UNIVERSITY OF Southampton
School of Engineering Sciences

Aerodynamics & Flight Mechanics Research Group

Modelling a Fuselage Profile in MATLAB

S. J. Newman

Technical Report AFM-11/06

January 2011

UNIVERSITY OF SOUTHAMPTON

SCHOOL OF ENGINEERING SCIENCES

AERODYNAMICS AND FLIGHT MECHANICS RESEARCH GROUP

Modelling a Fuselage Profile in MATLAB

by

S. J. Newman

AFM Report No. AFM 11/06

January 2011

Introduction

The following analysis is to produce a three dimensional picture of a generic fuselage shape using basic MATLAB commands. The fuselage is created by calculating the coordinates of a sequence of nodal points. These form a set of bulkheads down the length of the fuselage. For each bulkhead, the cross-section is defined by a polar equation based on the ROBIN fuselage profiles. These are based on the coordinate geometry of super ellipses. Having formed the grid of nodes, each set of four nodes – defines a set of patches. These are then plotted using the 'patch' command. The colour used for the patch plotting is first kept constant.

The analysis also examines the use varying the colour of the patch to indicate the projected frontal area and thereby providing a measure of the contribution to the drag of the fuselage.

Figure 1 - CH53 Super Stallion (Courtesy US Navy)

Specification of Fuselage

As example, the longitudinal variation of the size of the bulkheads is a piecewise function shown in Figure 2. The nose section is defined by an elliptical function. The central cabin is of constant dimension, and the tail section is as linear decrease to zero at the termination of the fuselage length.

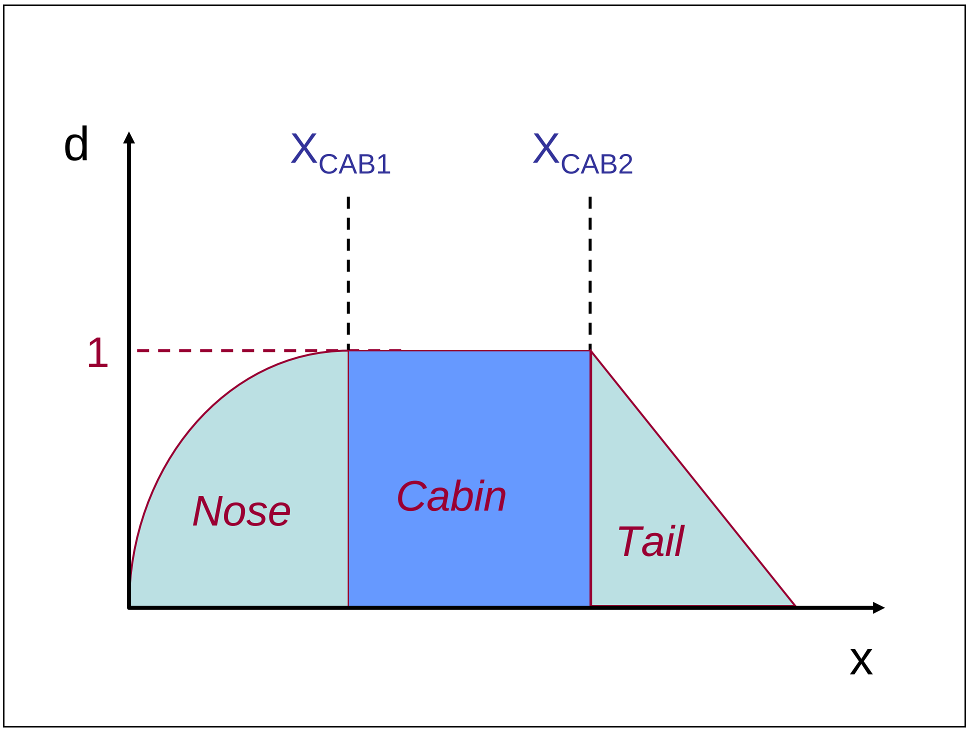

Figure 2 – Example Cross Section

The function is defined in (1).

$$d = \frac{1}{x_1}\sqrt{x(2x_1 - x)} \quad 0 \le x \le x_1$$
$$= 1 \quad x_1 \le x \le x_2 \qquad \textbf{(1.)}$$
$$= \frac{1-x}{1-x_2} \quad x_2 \le x \le 1$$

The cross-section dimensions are shown in Figure 3.

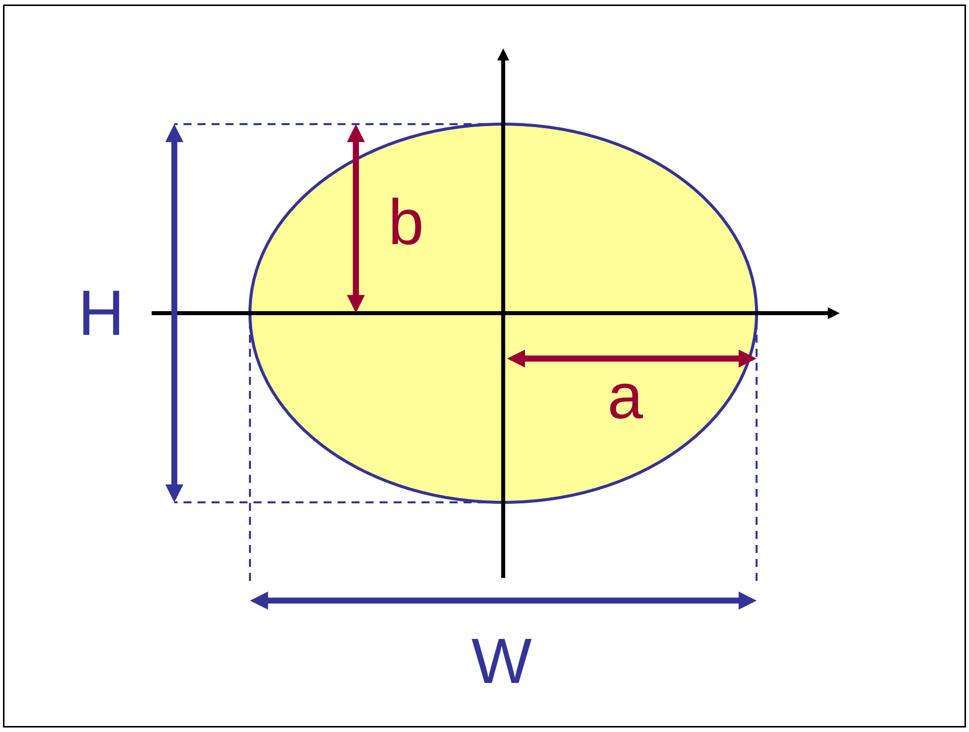

Figure 3 - Cross Section Dimensions

The fuselage bulkhead width and height come from (1) and the cross-section definition uses the semi major and minor axes as defined is (2).

$$a = \frac{W}{2}$$
$$b = \frac{H}{2} \qquad (2.)$$

At each cross-section, the shape is defined by a polar form - (r,ϕ) - with a Z_0 addition which permits a camber to be specified.

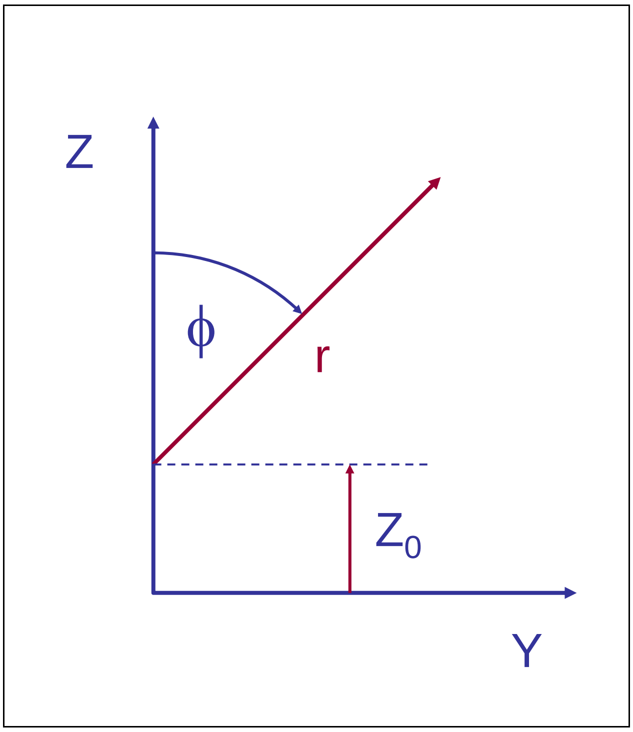

Figure 4 - Cross Section Specification

The super-ellipse profile is obtained using the equations shown in (3).

$$r^N \cdot \left\{ \left| \frac{\sin\phi}{a} \right|^N + \left| \frac{\cos\phi}{b} \right|^N \right\} = 1$$

$$r = \left\{ \left| \frac{\sin\phi}{a} \right|^N + \left| \frac{\cos\phi}{b} \right|^N \right\}^{-\frac{1}{N}} \quad \textbf{(3.)}$$

From this, the nodal points of the fuselage can now be calculated using (4).

$$\begin{aligned} X &= Specified \\ a &= a(X) \\ b &= b(X) \\ N &= N(X) \\ Y &= r \cdot \sin\phi \\ Z &= Z_0 + r \cdot \cos\phi \end{aligned} \quad \textbf{(4.)}$$

An example of this method is shown in Figure 5. The patches are facets each given a cyan colour with a blue edge.

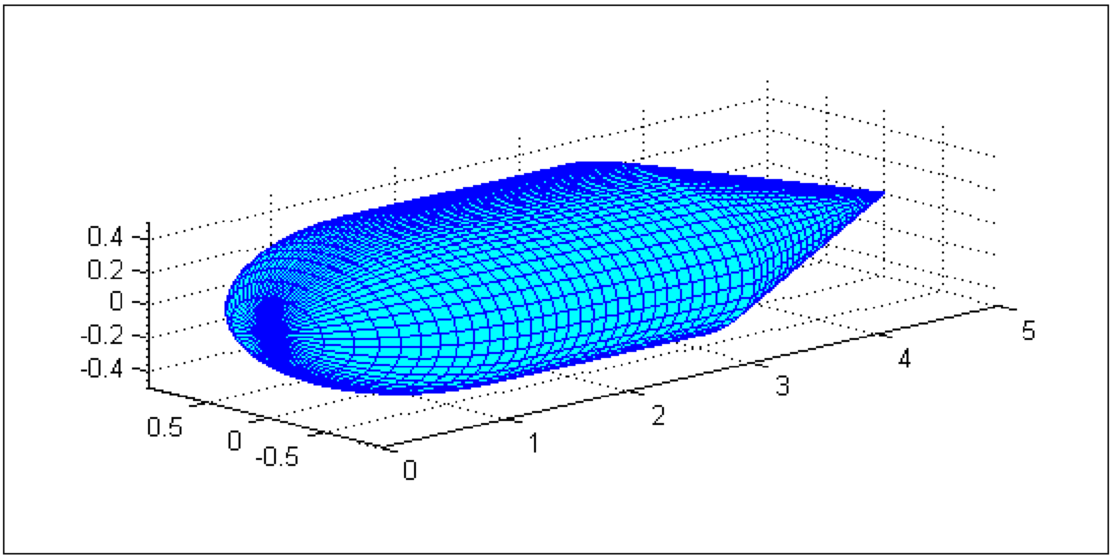

Figure 5 - Faceted Body with Fixed Patch Colour

Projected Frontal Area

One aspect of the body so created is the drag force. This can be viewed as the projected frontal area of each patch in a specified direction, or just the patch area unit vector resolved into the specified direction.

The area of the patch can be obtained by using the cross product of vectors defining two adjacent sides of the patch. There I also the potential problem of these vectors giving a zero cross product. This would happen at the node of the body where the patches are triangular. In this case the other two vectors can be used to obtain the cross product. The value taken can be either the sum of the two results or their maximum value. Figure 6 shows the generic patch geometry with the cross product.

The numbering system for the four corners of the patch are shown in Figure 7, together with the index notation used *(1 represents moving down the length of the body whilst j represents moving around the perimeter of a bulkhead)*.

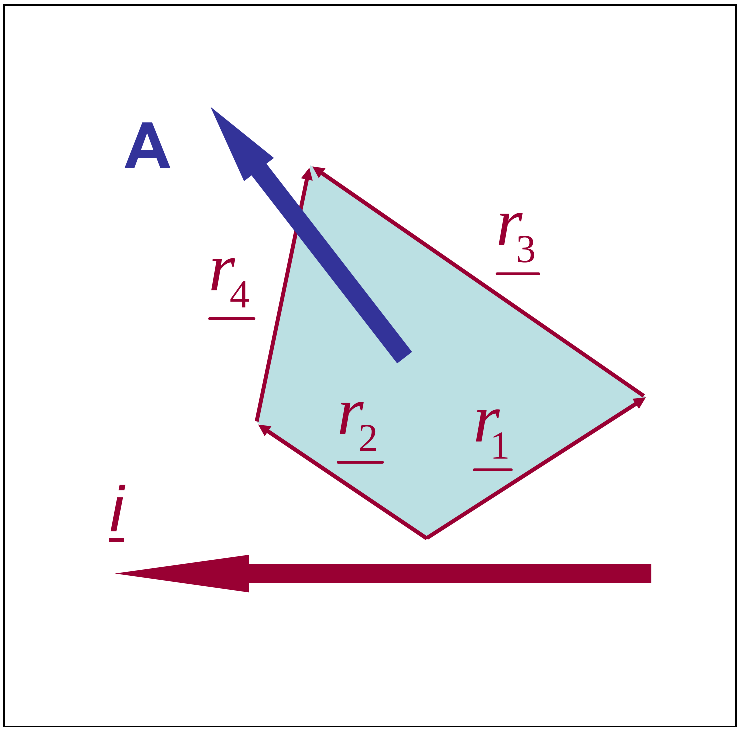

Figure 6 - Patch Definition

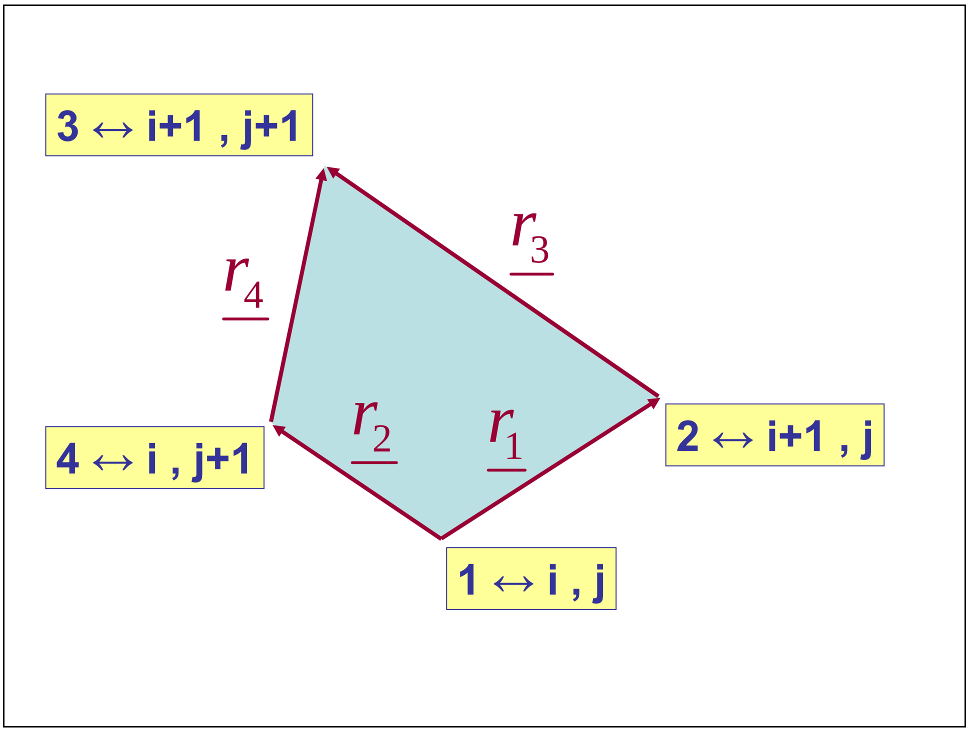

Figure 7 – Generic Patch Definition – Index Convention

With this convention the four defining vectors are:

$$\begin{aligned}
\underline{r}_1 &= (X_{i+1,j} - X_{i,j}, Y_{i+1,j} - Y_{i,j}, Z_{i+1,j} - Z_{i,j}) \\
\underline{r}_2 &= (X_{i,j+1} - X_{i,j}, Y_{i,j+1} - Y_{i,j}, Z_{i,j+1} - Z_{i,j}) \\
\underline{r}_3 &= (X_{i+1,j+1} - X_{i+1,j}, Y_{i+1,j+1} - Y_{i+1,j}, Z_{i+1,j+1} - Z_{i+1,j}) \\
\underline{r}_4 &= (X_{i+1,j+1} - X_{i,j+1}, Y_{i+1,j+1} - Y_{i,j+1}, Z_{i+1,j+1} - Z_{i,j+1})
\end{aligned} \qquad (5.)$$

If the direction required is defined by the unit vector $\underline{\hat{n}}$, then the projected patch area is given by:

$$\frac{\left|\left(\underline{r}_1 \wedge \underline{r}_2\right)\cdot \underline{\hat{n}}\right| + \left|\left(\underline{r}_4 \wedge \underline{r}_3\right)\cdot \underline{\hat{n}}\right|}{2} \qquad (6.)$$

If the direction unit vector has components:

$$\underline{\hat{n}} = (n_x, n_y, n_z) \qquad \textbf{(7.)}$$

We have the following results:

$$\underline{r_1} \wedge \underline{r_2} = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ X_{i+1,j} - X_{i,j} & Y_{i+1,j} - Y_{i,j} & Z_{i+1,j} - Z_{i,j} \\ X_{i,j+1} - X_{i,j} & Y_{i,j+1} - Y_{i,j} & Z_{i,j+1} - Z_{i,j} \end{vmatrix}$$

$$\textbf{(8.)}$$

$$\underline{r_4} \wedge \underline{r_3} = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ X_{i+1,j+1} - X_{i,j+1} & Y_{i+1,j+1} - Y_{i,j+1} & Z_{i+1,j+1} - Z_{i,j+1} \\ X_{i+1,j+1} - X_{i+1,j} & Y_{i+1,j+1} - Y_{i+1,j} & Z_{i+1,j+1} - Z_{i+1,j} \end{vmatrix}$$

$$\left(\underline{r_1} \wedge \underline{r_2}\right) \cdot \underline{\hat{n}} = \begin{vmatrix} n_x & n_y & n_z \\ X_{i+1,j} - X_{i,j} & Y_{i+1,j} - Y_{i,j} & Z_{i+1,j} - Z_{i,j} \\ X_{i,j+1} - X_{i,j} & Y_{i,j+1} - Y_{i,j} & Z_{i,j+1} - Z_{i,j} \end{vmatrix}$$

$$\textbf{(9.)}$$

$$\left(\underline{r_4} \wedge \underline{r_3}\right) \cdot \underline{\hat{n}} = \begin{vmatrix} n_x & n_y & n_z \\ X_{i+1,j+1} - X_{i,j+1} & Y_{i+1,j+1} - Y_{i,j+1} & Z_{i+1,j+1} - Z_{i,j+1} \\ X_{i+1,j+1} - X_{i+1,j} & Y_{i+1,j+1} - Y_{i+1,j} & Z_{i+1,j+1} - Z_{i+1,j} \end{vmatrix}$$

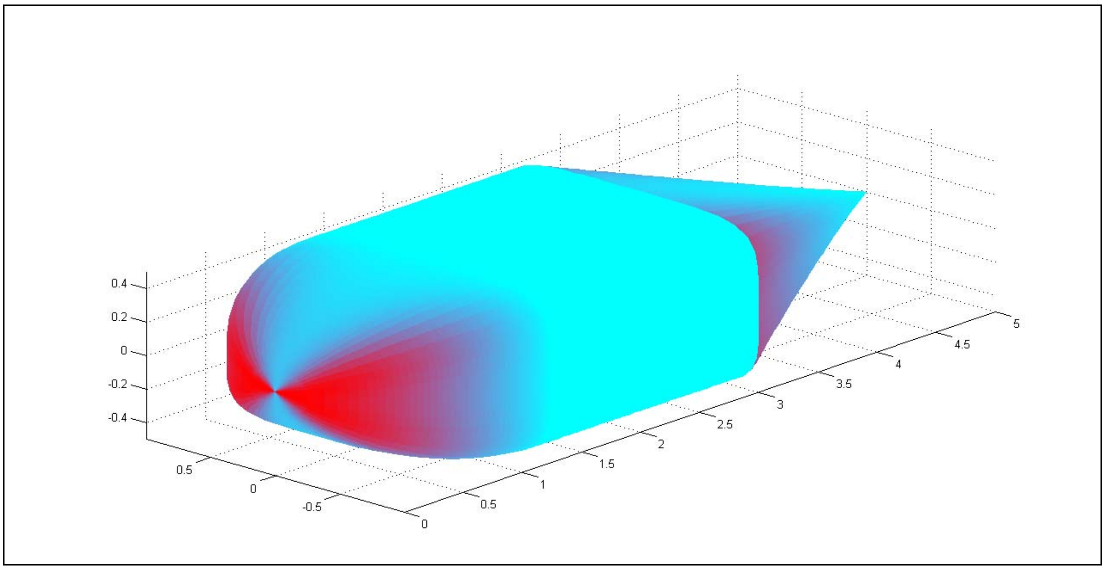

Figure 8 - Smooth Body with Patch Colour determined by Projected Frontal Area

Patch Direction Indication

If the direction of the patch is required without any scaling due to area then the results become:

$$\frac{\left|\left(\hat{\underline{r}}_1 \wedge \hat{\underline{r}}_2\right)\cdot \hat{\underline{n}}\right| + \left|\left(\hat{\underline{r}}_4 \wedge \hat{\underline{r}}_3\right)\cdot \hat{\underline{n}}\right|}{2} \tag{10.}$$

Where the hat symbol on the terms $r_1 - r_4$ refers to the appropriate unit vector.

$$\left(\hat{\underline{r}}_1 \wedge \hat{\underline{r}}_2\right)\cdot \hat{\underline{n}} = \frac{\begin{vmatrix} n_x & n_y & n_z \\ X_{i+1,j} - X_{i,j} & Y_{i+1,j} - Y_{i,j} & Z_{i+1,j} - Z_{i,j} \\ X_{i,j+1} - X_{i,j} & Y_{i,j+1} - Y_{i,j} & Z_{i,j+1} - Z_{i,j} \end{vmatrix}}{|r_1| \cdot |r_2|}$$

$$\left(\hat{\underline{r}}_4 \wedge \hat{\underline{r}}_3\right)\cdot \hat{\underline{n}} = \frac{\begin{vmatrix} n_x & n_y & n_z \\ X_{i+1,j+1} - X_{i,j+1} & Y_{i+1,j+1} - Y_{i,j+1} & Z_{i+1,j+1} - Z_{i,j+1} \\ X_{i+1,j+1} - X_{i+1,j} & Y_{i+1,j+1} - Y_{i+1,j} & Z_{i+1,j+1} - Z_{i+1,j} \end{vmatrix}}{|r_3| \cdot |r_4|} \tag{11.}$$

The following result is obtained for the same body:

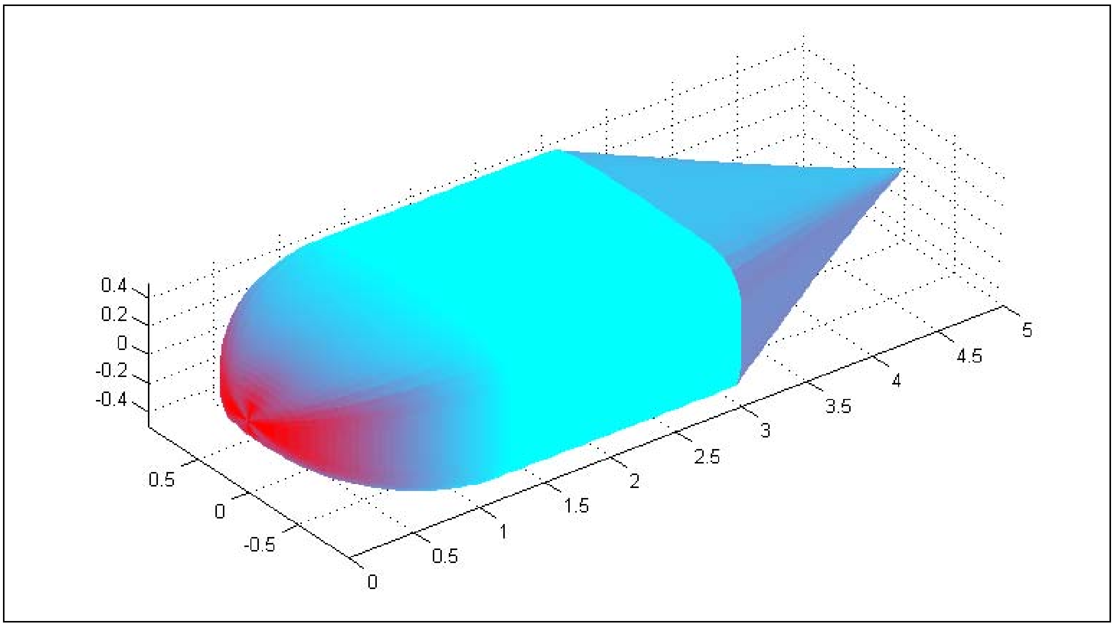

Figure 9 - Patch Orientation

Definition of Incident Direction

The incident direction can be specified by using the azimuth & elevation angles. These are shown in Figure 10.

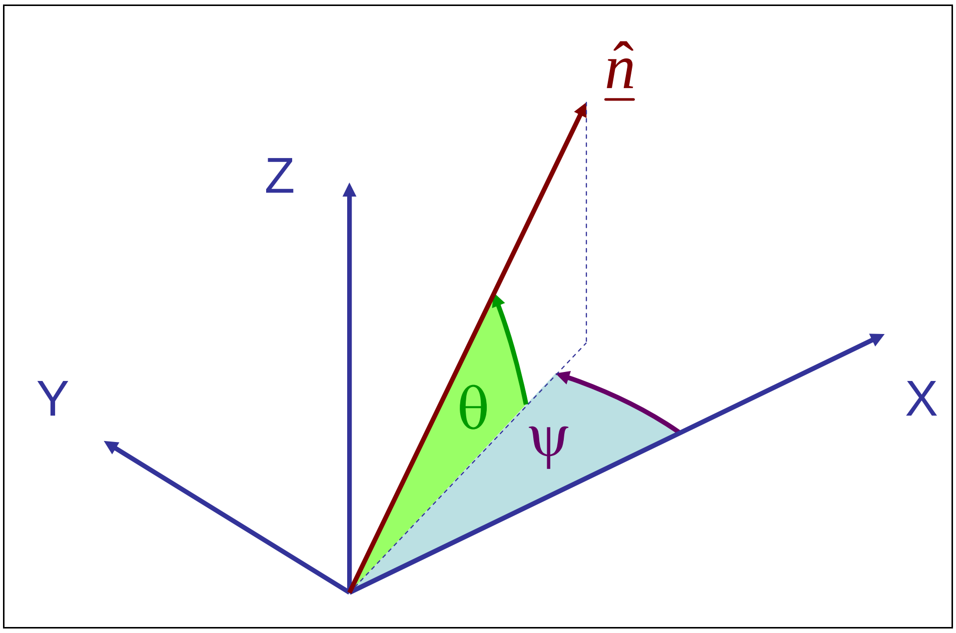

Figure 10 - Definition of Azimuth & Elevation Angles

The unit vector now becomes:

$$\underline{\hat{n}} = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \begin{bmatrix} \cos\theta \cdot \cos\psi \\ \cos\theta \cdot \sin\psi \\ \sin\theta \end{bmatrix} \qquad \textbf{(12.)}$$

MATLAB File

```
%
%   Program Fussy3
%
%   Grey Surface - Colour Coded Frontal Area
%
%   SJN 16th Feb 2006
%
clear
clf
colordef black
%-----------------------------------------------
nx=151; % No. of Stations
fuscmb=0; % Fuselage - Maximum Camber
fusl=5; % Fuselage - Overall Length
fusw=2; % Fuselage - Overall Width
fush=1; % Fuselage - Overall Height
xcab1=.25; % ND X Value of Cabin Front
xcab2=.6; % ND X Value of Cabin Rear
Nmin=2; % Minimum Value of Sharpness
Nmax=5; % Maximum Value of Sharpness (Cabin)
%-----------------------------------------------
gry=1; % Shade of Grey for Bulkheads
nphi=72; % No. of Bulkhead Azimuth Positions
%-----------------------------------------------
xnd=linspace(0,1,nx);
d=zeros(1,nx);
%-----------------------------------------------
% Axis Extent - Nearest Integer
nearint=1;
%-----------------------------------------------
%   Set Up Fuselage Profile
for i=1:nx
    xarg=xnd(i);
    if xarg<xcab1
        d(i)=(xarg*(2*xcab1-xarg))^.5/xcab1;
    elseif xarg<xcab2
        d(i)=1;
    else
        d(i)=(1-xarg)/(1-xcab2);
    end
end
%d=linspace(0,1,nx);% Cone Check
%-----------------------------------------------
x=xnd*fusl;
z0=fuscmb*sin(xnd*pi);
xmin=0;
xmax=nearint*(fix(fusl/nearint)+1);
ymin=-nearint*(fix(fusw/nearint)+1);
ymax=nearint*(fix(fusw/nearint)+1);
zmin=-nearint*(fix(fush/nearint)+1);
zmax=nearint*(fix(fush/nearint)+1);
phi=linspace(0,360,nphi)*pi/180;
```

```
cphi=cos(phi);
sphi=sin(phi);
%---------5------------------------------------
%   Set Up Fuselage Coordinate & Frontal Area Arrays
X=[];
Y=[];
Z=[];
frontalarea=[];
%-----------------------------------------------
%   Loop Down Fuselage Length
for i=1:nx
      a=fusw*d(i)/2;
      b=fush*d(i)/2;
    N=Nmin+(Nmax-Nmin)*d(i);
    if a*b~=0
        r=(abs((sphi/a)).^N+abs((cphi/b)).^N).^(-1/N);
      else
        r=0;
      end
      xxs=x(i)*ones(1,nphi);
      yxs=r.*sphi;
      zxs=r.*cphi+z0(i)*ones(1,nphi);
    X=[X;xxs];
    Y=[Y;yxs];
    Z=[Z;zxs];
end
%-----------------------------------------------
for i=1:nx-1
    for j=1:nphi-1
        farea1=(Y(i+1,j)-Y(i,j))*(Z(i,j+1)-Z(i,j))-(Z(i+1,j)-
Z(i,j))*(Y(i,j+1)-Y(i,j)); % Vector Product 1
        farea2=(Y(i+1,j+1)-Y(i+1,j))*(Z(i+1,j+1)-Z(i,j+1))-(Z(i+1,j+1)-
Z(i+1,j))*(Y(i+1,j+1)-Y(i,j+1)); % Vector Product 2
        frontalarea(i,j)=abs(farea1)+abs(farea2); % Sum Two Vector
Products
   end
end
areascale=max(max(frontalarea)); % Calculate Maximum Frontal Area Value
- Not to Screw Up Patch Colour Specification
%-----------------------------------------------
for i=1:nx-1
    for j=1:nphi-1
        px=[X(i,j),X(i+1,j),X(i+1,j+1),X(i,j+1)];
        py=[Y(i,j),Y(i+1,j),Y(i+1,j+1),Y(i,j+1)];
        pz=[Z(i,j),Z(i+1,j),Z(i+1,j+1),Z(i,j+1)];
        %-----------------------------------------------
        % Establish Patch Colour
        fa=frontalarea(i,j)/areascale;
        patchcolour=[fa,1-fa,1-fa^2];

patch(px,py,pz,patchcolour,'EdgeColor','none','EdgeLighting','phong');
        hold on
    %-----------------------------------------------
    end
end
%camlight('headlight')
%-----------------------------------------------
grid on
```

```
view(3)
```