Final Report for:

Surface Generation and Cartesian Mesh Support

NAG2-1458

Robert Haimes
Aerospace Computational Design Laboratory
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
haimes@mit.edu

Overview

This document serves the final report for the grant titled "Surface Generation and Cartesian Mesh Support". This completed work was in algorithmic research into automatically generating surface triangulations from CAD geometries. NASA's *OVERFLOW* and *Cart3D* simulation packages use surface triangulations as an underlying geometry description and the ability to automatically generate these from CAD files (without translation) substantially reduces both the wall-clock time and expertise required to get geometry out of CAD and into mesh generation. This surface meshing was exercised greatly during the Shuttle investigation during the last year with success. The secondary efforts performed in this grant involve work on a visualization system cut-cell handling for Cartesian Meshes with embedded boundaries.

Introduction

The computational steps traditionally taken for most engineering analysis suites (computational fluid dynamics -- CFD, structural analysis, heat transfer and etc.) are:

- Part Generation -- usually by employing a CAD system
- Grid Generation -- preparing the volume for the simulation
- Flow Solver -- producing the results at the specified operational point
- Post-processing Visualization -- interactively attempting to understand the results

For structural analysis, integrated systems can be obtained from a number of commercial vendors. These vendors couple directly to most common CAD systems and are executed from within the CAD GUI. It should be noted that the structural analysis problem is more tractable than CFD; there are fewer mesh topologies used and the grids are not as fine (this problem space does not have the length scaling issues of fluids).

For CFD, these steps have worked well in the past for simple steady-state simulations at the expense of much user interaction. The data was transmitted between phases via files. In most cases, the output from a CAD system could go to IGES or STEP files. The output from Grid Generators and Solvers do not really have standards though there are a couple of file formats that can be used for a subset of the meshing space (i.e. PLOT3D and CGNS data formats). The user would have to patch up the data or translate from one format to another to move to the next step. Sometimes this could take days (or longer). Specifically the problems with this procedure are:

- File based. Information flows from one step to the next via data files with formats specified for that procedure. File standards, when they exist, are inadequate. For example, geometry from CAD systems (transmitted via IGES files) is defined as disjoint surfaces and curves (as well as masses of other information of no interest for the Grid Generator). This is particularly onerous for modern CAD systems based on solid modeling. The part was a proper (watertight) solid and in the translation to IGES has lost this important characteristic. STEP is another standard for CAD data that exists and supports the concept of a solid. The problem with STEP is that a solid modeling geometry kernel is required to query and manipulate the data within this type of file.

- 'Good' Geometry. A bottleneck in getting results from a solver is the construction of proper geometry to be fed to the grid generator. Cartesian solvers can start from a closed tessellation of a solid. With 'good' geometry an unstructured grid can be constructed in minutes (even with a complex configuration). Adroit hybrid and multi-block methods are not far behind. This means that a ten million node steady-state solution can be computed on the order of hours (using current high performance computers) starting from this 'good' geometry. Unfortunately, the geometry usually transmitted from the CAD system is not the correct fidelity in the grid generator sense. The grid generator needs smooth closed manifold geometry. It can take a lot of interaction with the CAD output (sometimes by hand) before the process can begin.

- One-way Communication. All information travels on from one phase to the next. This makes procedures like node adaptation difficult when attempting to add or move nodes that sit on bounding surfaces (when the actual surface data has not been transmitted from the grid generation phase).

Until this process can be automated, more complex problems such as parametric studies, multi-disciplinary analysis or using the above procedure for design becomes prohibitive. There is also no way to easily deal with this system in a modular manner.

CAPRI

An alternative view to building analysis suites has been under development in the Department of Aeronautics & Astronautics at MIT. The Computational Analysis PRogramming Interface (*CAPRI*) [1] takes a geometry centric approach. This makes the actual geometry and topology (as well as a discretized view) accessible to all phases of the analysis. The connection to the geometry is made through an Application Programming Interface (API) and NOT a file system. This API isolates the top-level applications (grid generators, solvers and visualization components) from the geometry engine. *CAPRI*s approach allows the replacement of one geometry kernel with another, without effecting these top-level applications. This is a crucial advantage for NASA and large multi-division corporations where data from multiple CAD vendors are an every-day occurrence.

CAPRI's handling of CAD data is much better than STEP because when the data is queried, the same software is executed as used in the CAD system (the *CAPRI* application in some cases becomes part of the CAD system). Therefore, one analyzes the exact part that is in the CAD system. The geometry kernel loads the part so *CAPRI* is immune to changes made by the CAD vendors to their native files.

CAPRI uses the same idea as the commercial structural analysis codes but does not specify control. Software components of the CAD system are used, but the analysis suite, not the CAD operator, specifies the control of the software session. This also means that the license issues (may be) minimized and individuals need not have to know how to operate a CAD system in order to run the analysis suite.

Dual View of the Geometry

The *CAPRI* programming interface must not be overly complex; it is crucial that the geometry description be uncomplicated (but not too simple as to impair functionality). Most solid based computational geometry systems make the distinction between geometry and topology; *CAPRI* mixes these for a single data definition. The geometry and topology are defined in *CAPRI* in the following manner:

- Nodes are the simplest entities and are just points in 3 space.

- Edges are curves with a specified direction. Two unique Nodes bound each Edge. The Edge is parameterized with where the first Node having a lesser parameter value than the second bounding Node. The curve type and the actual parameterization are not exposed through *CAPRI*. From this definition, periodic curves (like circles) are not supported. It is *CAPRI*s responsibility to split these curves when found in the CAD data. (Note: this is a change in topology but not geometry.)

- Faces are parameterized surfaces with a given direction. Again the type of surface and the details of the parameterization are not exposed. The bounds of Face are Edges but the relationship is not as simple as the Edge-Node definitions. This is because Faces may be bounded by as few as 2 Edges and on the upper end there is no limit. The bounds of the Face are defined by closed set(s) of Edges. There may be one or more of these loops for each Face. Stored with each defining Edge is an orientation so that it is known whether to look at the Edge as specified or in the opposite sense. The loop is an ordered set that defines the orientation of the Face. The outer loop(s), specify the boundary of the surface, and traverse the Face in a right-handed manner -- defining the outward pointing normal (out of the volume). Any holes are specified by a left-handed traversal of Edges. Again *CAPRI* splits periodic surfaces.

- Boundaries. In *CAPRI*'s hierarchy the next entity is not from the CAD system, but an artificial object that is a collection of Faces. This is the connection to the analysis suite. Specific boundary conditions may be applied to the collection of faces.

- Finally, the Volume is closed by the complete set of the Boundaries.

CAPRI provides Edge and Face functions that given the parameter(s) a point is returned. Also, the inverse operation is available; given a point what are the nearest coordinates that exist on the geometric entity. Calls exist to get tangent, normal and curvature data. Complex operations are provided that assert whether a point is in/on an entity including the bounding volume.

It has been found that using even this simple data definition, the task of grid generation can still be daunting. In the absence of having a good CAD interface, a number of grid generators have developed techniques that can mesh directly from a closed triangular tessellation of a solid part. This is something that has been available -- the files generated for stereo lithography. *CAPRI* provides a manifold triangulation as a secondary view of the CAD part. Unlike the data for stereo lithography, *CAPRI*'s tessellation is coupled to the geometric/topologic definition. Every point on the Face tessellation contains the 3D coordinates, the [u,v] pair, and an indication if the point is a Node or on an Edge. All triangles are defined in a right-handed manner with the normal pointing out of the volume.

It is not suggested that one use this triangulation directly for the meshing process, but as a starting point. Using *CAPRI* is simple for those grid generators that can start from a triangulation; use the tessellation but before storing away any coordinates for those points that bound the domain, use *CAPRI* to snap the data to the geometry. *CAPRI*'s tessellation can be modified and enhanced to provide the characteristics required for most any meshing procedure [2].

Work Performed under this Contract

1. Improved Robustness for the CAD to surface triangulation

UniGraphics and *SoildWorks* (through *Parasolid*), *CATIA V4*, *CATIA V5* and *CV's CADDS-V* provide a closed, conformal tessellation of the solid. In these cases *CAPRI* uses the CAD kernel's result. *PTC's Pro/ENGINEER* provides a triangulation of individual surfaces but it does not close along bounding curves (Edges). *SDRC's OpenIDEAS* provides no triangulation at all!

CAPRI must perform a tessellation of the solid (for *ProE* and *IDEAS*) that meets the requirement that the object be closed in order that a consistent API is provided for all CAD back-ends. This is an exceedingly difficult task (remember the underlying geometry may only 'close' to a certain tolerance). When this work was started this task was one the unsolved applied Computational Geometry problems.

The best method to ensure that surfaces match is to start with a discretization of the bounding curves and 'grow' the triangulation inwards (and out from holes). In the past, an *ad hoc* series of techniques was used with the hope that one will provide a topologically and geometrically correct triangulation. These involve of the following methods; (a) assume the surface is flat, (b) triangulate by attaching nodes of similar normals [this works well for cylindrical surfaces], (c) seed interior points and (d) start from a hierarchal interior point distribution based on normals. Rarely, but at times, all these procedures fail and *CAPRI* does not provide a valid tessellation for the surface. These techniques did not have the appropriate handles to be able to adjust the quality of the resultant surface triangulation.

In fact, all current schemes used in surface triangulation require some point insertion criteria and invariably provide the points *a priori*.

The surface-meshing algorithm produced during this grant has the following characteristics:

- Provide a watertight tessellation for the solid in a robust manner
  Start with a discretization of the bounding Edges (trimming curves). These line segments provide at least one closed contour (there may be multiple outer loops) with the possibility of inner loops that reflect holes in the surface. Computation Geometry provides proofs that any 2D closed areas can be triangulated. This gives the algorithm the robustness required; we can always generate a starting triangulation in the parameter space of the surface, which is 2D [u,v]).
- Maintains a parametric and physical-space view of the triangulation
  A valid triangulation is always maintained in the surface's parameter-space. The initial triangulation may not be proper in physical-space (it may display creases and folds – the normals of adjacent triangles may not be pointing in a similar direction). Triangles are iteratively broken up in such a manner that the resultant tessellation is also valid in [u,v] and reflects an improvement in [x,y,z]. The operations on the triangulation can be thought

of as insertion in [u,v] based on criteria formed by the physical-space triangulation. The end result is one where the triangulation is valid in both parameter and physical spaces (which is also required by *CAPRI*).

- Point insertion locations automatically generated by user-defined quality parameters
  This algorithm will attempt to meet the user specified quality parameters. This is not guaranteed because of issues of model closure. As stated before the trimming curves may actually not be on the surface. This deviation is noted and point insertion is not allowed when the point gets to within a factor of this deviation from the bounds of the surface. This avoids generating creases in the tessellation. The adjustable quality parameters are:
  1. Maximum triangle side
     Any triangle whose side is larger than this value has that side bisected in [u,v]. This produces 4 triangles from 2 (the triangle's neighbor is also split) by adding a new point in the triangulation.
  2. Maximum angle between triangle neighbors
     When the angle between 2 facets is greater than the specified value the larger (in area) of the 2 triangles is split in 3. The center of that triangle in [u,v] is used for the new point and then 3 triangles are generated.
  3. Maximum distance between the triangle centroid and the underlying surface
     When a triangle does not meet this criterion, it is also split into 3 by the addition of the centroid point.
- Edge discretization is required to be consistent with surface quality parameters
  To get a model to display a fine tessellation overall it is necessary to generate the loop segments with the surface quality parameters. An additional step is also required that examines the surface curvature close to the bounding line.

A complete and detailed description of this method has being written in the form of a paper [4] and reflects the work done during year 1.

Specifically these 3 issues were handled during year 2:

- There was no checking of the bounding Edges in [u,v] when assembled to create the loops. It was therefore possible (do to the discrete nature of the result) that segments cross when the actual curves only come close. This situation causes the initial triangulator to fail. Software has been written to test for this condition and the remedy situation by either removing or adding points.
- It was believed that instead of inserting points at the center of the larger triangle, if a circumcenter insertion is performed then better triangulations will result. This has been tested and it appears not to improve the quality of the tessellation.
- The performance of the scheme has been addressed. The algorithm executes competing surface reconstructions after the insertion of each new point. Each series of triangle side swappings uses a different criteria (usually one in [u,v] and the other in physical-space). When the total number of triangles gets large, the time taken to perform this task grows geometrically. Recursive and partial swapping of diagonals has been implemented.

This work benefits users of both *OVERFLOW* and *Cart3D* by directly providing CFD ready surface triangulations that codes like *OVERGRID* (in *OVERFLOW*) or 'intersect' (in *Cart3D*) can use as input to create surface meshes. In addition, these surface tessellations can be improved via the surface triangulation codes as discussed in [2] for various other applications. The aim of this work is to permit users to read in native CAD geometry, and extract surface triangulations without actually having to interactively launch the CAD package. This substantially reduces the level of CAD expertise required for running simulations. These goals have been met.

Furthermore, a new software module that can be used with *Cart3D* and *OVERFLOW* has been created: SurfTriCAPRI. This application takes as input a CAD part and uses *CAPRI* to perform the geometry manipulation required to generate high fidelity triangulations. This allows for true hands-off meshing. SurfTriCAPRI has only been tested with *Pro/ENGINEER*.

During the final year the following tasks were completed:

- Quilting. The speed of cut-cell handling is proportional to the amount of geometry that has been intersected by the cell. CAPRI always reflects the CAD data to the application regardless of fidelity (that is, if a sliver Face exists then tiny high aspect ratio triangles will be seen in the tessellation). A Quilting package is being added to CAPRI that will allow sliver Faces to be integrated into neighboring (larger) Faces. The triangles associated with the Face will then be merged so less geometry is presented to the cut algorithms of *Cart3D*.
- Maintenance of SurfTriCAPRI. Additional CAD systems/Geometry Kernels have ported and checked-out. An *OpenCASCADE* port provides SurfTriCAPRI support without the requirement of licenses and allows for the import of STEP files.
- Support for SurfTriCAPRI beta-testers.
- Master-Model. In order to execute parametric studies and have the potential to perform automated design, access to CAPRI's Master-Model interface will be required. This enhanced API provides for queries of the list of design parameters and the "Feature Tree" structure. The ability to suppress nodes of the "Feature Tree" allows for defeaturing the geometry. Also, there is the capacity to set specific values of all design parameters and have the CAD system/geometry kernel rebuild the requested object(s). This has been accomplished in a design optimization setting by the *Cart3D* team. A library of parametric components has been built under *Pro/ENGINNER* and driven through CAPRI. A paper on this work has recently been presented [5].

Though not explicitly tasked in this final year, work continued on improving the quality and nature of CAPRI's triangulation algorithms. Some success has been had dealing with maintaining the quality of some surface triangulations while significantly reducing the counts. Attached to this report is the first draft of a paper "Watertight Anisotropic Surface Meshing Using Quadrilateral Patches" on the subject that will be submitted to one of the summer/fall Grid Conferences.

2. Cartesian (*Cart3D*) Visualization

Work is ongoing [3] but not completely finished in order to take the complex (cut) cells found in Cartesian systems and provide an underlying interpolation scheme. This is important for Lagrangean operations (e.g. streamlining and streaklining) and, in general, to perform the normal suite of scientific visualization tasks. The undertaking of breaking up a single complex (and usually concave) polyhedron into tetrahedra is almost complete. This result needs to be coupled with the visualization system *pV3*. The end result will provide a distributed visualization platform where the very large (order 10s to 100 million node cases) can be visualized at the desktop. This effort is unique in the visualization community and no other packages (commercial or research) currently address this issue.

The tetrahedral count has been reduced considerably by using face swapping during the triangle side piercing of the volume. Also, much effort has been placed in attempting to address the precision issues when the cut cell intersects a great deal of geometry. Tetrahedra of increasing small volumes tend to get generated under these circumstances. When this happens a number of crucial functions fail (i.e., which neighbor should be followed when tracing along a ray). Precision problems exist in the current scheme because there is no absolute method to determine if a new (constructed) point is 'too close' to an existing point. Relative techniques are used that are based on the magnitude of the weights that support the position of the prospective new point in the existing tetrahedron.

A 32-bit integer-based 3D mesh has been used to represent the cut cell space. All points in the cell will have an [i,j,k]. Point coincidence can now be determined by equality of 3 integer coordinates. This leaves more than enough bits in an IEEE double to accurately move though this discrete grid. This scheme also tends to fail but in many fewer instances. The failure mode is one where the point is moved onto the grid and then the target ray no longer intersects the tetrahedron.

Precision problems still persist in the current scheme because there is no absolute method to determine if a new (constructed) point is 'too close' to an existing point. Relative techniques are used that are based on the magnitude of the weights that support the position of the prospective new point in the existing tetrahedron. The 32-bit integer sub-mesh improved the situation but did not provide the level of robustness desired.

The use of exact arithmetic is under investigation in order to avoid the IEEE float-point precision issues.

3. Final Delivery

All pertinent software required and/or produced during this contract was delivered on-site during the week of 16 February 2004. Appropriate CAPRI libraries were built for both LINUX and SGI equipment for the following back-ends:

- Pro/ENGINEER
- OpenCASCADE
- Parasolid

A version of SurfTriCAPRI was built upon these libraries. Also a STEP reader was constructed that the OpenCASCADE port of SurfTriCAPRI could read.

References

[1] Haimes R., and Follen G., "Computational Analysis PRogramming Interface." Proceedings of the 6th International Conference on Numerical Grid Generation in Computational Field Simulations. Eds. Cross, Eiseman, Hauser, Soni and Thompson, July 1998.

[2] Aftosmis M., Delanaye M, and Haimes R., "Automatic Generation of CFD-Ready Surface Triangulations for CAD Geometry." AIAA-Paper 99-0776, Jan. 1999.

[3] Hendricks A., Haimes R., and Aftosmis M., "Three Dimensional Finite Element-Based Grid Generation for Cartesian Mesh Visualization." AIAA Paper 2001-0916, Jan. 2001.

[4] Haimes, R. and Aftosmis, M., "On Generating High-Quality *Water-Tight* Triangulations Directly from CAD", Proceedings of the 8th International Conference on Numerical Grid Generation in Computational Field Simulations, Honolulu, Hawaii, June 2002.

[5] Nemec, M., Aftosmis, M., and Pulliam, T., "Aerodynamic Design of Complex Configurations Using Cartesian Methods and CAD Geometry", AIAA Paper 2004-0113, Jan. 2004.

Watertight Anisotropic Surface Meshing Using Quadrilateral Patches

Robert Haimes
Massachusetts Institute of Technology
haimes@mit.edu
and
Michael J. Aftosmis
NASA Ames Research Center
aftosmis@nas.nasa.gov

Abstract

This paper presents a simple technique for generating anisotropic surface triangulations using unstructured quadrilaterals when the CAD entity can be mapped to a logical rectangle. Watertightness and geometric quality measures are maintained and are consistent with the CAPRI default tessellator. These triangulations can match user specified criteria for chord-height tolerance, neighbor triangle dihedral angle, and maximum triangle side length. This discrete representation has *hooks* back to the owning geometry and therefore can be used in conjunction with these entities to allow for easy enhancement or modification of the tessellation suitable for grid generation or other downstream applications.

Introduction

The premier design goal for the **C**omputational **A**nalysis **PR**ogramming **I**nterface (CAPRI) [1] is that geometry access be appropriate for CAE developers. The complete Computational Geometry (CG) perspective is avoided while maintaining full functionality. Early in the design and implementation of CAPRI, it became obvious that providing an Application Programming Interface (API) that only gives the programmer access to the geometry and topology of a solid part was insufficient. The burden of deciphering the CAD data and attempting to generate a discrete representation of the surfaces required for mesh generation was too great. Fortunately, many grid generation systems (used in CFD and other disciplines) can use STL (*Stereo Lithography*) files as input. Combining a discretized view of the solid part as well as it's geometry and topology can provide a complete, and easier to use, access point into the CAD data. A tessellation of the object that contains not only the mesh coordinates and supporting triangle indices but other data, such as the underlying CAD surface parameters (for each point), as well as the connectivity of the triangles, assists in traversing through and dissecting the CAD representation of a part.

An important aspect of CAPRI is that it provides CAD vendor neutral access to

all of the data obtained from the models that is to be passed back to the application. The triangulation generated by CAPRI is guaranteed to be watertight, regardless of the CAD kernel in use. Some CAD system geometry kernels can provide data of this quality (i.e., UniGraphics, Parasolid, CATIA and ComputerVision). Other CAD systems can provide the data, but it is not of sufficiently high quality to use. (For example, Pro/Engineer requires one to buy Pro/MESH to get a closed triangulation.) Finally, SDRC's Open I-DEAS API does not provide access to a triangulation at all. The fact that not all CAD systems provide such a tessellation has forced the development of a surface triangulator within CAPRI for CAD solid parts that *does* meet all of the quality requirements.

It should be noted that CAPRI's tessellations are not intended as the starting point for computational analysis (though they could be used in some cases). CAPRI sees only geometry, and it cannot anticipate the smoothness, resolution or other requirements of the downstream application(s). The triangulations approximate the geometry only; some processing of the tessellation is expected in order to refine the triangulation to a state suitable for the physical problem being investigated. The triangulation can be enhanced through either physical or parameter space manipulation, using point "snap" and (u, v) surface evaluations routines provided by the CAPRI API [2]. The triangulation technique used within CAPRI displays the following characteristics [3]:

- Robust. It is imperative that the scheme work for all possible topologies and provide a tessellation that can be used.
- Correct. The triangulation is of no use if it is not true to the CAD model. The tessellation must be logically correct; i.e. provide a valid triangulation in the parameter space (u, v) of the individual surface. It must also be geometrically correct; i.e. depict a surface triangulation that truly approximates the geometry. This involves ensuring all facets have a consistent orientation with no creases or abrupt changes in triangle normals. Correctness in both physical and parameter space allows CAPRI based application enhancement schemes to operate in either or both.
- Adjustable. To minimize the post-processing of CAPRI's tessellation for a specific discipline or analysis, some a priori adjustment of the resultant quality is available. It must be noted however, that any criteria may not be met (especially near the bounds of a CAD object) due to issues of closure and solid model accuracy. This goal may conflict with the more important characteristic of being watertight and having a smooth surface representation. The parameters are:
  - Maximum triangle side length. Any triangle sides (not on a CAD Edge) longer (in (x, y, z)) than a specified value are bisected.
  - Maximum dihedral angle between two triangles. Any two triangles on the same CAD Face whose (x, y, z) facet normals differ by more than the input value will be broken up.

- Chord-height tolerance. When the deviation between triangle center and actual surface (in (x, y, z)) is greater than the specified value then the triangle is subdivided by inserting the center point.

- No geometric translation. To truly facilitate hands-off grid generation, anything that requires user intervention must be avoided. All data maintained within CAPRI is consistent with the CAD's solid model representation. An alternate or translated representation is not used, because then the result will be something different than resides within CAD.
- Watertight. Triangulated CAD solids are closed and conformal; having this characteristic allows for meshing without "fixing" geometry. For the tessellation of a solid object, this means that all Edge (trimming) curves terminate at consistent coordinates of the bounding Nodes and a single discretization for Edge curves be used on both surfaces sharing the common Edge. Each triangle side in the tessellation is shared by exactly two triangles, and the star of each vertex is surrounded and bounded by a single closed loop of sides. The triangulation is everywhere locally manifold. In a manifold triangulation, there are no voids, cracks or overlaps of any triangles that make up the solid.

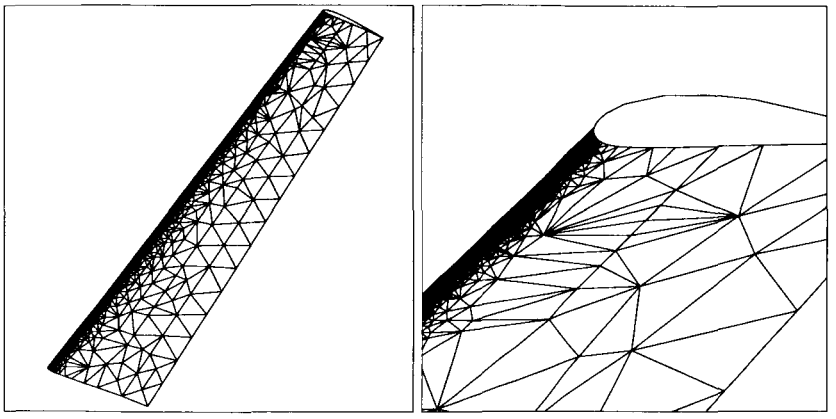

Figure 1. Isotropic Triangulation using default Tessellation scheme. # points = 6424, # triangles = 12639, Total CPU time = 3.8 sec.

As can be seen in Figure 1 the dihedral angle based refinement is sensitive to anisotropic curvature in the surface. Since the simple subdivision rules do not attempt to align inserted vertices with the direction of principle curvature, surface bends, cylindrical trailing edges, fillets and a surface with any linear feature require a large number of triangles to satisfy the angle metric. When combined with the isotropic nature of the MinMax triangulation in (u, v), this misalignment can make dihedral-angle based refinement expensive and provide results with high counts.

The CPU time to generate the complete tessellation of the solid (4 Faces) is 3.8 sec[1]. A careful timing analysis indicates that as the number of triangles increases on a Face, the proportion of time spent swapping for surface recovery grows rapidly. While triangulation speed has been improved through the use of recursive swaps, many triangles will still be required at anisotropic surface features without a more sophisticated site insertion strategy [3].

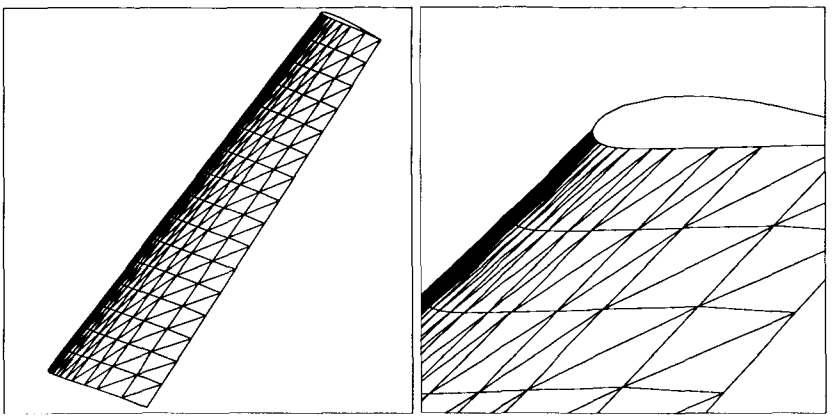

Figure 2. Triangulation based on a Trans-Finite Interpolation scheme. # points = 340, # triangles = 608, Total CPU time = 0.1 sec.

In an attempt to mitigate this problem a simple Trans-Finite Interpolation (TFI) scheme was implemented. The TFI procedure takes the (u, v)s along the bounds of the quadrilateral face and interpolates (u, v)s to interior points. These new parameter pairs can be used to evaluate to physical coordinates and therefore simply (and quickly) fill the Face with sub-quadrilaterals. These cells can be further subdivided to produce a triangular mesh. This initial scheme has the following restrictions:

1. Face must have only one bounding Loop (i.e. no holes)
2. The Loop must contain 4 Edges
3. Number of points found in opposing Edges must match

These limitations ensure few Faces can use this algorithm. But, the results are both enlightening and encouraging for Faces (with linear features) when the scheme can be employed. This can clearly be seen in Figure 2 where the resultant mesh is more regular, the counts are significantly reduced, the time to generate is a small fraction of the original, while satisfying the same geometric quality metrics. It should also be noted that the triangles generated are far from isotropic in particular those at the leading edge of the wing.

[1]All timings in this paper are generated on a 1.8GHz Pentium 4m running LINUX.

Another less obvious advantage of this type of scheme is consistency. Any change in geometry will produce an entirely different triangulation using the standard scheme. The topology of the TFI mesh is driven only by the bounds of the quadrilateral Face. Therefore, if the point count at the Edges remain constant then the interior triangulation is consistent. This can be useful in design settings when differencing is employed to determine parameter sensitivities. This is because point movement within a Face can be tracked.

Unstructured Quadrilateral Patch Fill

The easiest way to ensure a watertight triangulation of a solid is to first discretize the Edges that bound each Face. Face tessellations can then be performed using the Edge points and filling in the interior without regard to the neighboring Faces. In an attempt to capture more Faces using the TFI scheme it is obvious that the restrictions need to be relaxed.

Since we do not wish to change the manner in which the general triangulation scheme is done, it would helpful to find a TFI-like method that does not have restriction #3. This method must also be able to produce near-normal sub-quadrilateral elements near the bounds of the Face in (u, v) so that linear features (found at the Edges) can propagate into the Face in an anisotropic manner.

One Set of Opposite Sides Match

In this simplest case, one set of opposing quadrilateral sides match in point count – the other does not. If we assume that the largest of the mismatching sides is found at the right then the blocking that can be used to subdivide the Face is seen in Figure 3.

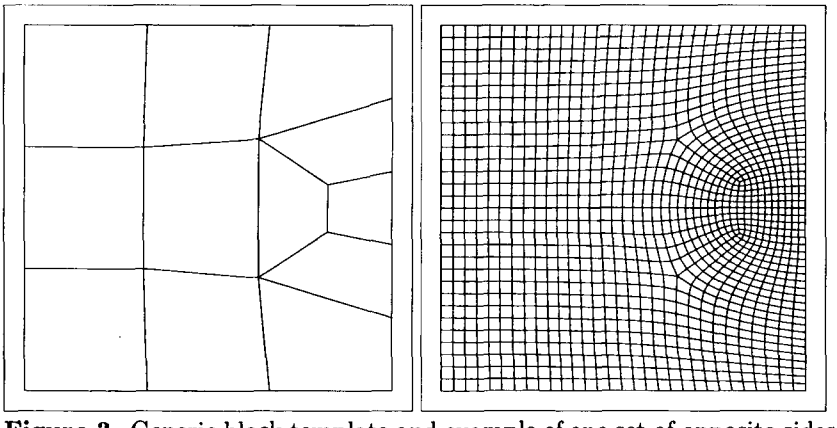

Figure 3. Generic block template and example of one set of opposite sides having the same point count.

By examining the block template (the left side of Figure 3) one can see that the additional points in the larger side are connected back to themselves by making a loop of elements. This loop is brought back to about 1/3 of the way in the opposite direction so that these elements do not penetrate too far to the left. Also the turning of the loop does not end up too close to the generating side so that the quads at the right side can be close to normal. The picture seen on the right-hand side of Figure 3 is a completed mesh using this block template. The size of each block is determined by either 1/3 of the appropriate side count or 1/2 of the difference between the opposing sides. Note that an odd difference is made even (by reducing the count by 1). This point is placed back into the final mesh by subdividing the appropriate sub-quad into 3 triangles instead of 2.

After each sub-block is populated, the result is improved by applying a Laplacian smoother.

It should be noted that the constraint of the side with the largest count being on the right is one of convenience. A set of up to 3 90^o rotations can take any configuration and place in the above situation.

Opposite Sides Differ by a Constant

Another simple case to consider is when opposing sides differ by the same count. The blocking can be found in the left-hand picture of Figure 4. Here we assume that the largest side is on the right and the largest side from the 2 others can be found at the bottom. Again this constraint is artificial. A set of up to 3 90^o rotations and a reflection can take any configuration and place in this setting.

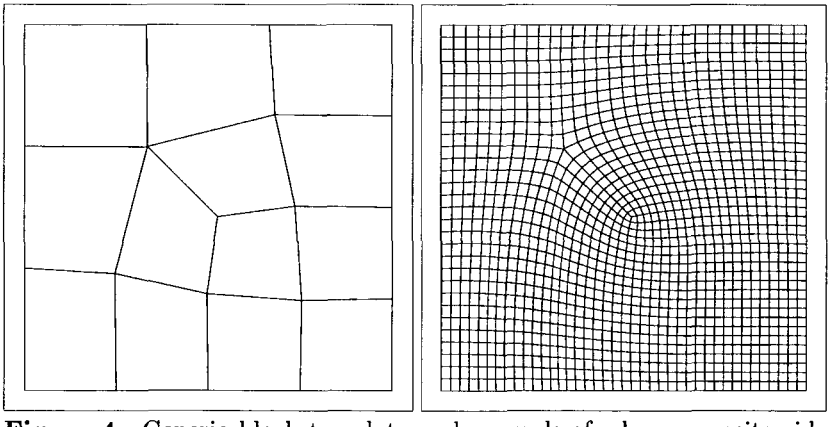

Figure 4. Generic block template and example of where opposite sides differ by the same point count.

For this case, it can be seen that the additional points generate elements that loop from the bottom and end up at the right quadrilateral side. The picture seen on

the right-hand side of Figure 4 is a completed mesh using this block template and is constructed in a similar manner to the first case.

General Case

The blocking for the general case can be found in the left-hand picture of Figure 5. This is, in fact, the combination of both of the simpler cases described above.

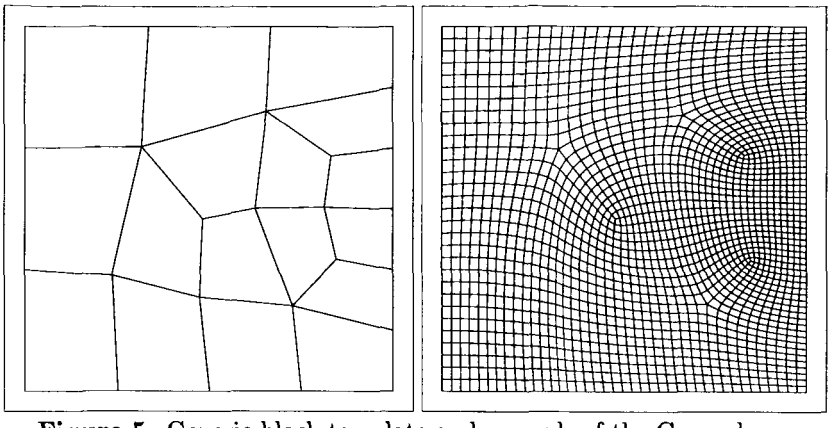

Figure 5. Generic block template and example of the General case.

As in the last case, we assume that the largest side is on the right and the largest side from the 2 others can be found at the bottom. This is not a constraint; a set of up to 3 90^o rotations and a reflection can take any 4 discretized sides and place in this configuration.

17 blocks are required in order to subdivide the original quadrilateral and both simpler cases can be seen imprinted in the blocking. The circular loop is obvious on the right as depicted in Figure 3 and the set of elements coming up from the bottom can still be tracked to the right of the original quadrilateral. Here it is clear that the elements get further broken up when the circular loop intersects this group of elements.

The picture on the right-hand side of Figure 5 is a completed mesh using this block template and is constructed in a similar manner to the first case; fill each block and then apply a Laplacian smoother. One can clearly see that local orthogonality has been maintained and those places that deviate from normal to the Face sides are far from those sides.

This scheme can deal with any 4 sides discretized with any number of points except for these conditions:

- A side has less than 4 points. This is because the basic method requires at least 3 blocks on a side unless the true TFI algorithm can be applied.

- The number of points on opposing sides differs greatly. It is possible to have situations where this scheme does not reduce the vertex/triangle count over the default triangulation. This occurs when there is a great disparity between opposing side counts. It has been found that when the side vertex count ratio (largest/smallest) gets greater than 3 the benefit begins to be minimized. This heuristic is used to limit the use of the anisotropic scheme.

Loops that do not have 4 Edges

In an attempt to further remove the constraints of this TFI-like anisotropic triangulation scheme we now look at restriction #2:

3 Edges

Under the limited set of circumstances that a Face with 3 Edges contains a degenerate Node; the Face can be viewed as a quadrilateral and the technique described in the previous section can be used. Degenerate mappings can at the tip of a conical surface and the pole of a spherical surface. This can be identified by a discontinuity in the values of (u, v) on either side of the Node.

When this situation is found, the following technique can be used:

- Create a virtual side at the degenerate Node. A new side is created with the same number of vertices as the side that is now opposing. The (u, v)s copied from the opposite side and appropriate parameter (the one not incrementing/decrementing) is set to that found with the Node.
- Perform TFI or **One Set of Opposite Sides Match** scheme.
- Deflate the virtual side. When the sub-element quadrilaterals are broken up, any triangles that have an edge on the virtual side are not included in the final tessellation.

More than 4 Edges

If one can determine sets of Edges that are part of a larger continuous curve in physical space, these can be considered a single quadrilateral side. If after analyzing all Edges there are 4 sides then the schemes described in this paper can be applied.

The technique used to examine each pair of Edges is simple and requires the following to be true:

- Is the Edge an isocline (have roughly a constant u or constant v)?
- Is the pair the same type of isocline? If so, then the pair can be considered part of a single quadrilateral side.

Example – A Turbine Blade

The following question needs to be asked; can the methods outlined in this paper show real benefits for actual CAD parts? To answer this question we will use an example from turbomachinery. The turbine blade part contains not only the aerodynamic shape but also the hub and tip casements including the *fir tree*. The full geometry (generated in Pro/ENGINEER) can be seen in the left-hand picture of Figure 6 and a blow-up of the hub/leading-edge region can be seen on the right.

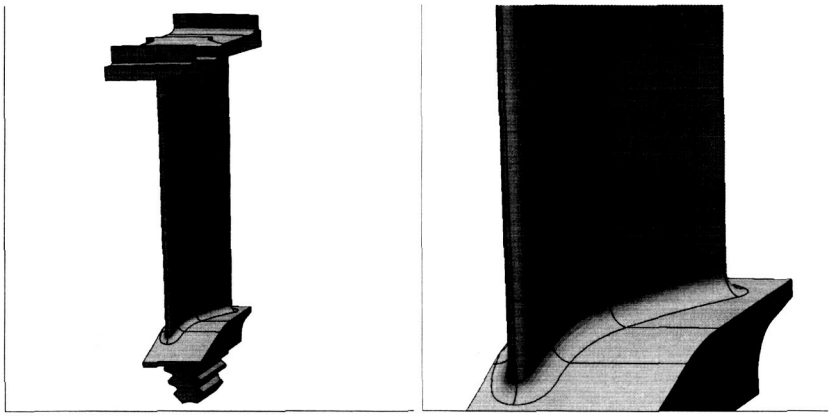

Figure 6. A turbine blade CAD part on the left. There are 94 Faces the represent the solid. On the right is a blow-up of the blade leading-edge hub junction where the fillets can be seen.

Figure 6 shows that the fillet between the hub and the aerodynamic shape as broken up into quadrilateral patches (this is seen with most all CAD systems). The set of trimming curves along the upper bounds of the fillets is a single entity as far as the aero shape is concerned (broken up to maintain the manifold aspect of the solid). It should also be noted that the aero shape is split at the leading edge into suction and pressure surfaces each reflected in a single CAD Face.

The left-hand side of Figure 7 shows the isotropic triangulation of the blade surfaces and the fillets for the hub blown-up view. The entire tessellation of the solid contains 66308 triangles and took 60.1 CPU seconds. On the right one can see the triangulation of the same Faces (all using the anisotropic scheme). The complete solid contains 22262 triangles and took only 2.53 CPU seconds being able to treat 56 of the 94 Faces as quadrilateral patches. From the performance improvement one can assume that the Faces that consumed most of the time for the isotropic triangulation were handled by the quadrilateral scheme (the suction and pressure surfaces).

The pressure surface is properly handled even though it is bounded by more

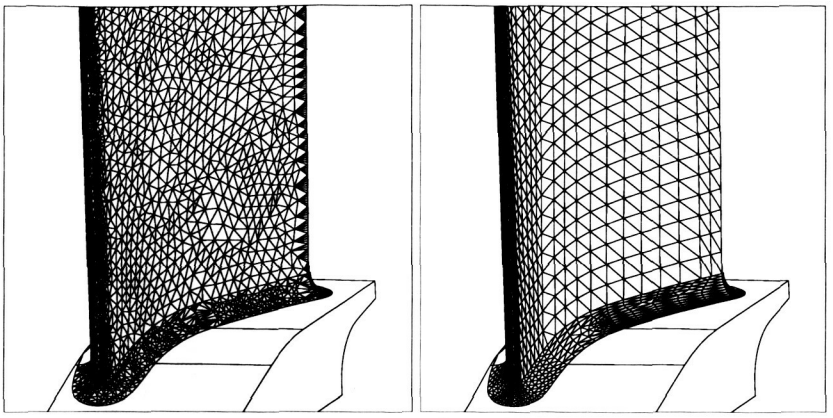

Figure 7. The surface mesh displayed on the Blade surface and some of the fillets. The picture on the left displays the results from the isotropic tessellation scheme. Seen on the right are the same Faces triangulated with the anisotropic method.

than 4 Edges (3 Edges can be seen at the mating with the fillet Faces). An abrupt spacing change can noted in this Face's triangulation and looks odd at first inspection. The location of this change is at the position where the fillet Faces are subdivided. Remember that the interior points of a TFI algorithm are driven by the spacing at the frame. In this case each Edge has been discretized separately and there is much more curvature near the leading edge producing a finer set of points. The fillet Edge interior to the leading edge sees much less curvature and hence displays a coarser spacing.

It is easy to imagine that both the performance gain and the ratio of quadrilateral Faces to total CAD Faces to be high in this example. In the area of performance this may be true; if the suction and pressure surfaces of the blade had cooling holes neither of the Faces could have used the anisotropic method. But, after a (non-rigorous) survey of CAD parts a ratio of around 50% is appears to be average.

Discussion

The anisotropic quadrilateral patching method described in this paper has been integrated into CAPRI, but it is not fully automatic. This is due to the possible situation where the quadrilateral approach cannot be applied and the default triangulation method must be used. Due to the isotropic nature (in (u, v)) of the tessellation scheme the Edge discretization must be finer in regions of high surface curvature so the geometry can be captured. That is the default. So without (hands-on) intervention fewer CAD Faces employ the quadrilateral scheme because the constraint based on **the number of points on opposing sides**

differ greatly becomes invoked.

Effort is underway to integrate the two surface meshing schemes so that it can become fully automatic. This is difficult because the Edge discretization is done before the Faces are tessellated (a requirement of the watertight attribute). The last phase of the Edge discretization is the examination of the local curvature of both of the Faces that touch the Edge. If curvature is found then the Edge tessellation continues to be enhanced. This is not only not necessary for the quadrilateral method, but can significantly reduce its quality if employed.

This also brings up the possibility of taking the current default surface triangulation and supporting anisotropic meshing. One of the swapping techniques used in the isotropic triangulation scheme drives the tessellation toward Delauney (Min-Max). This is done in the underlying surface's parameter space (u, v). Since the parameter space is artificial (i.e. not physical) it could actually be any 2D mapping. Therefore a transformation from a 2D space that could support an anisotropic stretching to the surfaces parameter space would be all that is required to achieve the anisotropy found in the quadrilateral patch method. One could base the transform on the spacing found at the (u, v) rectangular bounds for the Face.

Finally, if this is successful, then the last phase of the Edge discretization may be able to be removed. Or, could be modified so that the spacing reflects the transformed 2D space. This then could mitigate restriction #1.

References

[1] R. Haimes, and G. Follen. **C**omputational **A**nalysis **PR**ogramming **I**nterface. Proceedings of the 6^{th} International Conference on Numerical Grid Generation in Computational Field Simulations. University of Greenwich, June 1998.

[2] M. Aftosmis, M. Delanaye and R. Haimes. Automatic Generation of CFD-Ready Surface Triangulations from CAD Geometry. AIAA Paper 99-0776, January 1999.

[3] R. Haimes, and M. Aftosmis. On Generating High Quality "Water-tight" Triangulations Directly from CAD. Proceedings of the 8^{th} International Conference on Numerical Grid Generation in Computational Field Simulations. Honolulu HI, June 2002.