THE DEVELOPMENT OF A TOOL FOR SEMI-AUTOMATED GENERATION OF STRUCTURED AND UNSTRUCTURED GRIDS ABOUT ISOLATED ROTORCRAFT BLADES*

By Ramakrishnan Shanmugasundaram, Javier A. Garriz and Jamshid A. Samareh,
Computer Sciences Corporation
Hampton, VA

ABSTRACT

The grid generation used to model rotorcraft configurations for Computational Fluid Dynamics (CFD) analysis is highly complicated and time consuming. The highly complex geometry and irregular shapes encountered in entire rotorcraft configurations are typically modeled using overset grids. Another promising approach is to utilize unstructured grid methods. With either approach the majority of time is spent manually setting up the topology. For less complicated geometries such as isolated rotor blades, less time is obviously required. This paper discusses the capabilities of a tool called Rotor blade Optimized Topology Organizer and Renderer(ROTOR) being developed to quickly generate block structured grids and unstructured tetrahedral grids about isolated blades. The key algorithm uses individual airfoil sections to construct a Non-Uniform Rational B-Spline(NURBS) surface representation of the rotor blade. This continuous surface definition can be queried to define the block topology used in constructing a structured mesh around the rotor blade. Alternatively, the surface definition can be used to define the surface patches and grid cell spacing requirements for generating unstructured surface and volume grids. Presently, the primary output for ROTOR is block structured grids using O-H and H-H topologies suitable for full-potential solvers. This paper will discuss the present capabilities of the tool and highlight future work.

INTRODUCTION

The high degree of geometric complexity typically encountered in full rotorcraft configurations is such that even the generation of grids for inviscid flow simulations requires a large amount of user intervention. Overset grids have emerged as the method of choice for modeling entire configurations [1], driven in equal parts by the simplification of generating grids for individual rotorcraft components and advances in flow solver algorithms. The flow solver advances include all the pre-processing that overset methods require, for example, determining the interpolation stencils. Another approach that holds promise for rotorcraft and indeed any complex shape is utilizing unstructured grid techniques. Figure 1, for instance, depicts the triangular surface mesh on a simplified AH-64 Apache helicopter fuselage. Recent improvements to the basic "advancing front" algorithm used to produce the mesh shown in Figure 1 have resulted in the capability to generate viscous unstructured meshes, with or without anisotropic stretching to keep the total number of cells reasonable [2]. Such capability has already been demonstrated with favorable results, although mainly in fixed-wing applications to this point.

Regardless of whether a structured or unstructured approach is chosen to generate the grid(s), users are still required to manually set up the blocking/topology for multi-block structured grids or surface patching for unstructured grids. Far less intervention is obviously required when modeling an isolated rotor blade or propeller, as is done when performing computational studies of rotorcraft airfoils and blades[4]. This paper discusses the capabilities of a tool being developed with the intention of allowing the user to quickly generate block structured and unstructured tetrahedral grids about isolated blades.

At the heart of the tool being developed is an algorithm which uses the individual airfoil sections defining the blade and global blade characteristics such as twist, and sweep as input and constructs a Non-Uniform Rational B-Spline (NURBS) surface representation of the blade. This "geometry definition" can then be queried to obtain the mesh points describing the now continuous surface, and a block structured mesh constructed around it. Alternatively, the geometry definition can be queried to output surface patches and grid cell spacing specifications for use in generating unstructured surface and volume meshes. The common surface representation can just

*presented at Technical Specialists' Meeting for Rotorcraft Acoustics and Aerodynamics
Williamsburg, Virginia; October 28-30, 1997

as easily be used to write out input for linear (panel) codes for preliminary design/quick "check-out" of the blade aerodynamics. Presently, the tool is being written such that it's primary output will be block structured grids using O-H and H-H topologies suitable for use in full-potential solvers such as FPR and FPX[3]. As such, an approximate wake shape model will be implemented depending on whether the grid is designed for calculations in helicopter or propeller modes. Outputs for other codes and/or grid generators will follow as time permits. A schematic of the simple process described above is shown in Figure 2. Figure 3 shows the input blade airfoil sections for a sample case, while Figure 4 shows the NURBS surface representation built from the input sections.

The capability to generate grids with little or no user intervention could, of course, also be exploited to advantage in a design-and-optimization environment, where changes in design variables need to be reflected in the grid to be used by the analysis code(s).

When using typical grid generators, the grid topology is defined interactively around the geometry. The user defines the topology and generates the volume grid. Often, a significant portion of time is spent refining the grid to resolve regions of interest. If the blade is redefined, the topology must be modified and the complete volume grid regenerated. Similarly, if the volume grid is refined, the sequence of steps used to define the original volume grid must be repeated. Codes exist that exploit the fact that when modeling the flow field surrounding isolated rotor blades, the grid topology is normally simple and fixed. Codes such as GRGEN take as input the airfoil section, blade characteristics, and grid generation parameters and automatically generate a single block structured volume grid to be used in full-potential flow codes. Oftentimes, these grid generators lack adequate control of the block interior spacing to resolve regions of interest. It would be useful to modify the volume grid locally. However, since the grid topology is not available to the user, it is very difficult to make the necessary changes. The ROTOR tool allows the user to build the topology as a sequence of commands. The resulting topology is then passed to a volume grid generator. The main design philosophy behind the ROTOR tool is use of pre-existing tools and reusability.

The paper will demonstrate the present capabilities of the tool from input sections to volume grid on sample blade geometries. Future work and possible applications will also be discussed.

METHODOLOGY

The ROTOR tool has three integral components. The first component converts the input blade characteristics to a NURBS surface representation. The surface definition may be used to generate a block structured volume grid or an unstructured volume grid. The second component organizes the topology. The tool uses a grid command language to define the topology. A command set has been implemented to define an O-H structured block topology surrounding an isolated rotor blade defined by a single NURBS surface. The third component is the volume grid generator. The ROTOR tool writes out a command set to be used by the Coordinate and Sensitivity Calculator for Multi-Disciplinary Design Optimization(CSCMDO) [5] to generate the volume grid. CSCMDO is specifically designed to modify existing volume grids within a design optimization loop. Each component is a separate code bound together by a UNIX script. Eventually, the first and second components are expected to be joined so that maintenance of two separate codes will no longer be necessary.

Rotor Definition

The input to the tool is the rotor definition and the grid generation parameters. The rotor is defined as a series of airfoil sections at given span stations. Each section is defined by a shape and various properties. The various properties include chord, thickness ratio, twist, sweep, and center of rotation. A global rotor parameter of dihedral at the root may be specified. After applying each of the properties to each airfoil section, a set of three dimensional curves are created. The NURBS surface is created by interpolating each point on the curve and lofting a surface between the curves. The resultant surface quality is highly dependent upon the point distribution of each curve as well as the spanwise distance between span stations. At present a separate utility based on the routines found in the GridTool[6] code actually performs the creation of the NURBS

surface from discrete curve data. This utility writes out the surface definition in IGES(Initial Graphics Exchange Standard) format. The NURBS surface can now be used to generate an unstructured or structured grid.

Block Structured Volume

The ROTOR tool is primarily designed to generate a block structured grid by defining the topology using an ASCII input deck. The ROTOR tool command set defines geometric and topological entities in order to define the topology for a block structured grid. Commands exist to define surfaces, points, curves, blocks, edges and patches. The commands are defined in a hierarchy as shown in Figure 5 by first defining a configuration, followed by the geometry, and finally the topology. The command hierarchy is necessary so that dependent entities may be defined at their instantiation instead of waiting for the definition of the parent entity. For example, if a point is evaluated on a specified surface, the surface must be defined before the definition of the point. The ROTOR tool was designed so that additional commands could be easily added to the tool. Efforts were made to keep the command set concise and simple. Currently, there are two main commands **Create** and **Add...To**. These are used to create a configuration and to add entities to the database. There are additional data identifiers to specify the type data used to define the object attributes. Figure 6 shows a portion of the input deck used to define a zero twist rotor blade. There are five additional functions called **Evaluate**, **Offset**, **Arc3pt**, **Reference** and **Conic3pt**. Respectively, these functions are used to evaluate points on a surface, define a point by offsetting an existing point, create an arc, reference a point to be used as an endpoint to a curve and create a conic. These are used to define points by setting dependencies to existing points within the database. The input deck consists of a series of commands defining each object.

A configuration is first defined within the input deck. A configuration may have a title associated with it. Multiple configurations within a single input deck are possible. After defining the configuration, geometry and topology may be added to the configuration. Surfaces are the first entities defined within the input deck, followed by points, and finally curves. Currently, the surface definition is expected to be read in as an IGES file. Eventually, the functionality of defining a surface by a series of discrete curves and generating the NURBS surface representation is expected to be added to the main ROTOR code. At that time, improvements to surface interpolation routines will be made. Points are defined as raw coordinate values, a parametric position on a surface, or as an offset from a previously defined point. Curves are defined by as a series of two or more points. A point previously defined within the database may be referenced. Surface curves have a special reference and are defined by specifying the endpoints on the surface. A special function allows the creation of an arc by specifying three points.

Once the geometry is defined, the topological entities such as blocks, edges, and patches must be defined. As per the entity hierarchy, the blocks are defined first. At a minimum, a block is expected to describe the dimension and the default spacing at the corners. When a block is added to the configuration, the six faces of the blocks are automatically created. By default, these faces are referenced by the tags Kmin, Kmax, Imin, Imax, Jmin, and Jmax describing the relationship of the face to the computational domain. Optionally, the faces within a block may be renamed to a more meaningful tag using the **FaceNames** command. The edges and patches are added to the appropriate block. An edge is defined by a shape, a grid point distribution, and an application region within the governing block. The edge is associated on a specified face within the governing block. The shape of an edge is defined by specifying a list of pre-existing curves and the valid parametric range within each curve. Since edges are directional entities, one must be careful in ordering the parametric range.

For example, consider the edge curve definition for Kmin%EDGE1 within Figure 6. The edge EDGE1 is defined as part of the Kmin face of block OHBLOCK. The edge shape is defined by two curves CURVE8 and CURVE9. The edges start at the beginning of CURVE8(t=0), and continues to the end of CURVE8(t=1). It continues to the beginning of CURVE9(t=0) and finally ends at the ending of CURVE9(t=1). Alternatively, consider the shape definition of Jmin%EDGE2. The shape is defined by CURVE15 and CURVE1. However, the edge direction

starts at the beginning of CURVE15 (t=0) and ends at the beginning of CURVE1 (t=0).

The distribution of grid points along the edge is obtained by specifying the spacing at each of the endpoints. A hyperbolic tangent function is used to determine the distribution of interior points constrained by the endpoint spacing. The application region describes where within the governing block that the edge is defined.

A patch is defined by specifying the face of the governing block, and four edges. Currently, the four edges must be comprised of surface curves from the same surface. The shape of a patch will be determined by this surface. The application region for the patch specifies the computation range within the block that the patch affects. It will be determined by the region within the governing block where each of the edges have been defined.

Once the geometry and the topology is created, the code generates the necessary files to be used by the volume grid generator. The surface definition along with the discrete edge information for all 12 edges of each block is written out. Also, a set of commands to generate the volume grid within CSCMDO is also generated.

Volume Grid Generation

The ROTOR tool uses CSCMDO to generate the volume grid. The ROTOR tool is expected to generate a grid suitable for using in the full-potential code FPX. The initial step is to create an algebraic structured volume grid. Afterwards, block faces can be elliptically smoothed in order to obtain the necessary orthogonality and desired grid smoothness for FPX. CSCMDO is a general multi-block three dimensional volume grid generator which is specifically designed for Multidisciplinary Design Optimization (MDO). The volume grids are generated and controlled by using ASCII user input deck[5]. Within the ROTOR tool, CSCMDO is automatically run by the main script completely transparent to the user. The main advantage of using CSCMDO as the volume grid generator is reuse of technology. CSCMDO has been successful in generating volume multi-block structured volume grids for some time now. Another advantage is that CSCMDO was specifically designed to fit within a design optimization loop. Since the ROTOR tool is being designed to generate volume grids for isolated rotor blades using full-potential solvers, it should be straight forward to place the tool within a design optimization cycle. Also, CSCMDO has some additional capabilities such as elliptic smoothing of faces and grid deformations that could benefit the ROTOR tool. Currently, CSCMDO uses a discrete point representation of surfaces which is expected to be changed to a NURBS representation in the near future. The grid deformations within CSCMDO are used to update an existing volume grid to conform to a perturbation of the surface. It may be possible to utilize this technology to model the rotor wake in fixed wing or hover mode.

Experience has shown that the flow solver FPX is particularly sensitive at the trailing edge of the rotor blade. Figure 7 compares the face of grid which was elliptically smoothed using only Thomas-Middlecoff formulation to same grid that was also smoothed using a combination of LaPlace and Sorenson smoothing. The first grid resulted in a divergent solution using FPX. The second grid resulted in a convergent solution using FPX. However, the second grid required quite a bit of user-interaction to obtain. Consequently, further study to quantify what FPX considers as a valid grid is needed. Therefore, it is likely that some additional routines are needed to manage the smoothing within CSCMDO in order to satisfy FPX requirements. Presently, the volume grid generated by the ROTOR is algebraic only.

CONCLUDING REMARKS

As a proof of concept, an O-H topology was created for an untwisted V22 Tiltrotor rotor blade. An algebraic volume grid was generated. Using the input deck with some minor modifications, a grid around a twisted V22 rotor blade was created. Thus far, the ROTOR tool has shown some promising results. It can generate a NURBS surface representation from discrete airfoil definitions. The continuous surface representation of the rotor blade can be used to generate a structure block or unstructured volume grid. The primary purpose of the ROTOR tool is to generate a block structured volume grid to be used with full-potential solvers such as FPX. It has been demonstrated that the ROTOR tool can generate an algebraic grid with no user intervention beyond setting up the initial topology input deck.

However, several issues need to be resolved before the objective of using the resultant volume grid within FPX is reached.

FUTURE WORK

The scope of the project that initiated this paper is to obtain a tool that can be used to generate a volume grid about an isolated rotor and be used by the full-potential code FPX. Another aspect of the project to generate a surface representation that can be used in both unstructured and structured grid analysis. In order to reach these goals, additional work in resolving some of the issues relating to grid smoothness, aspect ratio, and orthogonality off the rotor surface need to be resolved. Additionally, the use of surface fitting as opposed to surface interpolation in order to create the NURBS surface representation of the isolated rotor blade is also being investigated. As a final phase of the project, an interactive Graphical User Interface (GUI) is to be developed in order to help set up the topology more easily.

ACKNOWLEDGMENTS

The authors would like to thank Dr. H. Jones and C. Burley of the Fluid Mechanics and Acoustics Division at NASA-Langley Research Center for initiating this project. The first author would like to thank William Jones of Computer Sciences Corporation for his suggestions, and the use of CSCMDO which serves as the structured volume grid generator for the ROTOR tool. The first author would also like to thank Carl Rogers for his suggestions and expertise in quantifying the grid quality measures that result in a valid grid for FPX.

REFERENCES

1. Meakin, R., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations," AIAA-93-3350-CP, 11th Computational Fluid Dynamics Conference, pp. 576-588, July 1993.

2. Pirzadeh, S., "Progress Toward A User-Oriented Viscous Unstructured Grid Generator," AIAA-96-0031, 34th Aerospace Sciences Meeting and Exhibit, January 1996.

3. Bridgeman, J.O., Prichard, D., and Caradonna, F.X., "The Development of a CFD Potential Method for the Analysis of Tilt-Rotors," presented at the AHS Technical Specialists Meeting on Rotorcraft Acoustics and Fluid Dynamics, Philadelphia, PA, October 15-17, 1991.

4. McCroskey, W. J.,"Some Recent Applications of Navier-Stokes Codes to Rotorcraft," presented at the Fifth Symposium on Numerical and Physical Aspects of Aerodynamic Flows, Long Beach, CA, January 13-16, 1992.

5. Jones, W.T. and Jamshid Samareh-Abolhassani, "A Grid Generation System for Multi-disciplinary Design Optimization," Proceedings of 12th AIAA Computational Fluid Dynamics Conference, AIAA-95-1689, San Diego, CA, June 20, 1995, pp. 9.

6. Samareh, Jamshid, "GridTool: A Surface Modeling and Grid Generation Too," Proceedings of the Workshop on Surface Modeling, Grid Generation, and Related Issues in CFD Solutions, NASA CP-3291, May 9-11, 1995.

FIGURES

1) Unstructured surface mesh around Apache Helicopter

2) Schematic of ROTOR tool process

3) Image showing input curve sections defining a BERP rotor blade

4) Image showing the NURBS surface obtained by ROTOR using input curves defining the BERP rotor

5) Diagram showing the entity definition hierarchy

6) Text showing portions of a sample ROTOR input file used to describe a rotor.

7) Image comparing two grids which were given as input to FPX full-potential flow solver

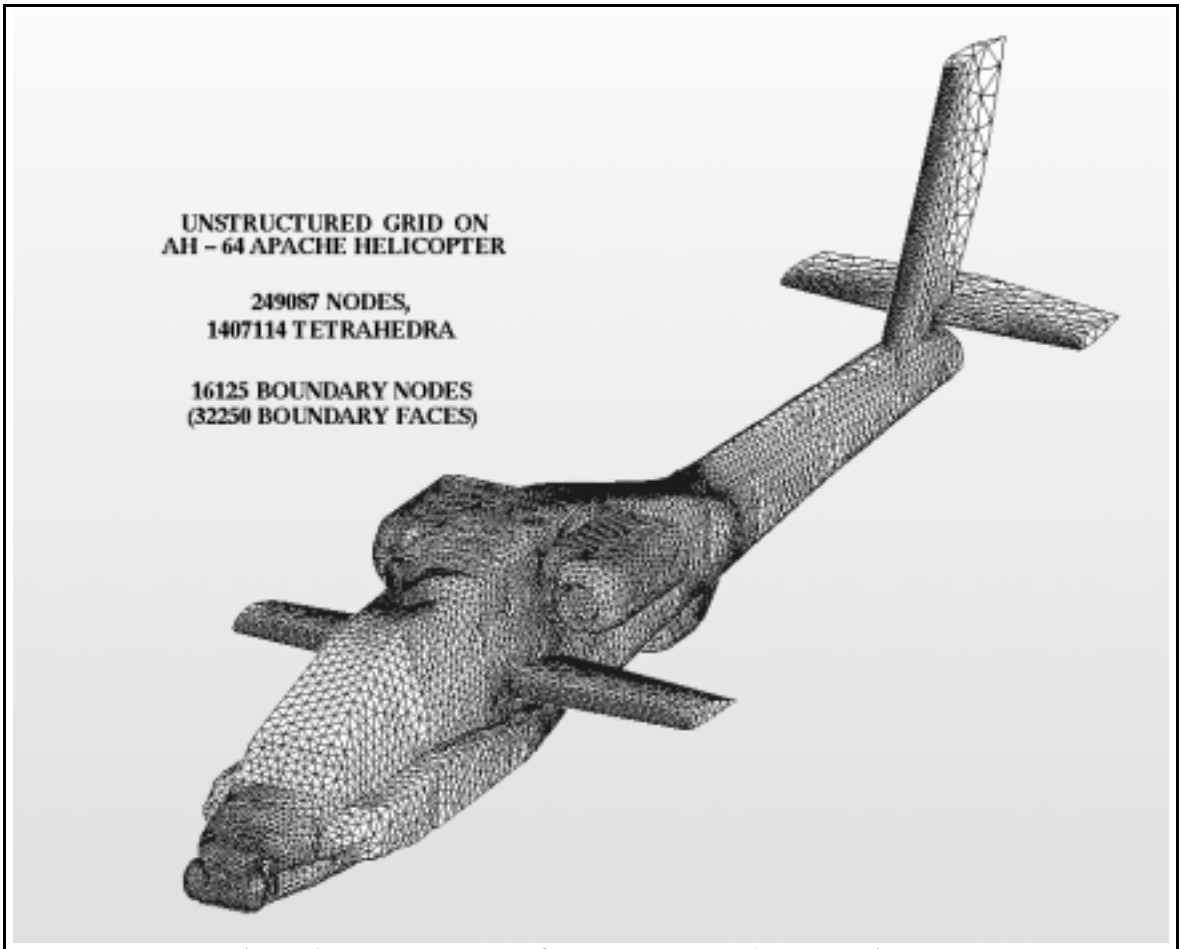


Figure 1: Unstructured surface mesh around Apache helicopter

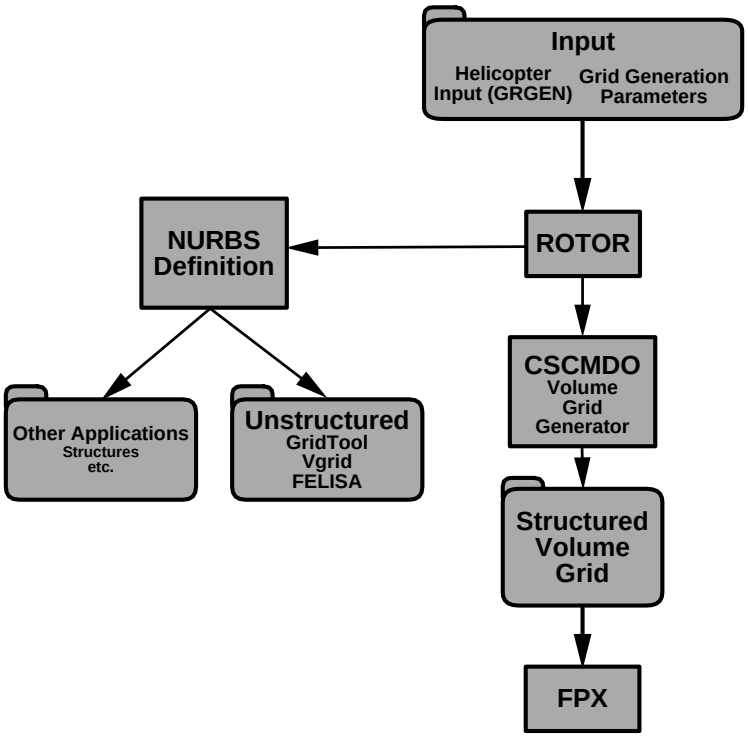


Figure 2: Schematic of ROTOR tool process

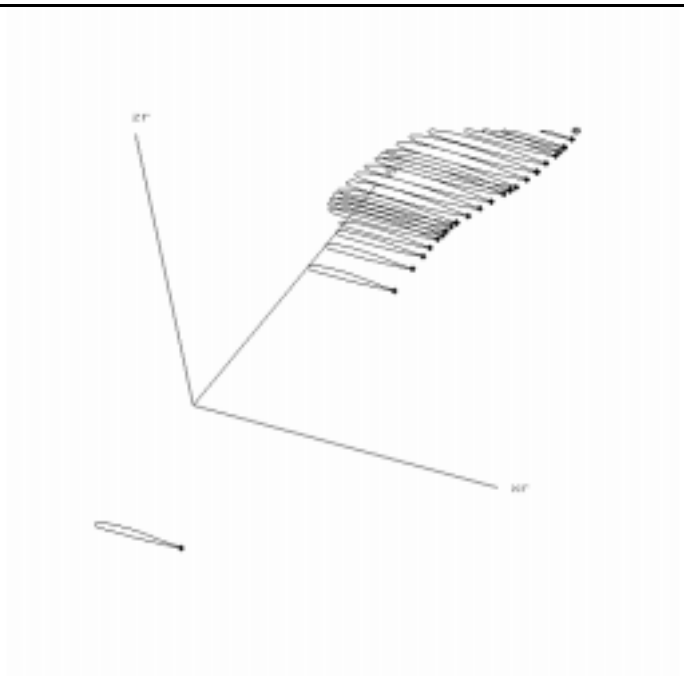


Figure 3: Image showing input curve sections defining BERP rotor blade

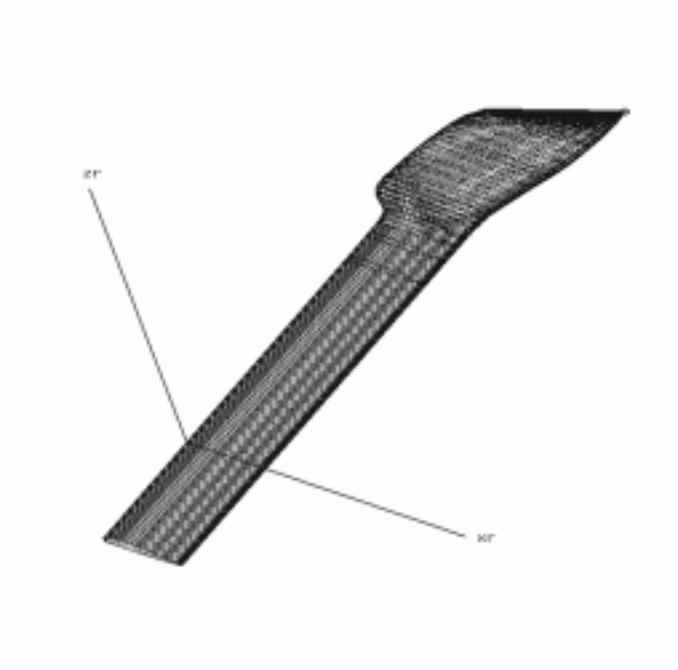


Figure 4: Image showing the NURBS surface obtained by ROTOR using input curves defining the BERP rotor

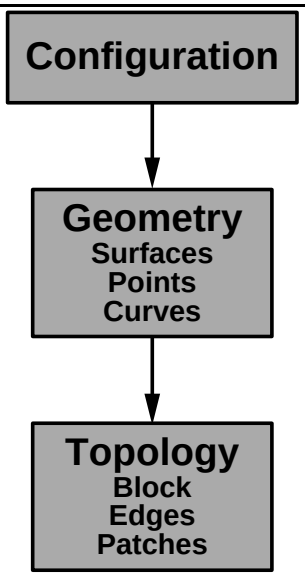
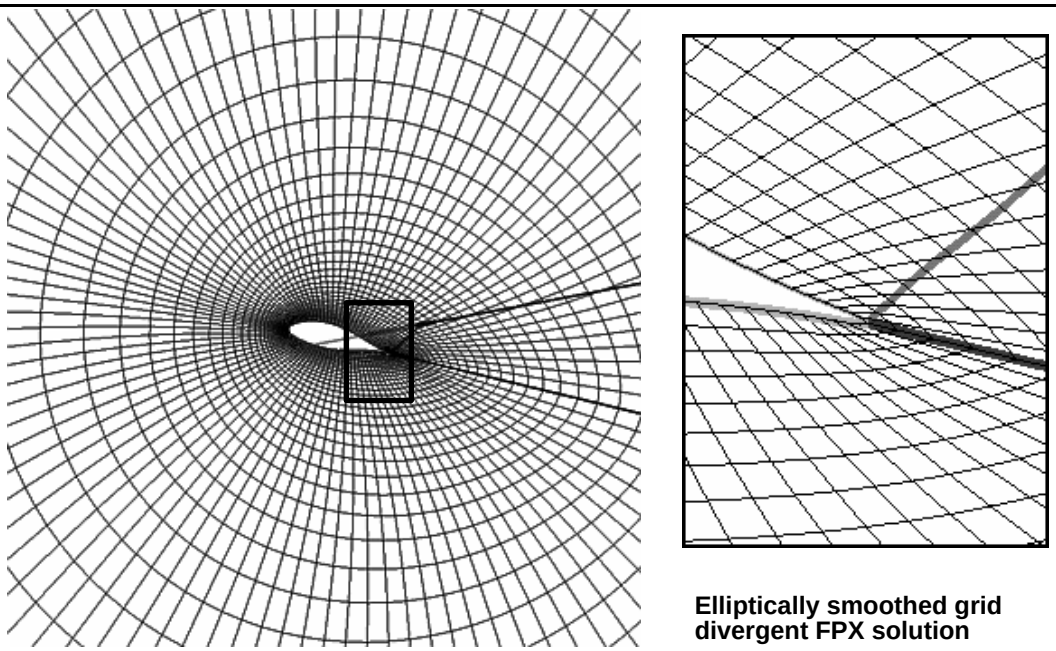


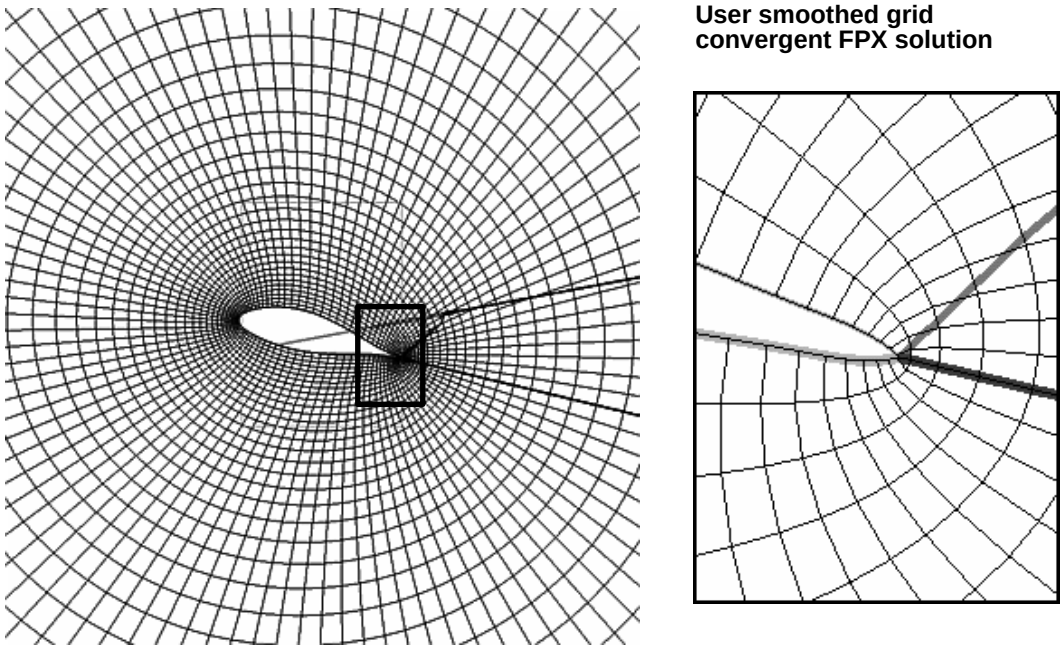
Figure 5: Diagram showing the ROTOR entity definition hierarchy

```
Create Config XV15 {
   Title Case1: Zero twist blade
}
Add Surface ROTOR To XV15 {
   ReadIGES Rotor.igs
}
Add Point Point1 To XV15 {
   XYZ( −5.0, 0.0, 0.0)
}

Add Point Point11 To XV15 {
   Evaluate Surface ROTOR At { UV(u1 = 0.5, u2 = 0.0) }
}

...

Add Curve CURVE1 To XV15 {
   Arc3Pt { Point2 Point16 Point1 }
}

...

Add Block OHBLOCK To XV15 {
   Dimension(101, 49, 37)
   Spacing {
      .01 .01 .01 .01
      .1  .1  .1  .1
   }
}

Add Edge Kmin%EDGE1 To OHBLOCK {
   Define Shape CURVE8 CURVE9 With {
     UV(0.0, 1.0) UV(0.0, 1.0)
   }
   ApplyToBlock { 1 101 1 1 }
}

Add Edge Jmin%EDGE2 To OHBLOCK {
   Define Shape CURVE15 CURVE1 With {
      UV(0.0, 1.0)  UV(1.0, 0.0)
   }
   ApplyToBlock { 37 37 1 101 }
}

...

Add Patch Kmin%Patch1 To OHBLOCK {
   Define Shape ROTOR With {
      Restriction { 0.0 1.0 0.0 1.0 }
   }
   ApplyToBlock { 1 101 1 37 }
}
```

Figure 6: Shows portions of a sample ROTOR input file used to describe a rotor



Elliptically smoothed grid divergent FPX solution



User smoothed grid convergent FPX solution

Figure 7: Image comparing two grids which were given as input to FPX full-potential flow solver