รายการอ้างอิง

Antonio J. Gil, 2006. Structural analysis of prestressed Saint Venant-Kirchhoff hyperelastic membranes subjected to moderate strains. Computers and structures. 84 : 1012-1028.

Argyris JH et al. 1974. A general method for the shape finding of light weight structures. International conference on tension structures. London.

Ataka N, Kozuka Y, 1984. A determination method of geodesic lines on the surface of membrane structures (in Japan). Summary of technical papers of the annual meeting AIJ.

Barnes MR, 1977. Form finding and analysis of tension structures by dynamic relaxation. PhD. thesis. University of London, 1977.

Broyden C.G., 1970. The convergence of a class of double-rank minimization algorithms. J. Inst. Maths. Applics. 6 : 76-90.

Bucholdt HA, McMillan BR. 1971. Iterative methods for the solution of pretensioned cable structures. IASS symposium on tension structures and space frames. Tokyo.

Charlie C.L. Wang, Shana S-F Smith, Matthew M.F. Yune, 2002. Surface flattening based on energy model. Computer-aided design. 34 : 823-833

Edwin K.P. Chong, Stanislaw H. Zak, 2001. An introduction to optimization. Canada : John Wiley & sons.

Fletcher R, 1970. A new approach to variable metric algorithms. Computer journal. 13 : 317-322.

Goldfarb D, 1970. A family of variable metric updates derived by variational means. Mathematics of computing. 24 : 23-26.

Grundig L, Ekert L, Moncrieff E, 1966. Geodesic and semi-geodesic lin algorithms for cutting pattern generation of architectural textile structures. Proceedings of Asia-Pacific conference on shell and spatial structures.

Ishii K, 1972. On development of curved surface of membrane structures. IASS international symposium on pneumatic structures.

Isler H, 1994. Concrete shells derived from experimental shapes. Struct Eng Int, 3 : 142-147.

J. McCartney, B.K. Hinds, B.L. Seow, 1999. Computer-aided design. 31 : 249-260.

Jae-Yeol Kim, Jang-Bog Lee,2002. A new technique for optimum cutting pattern generation of membrane structures. Engineering structures. 24 : 745-756.

Javier Bonet, Richard D. Wood, 1997. Nonlinear continuum mechanics for finite element analysis. United States of America : Cambrided university press.

Levy R, Spillers WR, 2003. Analysis of geometrically nonlinear structures. Netherlands : Kluwer academic publishers.

Kreyszig E, 1968. Differentialgeometrie. Akademische verlagsgesellschaft. Liepzig.

K.-U. Bletzinger, E. Ramm, 1999. A general finite element approach to the form finding of tensile structures by the updated reference strategy. Int. J. Space struct. 14 : 131-146.

Mark A. Abramson, 2001-2005. Genetic algorithm and direct search toolbox user's guide. The mathworks.

Maurin B, Motro R, 1977. Density methods and minimal forms computation. IASS colloquium on structural morphology. Notingham.

Moncrieff E, Grundig L, Stroel D, 1999. The cutting pattern generation of the pilgrim's tents for phase II of the Mina valley project. IASS international symposium. Madrid.

Nye JF, 1985. Physical properties of crystals. Oxford : Oxford university press.

Otto F., 1967. Tensile Structures. Cambridge, MA : MIT press.

Schek HJ, 1974. The force density method for form finding and computation of general networks. Comp. Meth. Appl. Mech. Eng. 3.

Shanno DF, 1970. Conditioning of quasi-newton methods for function minimization. Mathematics of computing. 24 : 647-656.

Siev A, Eidelman J, 1964. Stress analysis of prestressed suspended roofs. Proc. ASCE. 90 ST4 : 103-121.

Suzuki T, 1995. An analytical method of geodesic line in membrane structures. IASS symp.

Tabarrok B, Qin Z, 1993. Form finding and cutting pattern generation for fabric tension structures. Microcomputers in civil engineering. Elsevier science publishers Ltd. 8 : 377-384.

Wagner R, Reimann K, Alber R, Jentsch M and Kroplin B, 2004. Computation of tailored membranes. European congress on computational method in applied sciences and engineering.

ภาคผนวก

ภาคผนวก ก
โปรแกรม Matlab สำหรับการวิเคราะห์โครงสร้างเปลือกบางแบบไม่เชิงเส้น
โปรแกรมหลักสำหรับคำนวณค่าหน่วยแรงและการกระจัด

```
%Calculate disp and for CST
%-------------------------------------------------------------------------
%INPUT DATA
%-------------------------------------------------------------------------
Clear;
nn=load('C:\CuttingPattern\nn.txt');
nel=nn(1);        %number of elements
nnode=nn(2);      %number of nodes
ndof=nn(3);       %number of degree of freedoms per node
nnel=nn(4);       %number of node elements
nbc=nn(5);        %number of boundaryconditions
E=nn(6);          %young modulus
poisson=nn(7);    %poisson ratio
t=nn(8);          %thick (metre)
scalfactor=nn(9);    %factor for deformation shape
nglx=2;           %number of sampling points in x direction
ngly=2;           %number of sampling points in y direction
sdof=nnode*ndof;     %number of degree of freedoms
edof=nnel*ndof;      %number of degree of elements
%-------------------------------------------------------------------------
%INPUT COORDINATES
%-------------------------------------------------------------------------
gcoord=load('C:\CuttingPattern\gcoord.txt');
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=load('C:\CuttingPattern\node.txt');
%-------------------------------------------------------------------------
%INPUT BOUNDARY CONDITION
%-------------------------------------------------------------------------
bcdof=load('C:\CuttingPattern\bcdof.txt');  %node
bcval=load('C:\CuttingPattern\bcval.txt');  %value
%-------------------------------------------------------------------------
%INPUT FORCE VECTOR (kg)
%-------------------------------------------------------------------------
fini=load('C:\CuttingPattern\ff.txt');      %point load
%-------------------------------------------------------------------------
%INPUT INITIAL STRESSES
inistressG=load('C:\CuttingPattern\inistress.txt');
%-------------------------------------------------------------------------
%SIZE OF MATRICES
%-------------------------------------------------------------------------
gcoordupdate=gcoord;          %update coordinate matrix
total=zeros(sdof,1);          %total increment displacement
totaldisp=zeros(sdof,1);      %total iterative displacement
disp=zeros(sdof,1);           %diplacement
for increment=1:1
    displacement=zeros(sdof,100); %displacement per iterative
    Fin=zeros(sdof,1);            %internal force
    dispel=zeros(edof,1);         %element displacment
    totaldispel=zeros(edof,1);    %total element displacement
    index=zeros(edof,1);          %distribute dof. matrix
    bL=zeros(1,edof);             %transform linear strain matrix
    bNL=zeros(2,edof);            %transform non linear strain matrix
    tran=zeros(nnel*2,edof);      %trasnform local axis matrix
    [point,weight]=fpntwght2(nglx,ngly); %gauss point
```

```
    totaldisp=total;                %set new coordinate
    for iter=1:500
        kk=zeros(sdof,sdof);        %global stiffness matrix
        Fglobal=zeros(sdof,1);      %global force vector
        gstress=zeros(3*nel,1);     %global piola stress vector
        cauchystress=zeros(3*nel,1); %local cauchsy stress vector
        for i=1:nnode
            for j=1:3
                dispcoord(i,j)=disp(3*(i-1)+j); %distribute element
ccordiant
            end
        end
        gcoordupdate=gcoordupdate+dispcoord;    %update element…
coordinate
%-------------------------------------------------------------------------
%COMPUTATION OF STIFFNESS MATRIX
%-------------------------------------------------------------------------
        for iel=1:nel
            for i=1:nnel    %extract connect node
                nd(i)=node(iel,i);
                xcoord(i)=gcoord(nd(i),1);  %local element coordinate…
x vector
                ycoord(i)=gcoord(nd(i),2);  %local element coordinate…
y vector
                zcoord(i)=gcoord(nd(i),3);  %local element coordinate…
z vector
                xcoordupdate(i)=gcoordupdate(nd(i),1);  %local element…
coordinate x vector
                ycoordupdate(i)=gcoordupdate(nd(i),2);  %local element…
coordinate y vector
                zcoordupdate(i)=gcoordupdate(nd(i),3);  %local element…
coordinate z vector
            end
            index=feeldof(nd,nnel,ndof);
            for i=1:edof
                dispel(i)=disp(index(i));   %extract for element…
displacment
                totaldispel(i)=totaldisp(index(i));
            end

[tran,xprime,yprime,tran2D]=ftran(xcoord,ycoord,zcoord,nnel);
%tranformation matrix for initial config
[blank,xprimeupdate,yprimeupdate,tran2Dup]=ftran(xcoordupdate,ycoordupd
ate,zcoordupdate,1); %tranformation matrix for initial config
            dispel=tran*dispel; %transform to local displacement
            totaldispel=tran*totaldispel;
            kL=zeros(edof,edof);    %local linear stiffness matrix
            kNL=zeros(edof,edof);   %local nonlinear stiffness matrix
            ke=zeros(edof,edof);    %global stiffness matrix
            Finel=zeros(edof,1);    %local element force
            for intx=1:1    %integrate by Gausspoint
                x=point(intx,1);
                wtx=weight(intx,1);
                for inty=1:1
                    y=point(inty,2);
                    wty=weight(inty,2);
```

```
                    [shape,dr,ds]=fdrds(x,y);    %compute isoparametric…
3 nodes shape function
                    jacob=fjacob(nnel,dr,ds,xprime,yprime); %compute…
jacobian matrix
                    detjacob=det(jacob);
                    invjacob=inv(jacob);
                    [dhdx,dhdy]=fdxdy(nnel,dr,ds,invjacob); %compute…
dirivative by x and y
                    for i=1:3
                        inistress(i)=inistressG(3*(iel-1)+i); %local…
initial stress element
                    end

[strainel]=fstrain(dispel,totaldispel,dhdx,dhdy,nnel);  %compute…
element strain
                    [mat,tran2]=fmat(3,E,poisson,tran2D);    %compute…
constutitive element matrix

[tstresselpiola,vstresselpiola,mstressel]=fstressel(mat,strainel,inistr
ess');   %compute element stress

[dxdX]=fdefgrad(nnel,dr,ds,invjacob,xprimeupdate,yprimeupdate);
%compute element deformation gradient
                    [bL]=fkinlinear(dhdx,dhdy,totaldispel,nnel);
%compute linear kinematic matrix
                    [bNL]=fkinnonlinear(dhdx,dhdy); %compute nonlinear…
kinematic matrix
                    kL=kL+t*detjacob/2*bL'*mat*bL;  %compute linear…
stiffness
                    kNL=kNL+t*detjacob/2*bNL'*tstresselpiola*bNL;
%compute nonliear stiffness
                    k=kL+kNL;   %element stiffness…
                    Finel=Finel+t*detjacob/2*bL'*vstresselpiola;
%compute internal force of element
                end
            end
            ke=tran'*k*tran; %global element stiffness
            Fe=tran'*Finel; %global element force
            index=feeldof(nd,nnel,ndof);
            kk=feasmbl(kk,ke,index);
            for i=1:edof
                Fglobal(index(i))=Fglobal(index(i))+Fe(i);  %assemble…
force to global
            end
            cauchystressel=1/det(dxdX)*dxdX*mstressel*dxdX'; %trasfer…
piola stress to cauchy stress
            %cauchystressel=tran2Dup'*cauchystressel*tran2Dup;
%transfer to global stress

vcauchystressel=[cauchystressel(1,1);cauchystressel(2,2);cauchystressel
(1,2);];
            for i=1:3
                cauchystress((3*(iel-1)+i))=cauchystress((3*(iel-
1)+i))+vcauchystressel(i); %assemble cauchy stress vector
                gstress((3*(iel-1)+i))=gstress((3*(iel-
1)+i))+vstresselpiola(i);    %assemble piola stress vector
            end
```

```
        end
        inistressG=gstress'; %update piola stress
%-------------------------------------------------------------------------
%check the singular drilling dof
%-------------------------------------------------------------------------
%         for i=1:sdof
%             if (abs(kk(i,i))<1e-10)
%                 sum=0.0;
%                 for j=1:sdof
%                     sum=sum+abs(kk(i,j));
%                 end
%                 if (sum<1e-10)
%                     kk(i,i)=1;
%                 end
%             end
%         end
%-------------------------------------------------------------------------
%solve for displacement
%-------------------------------------------------------------------------
        ff=fini-Fglobal;
        %-----------------------------------------------------------------
        %check error
        %-----------------------------------------------------------------
        error=ff;   %residual force
        for i=1:nbc
            error(bcdof(i))=0;
        end
        if iter<2
            error(1)=1;
        end
        if norm(error,inf)<1e-7 %termiante at residual force < 1e-7
            break;
        end
        %-----------------------------------------------------------------
        if iter>1
            bcval=zeros(1,nbc); %set boundary at C(t) as zero
        end
        [kk,ff]=fbdc(kk,ff,bcdof,bcval);     %define boundary
        disp=kk\ff;
        for i=1:sdof
            displacement(i,iter)=disp(i);
        end
        totaldisp=totaldisp+disp;
    end
    total=totaldisp+total;
    for i=1:nel
        for j=1:3
            stressoutput(i,j)=cauchystress(3*(i-1)+j);
        end
    end
end
```

```
function [kk,ff]=fbdc(kk,ff,bcdof,bcval)
%apply boundary condition
n=length(bcdof);
sdof=size(kk);
for i=1:n
    id=bcdof(i);
    val=bcval(i);
    for j=1:sdof
        ff(j)=ff(j)-val*kk(j,id);
        kk(id,j)=0;
        kk(j,id)=0;
    end
    kk(id,id)=1;
    ff(id)=val;
end

function [dxdX]=fdefgrad(nnel,dr,ds,invjacob,xprimeupdate,yprimeupdate)
%determine deformation gradient
dxdX=zeros(2);
for i=1:nnel
    dhdx(i)=invjacob(1,1)*dr(i)+invjacob(1,2)*ds(i);
    dhdy(i)=invjacob(2,1)*dr(i)+invjacob(2,2)*ds(i);
end
dxdX(1,1)=dhdx(1)*xprimeupdate(1)+dhdx(2)*xprimeupdate(2)+dhdx(3)*xprim
eupdate(3);
dxdX(1,2)=dhdy(1)*xprimeupdate(1)+dhdy(2)*xprimeupdate(2)+dhdy(3)*xprim
eupdate(3);
dxdX(2,1)=dhdx(1)*yprimeupdate(1)+dhdx(2)*yprimeupdate(2)+dhdx(3)*yprim
eupdate(3);
dxdX(2,2)=dhdy(1)*yprimeupdate(1)+dhdy(2)*yprimeupdate(2)+dhdy(3)*yprim
eupdate(3);

function [shape,dr,ds]=fdrds(rvalue,svalue)
%shape function divided by dr and ds
%-------------------------------------------------------------------------
%shape function (linear triangle)
%-------------------------------------------------------------------------
shape(1)=1-rvalue-svalue;
shape(2)=rvalue;
shape(3)=svalue;
%-------------------------------------------------------------------------
%derivatives
%-------------------------------------------------------------------------
dr(1)=-1;
dr(2)=1;
dr(3)=0;
ds(1)=-1;
ds(2)=0;
ds(3)=1;
```

```
function [dhdx,dhdy]=fdxdy(nnel,dr,ds,invjacob)
%determine derivatives w.r.t. x and y
for i=1:nnel
    dhdx(i)=invjacob(1,1)*dr(i)+invjacob(1,2)*ds(i);
    dhdy(i)=invjacob(2,1)*dr(i)+invjacob(2,2)*ds(i);
end

function kk=feasmbl(kk,ke,index)
%assemble element matrices
edof=length(index);
for i=1:edof
    ii=index(i);
    for j=1:edof
        jj=index(j);
        kk(ii,jj)=kk(ii,jj)+ke(i,j);
    end
end

function index=feeldof(nd,nnel,ndof)
%arrange dof from element to global
edof=nnel*ndof;
jj=0;
for i=1:nnel
    start=(nd(i)-1)*ndof;
    for j=1:ndof
        jj=jj+1;
        index(jj)=start+j;
    end
end

function [Bglobal]=fglobalkinematrix(transtr,Blocal,edof);
%transform local stress to global stress for kinematic
matrixloc=zeros(6,edof);
for i=1:edof
    matrixloc(1,i)=Blocal(1,i);
    matrixloc(2,i)=Blocal(2,i);
    matrixloc(4,i)=Blocal(3,i);
end
Bglobal=transtr'*matrixloc;

function [jacob]=fjacob(nnel,dr,ds,xprime,yprime)
%determine jacobian 2 dimensions
jacob=zeros(2,2);
for i=1:nnel
    jacob(1,1)=jacob(1,1)+dr(i)*xprime(i);
    jacob(1,2)=jacob(1,2)+dr(i)*yprime(i);
    jacob(2,1)=jacob(2,1)+ds(i)*xprime(i);
    jacob(2,2)=jacob(2,2)+ds(i)*yprime(i);
end
```

```
function [bL]=fkinlinear(dhdx,dhdy,dispel,nnel);
%determine linear kinematic
ux=0;
uy=0;
vx=0;
vy=0;
wx=0;
wy=0;
for i=1:nnel
    ux=ux+dhdx(i)*dispel((3*(i-1)+1));
    uy=uy+dhdy(i)*dispel((3*(i-1)+1));
    vx=vx+dhdx(i)*dispel((3*(i-1)+2));
    vy=vy+dhdy(i)*dispel((3*(i-1)+2));
    wx=wx+dhdx(i)*dispel((3*(i-1)+3));
    wy=wy+dhdy(i)*dispel((3*(i-1)+3));
end
bL0=[dhdx(1) 0 0 dhdx(2) 0 0 dhdx(3) 0 0;
    0 dhdy(1) 0 0 dhdy(2) 0 0 dhdy(3) 0;
    dhdy(1) dhdx(1) 0 dhdy(2) dhdx(2) 0 dhdy(3) dhdx(3) 0];
bL1=[ux*dhdx(1) vx*dhdx(1) wx*dhdx(1) ux*dhdx(2) vx*dhdx(2) wx*dhdx(2)
ux*dhdx(3) vx*dhdx(3) wx*dhdx(3);
     uy*dhdy(1) vy*dhdy(1) wy*dhdy(1) uy*dhdy(2) vy*dhdy(2) wy*dhdy(2)
uy*dhdy(3) vy*dhdy(3) wy*dhdy(3);
     ux*dhdy(1)+uy*dhdx(1) vx*dhdy(1)+vy*dhdx(1) wx*dhdy(1)+wy*dhdx(1)
ux*dhdy(2)+uy*dhdx(2) vx*dhdy(2)+vy*dhdx(2) wx*dhdy(2)+wy*dhdx(2)
ux*dhdy(3)+uy*dhdx(3) vx*dhdy(3)+vy*dhdx(3) wx*dhdy(3)+wy*dhdx(3)];
 bL=bL0+bL1;

function [bNL]=fkinnonlinear(dhdx,dhdy)
%determine nonlinear kinematic
bNL=[dhdx(1) 0 0 dhdx(2) 0 0 dhdx(3) 0 0;
     dhdy(1) 0 0 dhdy(2) 0 0 dhdy(3) 0 0;
     0 dhdx(1) 0 0 dhdx(2) 0 0 dhdx(3) 0;
     0 dhdy(1) 0 0 dhdy(2) 0 0 dhdy(3) 0;
     0 0 dhdx(1) 0 0 dhdx(2) 0 0 dhdx(3);
     0 0 dhdy(1) 0 0 dhdy(2) 0 0 dhdy(3)];

function [point,weight]=fpntwght2(nglx,ngly);
% two points and weight coefficients
if nglx > ngly
    ngl=nglx;
else
    ngl=ngly;
end
point=zeros(ngl,2);
weight=zeros(ngl,2);
[pointx,weightx]=fpntwght1(nglx);
[pointy,weighty]=fpntwght1(ngly);
for intx=1:nglx
    point(intx,1)=pointx(intx);
    weight(intx,1)=weightx(intx);
end
for inty=1:ngly
    point(inty,2)=pointy(inty);
    weight(inty,2)=weighty(inty);
end
```

```
function [mat,tran2]=fmat(no,e,poisson,tran2D)
%determine constitutive equation
tran2=[tran2D(1,1)^2 tran2D(1,2)^2 tran2D(1,1)*tran2D(1,2);
       tran2D(2,1)^2 tran2D(2,2)^2 tran2D(2,1)*tran2D(2,2);
       2*tran2D(1,1)*tran2D(2,1) 2*tran2D(1,2)*tran2D(2,2)
(tran2D(1,1)*tran2D(2,2)+tran2D(1,2)*tran2D(2,1))];
if no==1
    mat=e/(1-poisson*poisson)*[1 poisson 0;poisson 1 0;0 0 (1-
poisson)/2];
elseif no==2
    mat=e/((1+poisson)*(1-2*poisson))*[(1-poisson) poisson 0;poisson
(1-poisson) 0;0 0 (1-2*poisson)/2];
else
    Ex=2725;
    Ey=8225;
    poissonx=0.29;
    poissony=0.87;
    G=712.5;
    mat=[Ex/(1-poissonx*poissony) poissony*Ex/(1-poissonx*poissony)
0;poissonx*Ey/(1-poissonx*poissony) Ey/(1-poissonx*poissony) 0;0 0 G];
    mat=tran2'*mat*tran2;
end

function [tstressel,vstressel,mstressel]=fstressel(mat,strainel,
%calculate element stress
inistress)
vstressel=mat*strainel+inistress;
tstressel=[vstressel(1) vstressel(3) 0 0 0 0;
           vstressel(3) vstressel(2) 0 0 0 0;
           0 0 vstressel(1) vstressel(3) 0 0;
           0 0 vstressel(3) vstressel(2) 0 0;
           0 0 0 0 vstressel(1) vstressel(3);
           0 0 0 0 vstressel(3) vstressel(2)];
mstressel=[vstressel(1) vstressel(3);
           vstressel(3) vstressel(2)];

function [strainel]=fstrain(dispel,totaldispel,dhdx,dhdy,nnel)
%calculation strain in element
totaldispel=totaldispel-dispel;
strainel=zeros(3,1);
xpart=0;
ypart=0;
zpart=0;
xparttotal=0;
yparttotal=0;
zparttotal=0;
for i=1:nnel
    xpart=xpart+dhdx(i)*dispel((3*(i-1)+1));
    ypart=ypart+dhdx(i)*dispel((3*(i-1)+2));
    zpart=zpart+dhdx(i)*dispel((3*(i-1)+3));
    xparttotal=xparttotal+dhdx(i)*totaldispel((3*(i-1)+1));
    yparttotal=yparttotal+dhdx(i)*totaldispel((3*(i-1)+2));
    zparttotal=zparttotal+dhdx(i)*totaldispel((3*(i-1)+3));
end
strainel(1,1)=xpart+xparttotal*xpart+yparttotal*ypart+zparttotal*zpart+
1/2*(xpart^2+ypart^2+zpart^2);
```

```
xpart=0;
ypart=0;
zpart=0;
xparttotal=0;
yparttotal=0;
zparttotal=0;
for i=1:nnel
    xpart=xpart+dhdy(i)*dispel((3*(i-1)+1));
    ypart=ypart+dhdy(i)*dispel((3*(i-1)+2));
    zpart=zpart+dhdy(i)*dispel((3*(i-1)+3));
    xparttotal=xparttotal+dhdy(i)*totaldispel((3*(i-1)+1));
    yparttotal=yparttotal+dhdy(i)*totaldispel((3*(i-1)+2));
    zparttotal=zparttotal+dhdy(i)*totaldispel((3*(i-1)+3));
end
strainel(2,1)=ypart+xparttotal*xpart+yparttotal*ypart+zparttotal*zpart+
1/2*(xpart^2+ypart^2+zpart^2);
uxpart=0;
uypart=0;
vxpart=0;
vypart=0;
wxpart=0;
wypart=0;
uxparttotal=0;
uyparttotal=0;
vxparttotal=0;
vyparttotal=0;
wxparttotal=0;
wyparttotal=0;
for i=1:nnel
    uxpart=uxpart+dhdx(i)*dispel((3*(i-1)+1));
    uypart=uypart+dhdy(i)*dispel((3*(i-1)+1));
    vxpart=vxpart+dhdx(i)*dispel((3*(i-1)+2));
    vypart=vypart+dhdy(i)*dispel((3*(i-1)+2));
    wxpart=wxpart+dhdx(i)*dispel((3*(i-1)+3));
    wypart=wypart+dhdy(i)*dispel((3*(i-1)+3));
    uxparttotal=uxparttotal+dhdx(i)*totaldispel((3*(i-1)+1));
    uyparttotal=uyparttotal+dhdy(i)*totaldispel((3*(i-1)+1));
    vxparttotal=vxparttotal+dhdx(i)*totaldispel((3*(i-1)+2));
    vyparttotal=vyparttotal+dhdy(i)*totaldispel((3*(i-1)+2));
    wxparttotal=wxparttotal+dhdx(i)*totaldispel((3*(i-1)+3));
    wyparttotal=wyparttotal+dhdy(i)*totaldispel((3*(i-1)+3));
end
strainel(3,1)=(uypart+vxpart)+(uxparttotal*uypart+uyparttotal*uxpart)+(
vxparttotal*vypart+vyparttotal*vxpart)+(wxparttotal*wypart+wyparttotal*
wxpart)+(uxpart*uypart+vxpart*vypart+wxpart*wypart);

function [tran,xprime,yprime,tran2D]=ftran(xcoord,ycoord,zcoord,n)
%transfrom matrix (triangle)
%-------------------------------------------------------------------------
%compute vector from side of triagle
%-------------------------------------------------------------------------
v12x=xcoord(2)-xcoord(1);
v12y=ycoord(2)-ycoord(1);
v12z=zcoord(2)-zcoord(1);
l12=sqrt(v12x^2+v12y^2+v12z^2);
v23x=xcoord(3)-xcoord(2);
```

```
v23y=ycoord(3)-ycoord(2);
v23z=zcoord(3)-zcoord(2);
l23=sqrt(v23x^2+v23y^2+v23z^2);
v13x=xcoord(3)-xcoord(1);
v13y=ycoord(3)-ycoord(1);
v13z=zcoord(3)-zcoord(1);
l13=sqrt(v13x^2+v13y^2+v13z^2);
%-------------------------------------------------------------------------
%compute cross product
%-------------------------------------------------------------------------
v1tx=v12y*v13z-v12z*v13y;
v1ty=v12z*v13x-v12x*v13z;
v1tz=v12x*v13y-v12y*v13x;
v1yx=v1ty*v12z-v1tz*v12y;
v1yy=v1tz*v12x-v1tx*v12z;
v1yz=v1tx*v12y-v1ty*v12x;
%-------------------------------------------------------------------------
%compute local axes
%-------------------------------------------------------------------------
vxx=v12x/l12;
vxy=v12y/l12;
vxz=v12z/l12;
vyx=v1yx/sqrt(v1yx^2+v1yy^2+v1yz^2);
vyy=v1yy/sqrt(v1yx^2+v1yy^2+v1yz^2);
vyz=v1yz/sqrt(v1yx^2+v1yy^2+v1yz^2);
vzx=v1tx/sqrt(v1tx^2+v1ty^2+v1tz^2);
vzy=v1ty/sqrt(v1tx^2+v1ty^2+v1tz^2);
vzz=v1tz/sqrt(v1tx^2+v1ty^2+v1tz^2);
%-------------------------------------------------------------------------
%compute tranformation matrix
%-------------------------------------------------------------------------
for i=1:n
    i1=(i-1)*3+1;
    i2=i1+1;
    i3=i2+1;
    tran(i1,i1)=vxx;
    tran(i1,i2)=vxy;
    tran(i1,i3)=vxz;
    tran(i2,i1)=vyx;
    tran(i2,i2)=vyy;
    tran(i2,i3)=vyz;
    tran(i3,i1)=vzx;
    tran(i3,i2)=vzy;
    tran(i3,i3)=vzz;
end
RRR=sqrt(vzz^2+vzy^2);
Tfab=[RRR -vzy*vzx/RRR -vzz*vzx/RRR;
      0 vzz/RRR -vzy/RRR;
      vzx vzy vzz;];
Tlocal=[vxx vxy vxz;
        vyx vyy vyz;
        vzx vzy vzz;];
Tfl=Tfab*Tlocal';
tran2D(1,1)=Tfl(1,1);
tran2D(1,2)=Tfl(1,2);
tran2D(2,1)=Tfl(2,1);
tran2D(2,2)=Tfl(2,2);
```

```
%-------------------------------------------------------------------------
%compute local coordinates
%-------------------------------------------------------------------------
zeta213=acos((l12^2+l13^2-l23^2)/(2*l12*l13));
xprime(1)=0;     yprime(1)=0;
xprime(2)=l12;     yprime(2)=0;
xprime(3)=l13*cos(zeta213);     yprime(3)=l13*sin(zeta213);
```

ภาคผนวก ข
โปรแกรม Matlab สำหรับการวิเคราะห์โครงสร้างเปลือกบางแบบไม่เชิงเส้น
โปรแกรมรับข้อมูลเข้าสำหรับการหาแบบตัดที่เหมาะสมที่สุด

```
%Input data for cutting pattern
%-------------------------------------------------------------------------
clear
nel=75; %number of elements
nnode=51;   %number of nodes
ndof=3; %number of degree of freedoms per node
nnel=3; %number of node elements
nbc=75; %number of boundary conditions
E=3e7;  %young modulus (ksc)
poisson=0.3;    %poisson ratio
t=0.08; %thick (centimeters)
scalfactor=1;   %factor for deformation shape
sdof=nnode*ndof;
%-------------------------------------------------------------------------
%INPUT COORDINATE
%-------------------------------------------------------------------------
gcoord=[];
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=[];
%-------------------------------------------------------------------------
%INPUT COORDINATE 3D
%-------------------------------------------------------------------------
gcoord3D=[];
gcoord3D=1*gcoord3D;
%-------------------------------------------------------------------------
%INPUT BOUNDARY CONDITIONS
%-------------------------------------------------------------------------
for i=1:nbc
    bcdof(i)=i;
end
bcval=zeros(1,nbc);
%-------------------------------------------------------------------------
%calculate displacement from 2D to 3D
%-------------------------------------------------------------------------
for i=1:nnode
    for j=1:3
        disbc(3*(i-1)+j)=gcoord3D(i,j)-gcoord(i,j);
    end
end
for i=1:nbc
    bcval(i)=disbc(i);
end
%-------------------------------------------------------------------------
%INPUT FORCE VECTOR (kg)
%-------------------------------------------------------------------------
ff=zeros(sdof,1);
%-------------------------------------------------------------------------
%INPUT INITIAL STRESS
%-------------------------------------------------------------------------
for i=1:nel
    inistress(3*(i-1)+1)=0.1;
    inistress(3*(i-1)+2)=0.1;
    inistress(3*(i-1)+3)=0;
end
%inistress=[];
```

```
stresstarget=[62.5;62.5;0];
%-------------------------------------------------------------------------
nn=[nel nnode ndof nnel nbc E poisson t scalfactor];
save('C:\CuttingPattern\nn.txt','nn','-ascii','-double');
save('C:\CuttingPattern\gcoord.txt','gcoord','-ascii','-double');
save('C:\CuttingPattern\gcoord3D.txt','gcoord3D','-ascii','-double');
save('C:\CuttingPattern\node.txt','node','-ascii','-double');
save('C:\CuttingPattern\bcdof.txt','bcdof','-ascii','-double');
save('C:\CuttingPattern\bcval.txt','bcval','-ascii','-double');
save('C:\CuttingPattern\ff.txt','ff','-ascii','-double');
save('C:\CuttingPattern\inistress.txt','inistress','-ascii','-double');
save('C:\CuttingPattern\stresstarget.txt','stresstarget','-ascii','-
double');
```

ภาคผนวก ค

โปรแกรม Matlab สำหรับการวิเคราะห์โครงสร้างเปลือกบางแบบไม่เชิงเส้น ฟังก์ชันเป้าหมายสมการที่ (3.6) สำหรับคำสั่ง gatool ของ Matlab

```
function deviat=objcut(impose)
%OBJECTIVE FUNCTION FOR CALCULATION STRESS
%-------------------------------------------------------------------------
%INPUT DATA
%-------------------------------------------------------------------------
nn=load('C:\CuttingPattern\nn.txt');
nel=nn(1);  %number of elements
nnode=nn(2);    %number of nodes
ndof=nn(3); %number of degree of freedoms per node
nnel=nn(4); %number of node elements
nbc=nn(5);  %number of boundaryconditions
E=nn(6);    %young modulus 'ksc)
poisson=nn(7);  %poisson ratio
t=nn(8);    %thick (metre)
scalfactor=nn(9);   %factor for deformation shape
nglx=2; %number of sampling points in x direction
ngly=2; %number of sampling points in y direction
sdof=nnode*ndof;    %number of degree of freedoms
edof=nnel*ndof; %number of degree of elements
%-------------------------------------------------------------------------
%INPUT COORDINATES
%-------------------------------------------------------------------------
gcoord=load('C:\CuttingPattern\gcoord.txt');
for i=1:nnode
    gcoord(i,1)=impose(1)*gcoord(i,1);
    gcoord(i,2)=impose(2)*gcoord(i,2);
end
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=load('C:\CuttingPattern\node.txt');
%-------------------------------------------------------------------------
%INPUT BOUNDARY CONDITION
%-------------------------------------------------------------------------
bcdof=load('C:\CuttingPattern\bcdof.txt');
bcval=load('C:\CuttingPattern\bcval.txt');
gcoord_ini=load('C:\CuttingPattern\gcoord3D.txt');
gcoord_ini=gcoord_ini-gcoord;
dispV=zeros(size(gcoord_ini,1),1);
for i=1:size(gcoord_ini,1);
    for j=1:size(gcoord_ini,2);
        dispV(3*(i-1)+j)=gcoord_ini(i,j);
    end
end
for i=1:nbc
    bcval(i)=dispV(i);
end
%-------------------------------------------------------------------------
%INPUT FORCE VECTOR (kg)
%-------------------------------------------------------------------------
fini=load('C:\CuttingPattern\ff.txt');
%-------------------------------------------------------------------------
%INPUT INITIAL  STRESSES
%-------------------------------------------------------------------------
inistressG=load('C:\CuttingPattern\inistress.txt');
stresstarget=load('C:\CuttingPattern\stresstarget.txt');
```

```
%-------------------------------------------------------------------------
%SIZE OF MATRICES
%-------------------------------------------------------------------------
gcoordupdate=gcoord;
total=zeros(sdof,1);
totaldisp=zeros(sdof,1);
disp=zeros(sdof,1);
for increment=1:1
    dispel=zeros(edof,1);
    totaldispel=zeros(edof,1);
    index=zeros(edof,1);
    bL=zeros(1,edof);
    bNL=zeros(2,edof);
    tran=zeros(nnel*2,edof);
    [point,weight]=fpntwght2(nglx,ngly);
    totaldisp=total;
    for iter=1:500
        kk=zeros(sdof,sdof);
        Fglobal=zeros(sdof,1);
        gstress=zeros(3*nel,1);
        cauchystress=zeros(3*nel,1);
        for i=1:nnode
            for j=1:3
                dispcoord(i,j)=disp(3*(i-1)+j);
            end
        end
        gcoordupdate=gcoordupdate+dispcoord;
%-------------------------------------------------------------------------
%COMPUTATION OF STIFFNESS MATRIX
%-------------------------------------------------------------------------
for iel=1:nel
            for i=1:nnel
                nd(i)=node(iel,i);
                xcoord(i)=gcoord(nd(i),1);
                ycoord(i)=gcoord(nd(i),2);
                zcoord(i)=gcoord(nd(i),3);
                xcoordupdate(i)=gcoordupdate(nd(i),1);
                ycoordupdate(i)=gcoordupdate(nd(i),2);
                zcoordupdate(i)=gcoordupdate(nd(i),3);
            end
            index=feeldof(nd,nnel,ndof);
            for i=1:edof
                dispel(i)=disp(index(i));
                totaldispel(i)=totaldisp(index(i));
            end

[tran,xprime,yprime,tran2D]=ftran(xcoord,ycoord,zcoord,nnel);

[blank,xprimeupdate,yprimeupdate,tran2Dup]=ftran(xcoordupdate,ycoordupd
ate,zcoordupdate,1);
            dispel=tran*dispel;
            totaldispel=tran*totaldispel;
            kL=zeros(edof,edof);
            kNL=zeros(edof,edof);
            ke=zeros(edof,edof);
            Finel=zeros(edof,1);
            for intx=1:1
```

```
                x=point(intx,1);
                wtx=weight(intx,1);
                for inty=1:1
                    y=point(inty,2);
                    wty=weight(inty,2);
                    [shape,dr,ds]=fdrds(x,y);
                    jacob=fjacob(nnel,dr,ds,xprime,yprime);
                    detjacob=det(jacob);
                    invjacob=inv(jacob);
                    [dhdx,dhdy]=fdxdy(nnel,dr,ds,invjacob);
                    for i=1:3
                        inistress(i)=inistressG(3*(iel-1)+i);
                    end

[strainel]=fstrain(dispel,totaldispel,dhdx,dhdy,nnel);
                    [mat,tran2]=fmat(3,E,poisson,tran2D);

[tstresselpiola,vstresselpiola,mstressel]=fstressel(mat,strainel,inistr
ess');

[dxdX]=fdefgrad(nnel,dr,ds,invjacob,xprimeupdate,yprimeupdate);
                    [bL]=fkinlinear(dhdx,dhdy,totaldispel,nnel);
                    [bNL]=fkinnonlinear(dhdx,dhdy);
                    kL=kL+t*detjacob/2*bL'*mat*bL;
                    kNL=kNL+t*detjacob/2*bNL'*tstresselpiola*bNL;
                    k=kL+kNL;
                    Finel=Finel+t*detjacob/2*bL'*vstresselpiola;
                end
            end
            ke=tran'*k*tran;
            Fe=tran'*Finel;
            index=feeldof(nd,nnel,ndof);
            kk=feasmbl(kk,ke,index);
            for i=1:edof
                Fglobal(index(i))=Fglobal(index(i))+Fe(i);
            end
            cauchystressel=1/det(dxdX)*dxdX*mstressel*dxdX';

vcauchystressel=[cauchystressel(1,1);cauchystressel(2,2);cauchystressel
(1,2);];
            for i=1:3
                cauchystress((3*(iel-1)+i))=cauchystress((3*(iel-
1)+i))+vcauchystressel(i);
                gstress((3*(iel-1)+i))=gstress((3*(iel-
1)+i))+vstresselpiola(i);
            end
        end
        inistressG=gstress';
%-------------------------------------------------------------------------
%solve for displacement
%-------------------------------------------------------------------------
ff=fini-Fglobal;
        %-----------------------------------------------------------------
        %check error
        %-----------------------------------------------------------------
        error=ff;
        for i=1:nbc
```

```
            error(bcdof(i))=0;
        end
        if iter<2
            error(1)=1;
        end
        if norm(error,inf)<1e-7
            break;
        end
        %-------------------------------------------------------------------
if iter>1
            bcval=zeros(1,nbc);
        end
        [kk,ff]=fbdc(kk,ff,bcdof,bcval);
        disp=kk\ff;
        totaldisp=totaldisp+disp;
    end
    total=totaldisp+total;
    for i=1:nel
        for j=1:3
            stressoutput(i,j)=cauchystress(3*(i-1)+j);
        end
    end
end
for i=1:nel
    for j=1:3
        target((3*(i-1)+j))=stresstarget(j);
    end
end
deviat=0.5*(cauchystress-target')'*(cauchystress-target'); %target
function
```

```
function ncoord=newcoord(impose)
%Calculate new coordinate by percent value
%-------------------------------------------------------------------------
%INPUT DATA
%-------------------------------------------------------------------------
nn=load('C:\CuttingPattern\nn.txt');
nel=nn(1);  %number of elements
nnode=nn(2);    %number of nodes
ndof=nn(3); %number of degree of freedoms per node
nnel=nn(4); %number of node elements
nbc=nn(5);  %number of boundaryconditions
E=nn(6);    %young modulus (ksc)
poisson=nn(7);  %poisson ratio
t=nn(8);    %thick (metre)
scalfactor=nn(9);   %factor for deformation shape
nglx=2; %number of sampling points in x direction
ngly=2; %number of sampling points in y direction
sdof=nnode*ndof;    %number of degree of freedoms
edof=nnel*ndof; %number of degree of elements
%-------------------------------------------------------------------------
%INPUT COORDINATES
%-------------------------------------------------------------------------
gcoord=load('C:\CuttingPattern\gcoord.txt');
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=load('C:\CuttingPattern\node.txt');
%-------------------------------------------------------------------------
%INPUT BOUNDARY CONDITIONS
%-------------------------------------------------------------------------
bcdof=load('C:\CuttingPattern\bcdof.txt');
bcval=load('C:\CuttingPattern\bcval.txt');
%-------------------------------------------------------------------------
%INPUT FORCE VECTOR (kg)
%-------------------------------------------------------------------------
fini=load('C:\CuttingPattern\ff.txt');
%-------------------------------------------------------------------------
%INPUT INITIAL STRESSES
inistressG=load('C:\CuttingPattern\inistress.txt');
%-------------------------------------------------------------------------
%INPUT Unknown (displacement)2* i-1)+
%-------------------------------------------------------------------------
for i=1:(nbc/3)
    bcval(3*(i-1)+1)=impose(2*(i-1)+1)*gcoord(i,1)-gcoord(i,1);
    bcval(3*(i-1)+2)=impose(2*(i-1)+2)*gcoord(i,2)-gcoord(i,2);
end
%-------------------------------------------------------------------------
%SIZE OF MATRICES
%-------------------------------------------------------------------------
gcoordupdate=gcoord;
total=zeros(sdof,1);
totaldisp=zeros(sdof,1);
disp=zeros(sdof,1);
for increment=1:1
    displacement=zeros(sdof,100);
    Fin=zeros(sdof,1);
    dispel=zeros(edof,1);
```

```
    totaldispel=zeros(edof,1);
    index=zeros(edof,1);
    bL=zeros(1,edof);
    bNL=zeros(2,edof);
    tran=zeros(nnel*2,edof);
    [point,weight]=fpntwght2(nglx,ngly);
    totaldisp=total;
    for iter=1:500
        kk=zeros(sdof,sdof);
        Fglobal=zeros(sdof,1);
        gstress=zeros(3*nel,1);
        cauchystress=zeros(3*nel,1);
        for i=1:nnode
            for j=1:3
                dispcoord(i,j)=disp(3*(i-1)+j);
            end
        end
        gcoordupdate=gcoordupdate+dispcoord;
%-------------------------------------------------------------------------
%COMPUTATION OF STIFFNESS MATRIX
%-------------------------------------------------------------------------
for iel=1:nel
            for i=1:nnel
                nd(i)=node(iel,i);
                xcoord(i)=gcoord(nd(i),1);
                ycoord(i)=gcoord(nd(i),2);
                zcoord(i)=gcoord(nd(i),3);
                xcoordupdate(i)=gcoordupdate(nd(i),1);
                ycoordupdate(i)=gcoordupdate(nd(i),2);
                zcoordupdate(i)=gcoordupdate(nd(i),3);
            end
            index=feeldof(nd,nnel,ndof);
            for i=1:edof
                dispel(i)=disp(index(i));
                totaldispel(i)=totaldisp(index(i));
            end

[tran,xprime,yprime,tran2D]=ftran(xcoord,ycoord,zcoord,nnel);

[blank,xprimeupdate,yprimeupdate,tran2Dup]=ftran(xcoordupdate,ycoordupd
ate,zcoordupdate,1);
            dispel=tran*dispel;
            totaldispel=tran*totaldispel;
            kL=zeros(edof,edof);
            kNL=zeros(edof,edof);
            ke=zeros(edof,edof);
            Finel=zeros(edof,1);
            for intx=1:1
                x=point(intx,1);
                wtx=weight(intx,1);
                for inty=1:1
                    y=point(inty,2);
                    wty=weight(inty,2);
                    [shape,dr,ds]=fdrds(x,y);
                    jacob=fjacob(nnel,dr,ds,xprime,yprime);
                    detjacob=det(jacob);
                    invjacob=inv(jacob);
```

```
                    [dhdx,dhdy]=fdxdy(nnel,dr,ds,invjacob);
                    for i=1:3
                        inistress(i)=inistressG(3*(iel-1)+i);
                    end

[strainel]=fstrain(dispel,totaldispel,dhdx,dhdy,nnel);
                    [mat,tran2]=fmat(3,E,poisson,tran2D);

[tstresselpiola,vstresselpiola,mstressel]=fstressel(mat,strainel,inistr
ess');

[dxdX]=fdefgrad(nnel,dr,ds,invjacob,xprimeupdate,yprimeupdate);
                    [bL]=fkinlinear(dhdx,dhdy,totaldispel,nnel);
                    [bNL]=fkinnonlinear(dhdx,dhdy);
                    kL=kL+t*detjacob/2*bL'*mat*bL;
                    kNL=kNL+t*detjacob/2*bNL'*tstresselpiola*bNL;
                    k=kL+kNL;
                    Finel=Finel+t*detjacob/2*bL'*vstresselpiola;
                end
            end
            ke=tran'*k*tran;
            Fe=tran'*Finel;
            index=feeldof(nd,nnel,ndof);
            kk=feasmbl(kk,ke,index);
            for i=1:edof
                Fglobal(index(i))=Fglobal(index(i))+Fe(i);
            end
            cauchystressel=1/det(dxdX)*dxdX*mstressel*dxdX';

vcauchystressel=[cauchystressel(1,1);cauchystressel(2,2);cauchystressel
(1,2);];
            for i=1:3
                cauchystress((3*(iel-1)+i))=cauchystress((3*(iel-
1)+i))+vcauchystressel(i);
                gstress((3*(iel-1)+i))=gstress((3*(iel-
1)+i))+vstresselpiola(i);
            end
        end
        inistressG=gstress';
%-------------------------------------------------------------------------
%solve for displacement
%-------------------------------------------------------------------------
ff=fini-Fglobal;
        %-------------------------------------------------------------------
        %check error
        %-------------------------------------------------------------------
        error=ff;
        for i=1:nbc
            error(bcdof(i))=0;
        end
        if iter<2
            error(1)=1;
        end
        if norm(error,inf)<1e-7
            break;
        end
```

```
        %-------------------------------------------------------------------
if iter>1
            bcval=zeros(1,nbc);
        end
        [kk,ff]=fbdc(kk,ff,bcdof,bcval);
        disp=kk\ff;
        for i=1:sdof
            displacement(i,iter)=disp(i);
        end
        totaldisp=totaldisp+disp;
    end
    total=totaldisp+total;
end
ncoord=gcoordupdate;
```

ภาคผนวก ง
โปรแกรม Matlab สำหรับการวิเคราะห์โครงสร้างเปลือกบางแบบไม่เชิงเส้น
โปรแกรมคำนวณรูปแบบโครงสร้าง

```
%Calculate form finding
%-------------------------------------------------------------------------
%INPUT DATA
%-------------------------------------------------------------------------
clear
nn=load('C:\formfinding\nn.txt');
nncable=load('C:\formfinding\nncable.txt');
nel=nn(1);  %number of triangle elements
nelcable=nncable(1);    %number of cable elements
nnode=nn(2);    %number of nodes triangle
nnodecable=nncable(2);  %number of node cable
ndof=nn(3); %number of degree of freedoms per node triangle
ndofcable=nncable(5);   %number of degree of freedoms per node triangle
nnel=nn(4); %number of node triangle elements
nnelcable=2;    %number of node cable elements
nbc=nn(5);  %number of boundaryconditions
E=nn(6);    %young modulus for surface
Ecable=nncable(3); %young modulus for cable
poisson=nn(7);  %poisson ratio
t=nn(8);    %thick (metre)
Acable=nncable(4);  %cable area
scalfactor=nn(9);   %factor for deformation shape
nglx=2; %number of sampling points in x direction
ngly=2; %number of sampling points in y direction
sdof=nnode*ndof;    %number of degree of freedoms triangle
edof=nnel*ndof; %number of degree of elements triangle
nglxcable=1;    %number of sampling points in x direction cable
sdofcable=nnodecable*ndofcable; %number of degree of freedoms cable
edofcable=nnelcable*ndofcable;  %number of degree of elements cable
%-------------------------------------------------------------------------
%INPUT COORDINATES
%-------------------------------------------------------------------------
gcoord=load('C:\formfinding\gcoord.txt');
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=load('C:\formfinding\node.txt');
nodecable=load('C:\formfinding\nodecable.txt');
%-------------------------------------------------------------------------
%INPUT BOUNDARY CONDITIONS
%-------------------------------------------------------------------------
bcdof=load('C:\formfinding\bcdof.txt');
bcval=load('C:\formfinding\bcval.txt');
%-------------------------------------------------------------------------
%INPUT FORCE VECTOR (kg)
%-------------------------------------------------------------------------
fini=load('C:\formfinding\ff.txt');
%-------------------------------------------------------------------------
%INPUT INITIAL STRESSES
inistressG=load('C:\formfinding\inistress.txt');
inistresscableG=load('C:\formfinding\inistresscable.txt');
%-------------------------------------------------------------------------
%SIZE OF MATRICES
%-------------------------------------------------------------------------
gcoordupdate=gcoord;
for increment=1:1000
    disp=zeros(sdof,1);
```

```
    totaldisp=zeros(sdof,1);
    inistress=zeros(3,1);
    gcoord=gcoordupdate;
    inistressG=load('C:\formfinding\inistress.txt');
    inistresscableG=load('C:\formfinding\inistresscable.txt');
    displacement=zeros(sdof,100);
    Fin=zeros(sdof,1);
    dispel=zeros(edof,1);
    dispelcable=zeros(edofcable,1);
    totaldispel=zeros(edof,1);
    totaldispelcable=zeros(edofcable,1);
    index=zeros(edof,1);
    bL=zeros(1,edof);
    bLcable=zeros(1,edofcable);
    bNL=zeros(2,edof);
    bNLcable=zeros(2,edofcable);
    tran=zeros(nnel*2,edof);
    trancable=zeros(nnelcable*2,edofcable);
    [point,weight]=fpntwght2(nglx,ngly);
    [pointcable,weightcable]=fpntwght1cable(nglxcable);
    for iter=1:500
        kk=zeros(sdof,sdof);
        Fglobal=zeros(sdof,1);
        globalstress=zeros(3*nel,1);
        globalstresscable=zeros(nelcable,1);
        cauchystress=zeros(3*nel,1);
        for i=1:nnode
            for j=1:3
                dispcoord(i,j)=disp(3*(i-1)+j);
            end
        end
        gcoordupdate=gcoordupdate+dispcoord;
%-------------------------------------------------------------------------
%COMPUTATION STIFFNESS MATRIX (triangle)
%-------------------------------------------------------------------------
for iel=1:nel
            for i=1:nnel
                nd(i)=node(iel,i);
                xcoord(i)=gcoord(nd(i),1);
                ycoord(i)=gcoord(nd(i),2);
                zcoord(i)=gcoord(nd(i),3);
                xcoordupdate(i)=gcoordupdate(nd(i),1);
                ycoordupdate(i)=gcoordupdate(nd(i),2);
                zcoordupdate(i)=gcoordupdate(nd(i),3);
            end
            index=feeldof(nd,nnel,ndof);
            for i=1:edof
                dispel(i)=disp(index(i));
                totaldispel(i)=totaldisp(index(i));
            end

[tran,xprime,yprime,tran2D]=ftran(xcoord,ycoord,zcoord,nnel);
[blank,xprimeupdate,yprimeupdate,tran2Dup]=ftran(xcoordupdate,ycoordupd
ate,zcoordupdate,1);
            dispel=tran*dispel;
            totaldispel=tran*totaldispel;
            kL=zeros(edof,edof);
```

```
            kNL=zeros(edof,edof);
            ke=zeros(edof,edof);
            Finel=zeros(edof,1);
            for intx=1:1
                x=point(intx,1);
                wtx=weight(intx,1);
                for inty=1:1
                    y=point(inty,2);
                    wty=weight(inty,2);
                    [shape,dr,ds]=fdrds(x,y);
                    jacob=fjacob(nnel,dr,ds,xprime,yprime);
                    detjacob=det(jacob);
                    invjacob=inv(jacob);
                    [dhdx,dhdy]=fdxdy(nnel,dr,ds,invjacob);
                    for i=1:3
                        inistress(i)=inistressG(3*(iel-1)+i);
                    end

[strainel]=fstrain(dispel,totaldispel,dhdx,dhdy,nnel);
                    [mat,tran2]=fmat(1,E,poisson,tran2D);

[tstressel,vstressel,mstressel]=fstressel(mat,strainel,inistress);

[dxdX]=fdefgrad(nnel,dr,ds,invjacob,xprimeupdate,yprimeupdate);
                    cauchystressel=1/det(dxdX)*dxdX*mstressel*dxdX';
                    [bL]=fkinlinear(dhdx,dhdy,totaldispel,nnel);
                    [bNL]=fkinnonlinear(dhdx,dhdy);
                    kL=kL+t*detjacob/2*bL'*mat*bL;
                    kNL=kNL+t*detjacob/2*bNL'*tstressel*bNL;
                    k=kL+kNL;
                    Finel=Finel+t*detjacob/2*bL'*vstressel;
                end
            end
            ke=tran'*k*tran;
            Fe=tran'*Finel;
            index=feeldof(nd,nnel,ndof);
            kk=feasmbl(kk,ke,index);
            for i=1:edof
                Fglobal(index(i))=Fglobal(index(i))+Fe(i);
            end

vcauchystressel=[cauchystressel(1,1);cauchystressel(2,2);cauchystressel
(1,2);];
            for i=1:3
                globalstress((3*(iel-1)+i))=globalstress((3*(iel-
1)+i))+vstressel(i);
                cauchystress((3*(iel-1)+i))=cauchystress((3*(iel-
1)+i))+vcauchystressel(i);
            end
        end
        inistressG=globalstress';
%-------------------------------------------------------------------------
%CABLE ELEMENT
%-------------------------------------------------------------------------
for iel=1:nelcable
            for i=1:nnelcable    %extract connect node
                nd(i)=nodecable(iel,i);
```

```
                xcoord(i)=gcoord(nd(i),1);
                ycoord(i)=gcoord(nd(i),2);
                zcoord(i)=gcoord(nd(i),3);
            end
            index=feeldof(nd,nnelcable,ndofcable);
            for i=1:edofcable
                dispelcable(i)=disp(index(i));
                totaldispelcable(i)=totaldisp(index(i));    %extract
for element displacement
            end
[trancable,xprime]=ftrancable(xcoord,ycoord,zcoord,nnelcable);
%tranformation matrix for C0
            dispelcable=trancable*dispelcable;
            totaldispelcable=trancable*totaldispelcable;    %transform
to local displacement
            kLcable=zeros(edofcable,edofcable); %local linear stiffness
matrix
            kNLcable=zeros(edofcable,edofcable);    %local nonlinear
stiffness matrix
            ke=zeros(edofcable,edofcable); %global stiffness matrix
            Finel=zeros(edofcable,1);   %local element force
            for intx=1:nglxcable    %integrate by Gausspoint
                x=pointcable(intx,1);
                wtx=weightcable(intx,1);
                [shape,dr]=fdrdscable(x);   %compute isoparametric two
nodes shape function
                jacob=fjacobcable(nnelcable,dr,xprime); %compute
jacobian matrix
                detjacob=det(jacob);
                invjacob=inv(jacob);
                [dhdx]=fdxdycable(nnelcable,dr,invjacob);   %compute
dirivative by x
                inistresscable=inistresscableG;

[stressel]=fstresscable(dispelcable,totaldispelcable,dhdx,nnelcable,Eca
ble,inistresscableG,iel);    %compute element strees

bLcable=fkinlinearcable(nnelcable,invjacob,dr,totaldispelcable,xprime);
%compute linear kinematic matrix
                bNLcable=fkinnonlinearcable(nnelcable,invjacob,dr);
%compute nonlinear kinematic matrix

kLcable=kLcable+bLcable'*Ecable*Acable*bLcable*wtx*jacob;    %compute
linear stiffness

kNLcable=kNLcable+Acable*bNLcable'*stressel*bNLcable*wtx*detjacob;
%compute nonliear stiffness
                k=kLcable+kNLcable;
                Finel=Finel+bLcable'*stressel*Acable*wtx*detjacob;
%compute internal force of element
            end
            ke=trancable'*k*trancable;
            Fe=trancable'*Finel;
            index=feeldof(nd,nnelcable,ndofcable);
            kk=feasmbl(kk,ke,index);
            for i=1:edofcable
```

```
                Fglobal(index(i))=Fglobal(index(i))+Fe(i);   %assemble
force to global
            end
            globalstresscable(iel)=globalstresscable(iel)+stressel;
%total stress vector
        end
        inistresscableG=globalstresscable';
%-------------------------------------------------------------------------
        ff=fini-Fglobal;
        %-----------------------------------------------------------------
        %check error
        %-----------------------------------------------------------------
        error=ff;
        for i=1:nbc
            error(bcdof(i))=0;
        end
        if iter<2
            error(1)=1;
        end
        if norm(error,inf)<1e-6
            break;
        end
        %-----------------------------------------------------------------
        if iter>1
            bcval=zeros(1,nbc); %set boundary C1 are zeros
        end
        [kk,ff]=fbdc(kk,ff,bcdof,bcval);
        disp=kk\ff;
        for i=1:sdof
            displacement(i,iter)=disp(i);
        end
        totaldisp=totaldisp+disp;
    end
    for i=1:nel
        for j=1:3
            stressoutput(i,j)=cauchystress(3*(i-1)+j);
        end
    end
end
```

```
function [shape,dr]=fdrds(rvalue)
%shape function divide by dr and ds cable
%-------------------------------------------------------------------------
%shape function line
%-------------------------------------------------------------------------
shape(1)=(1-rvalue)/2;
shape(2)=(1+rvalue)/2;
%-------------------------------------------------------------------------
%dirivative
%-------------------------------------------------------------------------
dr(1)=-1/2;
dr(2)=1/2;

function [dhdx]=fdxdy(nnel,dr,invjacob)
%determine dirivative with x
for i=1:nnel
    dhdx(i)=invjacob*dr(i);
end

function [jacob]=fjacob(nnel,dr,xprime)
%determine jacobain 2 dimensions cable
jacob=0;
for i=1:nnel
    jacob=jacob+dr(i)*xprime(3*(i-1)+1);
end

function [bL]=fkinlinear(nnel,invjacob,dr,totaldispel,xprime)
%determine linear cable
H=zeros(3,(3*nnel));
for i=1:3
    for j=1:nnel
        H(i,(j-1)*3+i)=dr(j);
    end
end
bL=invjacob^2*(xprime*H'*H+totaldispel'*H'*H);

function [bNL]=fkinnonlinear(nnel,invjacob,dr)
%determine nonlinear cable
H=zeros(3,(3*nnel));
for i=1:3
    for j=1:nnel
        H(i,(j-1)*3+i)=dr(j);
    end
end
bNL=invjacob*H;
```

```
function [mat]=fmat(no,e,poisson)
%determine constitutive equation cable
if no==1
    mat=e/(1-poisson*poisson)*[1 poisson 0;poisson 1 0;0 0 (1-
poisson)/2];
elseif no==2
    mat=e/((1+poisson)*(1-2*poisson))*[(1-poisson) poisson 0;poisson
(1-poisson) 0;0 0 (1-2*poisson)/2];
else
    mat=e/((1+poisson)*(1-2*poisson))*[(1-poisson) poisson poisson 0 0
0;poisson poisson (1-poisson) 0 0 0;0 0 0 (1-2*poisson)/2 0 0;0 0 0 0
(1-2*poisson)/2 0;0 0 0 0 0 (1-2*poisson)/2];
end

function [point1,weight1]=fpntwght1(ngl)
%one point and weight cocefficeint
point1=zeros(ngl,1);
weight1=zeros(ngl,1);
if ngl==1
    point1(1)=0.0;
    weight1(1)=2.0;
elseif ngl==2
    point1(1)=-0.577350269189626;
    point1(2)=-point1(1);
    weight1(1)=1.0;
    weight1(2)=weight1(1);
elseif ngl==3
    point1(1)=-0.774596669241483;
    point1(2)=0.0;
    point1(3)=-point1(1);
    weight1(1)=0.555555555555556;
    weight1(2)=0.888888888888889;
    weight1(3)=weight1(1);
elseif ngl==4
    point1(1)=-0.861136311594053;
    point1(2)=-0.339981043584856;
    point1(3)=-point1(2);
    point1(4)=-point1(1);
    weight1(1)=0.347854845137454;
    weight1(2)=0.652145154862546;
    weight1(3)=weight1(2);
    weight1(4)=weight1(1);
else
    point1(1)=-0.906179845938664;
    point1(2)=-0.538469310105683;
    point1(3)=0.0;
    point1(4)=-point1(2);
    point1(5)=-point1(1);
    weight1(1)=0.236926885056189;
    weight1(2)=0.478628670499366;
    weight1(3)=0.568888888888889;
    weight1(4)=weight1(2);
    weight1(5)=weight1(1);
end
```

```
function [stressel]=fstress(dispel,totaldispel,dhdx,nnel,E,inistress,
iel)
%calculation streess in element cable
totaldispel=totaldispel-dispel;
xpart=0;
ypart=0;
zpart=0;
xparttotal=0;
yparttotal=0;
zparttotal=0;
for i=1:nnel
    xpart=xpart+dhdx(i)*dispel((3*(i-1)+1));
    ypart=ypart+dhdx(i)*dispel((3*(i-1)+2));
    zpart=zpart+dhdx(i)*dispel((3*(i-1)+3));
    xparttotal=xparttotal+dhdx(i)*totaldispel((3*(i-1)+1));
    yparttotal=yparttotal+dhdx(i)*totaldispel((3*(i-1)+2));
    zparttotal=zparttotal+dhdx(i)*totaldispel((3*(i-1)+3));
end
strain=xpart+xparttotal*xpart+yparttotal*ypart+zparttotal*zpart+1/2*(xp
art^2+ypart^2+zpart^2);
stressel=E*strain+inistress(iel);

function [tran,xprime]=ftran(xcoord,ycoord,zcoord,n)
%transfrom matrix cable
%-------------------------------------------------------------------------
%compute vector on plane
%-------------------------------------------------------------------------
v12x=xcoord(2)-xcoord(1);
v12y=ycoord(2)-ycoord(1);
v12z=zcoord(2)-zcoord(1);
l12=sqrt(v12x^2+v12y^2+v12z^2);
v13x=0;
v13y=v12z;
v13z=-v12y;
l13=sqrt(v13x^2+v13y^2+v13z^2);
if l13<0.000000000001
    v13x=-v12z;
    v13y=0;
    v13z=v12x;
end
l13=sqrt(v13x^2+v13y^2+v13z^2);
%-------------------------------------------------------------------------
%compute vector Z
%-------------------------------------------------------------------------
vzx=v13x/l13;
vzy=v13y/l13;
vzz=v13z/l13;
%-------------------------------------------------------------------------
%cross product
%-------------------------------------------------------------------------
v1zx=vzy*v12z-vzz*v12y;
v1zy=-vzx*v12z+v12x*vzz;
v1zz=vzx*v12y-v12x*vzy;
l1z=sqrt(v1zx^2+v1zy^2+v1zz^2);
%-------------------------------------------------------------------------
%compute local axes
```

```
%-------------------------------------------------------------------------
vxx=v12x/l12;
vxy=v12y/l12;
vxz=v12z/l12;
vyx=v1zx/l1z;
vyy=v1zy/l1z;
vyz=v1zz/l1z;
%-------------------------------------------------------------------------
%compute tranformation matrix
%-------------------------------------------------------------------------
for i=1:n
    i1=3*i-2;
    i2=i1+1;
    i3=i2+1;
    j1=3*i-2;
    j2=j1+1;
    j3=j2+1;
    tran(j1,i1)=vxx;
    tran(j1,i2)=vxy;
    tran(j1,i3)=vxz;
    tran(j2,i1)=vyx;
    tran(j2,i2)=vyy;
    tran(j2,i3)=vyz;
    tran(j3,i1)=vzx;
    tran(j3,i2)=vzy;
    tran(j3,i3)=vzz;
end
%-------------------------------------------------------------------------
%compute local coordinates
%-------------------------------------------------------------------------
v12x=xcoord(2)-xcoord(1);
v12y=ycoord(2)-ycoord(1);
v12z=zcoord(2)-zcoord(1);
l12=sqrt(v12x^2+v12y^2+v12z^2);
xprime=[0 0 0 l12 0 0];
```

```
%Input data for cable program
%-------------------------------------------------------------------------
clear
nel=76; %number of elements
nnode=67;    %number of nodes
E=3e7;  %young modulus (kg/mxm)
A=0.25; %Area(mxm)
ndof=3; %number of degree of freedoms per node
sdof=nnode*ndof;
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=[];
%-------------------------------------------------------------------------
%STRESS
%-------------------------------------------------------------------------
inistress=zeros(nel,1);
for i=1:nel
    inistress(i)=5000;
end
%-------------------------------------------------------------------------
nn=[nel nnode E A ndof];
save('C:\Formfinding\nncable.txt','nn','-ascii','-double');
save('C:\Formfinding\nodecable.txt','node','-ascii','-double');
save('C:\Formfinding\inistresscable.txt','inistress','-ascii','-
double');
```

ภาคผนวก จ

โปรแกรม Matlab สำหรับการวิเคราะห์โครงสร้างเปลือกบางแบบไม่เชิงเส้น โปรแกรมคำนวณสมการที่ (3.6) สำหรับคำสั่ง fminunc ของ Matlab

```
%Transfer object in 3D to 2D by optimization tool box
%-------------------------------------------------------------------------
%Input data
%-------------------------------------------------------------------------
clear
nel=165;     %number of element
nnode=66;    %number of node
nn=[nel nbc];
save('C:\Flattening\nn.txt','nn','-ascii','-double');
%-------------------------------------------------------------------------
%INPUT COORDINATE (meters)
%-------------------------------------------------------------------------
gcoord=[];
save('C:\Flattening\gcoord.txt','gcoord','-ascii','-double');
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=[];
save('C:\Flattening\node.txt','node','-ascii','-double');
%-------------------------------------------------------------------------
%Optimizing program
%-------------------------------------------------------------------------
for i=1:nnode
    for j=1:2
        x0(2*i+j-2)=gcoord(i,j);     %starting guesss
    end
end
options = optimset('MaxFunEvals',40000,'maxiter',5000,'tolfun',1e-
20,'tolcon',1e-20,'tolx',2e-20);
[x,fval,exitflag,output,grad] = fmincon(@lengthfun,x0,options)
for i=1:nnode
    for j=1:2
        newcoord(i,j)=x(2*i+j-2);
    end
end
save('C:\Flattening\newcoord.txt','newcoord','-ascii','-double');
```

```
function [f,g]=lengthfun(x)
%ceate differential length between 3D and 2D function
%-------------------------------------------------------------------------
nn=load('C:\Flattening\nn.txt');
nl=nn(1);    %number of lengths
%-------------------------------------------------------------------------
%INPUT COORDINATE (meters)
%-------------------------------------------------------------------------
gcoord=load('C:\Flattening\gcoord.txt');
%-------------------------------------------------------------------------
%INPUT NODE CONNECTIVITY
%-------------------------------------------------------------------------
node=load('C:\Flattening\node.txt');
%-------------------------------------------------------------------------
for i=1:nl  %count length element
    for j=1:2   %count node per element
        nd(j)=node(i,j);
        for k=1:3
            a(3*j-3+k)=gcoord(nd(j),k);
        end
        for k=1:2
            b(2*j-2+k)=x(2*nd(j)-2+k);
        end
    end
    length(i)=sqrt((a(4)-a(1))^2+(a(5)-a(2))^2+(a(6)-a(3))^2);
%calculate length in x-y-z coordinate
    lengthinplane(i)=sqrt((b(3)-b(1))^2+(b(4)-b(2))^2); %calculate
length in plane x-y
    s(i)=length(i)-lengthinplane(i);
end
%-------------------------------------------------------------------------
f=1/2*(s)*(s)'; %target function
```

ภาคผนวก ฉ
ข้อมูลเข้าสำหรับตัวอย่างที่ 1

ตารางที่ ฉ1 พิกัดโครงสร้างเริ่มต้น

จุดต่อ	X	Y	Z
1	-120	-120	0
2	-60	-120	0
3	0	-120	0
4	60	-120	0
5	120	-120	0
6	120	-60	0
7	120	0	0
8	120	60	0
9	120	120	0
10	60	120	0
11	0	120	0
12	-60	120	0
13	-120	120	0
14	-120	60	0
15	-120	0	0
16	-120	-60	0
17	-60	-60	0
18	0	-60	0
19	60	-60	0
20	-60	0	0
21	0	0	0
22	60	0	0
23	-60	60	0
24	0	60	0
25	60	60	0

ตารางที่ ฉ2 สภาพเชื่อมโยงของชิ้นส่วนย่อยรูปสามเหลี่ยม

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
1	1	2	17;
2	2	3	18;
3	3	4	18;
4	4	5	19;
5	1	17	16;
6	2	18	17;
7	4	19	18;
8	5	6	19;
9	16	17	20;
10	17	18	21;
11	18	19	21;
12	19	6	22;
13	16	20	15;
14	17	21	20;
15	19	22	21;
16	6	7	22;
17	15	20	14;
18	20	21	23;
19	21	22	25;
20	22	7	8;
21	20	23	14;
22	21	24	23;
23	21	25	24;
24	22	8	25;
25	14	23	13;
26	23	24	12;
27	24	25	10;
28	25	8	9;
29	23	13	12;
30	24	11	12;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
31	24	10	11;
32	25	9	10;

ภาคผนวก ช
ข้อมูลเข้าสำหรับตัวอย่างที่ 2

ตารางที่ ช1 พิกัดโครงสร้างเริ่มต้น

จุดต่อ	X	Y	Z
1	0	-5	0
2	0.5	-4.5	0
3	1	-4	0
4	1.5	-3.5	0
5	2	-3	0
6	2.5	-2.5	0
7	3	-2	0
8	3.5	-1.5	0
9	4	-1	0
10	4.5	-0.5	0
11	5	0	0
12	4.5	0.5	0
13	4	1	0
14	3.5	1.5	0
15	3	2	0
16	2.5	2.5	0
17	2	3	0
18	1.5	3.5	0
19	1	4	0
20	0.5	4.5	0
21	0	5	0
22	-0.5	4.5	0
23	-1	4	0
24	-1.5	3.5	0
25	-2	3	0
26	-2.5	2.5	0
27	-3	2	0
28	-3.5	1.5	0
29	-4	1	0
30	-4.5	0.5	0
31	-5	0	0
32	-4.5	-0.5	0
33	-4	-1	0
34	-3.5	-1.5	0
35	-3	-2	0
36	-2.5	-2.5	0
37	-2	-3	0
38	-1.5	-3.5	0
39	-1	-4	0
40	-0.5	-4.5	0
41	0	-4.5	0
42	-0.5	-4	0
43	0	-4	0
44	0.5	-4	0
45	-1	-3.5	0
46	-0.5	-3.5	0
47	0	-3.5	0
48	0.5	-3.5	0
49	1	-3.5	0
50	-1.5	-3	0
51	-1	-3	0
52	-0.5	-3	0
53	0	-3	0
54	0.5	-3	0
55	1	-3	0
56	1.5	-3	0
57	-2	-2.5	0
58	-1.5	-2.5	0
59	-1	-2.5	0
60	-0.5	-2.5	0
61	0	-2.5	0
62	0.5	-2.5	0
63	1	-2.5	0

จุดต่อ	X	Y	Z
64	1.5	-2.5	0
65	2	-2.5	0
66	-2.5	-2	0
67	-2	-2	0
68	-1.5	-2	0
69	-1	-2	0
70	-0.5	-2	0
71	0	-2	0
72	0.5	-2	0
73	1	-2	0
74	1.5	-2	0
75	2	-2	0
76	2.5	-2	0
77	-3	-1.5	0
78	-2.5	-1.5	0
79	-2	-1.5	0
80	-1.5	-1.5	0
81	-1	-1.5	0
82	-0.5	-1.5	0
83	0	-1.5	0
84	0.5	-1.5	0
85	1	-1.5	0
86	1.5	-1.5	0
87	2	-1.5	0
88	2.5	-1.5	0
89	3	-1.5	0
90	-3.5	-1	0
91	-3	-1	0
92	-2.5	-1	0
93	-2	-1	0
94	-1.5	-1	0
95	-1	-1	0

จุดต่อ	X	Y	Z
96	-0.5	-1	0
97	0	-1	0
98	0.5	-1	0
99	1	-1	0
100	1.5	-1	0
101	2	-1	0
102	2.5	-1	0
103	3	-1	0
104	3.5	-1	0
105	-4	-0.5	0
106	-3.5	-0.5	0
107	-3	-0.5	0
108	-2.5	-0.5	0
109	-2	-0.5	0
110	-1.5	-0.5	0
111	-1	-0.5	0
112	-0.5	-0.5	0
113	0	-0.5	0
114	0.5	-0.5	0
115	1	-0.5	0
116	1.5	-0.5	0
117	2	-0.5	0
118	2.5	-0.5	0
119	3	-0.5	0
120	3.5	-0.5	0
121	4	-0.5	0
122	-4.5	0	0
123	-4	0	0
124	-3.5	0	0
125	-3	0	0
126	-2.5	0	0
127	-2	0	0

จุดต่อ	X	Y	Z
128	-1.5	0	0
129	-1	0	0
130	-0.5	0	0
131	0	0	0
132	0.5	0	0
133	1	0	0
134	1.5	0	0
135	2	0	0
136	2.5	0	0
137	3	0	0
138	3.5	0	0
139	4	0	0
140	4.5	0	0
141	-4	0.5	0
142	-3.5	0.5	0
143	-3	0.5	0
144	-2.5	0.5	0
145	-2	0.5	0
146	-1.5	0.5	0
147	-1	0.5	0
148	-0.5	0.5	0
149	0	0.5	0
150	0.5	0.5	0
151	1	0.5	0
152	1.5	0.5	0
153	2	0.5	0
154	2.5	0.5	0
155	3	0.5	0
156	3.5	0.5	0
157	4	0.5	0
158	-3.5	1	0
159	-3	1	0

จุดต่อ	X	Y	Z
160	-2.5	1	0
161	-2	1	0
162	-1.5	1	0
163	-1	1	0
164	-0.5	1	0
165	0	1	0
166	0.5	1	0
167	1	1	0
168	1.5	1	0
169	2	1	0
170	2.5	1	0
171	3	1	0
172	3.5	1	0
173	-3	1.5	0
174	-2.5	1.5	0
175	-2	1.5	0
176	-1.5	1.5	0
177	-1	1.5	0
178	-0.5	1.5	0
179	0	1.5	0
180	0.5	1.5	0
181	1	1.5	0
182	1.5	1.5	0
183	2	1.5	0
184	2.5	1.5	0
185	3	1.5	0
186	-2.5	2	0
187	-2	2	0
188	-1.5	2	0
189	-1	2	0
190	-0.5	2	0
191	0	2	0

จุดต่อ	X	Y	Z
192	0.5	2	0
193	1	2	0
194	1.5	2	0
195	2	2	0
196	2.5	2	0
197	-2	2.5	0
198	-1.5	2.5	0
199	-1	2.5	0
200	-0.5	2.5	0
201	0	2.5	0
202	0.5	2.5	0
203	1	2.5	0
204	1.5	2.5	0
205	2	2.5	0
206	-1.5	3	0
207	-1	3	0
208	-0.5	3	0
209	0	3	0
210	0.5	3	0
211	1	3	0
212	1.5	3	0
213	-1	3.5	0
214	-0.5	3.5	0
215	0	3.5	0
216	0.5	3.5	0
217	1	3.5	0
218	-0.5	4	0
219	0	4	0
220	0.5	4	0
221	0	4.5	0

ตารางที่ ช2 สภาพเชื่อมโยงของชิ้นส่วนย่อยรูปสามเหลี่ยม

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
1	1	41	40;
2	1	2	41;
3	40	42	39;
4	40	41	42;
5	41	43	42;
6	41	44	43;
7	41	2	44;
8	2	3	44;
9	39	45	38;
10	39	42	45;
11	42	46	45;
12	42	43	46;
13	43	47	46;
14	43	48	47;
15	43	44	48;
16	44	49	48;
17	44	3	49;
18	3	4	49;
19	38	50	37;
20	38	45	50;
21	45	51	50;
22	45	46	51;
23	46	52	51;
24	46	47	52;
25	47	53	52;
26	47	54	53;
27	47	48	54;
28	48	55	54;
29	48	49	55;
30	49	56	55;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
31	49	4	56;
32	4	5	56;
33	37	57	36;
34	37	50	57;
35	50	58	57;
36	50	51	58;
37	51	59	58;
38	51	52	59;
39	52	60	59;
40	52	53	60;
41	53	61	60;
42	53	62	61;
43	53	54	62;
44	54	63	62;
45	54	55	63;
46	55	64	63;
47	55	56	64;
48	56	65	64;
49	56	5	65;
50	5	6	65;
51	36	66	35;
52	36	57	66;
53	57	67	66;
54	57	58	67;
55	58	68	67;
56	58	59	68;
57	59	69	68;
58	59	60	69;
59	60	70	69;
60	60	61	70;
61	61	71	70;
62	61	72	71;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
63	61	62	72;
64	62	73	72;
65	62	63	73;
66	63	74	73;
67	63	64	74;
68	64	75	74;
69	64	65	75;
70	65	76	75;
71	65	6	76;
72	6	7	76;
73	35	77	34;
74	35	66	77;
75	66	78	77;
76	66	67	78;
77	67	79	78;
78	67	68	79;
79	68	80	79;
80	68	69	80;
81	69	81	80;
82	69	70	81;
83	70	82	81;
84	70	71	82;
85	71	83	82;
86	71	84	83;
87	71	72	84;
88	72	85	84;
89	72	73	85;
90	73	86	85;
91	73	74	86;
92	74	87	86;
93	74	75	87;
94	75	88	87;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
95	75	76	88;
96	76	89	88;
97	76	7	89;
98	7	8	89;
99	34	90	33;
100	34	77	90;
101	77	91	90;
102	77	78	91;
103	78	92	91;
104	78	79	92;
105	79	93	92;
106	79	80	93;
107	80	94	93;
108	80	81	94;
109	81	95	94;
110	81	82	95;
111	82	96	95;
112	82	83	96;
113	83	97	96;
114	83	98	97;
115	83	84	98;
116	84	99	98;
117	84	85	99;
118	85	100	99;
119	85	86	100;
120	86	101	100;
121	86	87	101;
122	87	102	101;
123	87	88	102;
124	88	103	102;
125	88	89	103;
126	89	104	103;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
127	89	8	104;
128	8	9	104;
129	33	105	32;
130	33	90	105;
131	90	106	105;
132	90	91	106;
133	91	107	106;
134	91	92	107;
135	92	108	107;
136	92	93	108;
137	93	109	108;
138	93	94	109;
139	94	110	109;
140	94	95	110;
141	95	111	110;
142	95	96	111;
143	96	112	111;
144	96	97	112;
145	97	113	112;
146	97	114	113;
147	97	98	114;
148	98	115	114;
149	98	99	115;
150	99	116	115;
151	99	100	116;
152	100	117	116;
153	100	101	117;
154	101	118	117;
155	101	102	118;
156	102	119	118;
157	102	103	119;
158	103	120	119;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
159	103	104	120;
160	104	121	120;
161	104	9	121;
162	9	10	121;
163	32	122	31;
164	32	105	122;
165	105	123	122;
166	105	106	123;
167	106	124	123;
168	106	107	124;
169	107	125	124;
170	107	108	125;
171	108	126	125;
172	108	109	126;
173	109	127	126;
174	109	110	127;
175	110	128	127;
176	110	111	128;
177	111	129	128;
178	111	112	129;
179	112	130	129;
180	112	113	130;
181	113	131	130;
182	113	132	131;
183	113	114	132;
184	114	133	132;
185	114	115	133;
186	115	134	133;
187	115	116	134;
188	116	135	134;
189	116	117	135;
190	117	136	135;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
191	117	118	136;
192	118	137	136;
193	118	119	137;
194	119	138	137;
195	119	120	138;
196	120	139	138;
197	120	121	139;
198	121	140	139;
199	121	10	140;
200	10	11	140;
201	31	122	30;
202	122	141	30;
203	122	123	141;
204	123	142	141;
205	123	124	142;
206	124	143	142;
207	124	125	143;
208	125	144	143;
209	125	126	144;
210	126	145	144;
211	126	127	145;
212	127	146	145;
213	127	128	146;
214	128	147	146;
215	128	129	147;
216	129	148	147;
217	129	130	148;
218	130	149	148
219	130	131	149;
220	131	132	149;
221	132	150	149;
222	132	133	150;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
223	133	151	150;
224	133	134	151;
225	134	152	151;
226	134	135	152;
227	135	153	152;
228	135	136	153;
229	136	154	153;
230	136	137	154;
231	137	155	154;
232	137	138	155;
233	138	156	155;
234	138	139	156;
235	139	157	156;
236	139	140	157;
237	140	12	157;
238	140	11	12;
239	30	141	29;
240	141	158	29;
241	141	142	158;
242	142	159	158;
243	142	143	159;
244	143	160	159;
245	143	144	160;
246	144	161	160;
247	144	145	161;
248	145	162	161;
249	145	146	162;
250	146	163	162;
251	146	147	163;
252	147	164	163;
253	147	148	164;
254	148	165	164;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
255	148	149	165;
256	149	150	165;
257	150	166	165;
258	150	151	166;
259	151	167	166;
260	151	152	167;
261	152	168	167;
262	152	153	168;
263	153	169	168;
264	153	154	169;
265	154	170	169;
266	154	155	170;
267	155	171	170;
268	155	156	171;
269	156	172	171;
270	156	157	172;
271	157	13	172;
272	157	12	13;
273	29	158	28;
274	158	173	28;
275	158	159	173;
276	159	174	173;
277	159	160	174;
278	160	175	174;
279	160	161	175;
280	161	176	175;
281	161	162	176;
282	162	177	176;
283	162	163	177;
284	163	178	177;
285	163	164	178;
286	164	179	178;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
287	164	165	179;
288	165	166	179;
289	166	180	179;
290	166	167	180;
291	167	181	180;
292	167	168	181;
293	168	182	181;
294	168	169	182;
295	169	183	182;
296	169	170	183;
297	170	184	183;
298	170	171	184;
299	171	185	184;
300	171	172	185;
301	172	14	185;
302	172	13	14;
303	28	173	27;
304	173	186	27;
305	173	174	186;
306	174	187	186;
307	174	175	187;
308	175	188	187;
309	175	176	188;
310	176	189	188;
311	176	177	189;
312	177	190	189;
313	177	178	190;
314	178	191	190;
315	178	179	191;
316	179	180	191;
317	180	192	191;
318	180	181	192;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
319	181	193	192;
320	181	182	193;
321	182	194	193;
322	182	183	194;
323	183	195	194;
324	183	184	195;
325	184	196	195;
326	184	185	196;
327	185	15	196;
328	185	14	15;
329	27	186	26;
330	186	197	26;
331	186	187	197;
332	187	198	197;
333	187	188	198;
334	188	199	198;
335	188	189	199;
336	189	200	199;
337	189	190	200;
338	190	201	200;
339	190	191	201;
340	191	192	201;
341	192	202	201;
342	192	193	202;
343	193	203	202;
344	193	194	203;
345	194	204	203;
346	194	195	204;
347	195	205	204;
348	195	196	205;
349	196	16	205;
350	196	15	16;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
351	26	197	25;
352	197	206	25;
353	197	198	206;
354	198	207	206;
355	198	199	207;
356	199	208	207;
357	199	200	208;
358	200	209	208;
359	200	201	209;
360	201	202	209;
361	202	210	209;
362	202	203	210;
363	203	211	210;
364	203	204	211;
365	204	212	211;
366	204	205	212;
367	205	17	212;
368	205	16	17;
369	25	206	24;
370	206	213	24;
371	206	207	213;
372	207	214	213;
373	207	208	214;
374	208	215	214;
375	208	209	215;
376	209	210	215;
377	210	216	215;
378	210	211	216;
379	211	217	216;
380	211	212	217;
381	212	18	217;
382	212	17	18;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
383	24	213	23;
384	213	218	23;
385	213	214	218;
386	214	219	218;
387	214	215	219;
388	215	216	219;
389	216	220	219;
390	216	217	220;
391	217	19	220;
392	217	18	19;
393	23	218	22;
394	218	221	22;
395	218	219	221;
396	219	220	221;
397	220	20	221;
398	220	19	20;
399	22	221	21;
400	221	20	21;

ภาคผนวก ซ
ข้อมูลเข้าสำหรับตัวอย่างที่ 3

ตารางที่ ช1 พิกัดโครงสร้างเริ่มต้น

จุดต่อ	X	Y	Z
1	204.73145	1.78E-05	100
2	307.09718	1.78E-05	100
3	409.46291	1.78E-05	100
4	511.82865	1.78E-05	100
5	614.19438	0.00013	100
6	716.56011	3.42E-05	100
7	818.92585	3.42E-05	100
8	921.29158	6.69E-05	100
9	1023.65731	8.32E-05	100
10	1074.84016	88.65142	100
11	1126.02301	177.3028	100
12	1177.20587	265.9541	100
13	1228.38872	354.6054	100
14	1177.20585	443.2567	100
15	1126.02299	531.9081	100
16	1074.84012	620.5594	100
17	1023.65726	709.2107	100
18	921.29153	709.2107	100
19	818.9258	709.2107	100
20	716.56007	709.2107	100
21	614.19434	709.2106	100
22	511.82858	709.2107	100
23	409.46285	709.2107	100
24	307.09712	709.2107	100
25	204.73139	709.2107	100
26	153.54854	620.5593	100
27	102.36569	531.908	100
28	51.18284	443.2567	100
29	-0.00001535	354.6053	100
30	51.18285	265.954	100
31	102.36572	177.3027	100
32	153.54858	88.65134	100
33	255.91431	88.65135	150
34	358.28004	88.65135	150
35	460.64577	88.65135	150
36	563.01151	88.65135	150
37	665.37724	88.65136	150
38	767.74297	88.65138	150
39	870.1087	88.65139	150
40	972.47444	88.65141	150
41	204.73144	177.3027	150
42	307.09717	177.3027	200
43	409.4629	177.3027	200
44	511.82863	177.3027	200
45	614.19439	177.3027	200
46	716.5601	177.3027	200
47	818.92583	177.3027	200
48	921.29156	177.3027	200
49	1023.65729	177.3027	150
50	153.54858	265.954	150
51	255.9143	265.954	200
52	358.28003	265.954	250
53	460.64576	265.954	250
54	563.01153	265.954	250
55	665.37728	265.954	249.9997
56	767.74296	265.9541	250
57	870.10869	265.9541	250
58	972.47442	265.9541	200
59	1074.84014	265.9541	150
60	102.36571	354.6053	150
61	204.73144	354.6053	200
62	307.09716	354.6053	250
63	409.46286	354.6053	300

จุดต่อ	X	Y	Z
64	511.82869	354.6054	300
65	614.19442	354.6054	300
66	716.56015	354.6054	300
67	818.92582	354.6054	300
68	921.29154	354.6054	250
69	1023.65727	354.6054	200
70	1126.02299	354.6054	150
71	153.54856	443.2567	150
72	255.91429	443.2567	200
73	358.28001	443.2567	250
74	460.64574	443.2567	250
75	563.01153	443.2567	249.9997
76	665.37726	443.2567	249.9997
77	767.74295	443.2567	250
78	870.10868	443.2567	250
79	972.4744	443.2567	200
80	1074.84013	443.2567	150
81	204.73141	531.908	150
82	307.09713	531.908	200
83	409.46287	531.908	200
84	511.8286	531.908	200
85	614.19438	531.908	199.9998
86	716.56008	531.908	200
87	818.92581	531.9081	200
88	921.29153	531.9081	200
89	1023.65726	531.9081	150
90	255.91426	620.5593	150
91	358.27999	620.5594	150
92	460.64572	620.5594	150
93	563.01147	620.5594	150
94	665.37721	620.5594	150
95	767.74294	620.5594	150
96	870.10867	620.5594	150
97	972.4744	620.5594	150

ตารางที่ ซ2 สภาพเชื่อมโยงชิ้นส่วนย่อยรูปสามเหลี่ยม

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
1	1	2	33;
2	2	3	34;
3	3	4	35;
4	4	5	36;
5	5	6	37;
6	6	7	38;
7	7	8	39;
8	8	9	40;
9	1	33	32;
10	2	34	33;
11	3	35	34;
12	4	36	35;
13	5	37	36;
14	6	38	37;
15	7	39	38;
16	8	40	39;
17	9	10	40;
18	32	33	41;
19	33	34	42;
20	34	35	43;
21	35	36	44;
22	36	37	45;
23	37	38	46;
24	38	39	47;
25	39	40	48;
26	40	10	49;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
27	32	41	31;
28	33	42	41;
29	34	43	42;
30	35	44	43;
31	36	45	44;
32	37	46	45;
33	38	47	46;
34	39	48	47;
35	40	49	48;
36	10	11	49;
37	31	41	50;
38	41	42	51;
39	42	43	52;
40	43	44	53;
41	44	45	54;
42	45	46	55;
43	46	47	56;
44	47	48	57;
45	48	49	58;
46	49	11	59;
47	31	50	30;
48	41	51	50;
49	42	52	51;
50	43	53	52;
51	44	54	53;
52	45	55	54;
53	46	56	55;
54	47	57	56;
55	48	58	57;
56	49	59	58;
57	11	12	59;
58	30	50	60;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
59	50	51	61;
60	51	52	62;
61	52	53	63;
62	53	54	64;
63	54	55	65;
64	55	56	66;
65	56	57	67;
66	57	58	68;
67	58	59	69;
68	59	12	70;
69	30	60	29;
70	50	61	60;
71	51	62	61;
72	52	63	62;
73	53	64	63;
74	54	65	64;
75	55	66	65;
76	56	67	66;
77	57	68	67;
78	58	69	68;
79	59	70	69;
80	12	13	70;
81	29	60	28;
82	60	61	71;
83	61	62	72;
84	62	63	73;
85	63	64	74;
86	64	65	75;
87	65	66	76;
88	66	67	77;
89	67	68	78;
90	68	69	79;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
91	69	70	80;
92	70	13	14;
93	60	71	28;
94	61	72	71;
95	62	73	72;
96	63	74	73;
97	64	75	74;
98	65	76	75;
99	66	77	76;
100	67	78	77;
101	68	79	78;
102	69	80	79;
103	70	14	80;
104	28	71	27;
105	71	72	81;
106	72	73	82;
107	73	74	83;
108	74	75	84;
109	75	76	85;
110	76	77	86;
111	77	78	87;
112	78	79	88;
113	79	80	89;
114	80	14	15;
115	71	81	27;
116	72	82	81;
117	73	83	82;
118	74	84	83;
119	75	85	84;
120	76	86	85;
121	77	87	86;
122	78	88	87;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
123	79	89	88;
124	80	15	89;
125	27	81	26;
126	81	82	90;
127	82	83	91;
128	83	84	92;
129	84	85	93;
130	85	86	94;
131	86	87	95;
132	87	88	96;
133	88	89	97;
134	89	15	16;
135	81	90	26;
136	82	91	90;
137	83	92	91;
138	84	93	92;
139	85	94	93;
140	86	95	94;
141	87	96	95;
142	88	97	96;
143	89	16	97;
144	26	90	25;
145	90	91	24;
146	91	92	23;
147	92	93	22;
148	93	94	21;
149	94	95	20;
150	95	96	19;
151	96	97	18;
152	97	16	17;
153	90	24	25;
154	91	23	24;

ชิ้นส่วนย่อย	จุดต่อที่ 1	จุดต่อที่ 2	จุดต่อที่ 3
155	92	22	23;
156	93	21	22;
157	94	20	21;
158	95	19	20;
159	96	18	19;
160	97	17	18;

ประวัติผู้เขียนวิทยานิพนธ์

นายวสันต์ ทองพูล เกิดวันที่ 13 ตุลาคม พ.ศ. 2521 มีภูมิลำเนาอยู่ที่ อำเภอท่าม่วง จังหวัดกาญจนบุรี เรียนชั้นมัธยมศึกตอนต้น และมัธยมศึกษาตอนปลายที่โรงเรียนวิสุทธรังษี จังหวัดกาญจนบุรี สำเร็จการศึกษาปริญญาบัณฑิต หลักสูตรวิศวกรรมศาสตร์บัณฑิต จากภาควิชาวิศวกรรมโยธา คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ บางเขน จังหวัดกรุงเทพมหานคร ในปีการศึกษา 2543 และเข้าศึกษาต่อในระดับปริญญามหาบัณฑิต หลักสูตรวิศวกรรมมหาบัณฑิต สาขาวิศวกรรมโครงสร้าง คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2546