An interactive geometry utility environment for multi-disciplinary computational engineering

Yao Zheng*,†, Nigel P. Weatherill‡ and Edward A. Turner-Smith

Department of Civil Engineering, University of Wales Swansea, Swansea SA2 8PP, U.K.

SUMMARY
A parallel simulation user environment (PSUE) has been developed for unstructured grid-based computational simulation. Arbitrary computer application software can be integrated into the environment to provide a multi-disciplinary engineering analysis capability within one unified computational framework. It provides an enhanced capability for complex and multiple problem definition, a graphics environment for guidance through the grid generation process with visual validation of each step, and robust and computationally efficient unstructured grid generation modules. This paper addresses an interactive geometry utility environment (IGUE), which is the primary part of the PSUE, providing sophisticated graphical user interfaces with geometric handling capability oriented to the unstructured grid technology. The IGUE is equipped with windowing functionality from the X-Window system, and its underpinning data structure is based on non-manifold topology. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: geometry modelling; non-manifold topology; grid generation; graphical user interface; scientific visualization

1. INTRODUCTION
Over the past decade, significant progress has been made in the development of algorithms for the construction of unstructured grids of triangles and tetrahedra. Such advances have progressed in parallel with algorithms for the problem solving in multi-disciplinary computational engineering such as computational fluid dynamics. Traditionally, the generation of grids has been an intricate process, in which data files are manually constructed and manipulated, and control files edited. In practice, the time-consuming aspects of the generation process are exacerbated since several grids are usually generated prior to the analysis cycle. Often geometries are defined with a combination of formats, including analytically and numerically defined components and, in some cases, the individual components may not fit together smoothly,
resulting in either overlaps or discontinuities. The user is often faced with the requirement to become intimately familiar with the geometry, being required to have detailed knowledge of co-ordinate values at key parts of the configuration. Although the algorithms for the generation of grids have advanced and matured, the process of building grids is still time consuming and far from automatic.

Advances in computer hardware, particularly in the area of computer graphics workstations, have introduced the possibility of reducing or eliminating some of the more tedious aspects of the user intensive tasks. Computer systems driven by hierarchical push button menu options, referred to as graphical user interfaces (GUI), can hide most of the actions of editing and file manipulation. However, the real advantage of the graphics workstation is the ability to visualize data, and in turn, to visually monitor the steps performed in the grid generation process. An immediate implication of this ability to visualize the consequences of user actions is that these systems are easy to use, and can reduce the level of expertise required from the user.

The parallel simulation user environment (PSUE) [1, 2] has been developed for such pre- and post-processing for unstructured grid-based computational simulation. The attractive aspect of the unstructured grid technology is the ease and speed with which complex geometrical configurations can be treated [3–5]. Arbitrary computer application software can be integrated into the environment to provide a multi-disciplinary engineering analysis capability within one unified computational framework. The PSUE harnesses state-of-the-art high performance computing and networking to provide a software environment for a diverse range of applications in computational field simulation. It is aimed to facilitate the optimization of design and to help reduce the number of production cycles in manufacturing industry.

The PSUE provides an enhanced capability for complex and multiple problem definition. Developments aim at integrating CAD and grid generating capabilities, addressing end-user requirements and data formats. This will include the provision of tools to correct invalid geometries, a graphics environment for guidance through the grid generation process with visual validation of each step, and robust and computationally efficient unstructured grid generation modules. The environment also provides the functionality for arbitrary engineering software applications to be integrated and analysis modules executed on computer platforms, including parallel computers.

At the present stage of development, the central functionality of the PSUE is grid generation, and the geometry manipulation capability is primary to the PSUE. The essential capability of the PSUE is furnished through its sub-environment—the interactive geometry utility environment (IGUE), which essentially consists of a geometry builder and a grid generation interface. The IGUE, as an enabling system, has been ergonomically designed, which leads the user through the generation procedure. Effort has been made to provide primary functionality and geometry utility, as well as the rich capability including feature visualization and grid control functionality.

2. OVERVIEW OF THE GEOMETRY UTILITY ENVIRONMENT

2.1. Parent environment and software architecture

A major design philosophy behind the PSUE is that modules are either ‘owned’ by the PSUE or ‘owned’ by the user. Modules owned by the PSUE are termed generic modules and are
closely coupled, through common data structures, for example, into the environment. Modules owned by the user are integrated into the PSUE through pre-defined format data channels, which take the form of files, pipe or socket transfer. This framework allows a user to capitalize on the generic functionality of the PSUE whilst maintaining the independence of utilizing their own modules.

During the development of the PSUE, the efficient performance of the system on large unstructured grids was of paramount importance. In this context, a large grid is of the order of 5 million tetrahedral elements. Hence, particular attention has been given to memory management, efficient algorithms and an effective use of high performance computing and networking. The functionality of the PSUE includes a geometry builder and geometry repair module, grid generation and statistics, numerical libraries, domain decomposition, parallel platforms, performance monitoring, database and data analysis. A general schematic of these modules is shown in Figure 1.

Whilst working within the PSUE, a user is faced with the possibility of being involved with many tasks at a given time. For example, whilst waiting for the execution of an application, the user may wish to progress the generation of another grid. Hence, at any given time, multiple processes can be running concurrently in the PSUE. The design of the PSUE, therefore, includes a memory management tool which allows the user complete control of data migration to modules in the PSUE. Plate 1 is a snapshot of the main session of the PSUE during a run time, in which the main PSUE manages the memory and controls individual sub-processes. Five geometry builders are opened with three iconized, while two grid generators and one grid analyser are invoked, all iconized. All these modules in various states are shown in the process control panel.

The memory manager provided allows the automatic transfer of data between processes so that users do not have to explicitly ‘move data’ between modules. Several different methods...
can be used to move data around the PSUE. The implementation allows for multiple modules of the same type to be open at one time, each which contains a different geometrical configuration, that can update each other or a different type of module at any given time. The PSUE ‘daemon’ is the centre of the memory manager. It connects to each of the modules and communicates using ‘sockets’ that transmit messages back and forth. The daemon uses a ‘shared memory’ mechanism to manage the memory. When data needs to be moved from one module to another, the daemon manages the memory, allowing fast transfer of the data between the modules.

2.2. The sub-environment—IGUE

The generation and analysis of unstructured grids is the result of modules coupled into the PSUE. The flowchart illustrated in Figure 2 shows the role of visual steering in this process. The visual steering of the grid generation and grid analysis allows the user control and guidance of each stage of the process [6]. The user can modify parameters or manipulate the geometry within the graphical environment and verify the result of their action using the visual steering tools.

As a primary part of the PSUE, the IGUE provides tools to manipulate the geometry definition, perform topological validation of the geometry, interact and repair geometry defects through operations such as surface reconstruction. Each of these operations prepares the geometry for grid generation. Further tools provide functionality to control the resulting grid point concentration by means of a background grid and grid sources. The visual steering allows the user to rapidly establish the required grid control by locking, for example, point sources to the geometrical feature which are identified as requiring additional grid point resolution. This reduces set-up time by alleviating the user of the necessity of an intimate knowledge of the geometry. The results of the grid generated by this technique can be qualitatively inspected within the IGUE, and a quantitative analysis of grid quality can be conducted and displayed within the associated grid statistics interface. This interface shares much functionality with the view mode of the geometry builder. The benefit of visual steering can be reaped from the iterative adjustment of grid control parameters, where changes can be visually explored after regeneration of the grid.
Plate 1. A selection of windows from the PSUE driving a typical user session.
Plate 2. Discovery of surface gaps on an aircraft engine intake.
Plate 3. Surface and volume grid of the domain of an aircraft wing with a repaired surface on the engine intake.
Plate 4. Surface reconstruction for a broken trefoil.
The IGUE enables the user to build simple geometries by means of creating points, curves, surfaces and so on. The corresponding data format is based on CAD input and utilizes IGES geometry files, although specialized data formats are also supported. The essential use of this capability is to build simple shapes, such as, for example, outer boundaries to close domains, and to provide the functionality to construct background grids and grid source data which are used to control the grid generation process.

3. GEOMETRY DESCRIPTIONS

3.1. Non-manifold topology

Prior to the discussion of non-manifold topology, some fundamental concepts are outlined as follows [7]. In \( n \)-dimensional Euclidean space \( E_n \), an open spherical neighbourhood of any point, denoted as \( p \), is defined as \( N(p, r) = \{ x | d(p, x) < r \} \), where \( d(p, x) \) is the Euclidean distance between points \( p \) and \( x \), and \( r \) is the radius of the neighbourhood concerned. All these kinds of neighbourhoods are the open subsets of \( E_n \), which serve as a basis of the topology. A mapping \( h \) from set \( A \) to \( B \), denoted as \( h: A \rightarrow B \), is a relationship associating points of \( A \) and points of \( B \). If a mapping is continuous and one to one, then it is a homeomorphism. Two point sets are named as homeomorphic and considered topologically equivalent if there exists a homeomorphic mapping between them. Points set \( A \) is called \( n \)-manifold, if for each point of \( A \), there exists a neighbourhood homeomorphic to an open neighbourhood in \( E_n \). The \( n \) is the inherent dimensionality of set \( A \). It is obvious that all open sets of \( E_n \) are \( n \)-manifolds.

In the field of geometric modelling, there is an increasing demand for considering generic geometric sets, which are decomposable into a number of components of the same or different dimensions. These components are not necessarily manifolds and non-manifolds are involved in many cases. An important kind of non-manifold is \( n \)-manifold with boundary, namely, the closure of a manifold, consisting of an \( n \)-dimensional interior and its lower-dimensional boundary. In a general geometry environment, such as the present environment under consideration, multi-dimensional, and non-manifold components are considered. Effort has been made by various groups of researchers into the investigation of data structures for non-manifold geometry and its associated modelling system [8–10]. Reference [10] presents the applications of this kind of geometry structure in a grid generation environment.

3.2. Data structures

In the IGUE, both general geometric entities and grid generation specific entities are included. Effort has been made to couple operations for both types, such that extended utilization of this environment for grid generation becomes a reality. The basic data types supporting data structures for general geometries are \texttt{coordT}, \texttt{vectT}, \texttt{GBCoords}, \texttt{GBVect}, \texttt{GBColor}, \texttt{GBBkgVal} and \texttt{GBSVal}.

The geometries consist of different layers of geometry entities, which are represented by vertices, points, curves, loops, surfaces and domains. Points, curves, loops, surfaces, domains, background grid and grid sources are stored as linear linked lists, so that these entities are able to be edited dynamically, such as deleting and appending an element of the lists. The
corresponding data structures for these entities are GBPoint, GBCurve, GBLoop, GBFace, GBDomain, GBBkgPoint, GBBkgElement, GBPSor, GBLSor and GBTSor, respectively. Vertices are stored in contiguous memory space in the form of the structure GBVertex, for the sake of performance in terms of speed. For efficiency of graphical rendering, there is an option to display an outer layer of a volume grid only. In order to find the inner part of the grid, edge information, and information of links between points and last edges are employed.

3.3. Geometric models for grid generation

The grid generators connected within this environment are of general purpose. They are able to generate surface and volume grids as well as planar isotropic, anisotropic and hybrid grids. The geometries for two-dimensional grid generation consist of curves, for surface gridding they are made up of surfaces and loops, and volume grids are based on their corresponding surface grids. In the present paper, three-dimensional examples are mainly emphasized. A surface model is necessary for surface gridding, and volume gridding requires the surface grid to define a close domain. Therefore, the main activity associated with the geometry preparation is to define a valid surface model.

Nowadays, real-life CAD models are represented in NURBS format [11], owing to the excellent mathematical and algorithmic properties, and the successful industrial applications of NURBS. However, many grid generators usually deal with geometries in forms of lower order, such as bi-cubic surfaces, due to historical reasons of the computational simulation community. The current version of grid generators integrated in the PSUE treat geometries represented by bi-cubic surfaces and cubic splines, although they could be extended straightforwardly to deal with NURBS entities. Currently, within the IGUE, all the surfaces are treated as bi-cubic surfaces, and the curves are defined as cubic splines. Though the conversion from NURBS to cubic splines is non-trivial, and not always possible due to the facts such as tolerancing issues, we describe surfaces models and address the associated geometry operations in a form of cubic spline without loss of generality.

Usually, a surface model does not consist of complete surfaces, there are intersections between surfaces, and ruled surfaces. Therefore, the geometry definition for gridding includes support surfaces, and loops with the appropriate specification of regions to be gridded. Figure 3 illustrates a support surface, loops, and the corresponding grid.
3.4. Underlying geometry data

Basically, a support surface can be specified by given co-ordinates of certain points on the surface, and a curve also can be given by specifying associated points. A loop consists of a group of curves, and, further, it is attached to a surface. This attachment is referred to as topology relation, which should be established prior to surface grid generation [12]. This information can be imported together with the basic surface and curve definition from external geometry modellers, in addition, it can be automatically generated within this environment or manually by utilizing a topology relation panel. For the automatic specification, an algorithm of surface topology abstraction is involved.

The present environment has been developed with an orientation towards grid generation for multi-disciplinary computational engineering, rather than as a generic geometric modelling tool. Surface description can be built externally and then imported into this environment. An example of data exchange format employed is IGES.

Apart from imported geometry data, geometric models can be built within this environment. A simple example of this type of utilization is building an outer boundary for a domain. However, the equipped functionality enables us to modify and validate the models, to enhance local features, and even to specify sophisticated controls to grid generation in various ways. Some special functionality and associated algorithms are addressed in the next section.

4. GEOMETRY OPERATIONS

4.1. Surface definition

An arbitrary surface is usually defined using a finite number of points. Within this geometry utility environment, the underlying surface definition is given by means of Coons patches, for geometry operations such as surface reconstruction, surface–surface intersection, and even the associated surface grid generation. The Coons patch technique uses the given data to construct a numerical model of the surface so that any point on the surface may be obtained in terms of two parameters. The Coons patch uses continuity of slopes of the surface as well as the point data. Hence this formulation results in a smooth representation of a three-dimensional surface.

The Coons patch technique is a method of surface representation which requires a surface mesh of quadrilaterals and for each patch the co-ordinates of the surrounding points are used to produce an equivalent parametric element in a bi-cubic form. Using the parametric patch, the problem is reduced from three dimensions to two dimensions, and the Cartesian co-ordinates anywhere on the surface may be calculated using the parametric patches.

The bi-cubic patch of parametric co-ordinates \((\xi, \eta)\) is given by

\[
X(\xi, \eta) = F(\eta)AF(\xi)^T \quad (0 < \xi < 1 \text{ and } 0 < \eta < 1)
\]

where \(X(\xi, \eta) = [x(\xi, \eta), y(\xi, \eta), z(\xi, \eta)]^T\), and \(F(u) = (F_1(u), F_2(u), F_3(u), F_4(u))\). Each component of \(F(u)\) has cubic properties, which ensures continuity and smoothness throughout the
Figure 4. Notations of parametric co-ordinate system \((\xi, \eta)\) and \((u, v)\) for a surface consisting of patches.

The surface. Actually, \(F(u)\) takes the form of Hermite blending functions, that is

\[
\begin{align*}
F_1(u) &= (1 + 2u)(u - 1)^2 \\
F_2(u) &= u^2(3 - 2u) \\
F_3(u) &= u(1 - u)^2 \\
F_4(u) &= u^2(u - 1)
\end{align*}
\]

(2)

The \(4 \times 4\) matrix \(A\) containing co-ordinates, tangents and twists on a patch can be expressed as

\[
A = \begin{bmatrix}
X(0,0) & X(0,1) & X_\eta(0,0) & X_\eta(0,1) \\
X(1,0) & X(1,1) & X_\eta(1,0) & X_\eta(1,1) \\
X_\xi(0,0) & X_\xi(0,1) & X_{\xi\eta}(0,0) & X_{\xi\eta}(0,1) \\
X_\xi(1,0) & X_\xi(1,1) & X_{\xi\eta}(1,0) & X_{\xi\eta}(1,1)
\end{bmatrix}
\]

(3)

where \(X = (x, y, z)^T\), \(X_\xi = \partial X / \partial \xi\), \(X_\eta = \partial X / \partial \eta\), and \(X_{\xi\eta} = \partial^2 X / \partial \xi \partial \eta\), and the co-ordinates \((\xi, \eta)\) take values of 1 or 0, depending upon the parametric position. Such a patch is often referred to as a tensor-product surface.

This \((\xi, \eta)\) co-ordinate system is for one patch only with \(0 < \xi < 1\) and \(0 < \eta < 1\). An overall parameter co-ordinate system can be introduced for a surface as composite patches, and is denoted as system \((u, v)\). Without loss of generality, it is convenient to assume that \(u\) and \(v\) take integral values at nodal points for the overall surface. Referring to Figure 4, the components of \(A\) can be rewritten in terms of the parametric co-ordinates as

\[
X_\xi(u_k,l, v_k,l) = f_{k,l} \frac{\partial X}{\partial \xi_{u_k,l}} = f_{k,l} X_u(u_k,l, v_k,l)
\]

(4)
\[
X_\eta(u_{k,l}, v_{k,l}) = g_{k,l}X_\eta(u_{k,l}, v_{k,l})
\]

\[
X_\xi(u_{k,l}, v_{k,l}) = f_{k,l}g_{k,l} \frac{\partial^2 X}{\partial \xi_k \partial \eta_l} = f_{k,l}g_{k,l}X_{uv}(u_{k,l}, v_{k,l})
\]

where \( u_{k,l} \) and \( v_{k,l} \) are parametric co-ordinates, and \( f_{k,l} \) and \( g_{k,l} \) are the curvilinear arc lengths along the \( u \) and \( v \) directions, respectively.

Using notations given in Figure 4, Equation (1) can be written as

\[
X(u, v) = F(\eta)BF(\xi)^T
\]

with the \( B \) matrix consisting of

\[
B = \begin{bmatrix}
X^{k,l} & X^{k+1,l} & f_{k,l}X^{k,l}_u & f_{k+1,l}X^{k+1,l}_u \\
X^{k,l+1} & X^{k+1,l+1} & f_{k,l+1}X^{k,l+1}_u & f_{k+1,l+1}X^{k+1,l+1}_u \\
g_{k,l}X^{k,l}_v & g_{k,l+1}X^{k+1,l}_v & (fg)_{k,l}X^{k,l}_{uv} & (fg)_{k+1,l}X^{k+1,l}_{uv} \\
g_{k,l+1}X^{k+1,l}_v & g_{k+1,l+1}X^{k+1,l+1}_v & (fg)_{k+1,l+1}X^{k+1,l+1}_{uv} & (fg)_{k+1,l+1}X^{k+1,l+1}_{uv}
\end{bmatrix}
\]

where \( X_u = \partial X/\partial u \), \( X_v = \partial X/\partial v \), \( X_{uv} = \partial^2 X/\partial u \partial v \), and the \( k \) and \( l \) subscripts indicate the position on the Cartesian grid.

### 4.2. Surface reconstruction

Apart from the geometry repair module, there are various functions available in the geometry builder which enable the user to manually modify the geometry. For a three-dimensional case, the main task is surface reconstruction. Plate 2 depicts a gap between the surfaces of an aircraft engine intake. This fault can be corrected using the geometry builder. Plate 3 shows surface and volume grid of the domain of an aircraft wing with a repaired surface on the engine intake.

For a missing section between two surfaces, surface reconstruction algorithms provide functionality to close the gap. It is required that both surfaces at either side of the gap should have the same number of divisions along the ‘open’ edges. Methods based on ‘bi-cubic patches’ or ‘Bezier curves’ may be chosen to fill the gap. There is also a choice between making an ‘additional surface’ (ending in three surfaces) or creating a ‘unified surface’ (ending up with one surface) [13]. Plate 4 illustrates a schematic of the surface reconstruction. In this plate, one example shows how the gap is filled with an additional surface, and the second and third examples show the approach which utilises a unified surface (coarse or fine descriptions).

#### 4.2.1. Additional surface to bridge a gap

Assume there is a gap between two surfaces, and the numbers of points on the two opposite edges are the equal to each other, then the Coons patch technique is used to build a connection surface to bridge the gap, in which new nodal points are located based on interpolation. Basically, the new surface can just connect two points on opposite edges by a straight line. In this case, the slopes at the two points are evaluated using backward and forward schemes. Introducing other points from the bottom and top edges in evaluating the slopes can improve the geometrical features of the connection surface in the sense of forcing the surface’s edges to have better continuity and smoothness. Different finite difference schemes result in a variation of performance in terms of curvatures.
It has been found that the best result can be made from such an approach using the backward scheme on the bottom edge and the forward scheme on the top edge.

4.2.2. Unified surface for the two surfaces and the gap. It has been noted that for the surface reconstruction it is not enough to reconstruct a surface to simply cover the gap, but a smooth and regular gridded surface is more desirable in certain circumstances. Therefore, it is valuable to include a feature which redistributes points over the two surfaces and the gap resulting in a unified surface. The procedure is based upon redistributing the boundary points using an arc length method, and then using a two-dimensional grid generator to establish a parametric representation of the surface. Having the original surface parametric grids underlining the redistributed surface, any points on the new redistributed surface can be defined, and located on the geometry surface. Therefore, the resulting unified surface has a high order of smoothness. There are two techniques employed for the two-dimensional gridding task in the redistribution procedure. One belongs to the family of algebraic grid generation systems, and the other uses the elliptic grid generation system.

**Algebraic grid generation:** The physical co-ordinates of the interior points within the rectangular transformed region can be determined directly by means of interpolation from the specified boundaries. Such an approach is referred to as algebraic grid generation. In the two-dimensional parametric domain, a linear uni-directional Lagrange interpolation function can be expressed in one of the curvilinear co-ordinate directions as

\[
R_a(\xi, \eta) = (1 - \xi)R(0, \eta) + \xi R(1, \eta)
\]  

which means that the interpolation \( R_a \) interpolates between the boundaries \( \eta = \text{const} \). Similarly, \( R_b(\xi, \eta) \) for the boundary \( \xi = \text{const} \) can be established. Since the boundary points should be preserved whatever scheme is used, a third interpolant \( R_c \) has been introduced to maintain the boundary values, which can be expressed as

\[
R_c = (1 - \xi)(1 - \eta)R(0, 0) + \xi(1 - \eta)R(1, 0) + (1 - \xi)\eta R(0, 1) + \xi \eta R(1, 1)
\]

Moreover, the required interpolant is \( R = R_a + R_b - R_c \), which can be given as

\[
R(\xi, \eta) = [(1 - \xi) \xi \begin{bmatrix} R(0, \eta) \\ R(1, \eta) \end{bmatrix} + [R(\xi, 0) \begin{bmatrix} R(0, 1) \\ R(1, 1) \end{bmatrix}]] \begin{bmatrix} (1 - \eta) \\ \eta \end{bmatrix}
\]

\[
-[(1 - \xi) \xi \begin{bmatrix} R(0, 0) \\ R(1, 0) \end{bmatrix} + [R(\xi, 1) \begin{bmatrix} R(0, 1) \\ R(1, 1) \end{bmatrix}]] \begin{bmatrix} (1 - \eta) \\ \eta \end{bmatrix}
\]  

(11)

This type of interpolation is referred to as transfinite interpolation. This idea can be extended to include boundary slope data, but that was not used in the present work as the elliptic grid generation system gives more reasonable results.

**Elliptic grid generation:** It is well known that the three categories of partial differential equations, that is, elliptic, parabolic and hyperbolic, can be used to generate a structured grid. However, the elliptic equations have natural smoothing effects which inhibits jumps or discontinuities. This makes these equations ideally suitable for grid generation. By using
elliptical equations data can be smoothed from the boundary into the interior of the domain, also the point distribution can be effectively controlled by using control functions. The interior points are relocated by means of an iterative procedure based on the boundary data.

Consider the two-dimensional physical space defined by parametric variables \( u \) and \( v \) associated with the surface definition. For the sake of convenience, \( x \) and \( y \) are used instead of \( u \) and \( v \) in this section. Denote \( \xi \) and \( \eta \) as regular co-ordinates of the transformed region related to the grid on the surface. In this two-dimensional space, Laplace’s equations can be written as

\[
\begin{align*}
\xi_{xx} + \xi_{yy} &= 0 \\
\eta_{xx} + \eta_{yy} &= 0
\end{align*}
\] (12)

The chain rule leads to a system of coupled non-linear elliptic partial equations in terms of \( x \) and \( y \) as follows:

\[
\alpha r_{\xi\xi} - 2\beta r_{\eta\eta} + \gamma r_{\eta\eta} = 0
\] (13)

where \( r = (x, y)^T \), and the coefficients are

\[
\begin{align*}
\alpha &= x_\eta^2 + y_\eta^2 \\
\beta &= x_\xi x_\eta + y_\xi y_\eta \\
\gamma &= x_\xi^2 + y_\xi^2
\end{align*}
\] (14)

To solve the system of equations (13), finite difference approximations are used, which, with \( h = x_i - x_{i-1} = 1 \), leads to

\[
\begin{align*}
\alpha_{i,j} &= \frac{1}{4}[((x_{i,j+1} - x_{i,j-1})^2 + (y_{i,j+1} - y_{i,j-1})^2] \\
\beta_{i,j} &= \frac{1}{4}[[(x_{i+1,j} - x_{i-1,j})(x_{i,j+1} - x_{i,j-1}) + (y_{i+1,j} - y_{i-1,j})(y_{i,j+1} - y_{i,j-1})] \\
\gamma_{i,j} &= \frac{1}{4}[(x_{i+1,j} - x_{i-1,j})^2 + (y_{i+1,j} - y_{i-1,j})^2]
\end{align*}
\] (15)

The corresponding residual can be represented as

\[
\begin{align*}
R_{i,j}^n &= \alpha_{i,j}(r_{\xi,i,j} - 2r_{\xi,i,j} + r_{\xi,i-1,j}) - \frac{1}{2}\beta_{i,j}(r_{\xi,i+1,j+1} - r_{\xi,i-1,j+1} - r_{\xi,i+1,j-1} + r_{\xi,i-1,j-1}) \\
&+ \gamma_{i,j}(r_{\xi,j+1} - 2r_{\xi,j} + r_{\xi,j-1})
\end{align*}
\] (16)

where \( r_{\xi,j}^n \) represents the unknown \( x \) and \( y \) at point \((i,j)\) at the iteration level \( n \). The solution procedure for Equation (13) is a linear successive point over-relaxation scheme

\[
r_{\xi,j}^{n+1} = r_{\xi,j}^n + \frac{\omega R_{i,j}^n}{2(\alpha + \gamma)}
\] (17)

where \( \omega \) is a relaxation parameter defined in the interval \([1, 2]\).

To ensure adequate control of the grid point spacing Equation (12) is augmented with source terms \( P \) and \( Q \). And the resulting transformed equations take the form

\[
\alpha(r_{\xi\xi} + Pr_{\xi\xi}) - 2\beta r_{\eta\eta} + \gamma(r_{\eta\eta} + Qr_{\eta\eta}) = 0
\] (18)
Noticing $\mathbf{r} = (x, y)^T$, $Q$ can be eliminated from Equation (18), and leads to

$$z[y_n(x_{\xi} + P_{\xi}) - x_n(y_{\eta} + P_{\eta})] = y_n^2 \left[ 2\beta \left( \frac{x_n}{y_n} \right)_{\xi} + \gamma \left( \frac{x_n}{y_n} \right)_{\eta} \right]$$

(19)

By assuming zero curvature on boundary $\eta = \text{const.}$, that is,

$$\left( \begin{array}{c} x_n \\ y_n \end{array} \right)_{\eta} = 0$$

(20)

and imposing the condition of orthogonality

$$x_{\xi}x_{\eta} + y_{\xi}y_{\eta} = 0$$

(21)

Equation (19) yields to

$$P = -\frac{\mathbf{r}_{\xi} \cdot \mathbf{r}_{\xi\eta}}{|\mathbf{r}_{\xi}|^2}$$

(22)

Similarly, we have

$$Q = -\frac{\mathbf{r}_{\eta} \cdot \mathbf{r}_{\eta\xi}}{|\mathbf{r}_{\eta}|^2}$$

(23)

Once values of $P$ and $Q$ have been obtained for each boundary point, their values at an interior grid point can be determined by linear interpolation along lines of constant $\eta$ and $\xi$. Then the corresponding $u$ and $v$ (namely, $x$ and $y$ here) can be obtained. This will further relocate the grid points on the surface on the base of the surface representation previously given by transfinite interpolation. This procedure ensures that the grid throughout the interior of domain reflects the point distribution on the boundaries.

4.2.3. Examples. If a geometry is created or imported, and it is found to have a section missing, a designed surface reconstruction panel allows the user to close the gap. The surfaces at either side of the gap can be selected. Presently, there is a restriction that both surfaces should have the same number of divisions along the ‘open’ edges. ‘Bi-cubic patches’ or ‘Bezier curves’ may be selected to specify an algorithm to fill the gap. There is a choice between making an ‘additional surface’ (ending in three surfaces) or creating a ‘unified surface’ (ending with one surface), which must be made prior to the surface reconstruction operation.

Plate 4 shows methods to reconstruct the trefoil with one section missing. An additional surface, and unified surfaces with coarse and fine descriptions are used to bridge the missing section as shown in Plate 4. The unified surfaces are constructed by means of elliptic gridding method. Plates 5 and 6 deal with a broken teapot with one section removed, where an additional surface is used to repair the teapot. There are two pairs of surfaces, either of which can be chosen as the basis for the surface reconstruction. It is observed that the choice of approach really makes a difference in the ability to capture surface curvature around the gap. As an additional feature, a gluing operation has been conducted for the examples shown in Plates 5 and 6.
Plate 5. Surface reconstruction for a broken teapot (Scheme 1).
Plate 6. Surface reconstruction for a broken teapot (Scheme 2).
Plate 7. Surface definition, intersection curves and surface grid of a teapot.
Plate 8. The effect of background grid settings on a two-dimensional grid.

4.3. Surface–surface intersection

The surface-to-surface intersection procedure is applicable for two arbitrary surfaces. Two parametric Coons patches are defined, one for each surface. Hence to calculate a point of intersection between the two surfaces, the problem has been reduced from one involving six degrees of freedom to one of only four. Since there are only three equations which define the problem, some way of knowing a fourth degree of freedom is required. Hence a further constraint must be given. The three equations may then be solved for the remaining degrees of freedom and a full solution obtained [14].

Assume the two parametric surfaces are \( R_1(s, t) \) and \( R_2(u, v) \), respectively. Referring to Equation (7), the two surfaces can then be expressed as

\[
R_1(s, t) = F(\sigma)B_1F(\tau)^T \\
R_2(u, v) = F(\eta)B_2F(\xi)^T
\]

(24)

From these equations, since \( R_1 \) and \( R_2 \) both are vectors containing \( x, y \) and \( z \), it is known that at any intersection point \( x_1 = x_2, \, y_1 = y_2 \) and \( z_1 = z_2 \). Therefore,

\[
F(\sigma)B_1F(\tau)^T - F(\eta)B_2F(\xi)^T = 0
\]

(25)

In these three equations, \( B_1 \) and \( B_2 \) are set up using \( x \) data, \( y \) data and \( z \) data, respectively. Because these are only three equations, and \( s, t, u, \) and \( v \) are four degrees of freedom, one value must be set to some specific value, say \( s_0 \), prior to solving the equations. Actually, the degree of freedom to be fixed can be chosen by means of trial and error. Since the length of a parametric element is unity, if the interval in parametric terms is set to unity, then the intersection points will be evaluated everywhere a grid line, of the chosen fixed parametric co-ordinate, crosses the second parametric surface. If the interval is less, more points will be found and vice versa.

Equation (25) is to be solved by means of the Newton–Raphson method, which requires initial estimates of the parametric co-ordinates to be evaluated. The center of the respective surfaces are used as a starting point for the first intersection point to be found. Then the previous intersection point gives the initial estimates for the calculation of the subsequent intersection points.

The approach based on the Newton–Raphson method consists of taking the initial estimates and calculating the derivatives of the Cartesian co-ordinates with respect to the parametric co-ordinates. On assembly into a matrix, these collectively become known as the tangent matrix. The construction of the tangent matrix is as follows. Since the residual is

\[
R = R_1(s_0, t) - R_2(u, v)
\]

(26)

where

\[
R_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \quad \text{and} \quad R_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}
\]
then there exists
\[
\begin{bmatrix}
\frac{dx_1}{dt} & \frac{dy_1}{dt} & \frac{dz_1}{dt} \\
\frac{dx_2}{du} & \frac{dy_2}{du} & \frac{dz_2}{du} \\
\frac{dx_2}{dv} & \frac{dy_2}{dv} & \frac{dz_2}{dv}
\end{bmatrix}
\begin{bmatrix}
\Delta t \\
\Delta u \\
\Delta v
\end{bmatrix} = 0
\]
(27)

Because \( K\Delta = -R \) and \( \Delta = [\Delta t, \Delta u, \Delta v]^T \), the tangent matrix can be expressed as
\[
K = \frac{\partial R}{\partial U} = \begin{bmatrix}
\frac{\partial R_1}{\partial t} & \frac{\partial R_1}{\partial u} & \frac{\partial R_1}{\partial v} \\
\frac{\partial R_2}{\partial t} & \frac{\partial R_2}{\partial u} & \frac{\partial R_2}{\partial v}
\end{bmatrix}
\]
(28)

or
\[
K = \begin{bmatrix}
\frac{\partial x_1}{\partial t} & \frac{\partial x_1}{\partial u} & \frac{\partial x_2}{\partial u} & \frac{\partial x_2}{\partial v} \\
\frac{\partial y_1}{\partial t} & \frac{\partial y_1}{\partial u} & \frac{\partial y_2}{\partial u} & \frac{\partial y_2}{\partial v} \\
\frac{\partial z_1}{\partial t} & \frac{\partial z_1}{\partial u} & \frac{\partial z_2}{\partial u} & \frac{\partial z_2}{\partial v}
\end{bmatrix}
\]
(29)

where
\[
R = \begin{bmatrix}
x_1 - x_2 \\
y_1 - y_2 \\
z_1 - z_2
\end{bmatrix}
\]

Now that a set of solvable equations is obtained, by using Gaussian elimination the increments of each of the parametric co-ordinates can be evaluated, and the guess values are updated accordingly. On iteration, the increments get smaller together with the residual. Once this residual has reached a desirable tolerance the intersection point is assumed to have been found. The process then starts again for the next intersection point where the fixed parametric co-ordinate is incremented, and the previous intersection point is used as the initial estimates for the solution. This process continues until all desired intersection points have been located.

Plate 7 depicts the surface definition, intersection curves and surface grid of a teapot. The intersection curves between the pot body and the handle, the pot body and the nozzle, have been established by means of the intersection procedures mentioned above. This is the baseline of the subsequent surface grid generation, as the intersection curves are required to model the teapot.

4.4. Topology abstraction

As mentioned above, a region on a support surface to be gridded is described in terms of a loop. Therefore, a loop consisting of curves should be attached to a particular support surface. This process is referred to as topology abstraction. A loop generator is provided to automatize loop construction and topology abstraction (assignment of relation between loops and surfaces) via a topology relation panel [12]. However, it is a challenging task to implement such a generator which works for arbitrary cases, as many factors affect the robustness of the
generator. Therefore, manual specification is necessary in some cases. Manual specification is also provided via the topology relation panel. During this specification, loops should be constructed first, then the relationship between loops and surfaces established. Figure 5 is a snapshot of the topology relation panel.

Often a CAD model comes with ambiguous information of edges. For instance, there are duplicated curves, curves partially overlapped, curves of a common location and different degree of discretization, intersecting curves without common supporting nodes, and curves which could not form a close loop to define a region (see Figure 6). For such cases, the loop generator highlights them graphically to prompt user involvement.

4.5. Geometry validation and repair

The time taken to prepare geometries for grid generation is presently causing major concerns within industrial groups. The raw CAD data is often not valid in context of topological and geometrical structures, due to such features as surface overlaps and discontinuities. Such a geometry is often referred to as a ‘dirty’ or an ‘invalid’ geometry. Prior to grid generation, it is necessary to examine the validity of the geometry. If the geometry is not valid, a repair phase must be implemented.

A geometry repair module is necessary to correct faults in the geometry definition, such as surface overlaps and discontinuities. It creates a valid geometry configuration suitable for later use in grid generation and analysis. Geometry repair is rather a complicated task and is yet to be systematically researched [15, 16]. Due to the availability of various forms of geometry definitions, individual research groups are considering their own particular geometry representation.

The parent environment, the PSUE, has a capability to detect topological and geometrical errors such as edges lying out of faces or gaps between two faces. A list of inconsistencies
is given to the user. Using repairing functionality the user is able to interactively modify the model in order to repair it prior to grid generation. A semi-automatic procedure is provided to correct geometries, although user interaction is required for ambiguous cases. There is an option to fill in holes with a new piece of surface, and to automatically zip up small holes and cracks. This facility is required to identify the dislocation of nodes in adjacent surfaces and to perform the necessary alignment. The geometry validation tool is required to provide the user with information showing the problems identified, and to visualize errors on the target as colour coded patches. Moreover, an option is provided to enable the user to interactively choose features which need repair.

5. GRID GENERATION ORIENTED FUNCTIONALITY

5.1. Background grid

Grids with varying point spacing are required for general field simulations. Using the grid generators incorporated in the PSUE, the grid spacing can be controlled by means of using a background grid and/or grid sources [4, 5]. During the process of grid generation, points are continually inserted into the domain and elements subsequently created, until the element sizes...
fall within the prescribed grid spacing. Plate 8 shows the effect of background grid settings on the two-dimensional grid. If the user does not specify relevant values, then a default setting will be used automatically.

A background grid which covers the entire domain consists of elements with spacing values assigned to individual nodes. The spacing values at locations other than the element nodes are interpolated from the values on the nodes. In the IGUE, a background grid can be represented as shaded elements with ellipses (ellipsoids) on grid points indicating spacing values. The colour shading is used to show the spacing variations, meanwhile the ellipses (ellipsoids) exhibit the differences between spacing values in $X$, $Y$ and $Z$ directions, which are useful for anisotropic grid generation. The user is able to create a particular background grid by modifying the default grid, or by direct construction. Background grid elements can be picked and the spacing values on the connected points can be reassigned. Also picked elements can be deleted and, of course, new elements can be created and new spacings assigned.

5.2. Grid sources

Apart from using a background grid, grid sources are usually used to control grid spacing. The IGUE provides functionality to create point, line and triangular sources, through the grid source panel, in conjunction with the editing tools. All types of sources can be created, deleted or modified at any stage.

A point source is assigned with two radii and a spacing value. Within the first radius elements will be generated which have a length scale that is similar to the spacing value specified. The element length scale then exponentially increases to a global spacing value outside the first radius. The incremental rate of expansion of point spacing is defined by the relationship shown in Figure 7. This expression prescribes the spacing at the outer radius to be twice that at the inner radius. Figure 7 illustrates the influence of a point source on the
point spacing variation. A colouring scheme allows the intensities to be displayed indicating the intensity variations of various sources. The procedure is similar for each type of source, i.e. point, line and triangular sources, with the capability of being able to specify different intensities and radii at each end of a line source or each corner of a triangular source.

A line source has a cylindrical range of influence, which is displayed as a thick line and two ends. A triangular source has a flattened triangular range of influence, which is represented as a shaded triangle and three corners. For a two-dimensional case, a point source is represented as two circles centered at a single point, while the ends of a line source and the corners of a triangular source are represented in a similar manner. However, in three-dimensional space, two spheres are used instead of two circles, and the spheres can be visualized in wireframe or solid modes. Translucent spheres can be used to represent the solid spheres, with a feature to show the geometry and the source setting in a clear way, such that the user can easily determine the ranges of influence sources on the grid to be generated.

Figure 8 shows the effect of spacing and influence sphere parameters of point and line sources. Plate 9 shows the effect of grid source settings on a resulting surface grid. Through the translucent spheres, the grid concentration can be observed, where the difference of grid spacing between the two radii are clearly exhibited. Plate 10 shows these ideas applied to a real industrial model, which is an F22 aircraft. In this example, the influence of grid sources results in a surface grid with appropriate grid concentration, which is suitable for a simulation of aerodynamic flow.
5.3. Grid generation

The PSUE features an unstructured grid generation capability, of which the attractive aspect is the ease and speed with which complex geometrical configurations can be treated [3–5]. The successful applications of this grid generation technology motivates the exploitation of the PSUE as a whole.

The grid generators which have been integrated within the PSUE are able to create unstructured planar, surface and volume grids. They are embedded within a common framework, that is, the grid generation interface, under the IGUE. The grid generation interface is initialized from the main PSUE or from the geometry builder. Under the grid generation interface, the data preparation facility can invoke the geometry builder for geometry importing or building, and background grid and source definition setup. It also allows for the geometry repair module to be initialized.

The grid generation interface allows the user to set-up grid types, parameters, and file manipulation. The parameters used in grid generation differ according to the different grid types to be created and they control the grid generation process. Within the interface, there are two computer platform options for the user to choose: local machine and remote machine. For the remote machine option, a remote access tool is invoked. Additionally, a parallel machine can be set-up by initiating parallel tools from the computing platform module. In this case, parallel grid generation can be executed with monitoring on a local host. The geometry/topology information region of the grid generation interface is able to give the number of each geometry/grid entity for the object concerned, and it is also capable of sending the present data to the PSUE daemon for memory management reasons.

6. COMPUTATIONAL APPLICATIONS

6.1. Metal forming

Modelling of industrial forming processes for manufacturing components is required in aerospace, automotive and related industrials. Ring rolling is a versatile metal forming process for manufacturing seamless annular forgings. Figure 9 depicts schematics of a radial-axial rolling mill with guide rolls [17]. Figure 10 shows a top view of tool surfaces and the finite element grid for the ring rolling. Plate 11 gives a visual representation within the IGUE of the geometry, surface and volume grids of the ring and tools during rolling, where the grids are cut to provide an internal view.

6.2. Aerodynamics

The use of the PSUE as a whole is demonstrated by use of an example from computational fluid dynamics. The example results in the simulation of compressible flow around the Dassault Falcon executive plane which has been performed using software developed for parallel architecture computers. The flow diagram of the use of the PSUE for this example is shown in Figure 11.

The geometry definition of the Dassault Falcon aircraft is imported into the IGUE. The geometry definition does not include the symmetry plane and farfield boundaries necessary for generating a grid suitable for flow simulation. Therefore, the geometry templates available
in the IGUE are used to create the appropriate farfield boundary, as either a hemisphere or half-cylinder, which is adjacent to the symmetry plane created for the geometry. This process is rapid and interactively provides immediate feedback on each stage of geometry creation. The completed geometry is passed through a surface topology extraction algorithm which uses the information on curves and surfaces to check the geometry definition for completeness.
Plate 10. Visual representation of the geometry, surface grid and grid sources for an F22 aircraft.
Plate 11. Visual representation of the geometry, surface and volume grid of ring and tools during rolling.
Plate 12. Visual representation of the geometry, surface and volume grid (with cutting plane) for a Dassault Falcon executive plane.
Plate 13. Partitioning the volume grid of the computational field for the Falcon aircraft in preparation for a parallel flow simulation.
Plate 14. Domain partition for Falcon aircraft with the flow simulation showing contours of pressure.
Figure 11. Schematic of the flow chart for a aerodynamics simulation.

An incomplete, invalid, geometry requires a geometry repair phase which flags topological inconsistencies and directs the user as to which repair processes are required for correction. The resulting valid geometry must then have grid control parameters associated with it, which control global and local grid point spacing through modifications to the default background grid and the creation of point, line and triangular grid sources. The visual interaction between grid sources and the geometry allow an intuitive control of grid spacing. The grid generation module is invoked after suitable grid control has been implemented. The grid generation module creates surface and volume grids, it also allows adjustments of certain grid parameters to be made to achieve a grid with the desired grid regularity. The volume grid created can be further improved by utilizing a grid cosmetics procedure which identifies less desirable tetrahedral and nodal connections and improves them by edge swapping or local grid regeneration techniques.

Plate 12 illustrates the Falcon aircraft within the IGUE with a surface grid and a part of the volume grid of the near computational field. The visualization of the volume grid is shown in the form of the triangles on the surface of the volume elements cut by a user specified cutting cylinder. The sophisticated visualization functionality, such as, the various cutting operations, enables the user to investigate the resultant grid in considerable detail. The seamless integration of the geometry builder and grid generation interface into the PSUE makes it possible to monitor the gridding process step by step.

The specification of boundary conditions is facilitated by the boundary condition editor, which is embedded into the geometry builder. The boundary conditions are specified at the geometry definition level, which are then mapped to the surface grid for subsequent use in the flow solution algorithm.
The flow simulation has been carried out using a parallel CFD code. Therefore, the whole of the volume grid has to be partitioned in a domain decomposition phase. Plate 13 shows the grid partition, where several inter-domain boundaries are presented, with colour coding indicating normal directions of the boundary elements. A CFD solution has been obtained, as shown in Plate 14, which shows the contours of pressure on the surface of the aeroplane. For a more accurate simulation, a grid adaptation procedure can be introduced, in which the grid concentration is controlled by the error estimate of the solution obtained.

7. CONCLUSIONS

The PSUE provides an enhanced capability for complex and multiple problem definition for unstructured grid-based computational simulation. It features visual steering of grid generation, which includes the provision of tools to correct invalid geometries, a graphics environment for guidance through the grid generation process with visual validation of each step, and robust and computationally efficient unstructured grid generation modules.

The essential geometric capability of the PSUE is provided through its sub-environment—IGUE. The IGUE is equipped with windowing functionality from the X-Window system (Motif) [18, 19], its underpinning data structure is based on non-manifold topology, and the 3D graphics rendering library is OpenGL [20, 21]. The non-manifold topology and the associated data structure provide a baseline of the IGUE. The IGUE features capability of various geometry operations including surface reconstruction, surface-to-surface intersection and topology abstraction. During the development of this environment, software engineering aspects have been considered in order to develop and maintain the code in a reasonable manageable manner.

ACKNOWLEDGEMENTS

The authors are grateful to the European Union for the financial support through the ESPRIT CAESAR project. The authors would like to thank the other members of the PSUE group, especially Dr M. J. Marchant, for their contributions. The authors also thank R. Said for his effort on the algorithm for surface reconstruction, and Dr O. Hassan for discussions and comments on the GUI aspects of the IGUE. The development of the IGUE has benefited from collaboration with the other key partners involved in the CAESAR project, in particular IPK (Berlin, Germany), Sowerby Research Centre, British Aerospace (U.K.), DASA (Germany), Odense (Denmark) and BASF (Germany). Their contribution is also gratefully acknowledged.

REFERENCES
