=sin(theta);
cpsi=cos(psir);
spsi=sin(psir);

%***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,stheta;0,1,0;-stheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=mroll*myaw*mpitch; % Reordered
%*****
%=====Invoke View
%
% Calculate Wing
%

```



```

wing=[wingx; wingy; wingz];
dwing=D*wing;
dwingx=dwing(1,:)+xcg(iframe);
dwingy=dwing(2,:)+ycg(iframe);
dwingz=dwing(3,:)+zcg(iframe);

fill3(dwingx,dwingy,dwingz,'c'); % Draw Wing

view(az, el);
hold on
%
% Calculate Tailplane
%
tailx=[-3*qtchd -3*qtchd qtchd qtchd]-(.75*fusl-3*qtchd)*[1 1 1
1];

taily=[-ts ts ts -ts];
tailz=[0 0 0 0];
tail=[tailx; tailey; tailz];
dtail=D*tail;
dtailx=dtail(1,:)+xcg(iframe);
dtaily=dtail(2,:)+ycg(iframe);
dtailz=dtail(3,:)+zcg(iframe);
fill3(dtaily,dtaily,dtailz,'c'); % Draw Tailplane

%
% Calculate Fuselage
%
t=linspace(0,pi,11);
r=sin(t).^.5;
[fusz,fusy,fusx]=cylinder(r);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-.75*ones(size(fusx)));

[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz); % Draw Fuselage
%
% Calculate Fin
%
tailr=.75*fusl;
tailf=tailr-tailchord;
finx=[-tailr -tailr -tailf -tailf];
finy=[0 0 0 0];
finz=[0 finh finh 0];
fin=[finx; finy; finz];
dfin=D*fin;
dfinx=dfin(1,:)+xcg(iframe);

```



```

dfiny=dfin(2,:)+ycg(iframe);
dfinz=dfin(3,:)+zcg(iframe);
fill3(dfinx,dfiny,dfinz,'c'); % Draw Fin
%
% Calculate Prop
%
for iprop=1:73
    propaz=2*pi*(iprop-1)/72;
    propx(iprop)=.25*fusl;
    propy(iprop)=propr*cos(propaz);
    propz(iprop)=propr*sin(propaz);
    prop=[propx'; propy'; propz'];
    dprop=D*prop;
    dpropx=dprop(1,:)+xcg(iframe);
    dpropy=dprop(2,:)+ycg(iframe);
    dpropz=dprop(3,:)+zcg(iframe);
end
fill3(dpropx,dpropy,dpropz,'w'); % Draw Prop
%
% Calculate Wing Tip Vortices - If Required
%
if tipvortex~=0
    dvortlx(1)=dwingx(1);
    dvortrx(1)=dwingx(2);
    dvortly(1)=dwingy(1);
    dvortry(1)=dwingy(2);
    dvortlz(1)=dwingz(1);
    dvortrz(1)=dwingz(2);

    dvortlx(2)=xmin;
    dvortrx(2)=xmin;
    dvortly(2)=dwingy(1);
    dvortry(2)=dwingy(2);
    dvortlz(2)=dwingz(1);
    dvortrz(2)=dwingz(2);
    %
    % Plot Wing Tip Vortices
    %

    plot3(dvortlx,dvortly,dvortlz,'linewidth',3,'Color','b');
    plot3(dvortrx,dvortry,dvortrz,'linewidth',3,'Color','b');
end

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y =
',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll =
',num2str(phid(iframe)), '\circ', ' Pitch =
',num2str(thetaf(iframe)), '\circ', ' Yaw =
',num2str(psid(iframe)), '\circ'];
title(heading,'Background','b','Margin',5,'Edgecolor','y');
grid on
m(iframe)=getframe(gcf);

```



```
%axis equal  
end  
movie2avi(m,'jim','Compression','Cinepak');
```



---

# Rotary Wing

---

## *Spot Turn*

---

```
%  
%     Flight Mechanics Simulation - Helicopter  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = -40;  
el = -10;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
nframe=181;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
mainx=zeros(73,1);  
mainy=mainx;  
mainz=mainx;  
  
tailx=zeros(73,1);  
taily=tailx;  
tailz=tailx;  
  
nosegx=zeros(73,1);  
nosegy=nosegx;  
nosegz=nosegx;  
  
mainglx=zeros(73,1);  
maingly=mainglx;  
mainglz=mainglx;
```



```

maingrx=zeros(73,1);
maingry=maingrx;
maingrz=maingrx;

xcg=zeros(nframe,1);
ycg=zeros(nframe,1);
zcg=zeros(nframe,1);
phid=zeros(nframe,1);
thetad=zeros(nframe,1);
psid=zeros(nframe,1);

%=====Specify Aircraft Components
% Fuselage Specification
fusl=3;
fusr=1;
% Tail Boom Specification
booml=4;
boomr=.2;
% Main Rotor Specification
mainr=2.5;
mainstn=0.;
mainh=1.;
% Tail Rotor Specification
tailr=.6;
tailstn=-3.:
tailoffst=0.2;
tailh=.5;
% Tricycle Undercarriage Specification
wheelr=0.25;
wheelh=-1.0;
nosewhlx=1.0;
mainwhlx=-.6;
mainwhly=.7;
%=====
%?????????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
%=====Set Up Circle Angles
angmovd=linspace(0,360,nframe);% 1 Cycle in Degrees
angmovr=angmovd*pi/180;% 1 Cycle in Radians

for iframe=1:nframe
%=====Specify Euler Angles in Degrees
%phid(iframe)=angmovd(iframe);
psid(iframe)=angmovd(iframe);
%thetad(iframe)=-15;
%=====Specify - CG Position
xcg(iframe)=0;
ycg(iframe)=0. ;
zcg(iframe)=0;
%=====
end
%=====

%?????????????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine

```



```

phir=phid(iframe)*pi/180;
thetar=thetad(iframe)*pi/180;
psir=psid(iframe)*pi/180;
%=====
cphi=cos(phir);
sphi=sin(phir);
ctheta=cos(theta);
stheta=sin(theta);
cpsi=cos(psir);
spsi=sin(psir);

***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,stheta;0,1,0;-stheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
*****


view(az, el);
hold on
%
% Calculate Fuselage
%
tf=linspace(-1,1,21);%+++++
rf=sqrt(1-tf.^2);%+++++
[fusz,fusy,fusx]=cylinder(rf);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-0.5*ones(size(fusx)));


[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz,'FaceColor','y','EdgeColor','none'); % Draw
Fuselage

%
% Calculate Tail Boom
%
t=linspace(0,pi,11);
r=sin(t).^5;
[boomz,boomy,boomx]=cylinder(r);
boomz=boomr*boomz;
boomy=boomr*boomy;
boomx=booml*(boomx-.75*ones(size(boomx)));


[nx,ny]=size(boomx);
for ix=1:nx
    for iy=1:ny
        pt=[boomx(ix,iy);boomy(ix,iy);boomz(ix,iy)];
    end
end

```



```

dpt=D*pt;
dboomx(ix,iy)=dpt(1)+xcg(iframe);
dboomy(ix,iy)=dpt(2)+ycg(iframe);
dboomz(ix,iy)=dpt(3)+zcg(iframe);
end
end
surf(dboomx, dboomy, dboomz, 'FaceColor', 'blue', 'EdgeColor', 'none'); %
Draw Tail Boom

%
% Calculate Main Rotor
%
for imain=1:73
    mainaz=2*pi*(imain-1)/72;
    mainz(imain)=mainh;
    mainx(imain)=mainr*cos(mainaz)+mainstn;
    mainy(imain)=mainr*sin(mainaz);
    main=[mainx'; mainy'; mainz'];
    dmain=D*main;
        dmainx=dmain(1,:)+xcg(iframe);
        dmainy=dmain(2,:)+ycg(iframe);
        dmainz=dmain(3,:)+zcg(iframe);
    end
fill3(dmainx,dmainy,dmainz,'c'); % Draw Main Rotor

%
% Calculate Tail Rotor
%
for itail=1:73
    tailaz=2*pi*(itail-1)/72;
    taily(itail)=tailoffst;
    tailx(itail)=tailr*cos(tailaz)+tailstn;
    tailz(itail)=tailr*sin(tailaz)+tailh;
    tail=[tailx'; taily'; tailz'];
    dtail=D*tail;
        dtailx=dtail(1,:)+xcg(iframe);
        dtaily=dtail(2,:)+ycg(iframe);
        dtailz=dtail(3,:)+zcg(iframe);
    end
fill3(dtailx,dtaily,dtailz,'r'); % Draw Tail Rotor

%
% Calculate Nose Gear
%
for inoseg=1:73
    nosegaz=2*pi*(inoseg-1)/72;
    nosegy(inoseg)=0;
    nosegx(inoseg)=wheelr*cos(nosegaz)+nosewhlx;
    nosegz(inoseg)=wheelr*sin(nosegaz)+wheelh;
    noseg=[nosegx'; nosegy'; nosegz'];
    dnoseg=D*noseg;
        dnosegx=dnoseg(1,:)+xcg(iframe);
        dnosegy=dnoseg(2,:)+ycg(iframe);
        dnosegz=dnoseg(3,:)+zcg(iframe);
    end

```



```

fill3(dnosegx,dnosegy,dnosegz,'r'); % Draw Nose Wheel

%
% Calculate Main Wheel Left
%
for imaingl=1:73
    mainglaz=2*pi*(imaingl-1)/72;
    maingly(imaingl)=mainwhly;
    mainglx(imaingl)=wheelr*cos(mainglaz)+mainwhlx;
    mainglz(imaingl)=wheelr*sin(mainglaz)+wheelh;
    maingl=[mainglx'; maingly'; mainglz'];

    dmaingl=D*maingl;
    dmainglx=dmaingl(1,:)+xcg(iframe);
    dmaingly=dmaingl(2,:)+ycg(iframe);
    dmainglz=dmaingl(3,:)+zcg(iframe);
end
fill3(dmainglx,dmaingly,dmainglz,'m'); % Draw Main Wheel Left

%
% Calculate Main Wheel Right
%
for imaingr=1:73
    maingraz=2*pi*(imaingr-1)/72;
    maingry(imaingr)=-mainwhly;
    maingrx(imaingr)=wheelr*cos(maingraz)+mainwhlx;
    maingrz(imaingr)=wheelr*sin(maingraz)+wheelh;
    maingr=[maingrx'; maingry'; maingrz'];

    dmaingr=D*maingr;
    dmaingrx=dmaingr(1,:)+xcg(iframe);
    dmaingry=dmaingr(2,:)+ycg(iframe);
    dmaingrz=dmaingr(3,:)+zcg(iframe);
end
fill3(dmaingrx,dmaingry,dmaingrz,'w'); % Draw Main Wheel Right

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y = ',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll = ',num2str(phid(iframe)), '\circ', ' Pitch = ',num2str(thetaf(iframe)), '\circ', ' Yaw = ',num2str(psi(iframe)), '\circ'];
title(heading,'Background','b','Margin',5,'Edgecolor','y');
grid on

m(iframe)=getframe(gcf);

%
%axis equal
end

movie2avi(m,'jim','Compression','None');

```



## Deck Landing

---

```
%  
%     Flight Mechanics Simulation - Helicopter - Deck Landing  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = 20;  
el = -20;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
napp=11; % Approach to Ship  
ntrans=11; % Transit  
ntd=11; % Descent & Touchdown on Deck  
nframe=napp+ntrans+ntd;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
mainx=zeros(73,1);  
mainy=mainx;  
mainz=mainx;  
  
tailx=zeros(73,1);  
taily=tailx;  
tailz=tailx;  
  
nosegx=zeros(73,1);  
nosegy=nosegx;  
nosegz=nosegx;  
  
mainglx=zeros(73,1);  
maingly=mainglx;  
mainglz=mainglx;  
  
maingrx=zeros(73,1);  
maingry=maingrx;  
maingrz=maingrx;  
  
xcg=zeros(nframe,1);  
ycg=zeros(nframe,1);
```



```

zcg=zeros(nframe,1);
phid=zeros(nframe,1);
thetad=zeros(nframe,1);
psid=zeros(nframe,1);

%=====Specify Aircraft Components
% Fuselage Specification
fusl=3;
fusr=1;
% Tail Boom Specification
booml=4;
boomr=.2;
% Main Rotor Specification
mainr=2.5;
mainstn=0.;
mainh=1.;
% Tail Rotor Specification
tailr=.6;
tailstn=-3.;
tailoffst=0.2;
tailh=.5;
% Tricycle Undercarriage Specification
wheelr=0.25;
wheelh=-1.0;
nosewhlx=1.0;
mainwhlx=-.6;
mainwhly=.7;

zcgondeck=0;
%=====Specify Deck + Markings & Hangar Door Dimensions
hhangar=3.5;
deckw=6; % Overall Deck Width
deckl=8; % Overall Deck Length
hdoorbottom=wheeh-wheelr;
zhangar=hhangar+hdoorbottom; % Z Coordinate of Hangar Roof
harpoonr=1.5;
hapi=1; % Height of Approach Path Indicator above Hangar Roof
decklight=1; % Deck Lighting ? (0 => NO)

s=deckw/2;
l=deckl/2;
deckx=[-1 -1 1 1];
decky=[-s s s -s];
deckz=(wheeh-wheelr)*ones(1,4);

s=deckw/2;
hgrx=l*ones(1,4);
hgry=[-s s s -s];
hgrz=[zhangar,zhangar,hdoorbottom,hdoorbottom];

buttlinex=zeros(1,2);
buttlincey=[-s s];
buttlinez=hdoorbottom*ones(1,2);

centrelinex=[-1 0];
centreliney=zeros(1,2);
centrelinez=hdoorbottom*ones(1,2);

```



```

ang=linspace(0,2*pi,72);
harpoonx=harpoonr*cos(ang)-harpoonr;
harpoony=harpoonr*sin(ang);
harpoonz=hdoorbottom*ones(1,72);

%=====
%?????????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
% Approach Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgapp=linspace(-2*mainr,0,napp);
ycgapp=(mainr+deckw/2)*ones(1,napp);
zcgapp=(zhangar-mainh)*ones(1,napp);
%=====
% Transit Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgtrans=zeros(1,ntrans);
ycgtrans=linspace(mainr+deckw/2,0,ntrans);
zcgtrans=(zhangar-mainh)*ones(1,napp);
%=====
% Touchdown Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgtd=zeros(1,ntd);
ycgtd=zeros(1,ntd);
h1=zhangar-mainh;
h2=zcgondeck;
zcgtd=h1*ones(1,ntd)+(h2-h1)*linspace(0,1,ntd);
%=====
% Screw the Whole Manoeuvre Together
xcg=[xcgapp,xcgtrans,xcgtd];
ycg=[ycgapp,ycgtrans,ycgtd];
zcg=[zcgapp,zcgtrans,zcgtd];
phid=zeros(1,nframe);
thetad=zeros(1,nframe);
psid=zeros(1,nframe);
%?????????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
    clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
    phir=phid(iframe)*pi/180;
    thetar=thetad(iframe)*pi/180;
    psir=psid(iframe)*pi/180;
%=====
    cphi=cos(phir);
    sphi=sin(phir);
    ctheta=cos(theta);
    stheta=sin(theta);
    cpsi=cos(psir);
    spsi=sin(psir);

```



```
%***** Rotation Matrix
mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,sttheta;0,1,0;-sttheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
%***** 

view(az, el);
hold on
%
% Calculate Deck Surface
%
fill3(deckx,decky,deckz,[.5 .5 .5],'EdgeColor',[.7 .7 .7]); %
Draw Deck Surface

%
% Calculate Deck Markings
%
plot3(buttlinex,buttliney,buttlinez,'w'); % Draw Butt Line
plot3(centrelinex,centreliney,centrelinez,'w'); % Draw Centre Line
plot3(harpoonx,harpoony,harpoonz,'w'); % Draw Harpoon Circle

%
% Calculate Hangar Door Surface
%
hgr=[hgrx; hgy; hgz];
dhgr=D*hgr;
dhgrx=dhgr(1,:);
dhgry=dhgr(2,:);
dhgrz=dhgr(3,:);
fill3(dhgrx,dhgry,dhgrz,.25*[.5 .5 .5],'EdgeColor',[.7 .7 .7]); %
Draw Hangar Door

%
% Calculate Fuselage
%
tf=linspace(-1,1,21);%+++++
rf=sqrt(1-tf.^2);%+++++
[fusz,fusy,fusx]=cylinder(rf);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-0.5*ones(size(fusx))));

[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz,'FaceColor','y','EdgeColor','none'); % Draw
Fuselage
```



```

%
% Calculate Tail Boom
%
t=linspace(0,pi,11);
r=sin(t).^.5;
[boomz,boomy,boomx]=cylinder(r);
boomz=boomr*boomz;
boomy=boomr*boomy;
boomx=booml*(boomx-.75*ones(size(boomx)));
nx,ny]=size(boomx);
for ix=1:nx
    for iy=1:ny
        pt=[boomx(ix,iy);boomy(ix,iy);boomz(ix,iy)];
        dpt=D*pt;
        dboomx(ix,iy)=dpt(1)+xcg(iframe);
        dboomy(ix,iy)=dpt(2)+ycg(iframe);
        dboomz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dboomx,dboomy,dboomz,'FaceColor','blue','EdgeColor','none'); %
Draw Tail Boom

%
% Calculate Main Rotor
%
for imain=1:73
    mainaz=2*pi*(imain-1)/72;
    mainz(imain)=mainh;
    mainx(imain)=mainr*cos(mainaz)+mainstn;
    mainy(imain)=mainr*sin(mainaz);
    main=[mainx'; mainy'; mainz'];
    dmain=D*main;
    dmainx=dmain(:,1)+xcg(iframe);
    dmainy=dmain(:,2)+ycg(iframe);
    dmainz=dmain(:,3)+zcg(iframe);
end
fill3(dmainx,dmainy,dmainz,'c'); % Draw Main Rotor

%
% Calculate Tail Rotor
%
for itail=1:73
    tailaz=2*pi*(itail-1)/72;
    taily(itail)=tailoffst;
    tailx(itail)=tailr*cos(tailaz)+tailstn;
    tailz(itail)=tailr*sin(tailaz)+tailh;
    tail=[tailx'; taily'; tailz'];
    dtail=D*tail;
    dtailx=dtail(:,1)+xcg(iframe);
    dtaily=dtail(:,2)+ycg(iframe);
    dtailz=dtail(:,3)+zcg(iframe);
end
fill3(dtailx,dtaily,dtailz,'r'); % Draw Tail Rotor
%
```



```

% Calculate Nose Gear
%
for inoseg=1:73
    nosegaz=2*pi*(inoseg-1)/72;
    nosegy(inoseg)=0;
    nosegx(inoseg)=wheelr*cos(nosegaz)+nosewhlx;
    nosegz(inoseg)=wheelr*sin(nosegaz)+wheelh;
    noseg=[nosegx'; nosegy'; nosegz'];

    dnoseg=D*noseg;
    dnosegx=dnoseg(1,:)+xcg(iframe);
    dnosegy=dnoseg(2,:)+ycg(iframe);
    dnosegz=dnoseg(3,:)+zcg(iframe);
end
fill3(dnosegx,dnosegy,dnosegz,'r'); % Draw Nose Wheel

%
% Calculate Main Wheel Left
%
for imaingl=1:73
    mainglaz=2*pi*(imaingl-1)/72;
    maingly(imaingl)=mainwhly;
    mainglx(imaingl)=wheelr*cos(mainglaz)+mainwhlx;
    mainglz(imaingl)=wheelr*sin(mainglaz)+wheelh;
    maingl=[mainglx'; maingly'; mainglz'];

    dmaingl=D*maingl;
    dmainglx=dmaingl(1,:)+xcg(iframe);
    dmaingly=dmaingl(2,:)+ycg(iframe);
    dmainglz=dmaingl(3,:)+zcg(iframe);
end
fill3(dmainglx,dmaingly,dmainglz,'m'); % Draw Main Wheel Left

%
% Calculate Main Wheel Right
%
for imaingr=1:73
    maingraz=2*pi*(imaingr-1)/72;
    maingry(imaingr)=-mainwhly;
    maingrx(imaingr)=wheelr*cos(maingraz)+mainwhlx;
    maingrz(imaingr)=wheelr*sin(maingraz)+wheelh;
    maingr=[maingrx'; maingry'; maingrz'];

    dmaingr=D*maingr;
    dmaingrx=dmaingr(1,:)+xcg(iframe);
    dmaingry=dmaingr(2,:)+ycg(iframe);
    dmaingrz=dmaingr(3,:)+zcg(iframe);
end
fill3(dmaingrx,dmaingry,dmaingrz,'w'); % Draw Main Wheel Right

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y = ',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll = ',num2str(phid(iframe)), '\circ', ' Pitch = ',num2str(thetaf(iframe)), '\circ', ' Yaw = ',num2str(psi(f)), '\circ'];

```



```
%title(heading,'Color','w','Background','b','Margin',5,'Edgecolor
','y');
title(heading,'Color','y');
grid on

if decklight~=0
% Add in Deck Lighting=====
ltx=[deckl/2,-deckl/2,-deckl/2];
lty=[0,-deckw/2,deckw/2];
ltz=[zhangar+hapi,hdoorbottom,hdoorbottom];
lightr=.2;
[xlight,ylight,zlight]=sphere(11);
for ilt=1:3
    xlt=lightr*xlight+ltx(ilt);
    ylt=lightr*ylight+lty(ilt);
    zlt=lightr*zlight+ltz(ilt);
    surf(xlt,ylt,zlt,'FaceColor','w','EdgeColor','none');

end
light('Position',[deckl/2 0 zhangar],'Style','local');
light('Position',[-deckl/2 -deckw/2 hdoorbottom],'Style','local');
light('Position',[deckl/2 deckw/2 hdoorbottom],'Style','local');
=====
end

m(iframe)=getframe(gcf);

%axis equal
end

%movie2avi(m,'jim','Compression','None');
```



## Deck Operations

---

```
%  
%     Flight Mechanics Simulation - Helicopter - Deck Landing  
%  
%=====Set Up Colours & Perspective View  
clear  
colordef black  
camproj('perspective');  
%=====Set Up View Angles  
az = 20;  
el = -20;  
%=====  
xmin=-5;  
xmax=5;  
ymin=-5;  
ymax=5;  
zmin=-5;  
zmax=5;  
%=====Define Number of Time Steps  
napp=11; % Approach to Ship  
ntrans=11; % Transit  
ntd=11; % Descent & Touchdown on Deck  
nframe=napp+ntrans+ntd;  
%=====Set Up Holding Arrays  
  
dpt=zeros(3,1);  
  
mainx=zeros(73,1);  
mainy=mainx;  
mainz=mainx;  
  
tailx=zeros(73,1);  
taily=tailx;  
tailz=tailx;  
  
nosegx=zeros(73,1);  
nosegy=nosegx;  
nosegz=nosegx;  
  
mainglx=zeros(73,1);  
maingly=mainglx;  
mainglz=mainglx;  
  
maingrx=zeros(73,1);  
maingry=maingrx;  
maingrz=maingrx;  
  
xcg=zeros(nframe,1);  
ycg=zeros(nframe,1);  
zcg=zeros(nframe,1);  
phid=zeros(nframe,1);
```



```

thetad=zeros(nframe,1);
psid=zeros(nframe,1);

%=====Specify Aircraft Components
% Fuselage Specification
fusl=3;
fusr=1;
% Tail Boom Specification
booml=4;
boomr=.2;
% Main Rotor Specification
mainr=2.5;
mainstn=0.;
mainh=1.;
% Tail Rotor Specification
tailr=.6;
tailstn=-3.;
tailoffst=0.2;
tailh=.5;
% Tricycle Undercarriage Specification
wheelr=0.25;
wheelh=-1.0;
nosewhlx=1.0;
mainwhlx=-.6;
mainwhly=.7;

zcgondeck=0;
%=====Specify Deck + Markings & Hangar Door Dimensions
hhangar=3.5;
deckw=6; % Overall Deck Width
deckl=8; % Overall Deck Length
hdoorbottom=wheelh-wheelr;
zhangar=hhangar+hdoorbottom; % Z Coordinate of Hangar Roof
harpoonr=1.5;
hapi=1; % Height of Approach Path Indicator above Hangar Roof
decklight=1; % Deck Lighting ? (0 => NO)

s=deckw/2;
l=deckl/2;
deckx=[-l -l l l];
decky=[-s s s -s];
deckz=(wheelh-wheelr)*ones(1,4);

s=deckw/2;
hgrx=l*ones(1,4);
hgry=[-s s s -s];
hgrz=[zhangar,zhangar,hdoorbottom,hdoorbottom];

buttlinex=zeros(1,2);
buttliney=[-s s];
buttlinez=hdoorbottom*ones(1,2);

centrelinex=[-1 0];
centreliney=zeros(1,2);
centrelinez=hdoorbottom*ones(1,2);

ang=linspace(0,2*pi,72);

```



```

harpoonx=harpoonr*cos(ang)-harpoonr;
harpoony=harpoonr*sin(ang);
harpoonz=hdoorbottom*ones(1,72);

%=====
%?????????????????????????????????????????????????????????????
%=====Set Up Motion Arrays
% Approach Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgapp=linspace(-2*mainr,0,napp);
ycgapp=(mainr+deckw/2)*ones(1,napp);
zcgapp=(zhangar-mainh)*ones(1,napp);
%=====
% Transit Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgtrans=zeros(1,ntrans);
ycgtrans=linspace(mainr+deckw/2,0,ntrans);
zcgtrans=(zhangar-mainh)*ones(1,napp);
%=====
% Touchdown Phase
%=====Specify Euler Angles in Degrees
% NUFFINK
%=====Specify - CG Position
xcgtd=zeros(1,ntd);
ycgtd=zeros(1,ntd);
h1=zhangar-mainh;
h2=zcgondeck;
zcgtd=h1*ones(1,ntd)+(h2-h1)*linspace(0,1,ntd);
%=====
% Screw the Whole Manoeuvre Together
xcg=[xcgapp,xcgtrans,xcgtd];
ycg=[ycgapp,ycgtrans,ycgtd];
zcg=[zcgapp,zcgtrans,zcgtd];
phid=zeros(1,nframe);
thetad=zeros(1,nframe);
psid=zeros(1,nframe);
%=====
%?????????????????????????????????????????????????????????????
%=====Perform Simulation
for iframe=1:nframe
    clf
%=====Convert Current Euler Angles to Radians + Sine *
Cosine
    phir=phid(iframe)*pi/180;
    thetar=thetad(iframe)*pi/180;
    psir=psid(iframe)*pi/180;
%=====
    cphi=cos(phir);
    sphi=sin(phir);
    ctheta=cos(thetar);
    stheta=sin(thetar);
    cpsi=cos(psir);
    spsi=sin(psir);
%***** Rotation Matrix

```



```

mroll=[1,0,0;0,cphi,-sphi;0,sphi,cphi];
mpitch=[ctheta,0,sttheta;0,1,0;-sttheta,0,ctheta];
myaw=[cpsi,-spsi,0;spsi,cpsi,0;0,0,1];
D=myaw*mpitch*mroll;
%***** ****

view(az, el);
hold on
%
% Calculate Deck Surface
%
fill3(deckx,decky,deckz,[.5 .5 .5],'EdgeColor',[.7 .7 .7]); %
Draw Deck Surface

%
% Calculate Deck Markings
%
plot3(buttlinex,buttliney,buttlinez,'w'); % Draw Butt Line
plot3(centrelinex,centreliney,centrelinez,'w'); % Draw Centre Line
plot3(harpoonx,harpoony,harpoonz,'w'); % Draw Harpoon Circle

%
% Calculate Hangar Door Surface
%
hgr=[hgrx; hgry; hgrz];
dhgr=D*hgr;
dhgrx=dhgr(1,:);
dhgry=dhgr(2,:);
dhgrz=dhgr(3,:);
fill3(dhgrx,dhgry,dhgrz,.25*[.5 .5 .5],'EdgeColor',[.7 .7 .7]); %
Draw Hangar Door

%
% Calculate Fuselage
%
tf=linspace(-1,1,21);%+++++
rf=sqrt(1-tf.^2);%+++++
[fusz,fusy,fusx]=cylinder(rf);
fusz=fusr*fusz;
fusy=fusr*fusy;
fusx=fusl*(fusx-0.5*ones(size(fusx)));

[nx,ny]=size(fusx);
for ix=1:nx
    for iy=1:ny
        pt=[fusx(ix,iy);fusy(ix,iy);fusz(ix,iy)];
        dpt=D*pt;
        dfusx(ix,iy)=dpt(1)+xcg(iframe);
        dfusy(ix,iy)=dpt(2)+ycg(iframe);
        dfusz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dfusx,dfusy,dfusz,'FaceColor','y','EdgeColor','none'); % Draw
Fuselage

%
% Calculate Tail Boom

```



```

%
t=linspace(0,pi,11);
r=sin(t).^.5;
[boomz,boomy,boomx]=cylinder(r);
boomz=boomr*boomz;
boomy=boomr*boomy;
boomx=booml*(boomx-.75*ones(size(boomx))));

[nx,ny]=size(boomx);
for ix=1:nx
    for iy=1:ny
        pt=[boomx(ix,iy);boomy(ix,iy);boomz(ix,iy)];
        dpt=D*pt;
        dboomx(ix,iy)=dpt(1)+xcg(iframe);
        dboomy(ix,iy)=dpt(2)+ycg(iframe);
        dboomz(ix,iy)=dpt(3)+zcg(iframe);
    end
end
surf(dboomx,dboomy,dboomz,'FaceColor','blue','EdgeColor','none'); %
Draw Tail Boom

%
% Calculate Main Rotor
%
for imain=1:73
    mainaz=2*pi*(imain-1)/72;
    mainz(imain)=mainh;
    mainx(imain)=mainr*cos(mainaz)+mainstn;
    mainy(imain)=mainr*sin(mainaz);
    main=[mainx'; mainy'; mainz'];

    dmain=D*main;
    dmainx=dmain(:,1)+xcg(iframe);
    dmainy=dmain(:,2)+ycg(iframe);
    dmainz=dmain(:,3)+zcg(iframe);
end
fill3(dmainx,dmainy,dmainz,'c'); % Draw Main Rotor

%
% Calculate Tail Rotor
%
for itail=1:73
    tailaz=2*pi*(itail-1)/72;
    taily(itail)=tailoffst;
    tailx(itail)=tailr*cos(tailaz)+tailstn;
    tailz(itail)=tailr*sin(tailaz)+tailh;
    tail=[tailx'; taily'; tailz'];

    dtail=D*tail;
    dtailx=dtail(:,1)+xcg(iframe);
    dtaily=dtail(:,2)+ycg(iframe);
    dtailz=dtail(:,3)+zcg(iframe);
end
fill3(dtailx,dtaily,dtailz,'r'); % Draw Tail Rotor

%
% Calculate Nose Gear
%

```



```

for inoseg=1:73
    nosegaz=2*pi*(inoseg-1)/72;
    nosegy(inoseg)=0;
    nosegx(inoseg)=wheelr*cos(nosegaz)+nosewhlx;
    nosegz(inoseg)=wheelr*sin(nosegaz)+wheelh;
    noseg=[nosegx'; nosegy'; nosegz'];

    dnoseg=D*noseg;
    dnosegx=dnoseg(1,:)+xcg(iframe);
    dnosegy=dnoseg(2,:)+ycg(iframe);
    dnosegz=dnoseg(3,:)+zcg(iframe);
end
fill3(dnosegx,dnosegy,dnosegz,'r'); % Draw Nose Wheel

%
% Calculate Main Wheel Left
%
for imaingl=1:73
    mainglaz=2*pi*(imaingl-1)/72;
    maingly(imaingl)=mainwhly;
    mainglx(imaingl)=wheelr*cos(mainglaz)+mainwhlx;
    mainglz(imaingl)=wheelr*sin(mainglaz)+wheelh;
    maingl=[mainglx'; maingly'; mainglz'];

    dmaingl=D*maingl;
    dmainglx=dmaingl(1,:)+xcg(iframe);
    dmaingly=dmaingl(2,:)+ycg(iframe);
    dmainglz=dmaingl(3,:)+zcg(iframe);
end
fill3(dmainglx,dmaingly,dmainglz,'m'); % Draw Main Wheel Left

%
% Calculate Main Wheel Right
%
for imaingr=1:73
    maingraz=2*pi*(imaingr-1)/72;
    maingry(imaingr)=-mainwhly;
    maingrx(imaingr)=wheelr*cos(maingraz)+mainwhlx;
    maingrz(imaingr)=wheelr*sin(maingraz)+wheelh;
    maingr=[maingrx'; maingry'; maingrz'];

    dmaingr=D*maingr;
    dmaingrx=dmaingr(1,:)+xcg(iframe);
    dmaingry=dmaingr(2,:)+ycg(iframe);
    dmaingrz=dmaingr(3,:)+zcg(iframe);
end
fill3(dmaingrx,dmaingry,dmaingrz,'w'); % Draw Main Wheel Right

axis equal
axis([xmin xmax ymin ymax zmin zmax]);

heading=['X = ',num2str(xcg(iframe)), ' Y = ',num2str(ycg(iframe)), ' Z = ', num2str(zcg(iframe)), ' Roll = ',num2str(phid(iframe)), '\circ', ' Pitch = ',num2str(thetaf(iframe)), '\circ', ' Yaw = ',num2str(psi(f)), '\circ'];

```



```
%title(heading,'Color','w','Background','b','Margin',5,'Edgecolor
','y');
title(heading,'Color','y');
grid on

if decklight~=0
% Add in Deck Lighting=====
ltx=[deckl/2,-deckl/2,-deckl/2];
lty=[0,-deckw/2,deckw/2];
ltz=[zhangar+hapi,hdoorbottom,hdoorbottom];
lightr=.2;
[xlight,ylight,zlight]=sphere(11);
for ilt=1:3
    xlt=lightr*xlight+ltx(ilt);
    ylt=lightr*ylight+lty(ilt);
    zlt=lightr*zlight+ltz(ilt);
    surf(xlt,ylt,zlt,'FaceColor','w','EdgeColor','none');

end
light('Position',[deckl/2 0 zhangar],'Style','local');
light('Position',[-deckl/2 -deckw/2 hdoorbottom],'Style','local');
light('Position',[deckl/2 deckw/2 hdoorbottom],'Style','local');
=====
end

m(iframe)=getframe(gcf);

%axis equal
end

%movie2avi(m,'jim','Compression','None');
```

