TICAM Report 02-30

Geometrical Modeling Package. Version 2.0

Dong Xue and Leszek Demkowicz

Texas Institute for Computational and Applied Mathematics
The University of Texas at Austin
Austin, TX 78712

Abstract

The report provides a short documentation for the second edition of the Geometrical Modeling Package (GMP) originally presented in [1]. The GMP can be used for modeling two-dimensional manifolds, both in $\mathbb{R}^2$ (for Finite Element (FE) computations), and in $\mathbb{R}^3$ (for Boundary Element (BE) computations), and three-dimensional manifolds in $\mathbb{R}^3$. Both explicit and implicit parameterizations are used.

The new edition has been implemented in FORTRAN90 with the following changes made, compared with [1]:

- A new data structure, based on user defined objects and supported by FORTRAN90, has been developed. With descriptive names allowed in FORTRAN90 and naturally defined geometrical objects, the logic of the code is easier to follow.
- The code supports the concatenation of two separate, geometrically compatible, objects. This helps dealing with more complicated manifolds which can be separated into disjoint pieces (sub-manifolds), with each piece modeled separately. In practice, GMP input files for non-trivial objects are prepared by writing small auxiliary programs. It is easier to write such programs for smaller, isolated sub-manifolds.
- Inconsistencies, related to the orientation of surfaces, and orientation of faces for prisms and hexahedrons, have been corrected.
- Transfinite interpolation prism and hexahedron have been added.

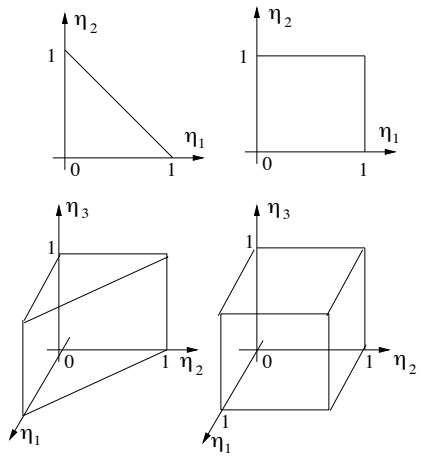

Figure 1: Reference triangle, rectangle,prism and hexahedron

1 Introduction

1.1 Motivation

The success of Finite Element (FE) or Boundary Element (BE) simulations depends much on a precise representation of the geometry of involved objects. The geometry issue becomes especially sensitive for adaptive methods where we strive for high accuracy of simulations, which may be completely offset by errors resulting from a poor geometry representation.

The original version of the Geometrical Modeling Package(GMP) was motivated by research on hp-adaptive discretizations. It has provided a foundation for a multi-block hp mesh generator that has been used in many projects. The small size of the package allows for maintaining a continuous interface with the adaptive codes to update the geometry information during mesh refinements.

The original version of GMP was written in Fortran 77, and it has shared all shortcomings of the language: names could not exceed a few characters and, therefore, were not communicative; the data structure had to be expressed in terms of multiple arrays; no dynamic allocation of memory was possible.

The new version of the package is based on a completely rewritten Fortran 90-like data structure. Additionally, a new important feature has been added to the code - the possibility of concatenating two geometrical objects into one. This feature comes handy when we need to work on complicated geometries that can be broken into disjoint objects which can first be studied individually.

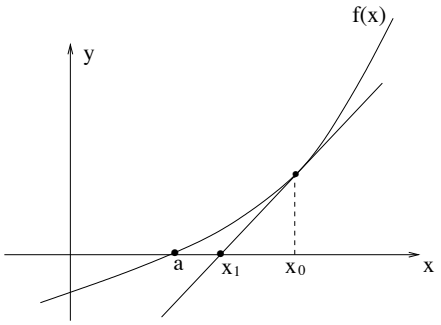

Figure 2: Newton-Raphson iterations

1.2 Main assumptions and parameterizations

Geometrical objects are classified into the following entities: *Points, Curves, Triangles, Rectangles, Prisms, Hexahedrons*. All entities are prescribed in a global Cartesian system of physical coordinates x_i, $i = 1,\ldots,N$, with $N = 2$ for two dimensional problems, and $N = 3$ for three dimension problems.

Each of the entities has a corresponding catalog of objects. For example, the simplest entity, the *points*, contain not only the usual geometrical points uniquely defined by their physical coordinates, but also the implicit points defined by three intersecting surfaces.

Mathematically, each of the geometrical objects is identified with its corresponding parameterization. More specifically, *e.g.,* a curve is a transformation x_c from $[0,1]$ into $\mathbb{R}^N$ *i.e.,*

$$x_c : [0,1] \to \mathbb{R}^N \qquad N = 2,3$$

We define the remaining entities in the same manner. The reference objects are shown in Figure 1. If two or more mappings are used to parameterize a specific geometric object, we shall treat them as describing separate objects.

Conceptually, the parameterizations are classified into two classes: *explicit parameterizations* , and *implicit parameterizations*. In the first case, a mapping is defined *explicitly* by a specific formula. The simplest examples include objects which are characterized uniquely by entities of lower dimension, and a specific interpolation rule, *e.g.,* a segment of straight line is defined by its endpoints, a plane triangle is defined by its edges. In the second case the mappings are defined *implicitly* by specifying systems of nonlinear equations to be solved for coordinates x, y, z.

The structure of our code is open. New definitions can be easily added to the catalogs to enhance the existing capabilities of the code.

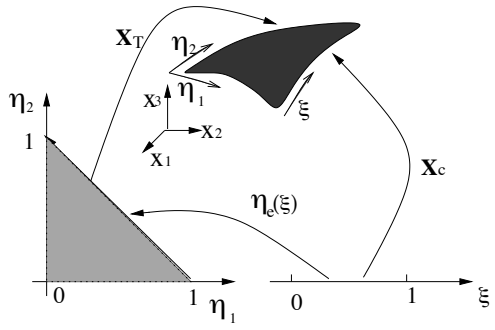

Figure 3: Compatibility of parameterizations for a triangle

1.3 Newton-Raphson technique

For *implicit* parameterizations, in order to assess the value of the mapping for some specific choice of reference coordinates, a nonlinear system of algebraic equations has to be solved using typically a few Newton-Raphson iterations [3].

Let $f : \mathbb{R} \to \mathbb{R}$ be a nonlinear C^1 function illustrated in Fig. 1.2. In the classical Newton method, we iterate towards root $x = a$, starting with an initial approximation x_0, and approximating function $f(x)$ in a neighborhood of x_0 with its tangent line at x_0,

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0). \tag{1.1}$$

The root of the linear approximation $f(x_0) + f'(x_0)(x - x_0) = 0 \Rightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)}$ provides the next iterate toward finding a root of $f(x)$, provided $f'(x_0) \neq 0$.

The Newton-Raphson method is a generalization of the Newton method to a system of algebraic equations represented by a vector-valued function $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$. In order to find a solution to the system $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$, given a starting approximation $\boldsymbol{x}_0$, we compute again the linear approximation as

$$\boldsymbol{f}(\boldsymbol{x}) \approx \boldsymbol{f}(\boldsymbol{x}_0) + \boldsymbol{f}'(\boldsymbol{x}_0)(\boldsymbol{x} - \boldsymbol{x}_0). \tag{1.2}$$

where $\boldsymbol{f}'(\boldsymbol{x}_0)$ is the $N \times N$ Jacobian matrix of first order derivatives $[\boldsymbol{f}'(\boldsymbol{x}_0)]_{ij} = \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}_0)$ The next approximation $\boldsymbol{x}_1$ is obtained from

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 - [\boldsymbol{f}'(\boldsymbol{x}_0)]^{-1}\boldsymbol{f}(\boldsymbol{x}_0) \tag{1.3}$$

provided $\boldsymbol{f}'(\boldsymbol{x}_0)$ is not singular.

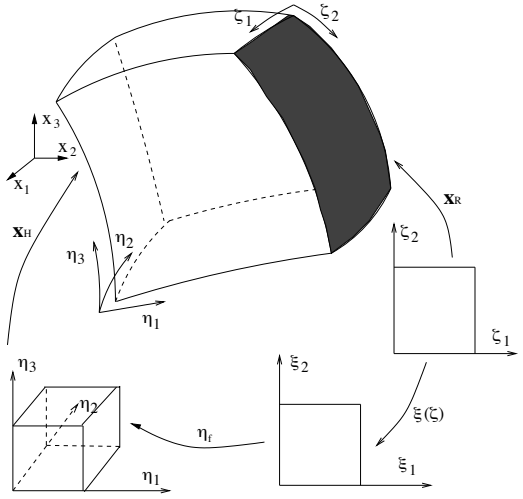

Figure 4: Compatibility of parameterizations for a hexahedron

2 Compatibility of parameterizations

One matter must be very strongly emphasized –*the compatibility of parameterizations*. We begin with a discussion of a typical example - a triangle shown in Fig. 3. Assume that the reference triangle is mapped onto the physical triangle using a map $\boldsymbol{x}_T : (\eta_1, \eta_2) \longrightarrow \boldsymbol{x}_T(\eta_1, \eta_2)$.

Each of the physical triangle edges, a curve, comes with its own *global orientation* (illustrated with the arrow). On the other side, the natural parameterizations $\boldsymbol{\eta} = \eta_e(\xi)$ for the reference triangle edges e,

$$\left\{ \begin{array}{l} \eta_1 = \xi \\ \eta_2 = 0 \end{array} \right. \qquad \left\{ \begin{array}{l} \eta_1 = 1 - \xi \\ \eta_2 = \xi \end{array} \right. \qquad \left\{ \begin{array}{l} \eta_1 = 0 \\ \eta_2 = 1 - \xi \end{array} \right. \tag{2.4}$$

induce *local* orientations for the edges. Consider edge e of the triangle coinciding in the physical space with a curve e, and let $\boldsymbol{x}_c(\xi)$ denote the parameterization for the curve provided by GMP. We request that

$$\boldsymbol{x}_T(\boldsymbol{\eta}_e(\xi')) = \boldsymbol{x}_c(\xi) \tag{2.5}$$

where $\xi' = \left\{ \begin{array}{ll} \xi & \text{if local and global parameterizations are compatible} \\ 1 - \xi & \text{otherwise} \end{array} \right.$
A similar condition is enforced for rectangles.

The situation is more complicated for hexahedrons and prisms. We shall discuss the hexahedron case shown in Fig. 4 first. The *local orientation* of a face can be illustrated with a local face system of coordinates.

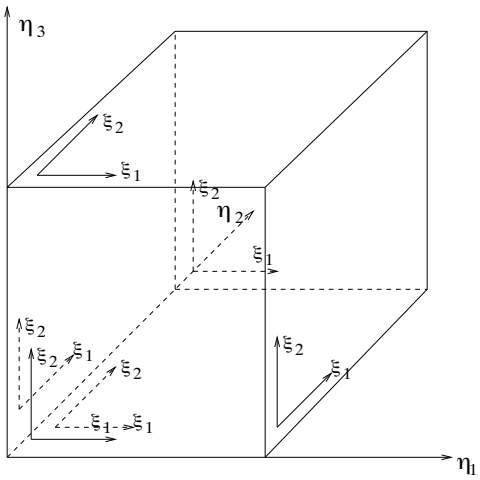

Figure 5: Local face orientations in reference hexahedron

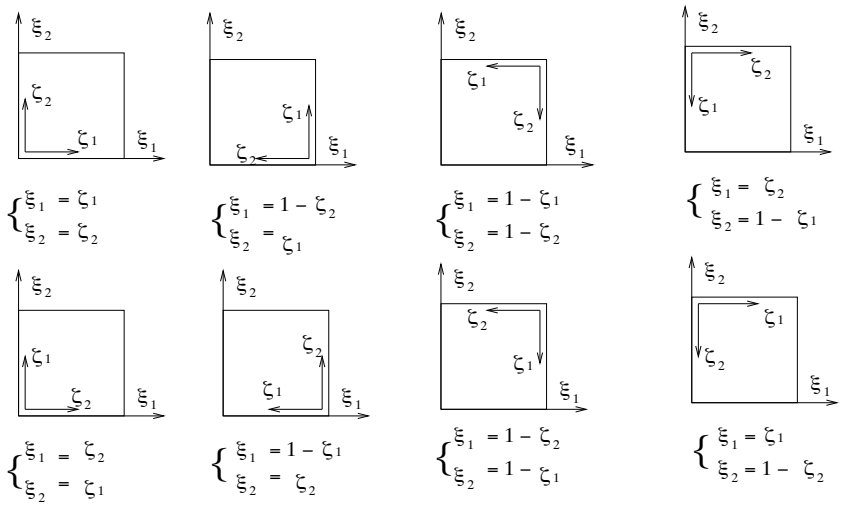

Figure 6: Transformations between local and global reference coordinates for a rectanglar face

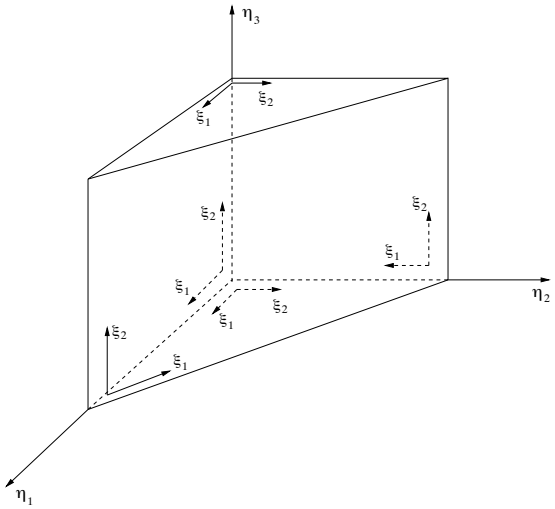

Figure 7: Local face orientations in reference prism

Fig 5 shows the local orientations for each of six faces of the reference hexahedron, corresponding to the lexicographic coordinates,

$$\begin{array}{lcl} \text{bottom} & : & \eta_1 = \xi_1, \quad \eta_2 = \xi_2, \quad \eta_3 = 0 \\ \text{top} & : & \eta_1 = \xi_1, \quad \eta_2 = \xi_2, \quad \eta_3 = 1 \\ \text{front} & : & \eta_1 = \xi_1, \quad \eta_2 = 0, \quad \eta_3 = \xi_2 \\ \text{right} & : & \eta_1 = 1, \quad \eta_2 = \xi_1, \quad \eta_3 = \xi_2 \\ \text{rear} & : & \eta_1 = \xi_1, \quad \eta_2 = 1, \quad \eta_3 = \xi_2 \\ \text{left} & : & \eta_1 = 0, \quad \eta_2 = \xi_1, \quad \eta_3 = \xi_2 \end{array} \tag{2.6}$$

There are now eight possible transformations that relate local and global reference coordinates for of a face [1]. There are depicted in Fig. 6.

Let $\boldsymbol{x}_R$ be now the parameterization for one of the hexahedron faces f. We request that

$$\boldsymbol{x}_H(\boldsymbol{\eta}_f(\boldsymbol{\xi}(\boldsymbol{\zeta}))) = \boldsymbol{x}_R(\boldsymbol{\zeta}) \tag{2.7}$$

where $\boldsymbol{\xi}(\boldsymbol{\zeta})$ reflects the *orientation* of the face, and corresponds to one of the eight cases depicted above, $\boldsymbol{\eta}_f(\boldsymbol{\xi})$ is the local coordinates of the face, and $\boldsymbol{x}_H(\boldsymbol{\eta})$ is the parameterization of the hexahedron.

We enforce the same rule for the prisms. The local parameterization for the five faces of the reference prism are as follows:

[1] In FE computations we talk about the orientation of the face.

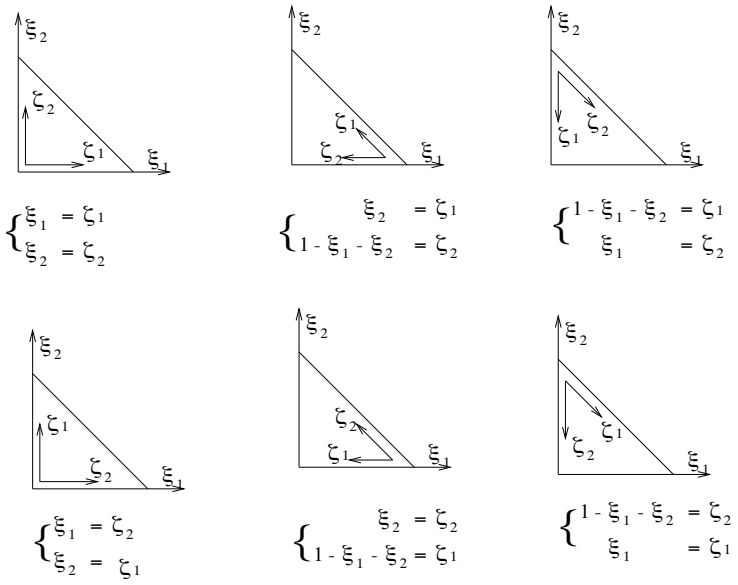

Figure 8: Transformations between local and global coordinates for a triangular face

$$
\begin{array}{lcllll}
\text{bottom} & : & \eta_1 = \xi_1, & \eta_2 = \xi_2, & \eta_3 = 0 \\
\text{top} & : & \eta_1 = \xi_1, & \eta_2 = \xi_2, & \eta_3 = 1 \\
\text{front} & : & \eta_1 = 1 - \xi_1, & \eta_2 = \xi_1, & \eta_3 = \xi_2 \\
\text{right} & : & \eta_1 = 0, & \eta_2 = 1 - \xi_1, & \eta_3 = \xi_2 \\
\text{left} & : & \eta_1 = \xi_1, & \eta_2 = 0, & \eta_3 = \xi_2
\end{array}
\tag{2.8}
$$

They are depicted in Fig. 7, and the transformations reflecting different triangle face orientations are shown in Fig. 8.

3 Data structure

We define seven different geometrical objects listed in Table 1. Each of the definitions includes a complete connectivity information, independent of a particular *Type* of the entity, and two dynamically allocated arrays (pointers) for storing a number of integer and real attributives of the object. The geometrical objects are placed then in separate seven global arrays with the same name [2].

Additionally, a number of global parameters is introduced:

[2]We add 's' to the names, *e.g.,* SURFACES for storing *surface* objects

Entity	*Character*	*Integer*	*Integer Pointer*	*Real Pointer*
Surface	Type		Idata	Rdata
Point	Type	NrCurv	CurvNo , Idata	Rdata
Curve	Type	EndPoNo(2) , NrFig	FigNo , Idata	Rdata
Triangle	Type	EndgeNo(3) , BlockNo(2)	Idata	
Rectangle	Type	EdgeNo(4) ,BlockNo(2)	Idata	
Prism	Type	FigNo(5)	Idata	
Hexahedron	Type	FigNo(6)	Idata	

Table 1: The data in module for different entities

NDIM - dimension of the problem
= 2 for plane problems
= 3 for space problems
MANDIM - dimension of the manifold (2 or 3)
NRPOINT - number of points
NRCURVE - number of curves
NRTRIAN - number of triangles
NRRECTA - number of rectangles
NRPRISM - number of prisms
NRHEXAS - number of hexahedrons
NRSURFS - number of surfaces

4 Catalog of Geometrical Entities

We review definitions of different geometrical entities supported in the code.

4.1 Catalog of surfaces

The catalog of surfaces is summarized in Table 2. Each surface is oriented with the unit vector of the gradient of the surface equation.

- **Plane normal to a given vector and passing through a point (Type ='VecPt')**

  Equation:

$$(x - x_0)a + (y - y_0)b + (z - z_0)c = 0 \tag{4.9}$$

Description of surfaces	Type	Rdata/Description
Plane normal to a given vector and passing through a point	*'VecPt'*	Rdata(1:3) the point coordinates
		Rdata(4:6) the vector components
Plane passing through three points	*'ThrPt'*	Rdata(1:3) the 1st point coordinates
		Rdata(4:6) the 2nd point coordinates
		Rdata(7:9) the 3st point coordinate
Sphere	*'Sphere'*	Rdata(1:3) the center coordinates
		Rdata(4) radius of the sphere
Infinite cylinder	*'Cylinder'*	Rdata(1:3) center point coordinates for the base of the cylinder
		Rdata(4:6) vector parallel to the cylinder axis
		Rdata(7) radius of the cylinder

Table 2: Data for different types of surfaces

x_0, y_0, z_0 - coordinates of the point
a, b, c - components of the normal vector

Unit vector $\boldsymbol{n} = (n_1, n_2, n_3)$ of (a, b, c) specifies the orientation.

- **Plane passing through three points (Type = 'ThrPt')**

  Equation:

$$(x - x_1)a_1 + (y - y_1)a_2 + (z - z_1)a_3 = 0 \tag{4.10}$$

  where,

$$\mathbf{a} = (\mathbf{x}_2 - \mathbf{x}_1) \times (\mathbf{x}_3 - \mathbf{x}_2) \tag{4.11}$$

  with

x_1, y_1, z_1 - coordinates of the first point $\mathbf{x}_1$
x_2, y_2, z_2 - coordinates of the first point $\mathbf{x}_2$
x_3, y_3, z_3 - coordinates of the first point $\mathbf{x}_3$.

Orientation is specified by unit vector of vector $\mathbf{a}$, *i.e.,* the order in which the three points $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}$ are listed, implies the orientation of the plane.

- **Sphere (Type = 'Sphere')**

  Equation:

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = 0 \tag{4.12}$$

  x_0, y_0, z_0 - coordinate of the center of the sphere
  r - radius of the sphere

  The external unit vector $\boldsymbol{n} = (\frac{x}{r}, \frac{y}{r}, \frac{z}{r})$ defines the orientation of the sphere.

- **Infinite cylinder (Type = 'Cylinder')**

  Equation:

$$x_1^2 + y_1^2 - r^2 = 0 \tag{4.13}$$

  where,

$$\mathbf{x}_1 = A(\mathbf{x} - \mathbf{x}_0) \tag{4.14}$$

  with

  A - a transformation matrix whose third column coincides with the unit vector of vector $\mathbf{d}(d_1, d_2, d_3)$ explained below, and the first two are vectors orthogonal to $\mathbf{d}$ and orthogonal to each other
  x_0, y_0, z_0 - coordinates of a point on the cylinder axis
  d_1, d_2, d_3 - a vector parallel to the axis of the cylinder
  r - radius of the cylinder

  The unit vector of gradient of Equation (4.1.5) wrt (x, y, z) defines the orientation of the cylinder.

4.2 Catalog of points

The points supported by the package are listed in Table 3. All points share the same connectivity information.

Description	Type	Attribute
Regular point	*'Regular'*	NrCurv, CurvNo(1:Nrcurv), Rdata(3)
Implicit point	*'Implicit'*	NrCurv, CurvNo(1:Nrcurv), Rdata(3), Idata(3)

Table 3: Data for different types of points

NrCurv - number of curves that meet at the point
CurvNo(1:Nrcurv) - curves' numbers for curves that meet at the point

- **Regular point (Type = 'Regular')**

  Rdata(3) - coordinates of the point

- **Implicit point (Type = 'Implicit')**

  Rdata(3) - coordinates of a starting point for New-Raphson iterations
  Idata(3) - the intersecting surfaces' numbers

4.3 Catalog of curves

The curves supported by the package are listed in Table 4. All curves share the same connectivity information:

EdgePoNo(2) - the endpoints number, listed in order consistent with the curve orientations
NrFig - number of rectangles and triangles adjacent to the curve
FigNo(1:NrFig) - a list of all the nicknames of the rectangles and triangles that are adjacent to the curve.

Nicknames for triangles are defined as :

$$triangle's number * 10 + 1 \tag{4.15}$$

Nicknames for rectangle are defined as :

$$rectangle's number * 10 + 2 \tag{4.16}$$

Additionally, the nicknames are premultiplied with ± 1 sign factor indicating whether the local orientation of the corresponding edge of the figure is consistent with the global orientation of the curve.

Description of curves	Type	Attribute
Segment of straight line	*'Seglin'*	EdgePoNo(2), NrFig, FigNo(1:NrFig)
Quarter of a circle	*'QuaCir'*	EdgePoNo(2), NrFig, FigNo(1:NrFig), Rdata(NDIM)
Segment of a circle	*'SegCir'*	EdgePoNo(2), NrFig, FigNo(1:NrFig), Rdata(NDIM)
Implicit curve	*'ImpCur'*	EdgePoNo(2), NrFig, FigNo(1:NrFig), Rdata(NDIM), Idata(4)

Table 4: Data for different types of curves

- **Segment of straight line (Type ='Seglin')**
- **Quarter of a circle (Type = 'QuaCir')**

  Rdata((NDIM) - coordinates of the circle center

- **Segment of a circle (Type = 'SegCir')**

  Rdata((NDIM) - coordinates of the circle center

- **Implicit curve (Type = 'ImpCir')**

  Rdata((NDIM) - coordinates of a starting point for Newton-Raphson iterations
  Idata(4) - intersecting surfaces' numbers

  Parameterization:

  What we know is the starting point, and the four intersecting surfaces that constitute the curve as shown in Fig.9. Denoting the surface equations by $\varphi_i(x, y, z) = 0$, with $i = 1, \ldots, 4$ we solve the following system of *nonlinear* equations,

$$\begin{aligned} \varphi_1(x, y, z) &= 0 \\ \varphi_2(x, y, z) &= 0 \\ (1 - \xi)\varphi_3(x, y, z) + \xi\varphi_4(x, y, z) &= 0 \end{aligned} \tag{4.17}$$

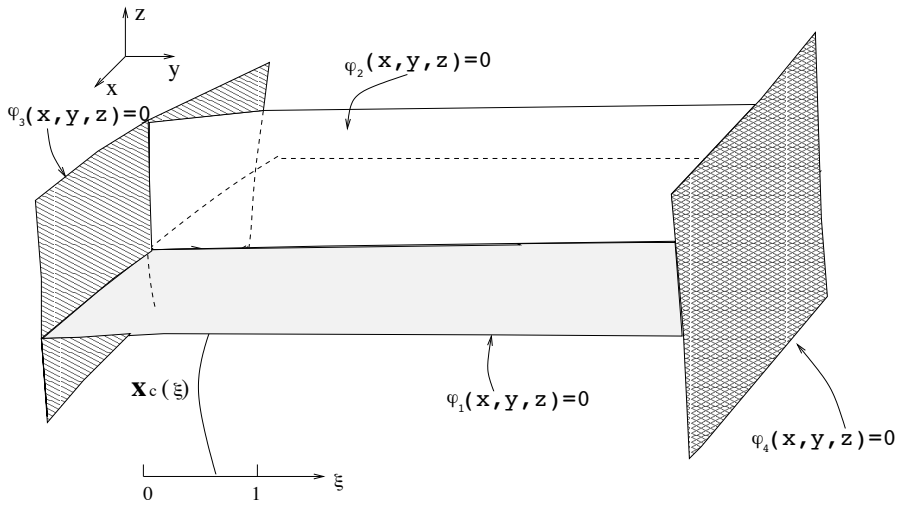

Figure 9: Implicit curve(Four Surfaces define the curve)

The physical coordinates can be now expressed as mapping $x(\xi), y(\xi), z(\xi)$ with $\xi \in [0, 1]$. By differentiating the equation above wrt reference coordinate ξ, we obtain,

$$\begin{aligned} & \frac{\partial \varphi_1}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial \varphi_1}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial \varphi_1}{\partial z}\frac{\partial z}{\partial \xi} = 0 \\ & \frac{\partial \varphi_2}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial \varphi_2}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial \varphi_2}{\partial z}\frac{\partial z}{\partial \xi} = 0 \\ -\varphi_3 + (1-\xi) & \left[\frac{\partial \varphi_3}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial \varphi_3}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial \varphi_3}{\partial z}\frac{\partial z}{\partial \xi}\right] + \\ \varphi_4 + \qquad \xi & \left[\frac{\partial \varphi_4}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial \varphi_4}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial \varphi_4}{\partial z}\frac{\partial z}{\partial \xi}\right] = 0 \end{aligned} \tag{4.18}$$

Once the coordinates x, y, z are determined, the linear system above is solved for the derivatives $\frac{\partial x}{\partial \xi}, \frac{\partial y}{\partial \xi}, \frac{\partial z}{\partial \xi}$.

4.4 Catalog of triangles

The triangles supported by the package are listed in Table 5. All triangles share the same connectivity information:

EdgeNo(3) - curves' numbers that constitute edges of the triangle with ± 1 sign factor indicating whether the global orientation of the curve is consistent with the local orientation for the corresponding triangle edge.

BlockNo(2) - nicknames for the prisms adjacent to the triangle.

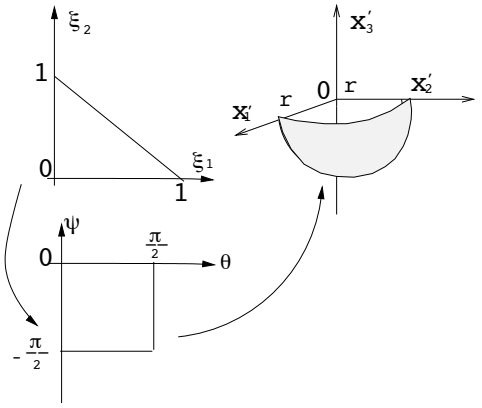

Figure 10: Parameterization of a spherical triangle

The nicknames are defined as,

$$prism'snumber * 10 + norient \tag{4.19}$$

where $norient = 0, 1, \ldots 7$ indicates the global orientation of the triangle as seen from the adjacent prism.

Description of triangles	Type	Attributes
Plane triangle	*'PlaneTri'*	EdgeNo(3), BlockNo(2)
Spherical triangle	*'SpherTri'*	EdgeNo(3), BlockNo(2)
Quarter of a circular	*'QtCirTri'*	EdgeNo(3), BlockNo(2)
Part of Sperical triangle	*'PaSphTri'*	EdgeNo(3), BlockNo(2)
Implicit triangle	*'ImpliTri'*	EdgeNo(3), BlockNo(2), Idata(4)
Implicit Spherical	*'ImSphTri'*	EdgeNo(3), BlockNo(2)

Table 5: Data for different types of triangles

- **Plane triangle (Type = 'PlaneTri')**

  Note that all edges of the triangle must be *segments of straight line*.

- **Spherical triangle (Type ='SpherTri')**

  Note that all edges of the triangle must be *quarters of circles*.

  Parameterization:

  Given an octant of a sphere, shown in Fig. 10 , we construct the corresponding parameterization, mapping the reference triangle onto the octant, by introducing auxiliary spherical coordinates θ, ψ and considering the composition of two *singular* mappings:

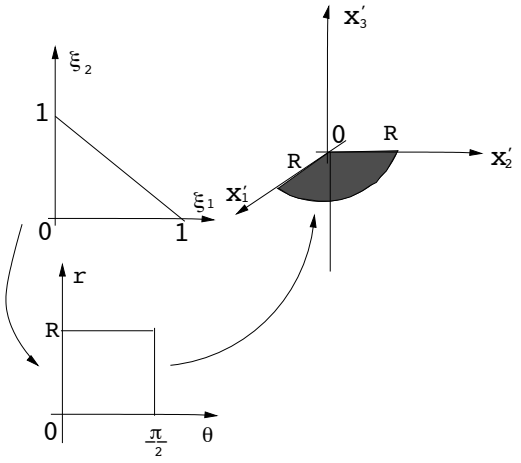

Figure 11: Parameterization of a circular triangle

– inverse map to the map, describing Cartesian product $(0, \frac{\pi}{2}) \times (-\frac{\pi}{2}, 0)$ in (θ, ψ) plane onto the reference triangle,

$$\begin{aligned} \xi_1 &= \frac{4}{\pi}\theta(\frac{1}{\pi}\psi + \frac{1}{2}) \\ \xi_2 &= (\frac{4}{\pi}\theta + 2)(\frac{1}{\pi}\psi + \frac{1}{2}) \end{aligned} \tag{4.20}$$

– the usual spherical coordinates parameterizations,

$$\begin{aligned} x_1' &= r\cos\psi\cos\theta \\ x_2' &= r\cos\psi\sin\theta \\ x_3' &= r\sin\psi \end{aligned} \tag{4.21}$$

It can be checked that the resulting composition is C^1 with bounded derivatives. By combining the map with a rigid rotation, we can parameterize an arbitrary octant of a sphere.

- **Circular triangle (Type = 'QtCirTri')**

  Note that two edges of the triangle must be *straight line segments*, and the remaining edge must be *a quarter of a circle*.

  Parameterization:

  Given a quadrant of a circle shown in Fig. 11, we construct the corresponding parameterization, mapping the reference triangle onto the circular triangle, by introducing auxiliary coordinates θ, r and considering the composition of two *singular* mappings:

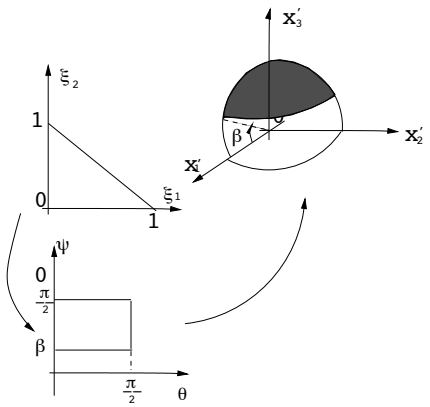

Figure 12: Parameterization for a part of a spherical triangle

– inverse map to the transformation mapping Cartesian product $(0, \frac{\pi}{2}) \times (0, \mathbb{R})$ in (θ, r) plane onto the reference triangle,

$$\begin{aligned} \xi_1 &= \frac{r}{R}\frac{2\theta}{\pi} \\ \xi_1 &= \frac{r}{R}(1 - \frac{2\theta}{\pi}) \end{aligned} \tag{4.22}$$

– the usual polar coordinates parameterization,

$$\begin{aligned} x_1' &= r\cos\theta \\ x_1' &= r\sin\theta \\ x_3' &= 0 \end{aligned} \tag{4.23}$$

By combining the inverse of the first map with the second map, and then adding a rigid rotation in $\mathbb{R}^3$, we can parameterize an arbitrary quadrant of a circular disc, located in $\mathbb{R}^3$.

- **Part of a spherical triangle (Type ='PaSphTri')**

  Parameterization:

  In the same way as with the octant of a sphere, see Fig. 12, we construct the corresponding parameterization, mapping the reference triangle onto the spherical triangle, by introducing auxiliary spherical coordinates (θ, ψ) and considering the composition of two *singular* mappings:

  – inverse map to the transformation mapping Cartesian product $(0, \frac{\pi}{2}) \times (\beta, \frac{\pi}{2})$ in (θ, ψ)

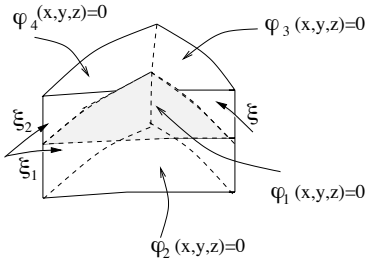

Figure 13: Implicit triangle

plane onto the reference triangle,

$$\begin{aligned} \theta &= \frac{\xi_1 * \frac{\pi}{2}}{\xi_1 + \xi_2} \\ \psi &= (\beta - \frac{\pi}{2} * (\xi_1 + \xi_2) + \frac{\pi}{2} \end{aligned} \tag{4.24}$$

– the usual spherical coordinates parameterization,

$$\begin{aligned} x_1^1 &= r\cos\psi\cos\theta \\ x_2^1 &= r\cos\psi\sin\theta \\ x_3^1 &= r\sin\psi \end{aligned} \tag{4.25}$$

- **Implicit triangle (Type ='ImpliTri')**

  Parameterization:

  The triangle is defined by four surfaces, surface $\varphi_1(x,y,z)$, the triangle is located on, and three surfaces $\varphi_i(x,y,z) = 0, \quad i = 2,\ldots,4$, defining the edges of the triangle, see Fig. 13. We solve the following nonlinear system of equations,

$$\begin{aligned} \varphi_1(x,y,z) &= 0 \\ (1 - f(\xi))\varphi_2(x,y,z) + f(\xi)\varphi_4(x,y,z) &= 0 \\ \xi_1[(1 - f_1(\xi_1+\xi_2))\varphi_5(x,y,z) + f_1(\xi_1+\xi_2)\varphi_3(x,y,z)] &+ \\ \xi_2[(1 - f_2(\xi_1+\xi_2))\varphi_5(x,y,z) + f_2(\xi_1+\xi_2)\varphi_3(x,y,z)] &= 0 \end{aligned} \tag{4.26}$$

  where

  • $\varphi_5(x,y,z)$ is the equation of the degenerated sphere with center at the first vertex of the triangle,

$$\varphi_5(x,y,z) = (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \tag{4.27}$$

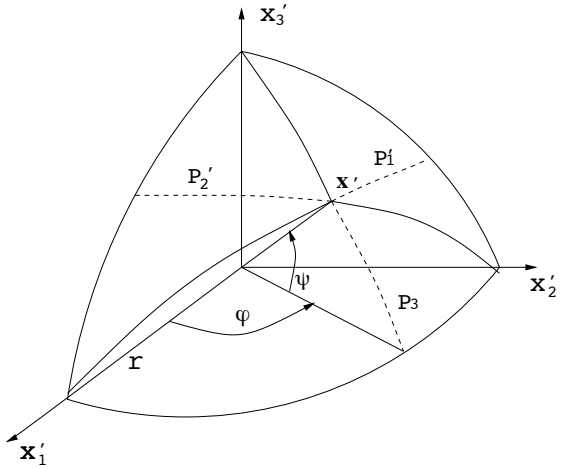

Figure 14: Implicit spherical triangle

with x_1, y_1, z_1 being the coordinates of the first vertex.

- $\xi = \frac{1}{2}\frac{\xi_2 - \xi_1}{\xi_2 + \xi_1} + \frac{1}{2}$
- $f(\xi), f_1(\xi), f_2(\xi)$ are *stretching functions* determined by requesting the compatibility of the triangle parameterization with the existing, specified parameterizations of its edges:

$$\begin{aligned} (1 - f(\xi))\varphi_2(\mathbf{x}_c^2(\xi)) + f(\xi)\varphi_4(\mathbf{x}_c^2(\xi)) &= 0 \\ (1 - f_1(\xi_1))\varphi_5(\mathbf{x}_c^1(\xi_1)) + f_1(\xi_1)\varphi_3(\mathbf{x}_c^1(\xi_1)) &= 0 \\ (1 - f_2(\xi_2))\varphi_2(\mathbf{x_c}^3(\xi_2)) + f_2(\xi_2)\varphi_3(\mathbf{x}_c^3(\xi_2)) &= 0 \end{aligned} \tag{4.28}$$

where $\mathbf{x}_c^1(\xi_1), \mathbf{x}_c^2(\xi), \mathbf{x}_c^3(\xi_2)$ denote the parameterizations of the edges.

From the equation above, we get the physical coordinates in terms of reference triangle coordinates ξ_1, ξ_2, ξ_3 and the parameterization map is defined now as
$x(\xi_1, \xi_2), y(\xi_1, \xi_2), z(\xi_1, \xi_2)$. As in the case of the *implicit* curve definition, the two systems of equations are differentiated with respect to the reference coordinates, to yield a corresponding linear system of equations for derivatives of x, y, z with respect to ξ_1, ξ_2.

- **Implicit spherical triangle by area coordinates (Type = 'ImSphTri')**

  Parameterization:

  The area coordinates for a point $\mathbf{x}'$ on the octant of a sphere are defined as $\lambda_i = \frac{P_i}{P}$ where P_i are the areas of the curvilinear triangles determined by geodesics passing through $\mathbf{x}'$, and P is the total area of the triangle (Fig. 14),

$$P = P_1 + P_2 + P_3 = \frac{\pi r^2}{2} \tag{4.29}$$

With φ and χ being the usual spherical coordinates, the formulas for P_1 and P_2 read as follows,

$$\begin{aligned} \varphi + \arcsin\Big(\frac{\sin\psi}{\sqrt{\sin^2\psi + \cos^2\varphi\cos^2\psi}} - \\ \arcsin\Big(\frac{\sin\varphi\sin\phi}{\sqrt{\sin^2\psi + \cos^2\varphi\cos^2\psi}} &= (\xi_1+\xi_2)\tfrac{\pi}{2} \\ \varphi + \arcsin\Big(\frac{\cos\varphi\sin\psi}{\sqrt{\sin^2\psi + \cos^2\varphi\cos^2\psi}} - \\ \arcsin\Big(\frac{\sin\phi}{\sqrt{\sin^2\psi + \cos^2\varphi\cos^2\psi}} &= \xi_2\tfrac{\pi}{2} \end{aligned} \tag{4.30}$$

By relating the area coordinates to the reference coordinates in the usual way,

$$\lambda_1 = 1 - \xi_1 - \xi_2, \quad \lambda_2 = \xi_1, \quad \lambda_3 = \xi_2$$

the system of equations is solved for φ and χ, which in turn determines coordinates $x_i', i = 1, 2, 3$ and,upon adding a possible translation and rotation, the physical coordinates x, y, z. As usual, by differentiating the equations, one obtains the corresponding system of equations for derivatives of x, y, z with respect to the reference coordinates.

4.5 Catalog of rectangles

The rectangles supported by the package are listed in Table 6. All rectangles share the same connectivity information:

EdgeNo(4) - curves' numbers that constitute edges of the rectangle with ± 1 sign factor indicating whether the global orientation of the curve is consistent with the local orientation of the corresponding edge of the rectangle.

BlockNo(2) - nicknames for the prisms and hexas that are adjacent to the rectangle.

Recall the definition of the nickname for a prism(hexahedron),

$$prism's(hexahedra's)number * 10 + norient \tag{4.31}$$

where $norient = 0, 1, \ldots 7$ indicates the global orientation of the triangle as seen from the adjacent prism (hexahedra).

Description of rectangles	Type	Attributes
Bilinear quadrilateral	*'BilQua'*	EdgeNo(4), BlockNo(2)
Transfinite interpolation rectangle	*'TraQua'*	EdgeNo(4), BlockNo(2)
Cylindrical rectangle	*'CylRec'*	EdgeNo(4), BlockNo(2)
Implicit rectangle	*'ImpRec'*	EdgeNo(4), BlockNo(2), Idata(5)

Table 6: The data in different types of rectangles

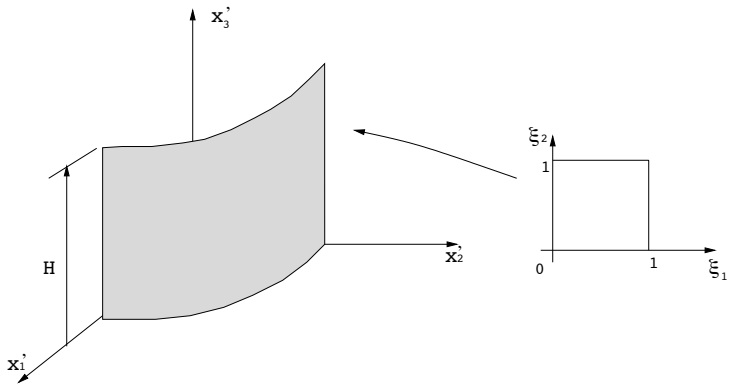

Figure 15: Parameterization of a cylindrical rectangle

- **Bilinear quadrilateral (Type = 'BilQua')**

  Note that all edges of the quadrilateral must be *segments of straight* line.

- **Transfinite interpolation with linear blending functions**

  quadrilateral (Type='TraQua')

  Formulas for the curves constituting edges of the quadrilateral are extended to the whole reference rectangle using the classical transfinite interpolation and linear blending functions [5] [6].

- **Cylindrical rectangle (Type = 'CylRec')**

  Note that two opposite edges of the quadrilateral must be *segments of a straight line* and the remaining two edges must be *quarters of circles*.

  Parameterization:

  The usual, cylindrical coordinates parameterization, see Fig. 15

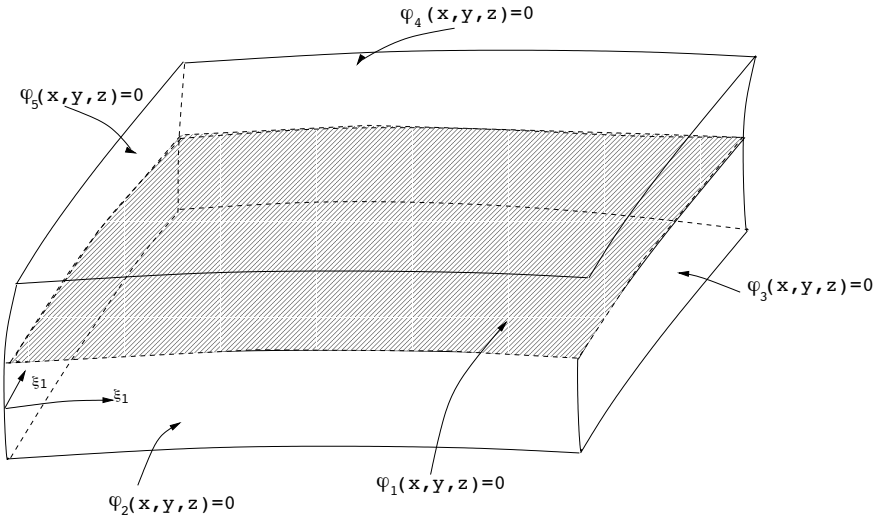

Figure 16: Implicit rectangle

$$\begin{aligned} x_1' &= r\cos(\frac{\pi}{2}\xi_1) \\ x_2' &= r\sin(\frac{\pi}{2}\xi_1) \\ x_3' &= H\xi_2 \end{aligned} \tag{4.32}$$

is superimposed with a rigid body motion.

- **Implicit rectangle (Type = 'ImpRec')** The rectangle lies on a given surface with its four edges cut off by four additional surfaces, see Fig. 16. Parameterization:

  Denoting the surface equations by $\varphi_i(\mathbf{x}) = 0, i = 1,\ldots,5$, we introduce the following *nonlinear* equations:

$$\begin{aligned} \varphi_1(\mathbf{x}) &= 0 \\ (1-\xi_2)(1-f_1(\xi_1))\varphi_5(\mathbf{x}) + f_1(\xi_1)\varphi_3(\mathbf{x}) & \\ \xi_2(1-f_3(\xi_1))\varphi_5(\mathbf{x}) + f_3(\xi_1)\varphi_3(\mathbf{x}) &= 0 \\ (1-\xi_1)(1-f_4(\xi_2))\varphi_2(\mathbf{x}) + f_4(\xi_2)\varphi_4(\mathbf{x}) & \\ +\xi_1(1-f_2(\xi_2))\varphi_2(\mathbf{x}) + f_2(\xi_2)\varphi_4(\mathbf{x}) &= 0 \end{aligned} \tag{4.33}$$

  where $f_i(\xi), i = 1,\ldots,4$ are the *stretching functions* determined by requesting the compatibility of the rectangle parameterization with the *existing*, specified parameterizations for its

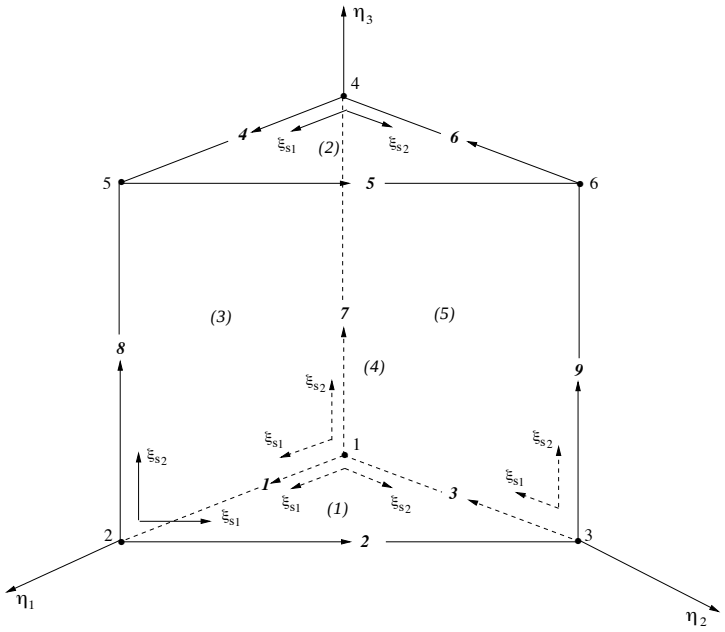

Figure 17: Reference prism

edges:

$$
\begin{aligned}
(1 - f_1(\xi_1))\varphi_5(\mathbf{x}_c^1(\xi_1)) + f_1(\xi_1)\varphi_3(\mathbf{x}_c^1) = 0 \\
(1 - f_2(\xi_1))\varphi_5(\mathbf{x}_c^3(\xi_1)) + f_2(\xi_1)\varphi_3(\mathbf{x}_c^3) = 0 \\
(1 - f_3(\xi_2))\varphi_2(\mathbf{x}_c^4(\xi_2)) + f_3(\xi_2)\varphi_4(\mathbf{x}_c^4) = 0 \\
(1 - f_4(\xi_2))\varphi_2(\mathbf{x}_c^2(\xi_2)) + f_4(\xi_2)\varphi_4(\mathbf{x}_c^2) = 0
\end{aligned}
\tag{4.34}
$$

with $\mathbf{x}_c^1(\xi_1), \mathbf{x}_c^2(\xi_2), \mathbf{x}_c^3(\xi_1), \mathbf{x}_c^4(\xi_2)$ being the parameterizations of the edges.

From the equation above, we can get the physical coordinates in terms of parameterization. The parameterization map is $\boldsymbol{x}(\xi_1, \xi_2)$. Derivatives $\frac{\partial x_i}{\partial \xi_j}$ are determined the same way as for *implicit curve* and *implicit triangle*.

4.6 Catalog of Prisms

All prisms supported by the package, see Fig. 17, share the same connectivity information:

Type - Type of the prism
FigNo(1:5) - Nicknames of figures that constitute faces of the prism,

the nickname of a figure is defined as,

$$figure's \quad number * 10 + norient \tag{4.35}$$

where $norient = 0, 1, \ldots 5(or 7)$ indicates the global orientation of the figure as seen from the adjacent prism.

- **Transfinite interpolation prism with linear blending functions**

  The parametrization of the prism is obtained by adding its vertex, edge and surface contributions. We introduce the following notation,

$$\boldsymbol{\phi}(\eta_1, \eta_2, \eta_3) = \sum_1^6 \boldsymbol{x}_v \psi_v + \sum_1^9 \tilde{\boldsymbol{\phi}}_e \psi_e + \sum_1^5 \tilde{\boldsymbol{\phi}}_s \psi_s \tag{4.36}$$

  where

  – $\boldsymbol{x}_v$ denote the global physical coordinates of vertex $\boldsymbol{v}$.

Vertex Number	ψ_v	Edge Number	ψ_e	Surface Number	ψ_s
(1)	$(1-\eta_1-\eta_2)(1-\eta_3)$	(1)	$\eta_1(1-\eta_1-\eta_2)(1-\eta_3)$	(1)	$1-\eta_3$
(2)	$\eta_1(1-\eta_3)$	(2)	$\eta_1\eta_2(1-\eta_3)$	(2)	η_3
(3)	$\eta_2(1-\eta_3)$	(3)	$\eta_2(1-\eta_1-\eta_2)(1-\eta_3)$	(3)	$\eta_1(1-\eta_1-\eta_2)$
(4)	$(1-\eta_1-\eta_2)\eta_3$	(4)	$\eta_1(1-\eta_1-\eta_2)\eta_3$	(4)	$\eta_1\eta_2$
(5)	$\eta_1\eta_3$	(5)	$\eta_1\eta_2\eta_3$	(5)	$\eta_2(1-\eta_1-\eta_2)$
(6)	$\eta_2\eta_3$	(6)	$\eta_2(1-\eta_1-\eta_2)\eta_3$		
		(7)	$1-\eta_1-\eta_2$		
		(8)	η_1		
		(9)	η_2		

Table 7: The Prism blending functions for vertex, edges and surfaces

  – ψ_v is the corresponding vertex bilinear blending function. It can be expressed as $\psi_v = \lambda_i \chi_j$, where

    * λ_i is the affine coordinate for the vertex with respect to the bottom or top triangle.
    * χ_j is the 1D shape function of η_3.

    The vertex blending functions are listed in Table 7.

  – $\tilde{\boldsymbol{\phi}}_e$ is the edge modified bubble function. For the three vertical edges, *i.e., edges 7,8,9*,

$$\tilde{\boldsymbol{\phi}}_e = \boldsymbol{\phi}_e(\xi_e) \tag{4.37}$$

for the edges on the bottom and top faces, *i.e., edges 1,2,3,4,5,6*,

$$\tilde{\boldsymbol{\phi}}_e = \frac{\boldsymbol{\phi}_e(\xi_e)}{\xi_e(1-\xi_e)} \tag{4.38}$$

where

* ξ_e is the local parameter for the edge, see the 3 vertical edges, *i.e., edges 7,8,9*,

$$\xi_e = \eta_3 \tag{4.39}$$

For the edges on the bottom and top surfaces, *i.e., edges 1,2,3,4,5,6*,

$$\xi_e = \frac{\lambda_{i+1} - \lambda_i + 1}{2} \tag{4.40}$$

where λ_{i+1} and λ_i are the affine coordinates for the two end points of the edge. The nine edge parameterizations are listed in Table 8

Edge Number	ξ_e	Surface Number	$\boldsymbol{\xi}_s$
(1)	$\frac{2\eta_1+\eta_2}{2}$	(1)	(η_1, η_2)
(2)	$\frac{\eta_2-\eta_1+1}{2}$	(2)	(η_1, η_2)
(3)	$\frac{2-\eta_1-2\eta_2}{2}$	(3)	(η_1, η_3)
(4)	$\frac{2\eta_1+\eta_2}{2}$	(4)	$(\frac{\eta_2-\eta_1+1}{2}, \eta_3)$
(5)	$\frac{\eta_2-\eta_1+1}{2}$	(5)	$(1-\eta_2, \eta_3)$
(6)	$\frac{2-\eta_1-2\eta_2}{2}$		
(7)	η_3		
(8)	η_3		
(9)	η_3		

Table 8: The parameterization for edges and surfaces

* $\boldsymbol{\phi}_e$ is the edge bubble function,

$$\boldsymbol{\phi}_e(\xi_e) = \hat{\boldsymbol{\phi}}_e(\xi_e) - \boldsymbol{x}_{v_i}(1-\xi_e) - \boldsymbol{x}_{v_j}\xi_e \tag{4.41}$$

where

  · $\hat{\boldsymbol{\phi}}_e(\xi_e)$ is the GMP curve parameterization adjusted for orientation.

  · $\boldsymbol{x}_{v_i}, \boldsymbol{x}_{v_j}$ are the coordinates of the two end points of the edge.

– ψ_e is the corresponding edge blending function. For the three vertical edges, *i.e., edges 7,8,9*,

$$\psi_e = \lambda_i \tag{4.42}$$

where λ_i is the affine coordinate for one of the end points of the edge. For the edges on the bottom and top faces, *i.e., edges 1,2,3,4,5,6*,

$$\psi_e = \lambda_i \lambda_{i+1} \chi_j \tag{4.43}$$

where

* λ_i and λ_{i+1} are the affine coordinates for the two end points of the edge.
* χ_j is the 1D shape function in terms of η_3.

The nine edge blending functions are listed in Table 7.

– $\tilde{\boldsymbol{\phi}}_s$ is the surface modified bubble function. For the top and bottom surface, *i.e., surfaces 1, 2*,

$$\tilde{\boldsymbol{\phi}}_s = \boldsymbol{\phi}_s(\boldsymbol{\xi}_s) \tag{4.44}$$

for the three vertical faces, *i.e., faces 3, 4, 5*,

$$\tilde{\boldsymbol{\phi}}_s = \frac{\boldsymbol{\phi}_s(\boldsymbol{\xi}_s)}{\xi_{s_1}(1-\xi_{s_1})} \tag{4.45}$$

where

* $\boldsymbol{\xi}_s$ is the local parameter for the face, $\boldsymbol{\xi}_s = (\xi_{s_1}, \xi_{s_2})$. The five faces parameterizations are listed in Table 8.
* $\boldsymbol{\phi}_s$ is the surface bubble function,

$$\boldsymbol{\phi}_s(\boldsymbol{\xi}_s) = \hat{\boldsymbol{\phi}}_s(\boldsymbol{\xi}_s) - \sum_{i=1}^{I} \boldsymbol{x}_{v_i} \psi_{v_i} - \sum_{i=1}^{I} \boldsymbol{\phi}_{e_i} \psi_{e_i} \tag{4.46}$$

  where $I = 3$ for a triangle, $I = 4$ for a rectangle.
  - $\hat{\boldsymbol{\phi}}_s(\boldsymbol{\xi}_s)$ is the surface parametrization for the face, provided by GMP, and adjusted for orientation.
  - $\boldsymbol{x}_{v_i}$ are the physical coordinates of the face vertices.
  - ψ_{v_i} are the corresponding vertex blending function, restricted to the face.
  - $\boldsymbol{\phi}_e$ are the bubble functions for each of the four edges on the face.
  - ψ_e is the corresponding edge blending function, restricted to the face.

– ψ_s is the corresponding face blending function. For the top and bottom face, *i.e., faces 1, 2*,

$$\psi_s = \chi_j \tag{4.47}$$

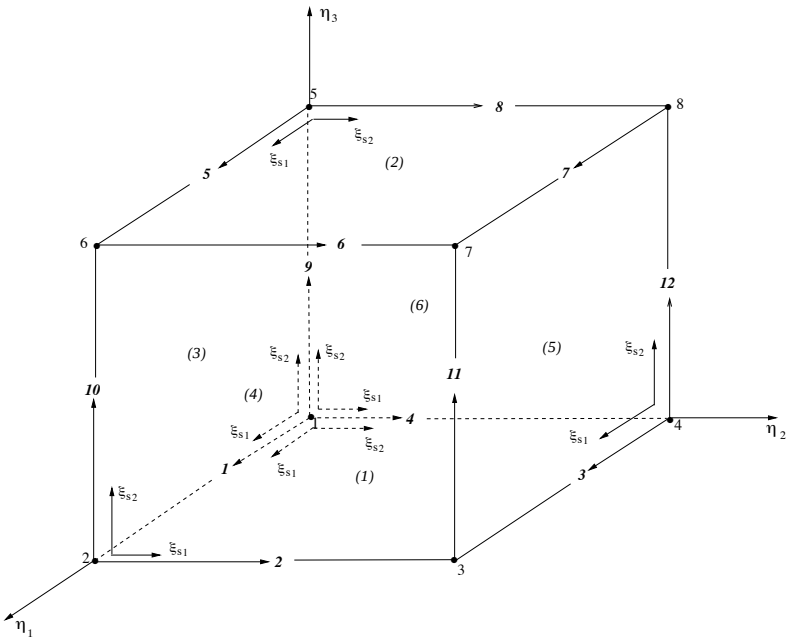

Figure 18: Reference hexahedron

where χ_j is 1D shape function in terms of η_3. for the three side vertical faces, *i.e., faces 3, 4, 5*,

$$\psi_s = \lambda_i \lambda_{i+1} \tag{4.48}$$

where λ_i and λ_{i+1} are the two affine coordinates for the end points of top or bottom surface edge.

The five blending functions for the surfaces are listed in Table 7

Formula (4.6.28) implies the corresponding formula for the derivatives.

4.7 Catalog of hexahedrons

All hexahedrons supported by the package, see Fig. 18, share the same connectivity information:

Type - Type of the Hexahedron.
FigNo(1:6) - Nicknames of the figures that constitute faces of the Hexahedron.

The nicknames for the adjacent figures are defined as,

$$figure's \quad number * 10 + norient \tag{4.49}$$

where $norient = 0, 1, \ldots 7$ indicates the global orientation of the figure as seen from the adjacent hexahedron.

- **Transfinite interpolation hexahedron with linear blending functions**

  The parametrization for the hexahedron is obtained by adding its vertex, edge and faces contributions. We introduce the following notation,

$$\boldsymbol{\phi}(\eta_1, \eta_2, \eta_3) = \sum_{1}^{8} \boldsymbol{x}_v \psi_v + \sum_{1}^{12} \boldsymbol{\phi}_e \psi_e + \sum_{1}^{6} \boldsymbol{\phi}_s \psi_s. \tag{4.50}$$

  – $\boldsymbol{x}_v$ denote the global physical coordinates of the vertex v.

  – ψ_v is the corresponding vertex blending function. It can be expressed as

$$\psi_v = \chi_i \chi_j \chi_k \tag{4.51}$$

  where χ_i, χ_j and χ_k are the 1D shape linear functions of η_1, η_2 or η_3.

  The eight vertex trilinear blending functions are listed in Table 9.

Vertex Number	ψ_v	Edge Number	ψ_e	Surface Number	ψ_s
(1)	$(1-\eta_1)(1-\eta_2)(1-\eta_3)$	(1)	$(1-\eta_2)(1-\eta_3)$	(1)	$1-\eta_3$
(2)	$\eta_1(1-\eta_2)(1-\eta_3)$	(2)	$\eta_1(1-\eta_3)$	(2)	η_3
(3)	$\eta_1\eta_2(1-\eta_3)$	(3)	$\eta_2(1-\eta_3)$	(3)	$1-\eta_2$
(4)	$(1-\eta_1)\eta_2(1-\eta_3)$	(4)	$(1-\eta_1)(1-\eta_3)$	(4)	η_1
(5)	$(1-\eta_1)(1-\eta_2)\eta_3$	(5)	$(1-\eta_2)\eta_3$	(5)	η_2
(6)	$\eta_1(1-\eta_2)\eta_3$	(6)	$\eta_1\eta_3$	(6)	$1-\eta_1$
(7)	$\eta_1\eta_2\eta_3$	(7)	$\eta_2\eta_3$		
(8)	$(1-\eta_1)\eta_2\eta_3$	(8)	$(1-\eta_1)\eta_3$		
		(9)	$(1-\eta_1)(1-\eta_2)$		
		(10)	$\eta_1(1-\eta_2)$		
		(11)	$\eta_1\eta_2$		
		(12)	$(1-\eta_1)\eta_2$		

Table 9: The hexahedron blending functions for vertex, edges and surfaces

  – $\boldsymbol{\phi}_e$ is the edge bubble function. It can be expressed as:

$$\boldsymbol{\phi}_e(\xi_e) = \hat{\boldsymbol{\phi}}_e(\xi_e) - \sum_{i=1}^{2} \boldsymbol{x}_{v_i} \psi_{v_i} \tag{4.52}$$

  where

* ξ_e is the local parameter for the edge, see Table 10.
* $\hat{\boldsymbol{\phi}}_e(\xi_e)$ is the GMP parametrization for the edge, adjusted for orientation,
* $\boldsymbol{x}_{v_i}$ are the physical coordinates of the edge end points, $i = 1, 2$ listed in order corresponding to the edge local parametrization,
* ψ_{v_i} are the corresponding vertex blending functions, restricted to the edge.

– ψ_e is the corresponding edge bilinear blending function, see Table 9.

Edge Number	ξ_e	Surface Number	$\boldsymbol{\xi}_s$
(1)	η_1	(1)	(η_1, η_2)
(2)	η_2	(2)	(η_1, η_2)
(3)	η_1	(3)	(η_1, η_3)
(4)	η_2	(4)	(η_2, η_3)
(5)	η_1	(5)	(η_1, η_3)
(6)	η_2	(6)	(η_2, η_3)
(7)	η_1		
(8)	η_2		
(9)	η_3		
(10)	η_3		
(11)	η_3		
(12)	η_3		

Table 10: The parameterization of the edges and surfaces

– $\boldsymbol{\phi}_s$ is the face bubble function. It can be expressed as:

$$\boldsymbol{\phi}_s(\boldsymbol{\xi}_s) = \hat{\boldsymbol{\phi}}_s(\boldsymbol{\xi}_s) - \sum_{i=1}^{4} \boldsymbol{x}_{v_i} \psi_{v_i} - \sum_{i=1}^{4} \boldsymbol{\phi}_{e_i} \psi_{e_i} \tag{4.53}$$

where

* $\boldsymbol{\xi}_s$ is the local parametrization for the face, $\boldsymbol{\xi}_s = (\xi_{s_1}, \xi_{s_2})$, see Table 10,
* $\hat{\boldsymbol{\phi}}_s(\boldsymbol{\xi}_s)$ is the GMP parameterization for the face, adjusted for orientation.
* $\boldsymbol{x}_{v_i}$ are the physical coordinates of the face vertices.
* ψ_{v_i} are the corresponding vertex blending functions, restricted to the face.
* $\boldsymbol{\phi}_{e_i}$ are the bubble functions for the four edges of the face,
* ψ_{e_i} are the corresponding edge blending functions, restricted to the face.

– ψ_s is the corresponding face linear blending function, see Table 9.

Last, we should adjust edge and face parameterizations for orientation. In the construction of the transfinite interpolations defined in this section, we refer to parameterizations of edges and faces in terms of *local coordinates*. The GMP parameterizations for edges and faces are provided in terms of *global coordinates*, and adujsting for the orientation of edges and faces involves transforming local edge and face coordinates into the global ones.

5 Concatenation

Now, we discuss how to concatenate two different geometrical objects; see Fig. 20 for the example of concatenation of two bricks.

When concatenating two different geometrical objects, we make the following assumptions:

1. The two objects to be concatenated have been described by two *separate* input files, *input1* and *input2*.

2. Besides the input files for the two objects, an extra input file *interface* is provided, which lists all the geometrical entities, *i.e.,* surfaces, points, curves, triangles and rectangles, from objects1 that lie on the interface.

3. The two objects are described in the *same* system of physical coordinates $x_i, i = 1, \ldots, N$.

4. All entities on the interface that belong to both objects must have the same *types* and *orientations*.

We describe now the concatenation algorithm.

5.1 Constructing the connectivities for the interface

Step1: We begin by inputing the geometrical data stored in files *input1* and *input2* and storing them in two different data structure moduli *GMP1.f* and *GMP2.f*. This is done in routines *input_geometry1.f* and *input_geometry2.f* which are exact copies of routine *input_geometry.f* except for the different data structure moduli.

Step2: We loop through all geometrical entities corresponding to the interface and referred to in terms of object1 data structure, and perform a global search through entities of *object2* to identity the corresponding entity numbers in object2. This is done in routine *input_inter.f*.

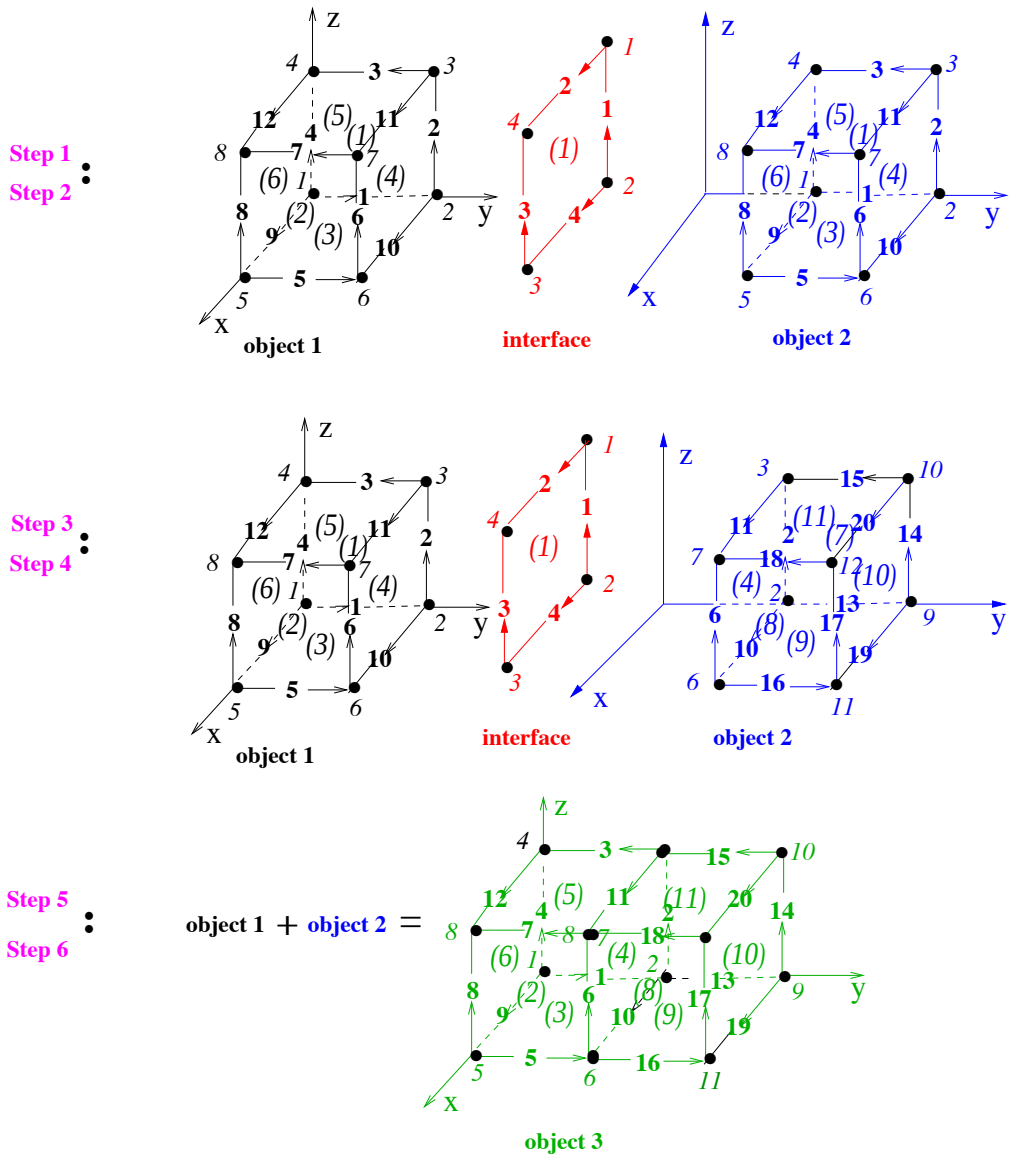

Figure 19: Concatenation of two bricks

Upon its execution, for each interface entity, we know the corresponding numbers in both data structures.

For example, from input file *interface*, see Fig. 19, we know that the 1st point on interface is the 3rd point in object1. The point's type is *regular* and it coordinates are (1, 1, 0). By looping through all the points of object2, we can locate the point with the same type and the same coordinates, *i.e.,* the 4th point in object2. Then we store the connectivity data for point 1 on the interface as INTPOINTS(1,1) = 3 and INPOINTS(2,1) = 4.

5.2 Changing the data structure for object2

Step3: Knowing all the connectivities for the interface, we can modify now the geometrical data structure in object2. This is done in routine *changeoj2.f* which loops through all the geometrical entities in object2 and changes their numbers according to the connectivity information. If an entity from object2 lies on the interface, we change its number to the corresponding number of matching entity in object1, otherwise we assign it a new number equal to the current total number of the entities plus one.

For example, see Fig. 19. we know the 4th point in object2 is on interface, and we change the number from 4 = INTPOINTS(2,1) to 3 = INTPOINTS(1,1). We also know the 6th point in object2 is not on interface. This point is the 5th point in object2 that is not on interface, and there are totally 8 points in object1. Therefore, the point number is changed from 3 to 11, *i.e.,* 8 + 3 = 11.

Along with the change of numbering, we change all the connectivity information in object2.

For example, the curves' numbers that meet at the point 11 in object2 are POINTS1(11)%CurvNo(1:3) $= (5, 6, 10)$. In order to concatenate, we change the point's connectivity to POINTS1(11)%CurvNo(1:3) $= (16, 17, 9)$, see Fig. 19.

Step4: We list the entities on the interface only once and update their connectivity information. For example, number of curves meeting at the point 7 on their interface is changed form 3 to 4, and the curve's numbers change from
POINTS1(7)%CurvNo(1:3) = (6,7,11) to POINTS1(7)%CurvNo(1:4) = (6,7,11,18), see Fig. 19.

5.3 Concatenating the objects

Step5: Now, after we have modified the data structure of object2 , we concatenate the two objects by looping through all the entities in the two objects separately, outputing the concatenated

data to file *input3*. This is done in routine *joint.f*.

Step6: We read in the geometrical data from file *input3* and store them into a single module *GMP.f*. This is done in routine *input_geometrical.f*. As a result, we get the concatenated object shown in Fig. 19.

6 Examples

We present now a few typical examples corresponding to various research projects at TICAM.

6.1 Example1. Implicit Rectangle

Figure 20 shows an implicit rectangle made up of 5 surfaces. The input files is as following:

INPUT DATA	REMARK
3 2	dimension of the problem and manifold
5	number of surfaces
Cylinder	type of the surface 1
0.0 0.0 0.0	coordinates of the base
0.0 1.0 0.0	coordinate of the parallel vector
8	radius
VecPt	type of surface 2
0.0 0.0 0.0	coordinate of the point crossed by plane
0.0 0.0 -1.0	coordinate of the normal vector
VecPt	type of surface 3
0.0 0.0 0.0	coordinate of the point crossed by plane
0.0 1.0 0.0	coordinate of the normal vector
Cylinder	type of surface 4
0.0 0.0 0.0	coordinates of the base
0.0 0.0 1.0	coordinate of the parallel vector
4	radius
VecPt	type of surface 5

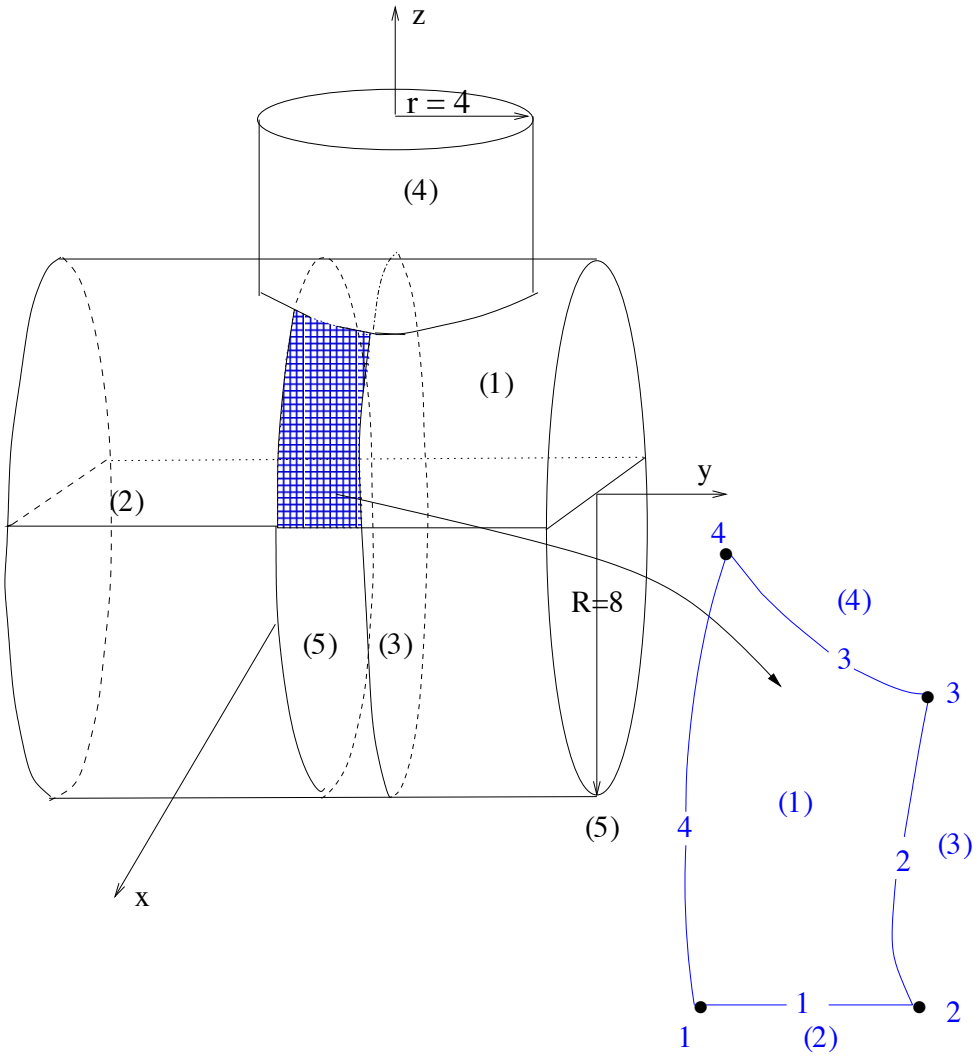

Figure 20: Implicit Rectangle

0.0 -2.0 0.0	coordinate of the point crossed by plane
0.0 1.0 0.0	coordinate of the normal vector
4	number of points
Implicit	type of point 1
2	number of curves
4 1	curves' number
1 2 5	surfaces' number
8.0 -2.0 0.0	coordinate of the point
Implicit	type of point 2
2	number of curves
1 2	curves' number
1 2 3	surfaces' number
8.0 0.0 0.0	coordinate of the point
Implicit	type of point 3
2	number of curves
2 3	curves' number
1 3 4	surfaces' number
8.0 0.0 6.92	coordinate of the point
Implicit	type of point 4
2	number of curves
3 4	curves' number
1 4 5	surfaces' number
3.464 -2.0 7.746	coordinate of the point
4	number of curves
ImpCir	type curve 1
1 2	endpoints numbers
1	number of figure
12	figure's number
1 2 5 3	surfaces' number

ImpCir	type of curve 2
2 3	endpoints numbers
1	number of figure
12	figure's number
1 3 2 4	surfaces' number
ImpCir	type of curve 3
3 4	endpoints numbers
1	number of figure
12	figure's number
1 4 3 5	surfaces' number
ImpCir	type of curve 4
4 1	endpoints numbers
1	number of figure
12	figure's number
1 5 4 2	surfaces' number
0	number of triangles
1	number of rectangles
ImpRec	type of rectangle 1
1 2 3 4	sides' numbers
1 2 3 4 5	surfaces' numbers

6.2 Example2. Fichera's corner

6.3 Eample3. Mock0 model with a tower

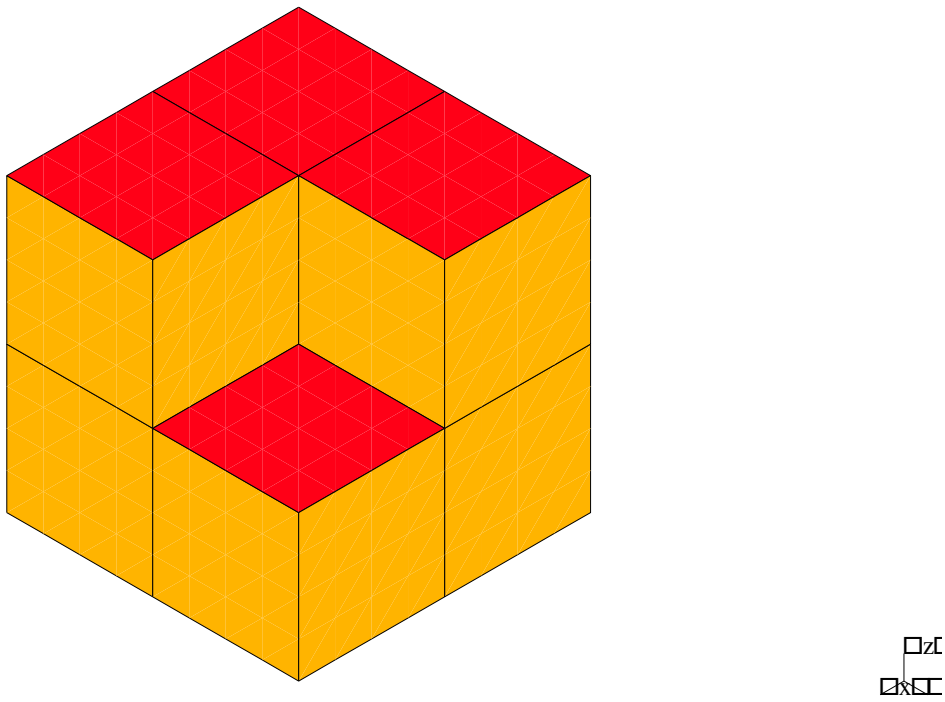

Figure 21: Fichera's corner

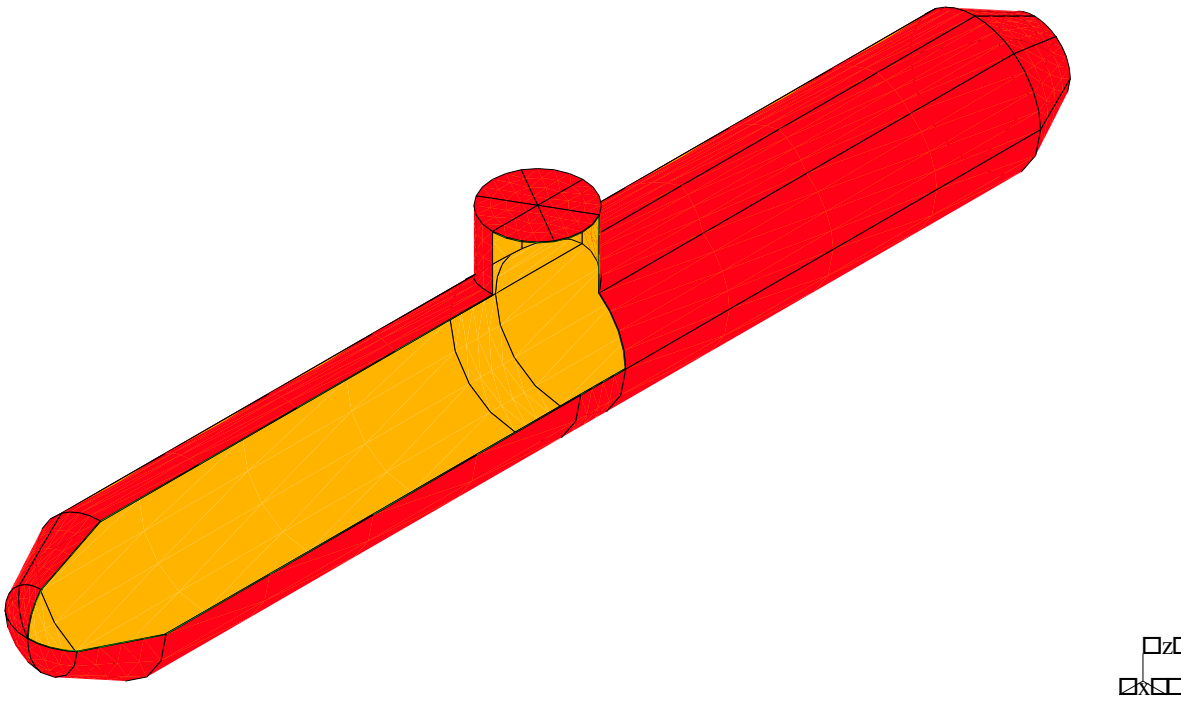

Figure 22: Mock0 model with a tower

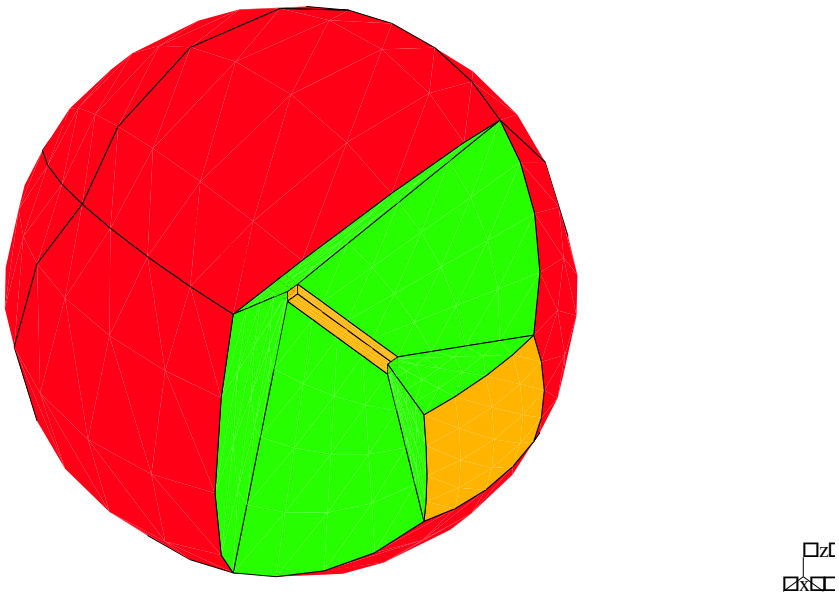

Figure 23: Dipole antenna

6.4 Example4. A dipole antenna enclosed in a truncating sphere

6.5 Example5. Cylindrical shell with spherical incaps

7 Conclusions

The purpose of this work is to review and update the technology of the Geometrical Modeling Package. In this work, we have added a number of important features to the GMP. Firstly, we have completely rewritten the code, introducing a Fortran 90-like data structure, so that the logic of the code is easier to follow. Secondly, we have added an option for the concatenation of two separate geometrically compatible objects. This helps us to deal with more complicated manifolds. Last but not least, we have corrected the inconsistencies in the original GMP related to the orientation of faces, and have added the transfinite parameterizations for prisms and hexahedrons.

The source code, along with input files for the presented examples can be download from *http://www.ticam.utexas.edu/ cynthia/paper/project.html*

The new GMP will replace soon the original package in our 2D and 3D hp codes [2] [4]

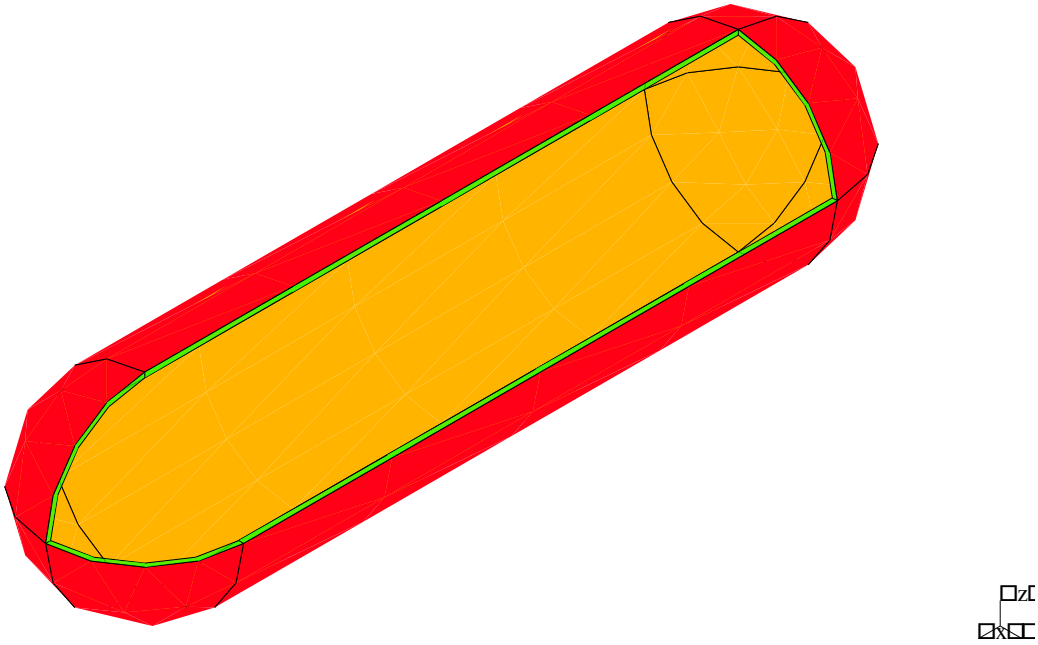

Figure 24: Cylindrical shell with spherical incaps

References

[1] L. Demkowicz, A. Bajer, and K. Banaś, "Geometrical Modeling Package", *TICOM Report 92-06*, 1992.

[2] L. Demkowicz, "2D hp-Adaptive Finite Element Package (2Dhp90). Version 2.0", *TICAM Report 02-06*, 2002.

[3] H. Schwetlick, *Numerische Losung nichtlinearer Gleichungen*, Deutscher Verlag der Wissenschaften, Berlin 1979.

[4] L. Demkowicz, D. Pardo, W. Rachowicz, "3D hp-Adaptive Finite Element Package (3Dhp90). Version 2.0, The Ultimate Data Structure for Three Dimensional, Anisotropic hp Refinements", *TICAM Report 02-24*, 2002.

[5] W. J. Gordon, and C. A. Hall, "Transfinite Element Methods: Blending Function Interpolation over Arbitary Curved Element Domain", *Numer. Math.* . No. 21, pp. 109 - 129, 1973.

[6] W. J. Gordon, "Blending Function Methods of Bivariate and Multivariate Interpolation and Approximation", *SIAM J. Numer. Anal.* . No. 8, pp. 158-177, 1971.