40

A High Performance Thin Shell Triangle

TABLE OF CONTENTS

§40.1. Introduction

The HPSHEL3 thin shell element is formed by combining the AQR high performance plate bending triangle [182] with the optimal ANDES (Assumed Natural DEviatoric Strains) formulation of a plane stress (membrane) triangle with drilling degrees of freedom presented in [6,75,90].[1] The element has 3 nodes and 18 degrees of freedom. The implementation of this element is described in this Chapter to support final AFEM projects that rely on that element.

This document describes the computation of the HPSHEL3 element stiffness as implemented in both *Mathematica* and Fortran 77. It also reports simple numeric tests that may be used to verify the element. The description is oriented towards computer implementation, avoiding theoretical derivations.

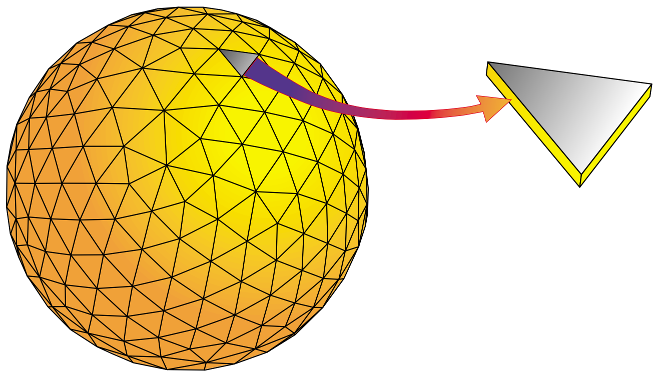

FIGURE 40.1. The HPSHEL thin shell element provides a faceted discretization of a shell structure.

§40.2. Element Overview

The HPSHEL3 shell element is geometrically a flat triangle that can be used to discretize shell structures as illustrated in Figure 40.1. The element geometry is defined by its position of its three corners in a global, right-handed rectangular Cartesian coordinate (RCC) system denoted by (x, y, z), as shown in Figure 40.1.

The element has three nodes located at the corners. It has six degrees of freedom at each node n: three translations (u_{xn}, u_{yn}, u_{zn}) and three rotations: $(\theta_{xn}, \theta_{yn}, \theta_{zn})$, $n = 1, 2, 3$, for a total of 18 degrees of freedom. The freedom configuration is pictured in Figure 40.2. The element is obtained by combining a membrane (plane stress) and a Kirchhoff plate bending component.

[1] Reference [90] is reproduced in Chapter 29 of these Notes.

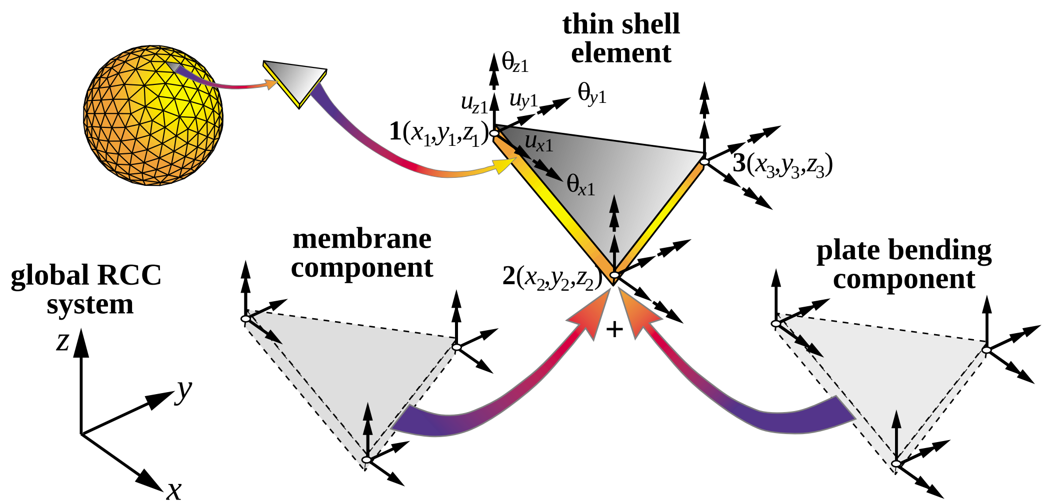

FIGURE 40.2. The HPSHEL3 triangular element, showing geometry and freedom configuration. The element is constructed by combining a membrane (plane stress) and a Krichhoff plate bending component.

§40.3. Element Implementation in *Mathematica*

§40.3.1. Localization

The first operation carried out on the shell element is transformation to a *local* coordinate system $\{\bar{x}, \bar{y}, \bar{z}\}$. For this the corners of the individual triangle are given the local numbers 1,2,3. The triangle geometry is fully defined by giving the corner global coordinates $\{x_i, y_i, z_i\}$, $i = 1, 2, 3$. Assuming that those points are not collinear, they will define the element midsurface plane.

The local frame is selected as illustrated in Figure 40.3:

1. The $\bar{x}$ axis is chosen parallel to side 1→2, with origin at the triangle centroid 0.
2. The $\bar{z}$ axis is normal to the plane of the triangle. Its direction is chosen so that the signed area of the triangle is positive when traversed 1→2→3.
3. The $\bar{y}$ axis is normal to $\bar{x}$ and $\bar{z}$ and forms a right-handed RCC system $\{\bar{x}, \bar{y}, \bar{z}\}$; thus $\bar{z} = \bar{x} \times \bar{y}$. It follows that $\{\bar{x}, \bar{y}\}$ span the triangle plane.

The *Mathematica* module `SM3LocalSystem`, listed in Figure 40.4, performs the global-to-local transformation.[2] The module is invoked as

```
{ncoorbar,dcm,status} = SM3LocalSystem[ncoor]
```
(40.1)

The arguments are

`ncoor` Global node coordinates of element corners stored as the two-dimensional list `{{x1,y1,z1},{x2,y2,z2},{x3,y3,z3}}`.

The function returns are

[2] The commented-out red-colored statements use built-in *Mathematica* functions forms of vector operators such as `Cross[]` for cross product and `.` (period) for dot product. These were replaced by inlined code to speed up the computations.

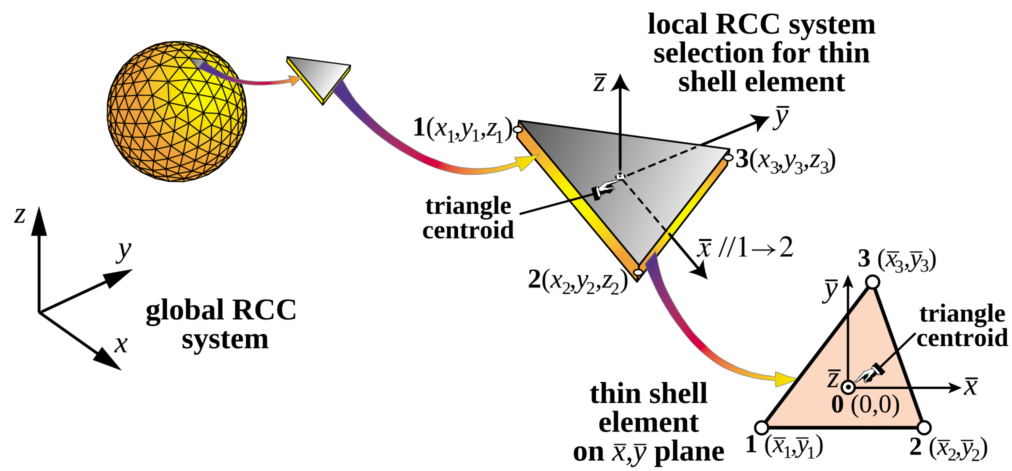

FIGURE 40.3. Selecting the local frame $\{\bar x, \bar y, \bar z\}$.

`ncoorbar` Local node coordinates of element corners stored as the two-dimensional list `{{xbar1,ybar1,0},{xbar2,ybar2,0},{xbar3,ybar3,0}}`.

`dcm` Direction cosine matrix containing the partials of the local system axes with respect to the global ones. More precisely, `dcm[[i,j]]` contains $\partial \bar x_i/\partial x_j$, i, $j = 1, 2, 3$.

`status` A status character string. Blank if no error detected, else error message.

For a simple numerical test of the module, take the triangle corners 1,2,3 as being located at {1, 4, 5}, {2, 6, 7} and {3, 3, 11}, respectively. Hence `ncoor` is `{{1,4,5},{2,6,7},{3,3,11}}`. The call `SM3LocalSystem[ncoor]` returns the local coordinate array `ncoorbar = {{-7/3,-5/3,0},{2/3,-5/3,0},{5/3,10/3,0}}`; thus $\bar x_1 = -7/3$, etc. The returned direction coside matrix is `dcm = {{1/3,2/3,2/3},{2/15,-11/15,2/3},{14/15,-2/15,-1/3}}`. The `status` flag returns blank. Figure 40.5 shows the test in which the foregoing `ncoor` are supplied as floating point format, and so is the output printed in the bottom cell.

(Chapter in progress)

```
SM3ShellLocalSystem[ncoor_]:=Module[{x1,y1,z1,x2,y2,z2,x3,y3,z3,
   x21,y21,z21,x32,y32,z32,xlr,ylr,zlr,x0,y0,z0,xyz10,xyz20,xyz30,
   dx=dy=dz={0,0,0},ncoorbar=dcm=Table[0,{3},{3}],status=" "},
   {{x1,y1,z1},{x2,y2,z2},{x3,y3,z3}}=ncoor;
    {x0,y0,z0}={x1+x2+x3,y1+y2+y3,z1+z2+z3}/3;
   {x21,y21,z21,x32,y32,z32}={x2-x1,y2-y1,z2-z1,x3-x2,y3-y2,z3-z2};
    xlr=Sqrt[x21^2+y21^2+z21^2]; If [xlr<=0,
        status="Nodes 1-2 coincide"; Return[{ncoorbar,dcm,status}]];
    dx={x21,y21,z21}/xlr;
 (*dz=Cross[{x21,y21,z21},{x32,y32,z32}]; too slow: inlined *)
    dz={y21*z32-z21*y32,z21*x32-x21*z32,x21*y32-y21*x32};
    zlr=Sqrt[dz[[1]]^2+dz[[2]]^2+dz[[3]]^2];
    If [zlr<=0, status="Nodes 1-2-3 are collinear";
                Return[{ncoorbar,dcm,status}]];
    dz=dz/zlr; (* dy=Cross[dz,dx]; too slow: inlined *)
    dy[[1]]=dz[[2]]*dx[[3]]-dz[[3]]*dx[[2]];
    dy[[2]]=dz[[3]]*dx[[1]]-dz[[1]]*dx[[3]];
    dy[[3]]=dz[[1]]*dx[[2]]-dz[[2]]*dx[[1]];
    ylr=Sqrt[dy[[1]]^2+dy[[2]]^2+dy[[3]]^2]; dy=dy/ylr;
    xyz10={x1-x0,y1-y0,z1-z0}; xyz20={x2-x0,y2-y0,z2-z0};
    xyz30={x3-x0,y3-y0,z3-z0}; dcm={dx,dy,dz};
 (*ncoorl={{dx.xyz10,dy.xyz10,dz.xyz10},{dx.xyz20,dy.xyz20,dz.xyz20},
           {dx.xyz30,dy.xyz30,dz.xyz30}}; too slow: inlined *)
    ncoorbar={
     {dx[[1]]*xyz10[[1]]+dx[[2]]*xyz10[[2]]+dx[[3]]*xyz10[[3]],
      dy[[1]]*xyz10[[1]]+dy[[2]]*xyz10[[2]]+dy[[3]]*xyz10[[3]],0},
     {dx[[1]]*xyz20[[1]]+dx[[2]]*xyz20[[2]]+dx[[3]]*xyz20[[3]],
      dy[[1]]*xyz20[[1]]+dy[[2]]*xyz20[[2]]+dy[[3]]*xyz20[[3]],0},
     {dx[[1]]*xyz30[[1]]+dx[[2]]*xyz30[[2]]+dx[[3]]*xyz30[[3]],
      dy[[1]]*xyz30[[1]]+dy[[2]]*xyz30[[2]]+dy[[3]]*xyz30[[3]],0}};
     Return[{ncoorbar,dcm,status}]];
```

FIGURE 40.4. Global-to-local system transformation module.

```
ncoor=N[{{1,4,5},{2,6,7},{3,3,11}}];
Print["ncoor=",ncoor//MatrixForm];
{ncoorbar,dcm,status}=SM3ShellLocalSystem[ncoor];
Print["ncoorbar=",ncoorbar//MatrixForm];
Print["dcm=",dcm//MatrixForm];
```

$$\text{ncoor} = \begin{pmatrix} 1. & 4. & 5. \\ 2. & 6. & 7. \\ 3. & 3. & 11. \end{pmatrix}$$

$$\text{ncoorbar} = \begin{pmatrix} -2.33333 & -1.66667 & 0 \\ 0.666667 & -1.66667 & 0 \\ 1.66667 & 3.33333 & 0 \end{pmatrix}$$

$$\text{dcm} = \begin{pmatrix} 0.333333 & 0.666667 & 0.666667 \\ 0.133333 & -0.733333 & 0.666667 \\ 0.933333 & -0.133333 & -0.333333 \end{pmatrix}$$

FIGURE 40.5. Global-to-local system transformation module test.