AUTOMATIC STRUCTURED GRID GENERATION USING GRIDGEN (SOME RESTRICTIONS APPLY)

John R. Chawner and John P. Steinbrenner
Pointwise, Inc.
Bedford, Texas

OVERVIEW

The authors have noticed in the recent grid generation literature an emphasis on the automation of structured grid generation. The motivation behind such work is clear; grid generation is easily the most despised task in the grid - analyze - visualize triad of computational analysis (CA). However, because 1) grid generation is closely coupled to both the design and analysis software and because 2) quantitative measures of grid quality are lacking, "push button" grid generation usually results in a compromise between speed, control, and quality. Overt emphasis on automation obscures the substantive issues of providing users with flexible tools for generating and modifying high quality grids in a design environment. In support of this paper's tongue - in - cheek title, many features of the Gridgen software are described. Gridgen is by no stretch of the imagination an automatic grid generator. Despite this fact, the code does utilize many automation techniques that permit interesting regenerative features.

INTRODUCTION

Gridgen [1] is a software system for generation of 3D, multiple block, structured grids. The system is comprised of two codes: Gridgen (see Figure 1), an interactive program that provides capabilities ranging from geometry model import through volume grid initialization and analysis software pre-processing; and Gridgen3D, a batch program for volume grid refinement. Gridgen is a monolithic program in the sense that it consists of a single process and a single graphics window. Gridgen's window, however, is repositionable and resizable. Gridgen also manages its own non-overlapping windows inside the main window. User interaction with Gridgen is directed through text - based button menus activated via the mouse or keyboard "hot keys". A 3D, graphical image of the grid is always present and may be easily panned, zoomed, rotated, and otherwise customized by the user.

The manner in which grids are constructed using Gridgen is is based on a hierarchical paradigm, beginning with the geometry model, and proceeding to curve, surface, and volume elements as shown in Figure 2. The geometry model, provided as input to Gridgen, serves as the foundation for the hierarchy. The user constructs curves which are in turn used to build the topological surface and volume components. The grid for each of the hierarchical components is an implicit but separate part of the component. The result of maintaining a distinction between the geometry model, the hierarchical components, and the grid allows changes to be rapidly propagated either forward or backward throughout the grid system.

Users operate Gridgen in much the same manner as a product designer operates a computer aided design (CAD) system. After importing the geometry model (database), the user draws curves (connectors), assigns a number of points to each connector, and distributes those points along the connector using a variety of distribution functions. The connectors are then selected as the boundaries of surface grids (domains). Transfinite interpolation (TFI), elliptic partial differential equation (PDE), and hyperbolic PDE methods are available for controlling the distribution of grid points on the domains. The domains are

then selected as the boundaries of volume grids (blocks). Again, TFI, elliptic PDE, and hyperbolic PDE methods are available for grid point control within the blocks. Finally, the user sets boundary conditions and exports formatted data for the chosen analysis software.

From the description above it should be apparent that grid generation using Gridgen is a user - in - the - loop task. Therefore, one of the goals of the software design has been to *automate* as much of the grid generation minutia and bookkeeping as possible so that the user may concentrate on topology and grid quality. This is made possible through Gridgen's data hierarchy. The data hierarchy maintains the inter-relationships between the 1D, 2D, and 3D grid components (connectors, domains, and blocks, respectively) so that the user's changes may be propagated up or down the hierarchy *automatically*. Some of these *automation* tools are described in the following sections.

EXAMPLE GRID

Gridgen's *automation* features will be demonstrated in the context of grid generation for an external automotive shape. This bluff body, described in Reference [2] and illustrated in Figures 3, 4, and 5, was the subject of a wind tunnel test that studied its wake structure relative to the base slant angle. This vehicle was also used as the basis for an evaluation of computational fluid dynamics (CFD) software for automative shapes (Reference [3]). The blocking system and surface grids created using Gridgen are shown in Figure 6.

DATABASE

Database is Gridgen's term for the geometry model of the object on and around which the grid is to be generated. Gridgen's database capability is based mathematically on nth degree, rational, Bezier curve and surface geometry. The database also includes relational data such as grouping, color, names, etc. Since Gridgen is dedicated to the generation of hexahedral grids, the user's CAD software may be used to create the database. Gridgen can import the database from several industry standard file types, including those listed below.

- PLOT3D (bilinear surfaces)
- Patran V2.5 Neutral[1]
- IGES[2]

The bilinear database surfaces for the bluff-body grid were created from the drawings and tabular data in Reference [2]. The surfaces in this particular database were stored in two files: one containing the five surfaces for the fore- and mid-bodies, and one containing the four aft-body surfaces.

Gridgen offers several methods for placing grid points on the database including curves that may be drawn directly on the surfaces ("Line on DB" and "Curve on DB" segment types), projection of curve grids, and projection of surface grids. More importantly, Gridgen *automatically* maintains the adherence of grid components to the database. This relationship between the grid and the database is maintained via the database entity name. Each entity in Gridgen's database has a unique name and every grid component

[1] Packet types 32 and 33

[2] Entities 000, 100, 102, 104, 106, 110, 116, 124, 126, 128, 212, 308, 314, 402, 406, 408

that adheres to the database saves the name of the entity and the database coordinates (see Figure 7). The utility of this feature is explained in the following paragraphs.

Gridgen provides several commands that allow changes to be made interactively to the database. The entire list of database commands is listed below.

- Import
- Export
- Examine
- Name
- Group
- Ungroup
- Reduce
- Copy
- Delete
- Translate
- Scale
- Rotate
- Intersect

When the shape modification commands (Translate, Scale, and Rotate) are applied by the user, all of the grid elements adhering to the modified entities (as determined by the data stored in Figure 7) are also modified *automatically.*

Changes to the grid based on parametric variations of the database can also be made in a "hands off" mode. As before, let's assume that a grid for the 40° base slant has already been created and that we wish to create a grid for the 20° base slant. Recall that Gridgen associates grid components with database entities via the entity name. Our goal, therefore, will be to create a database for the 20° base slant case and have Gridgen think that it's the same as the 40° configuration. In order to accomplish this it's important to understand Gridgen's convention for naming database entities. For IGES files it's simple: Gridgen uses the names stored in the file. For PLOT3D and Patran files, which don't contain entity names, Gridgen constructs the name from a text string of the form **basename-type-number** where

basename is the basename of the file from which the entity was imported,
type is a descriptive term for the type of entity (e.g, **psurface** for a parametric surface),
number is the sequence number of the entity in the file

Since the user knows that a variation of the base slant angle will require new several aft databases, the database files can be created ahead of time and named **aftxx.net** where the **xx** is the base slant angle, ie **aft40.net**. When the baseline 40° case is first generated we copy the 40° database file to a generic file

```
cp aft40.net Aft.net
```

Then, the baseline grid will be generated with references to database surfaces called **Aft-psurface-n** and the grid is saved to a Gridgen file. In order to generate grids for the other base slant angles the following steps can be used.

1. Copy the 20° database file to the same generic name used for the baseline case, ie.

   ```
   cp aft20.net Aft.net
   ```

2. Import into Gridgen the database files **Body.net** and **Aft.net**.
3. Import the Gridgen file that was created for the 40° case.

Since Gridgen doesn't notice a difference between the current and previous database configuration (ie, a database entity called **Aft-psurface-1** has been imported and there are grid references to that entity) the

grid is *automatically* regenerated in terms of the new database through the simple act of importing the files. Therefore, with a little forethought, the user can parametrically change the shape of the database without requiring significant grid rework. An example of this is shown in Figure 8 which shows how the 40° grid was changed into the 20° grid simply by importing a new database file.

CONNECTORS

The one dimensional grid element in Gridgen's data hierarchy is the connector. Each connector consists of three sets of information: the 3D curve shape, the number of points on the connector (dimension), and parameters describing the distribution of grid points along the connector.

The shape is defined interactively by how the user draws the connector. Each connector is made up of one or more segments where a segment is a primitive curve type including those listed below.

- Line (a 3D polyline)
- Curve (a 3D cubic Bezier curve)
- Akima Curve (a 3D cubic Bezier curve with slopes defined via Akima's method)
- Line on DB (same as a Line but constrained to lie on a database surface)
- Curve on DB (same as a Curve but constrained to the database)
- Akima Curve on DB (same as an Akima Curve but constrained to the database)
- Circular Arc
- Conic Section

Mathematically, the segments are represented by cubic, rational, Bezier curves. This underlying similarity allows the segments to be converted with a single Gridgen command from a Line, to a Curve, to an Akima Curve, and even to a generalized Bezier segment as shown in Figure 9. The Bezier segment affords the user the opportunity to edit the slope point of the Bezier control polygon. This flexibility is provided so that the Gridgen user can create connectors with the desired shapes. Editing control points is only one of Gridgen's connector modification commands which are listed below.

- Add Segment
- Insert Segment
- Erase Segment
- Edit Control Points
- Translate
- Stretch
- Rotate
- Project onto DB
- Split
- Join

The user assigns a number of grid points (called the dimension) to each connector by either typing a numerical value or by graphically picking a series of existing connectors and using the sum of their dimensions for the current connector.

The distribution of grid points along the connector is defined by several parameters, some of which are shown in Figure 10. Breakpoints are locations along the connector at which a grid point will lie and at which the grid point spacing will be explicitly defined by the user. Breakpoints also divide the connector into subconnectors. In each subconnector the user controls the number of grid points (subject to the total connector dimension), clustering at each breakpoint, and distribution function type which may be selected from this list:

- hyperbolic tangent
- monotonic rational quadratic spline
- geometric progression
- copied from other connector(s)

Although each of the three attributes of a connector (shape, dimension, distribution) are specified separately, they are all coupled. This allows the user to change any one of the attributes and the other two will be updated *automatically*. This coupling of the connector's attributes allows the user to edit the grid without reworking the whole connector which can make the user much more efficient since creating connectors is the most labor intensive task in Gridgen. This connector updating process occurs whenever the database is modified (as described in the previous section). As will be shown in the following sections, changes to the connectors are propagated *automatically* up the grid component hierarchy to domains and blocks.

DOMAINS

Surface grids (domains) are created by the user's selection of the connectors that comprise the domain's perimeter. [3] Any number of connectors may be used on the perimeter but they must be arranged into four edges such that opposite edges have the same number of grid points (ie, the domain must map into an $I \times J$ computational rectangle). Once the domain perimeter has been defined Gridgen will *automatically* create the surface grid points by applying an algebraic grid method called transfinite interpolation (TFI). This method uses the known grid point data $\vec{r} = [\ x\ y\ z\ ]^T$ on each connector (obtained from each connector's grid point distribution) to interpolate $\vec{r}$ at each interior grid point. Furthermore, when Gridgen detects that all of the connectors used to bound the domain lie on the same database surface the TFI algorithm will be applied *automatically* in the parametric space of the database surface. In this case, parametric TFI uses boundary data of the form $\vec{u} = [\ u\ v\ ]^T$ where u and v are the parametric (domain space) coordinates of the database surface (see Figure 7). This interpolation yields (u, v) coordinates at each interior grid point which are then evaluated in terms of the database surface resulting in $\vec{r} = [\ x\ y\ z\ ]^T$. Parametric TFI ensures that the surface grid points will adhere to the database without any need for projections.

This same type of *auto*-TFI logic is applied when connectors that belong to domains or the domains themselves are modified. Domain modification commands include those shown below.

- Translate
- Stretch
- Rotate
- Project onto DB
- Split
- Join

The new connector shape change is used to re-TFI the domain surface grid *automatically* using whatever TFI method was last applied to the domain by the user. This type of modification, where changes propagate up Gridgen's data hierarchy (see Figure 2) is called forward editing.

Surface grid quality (smoothness, clustering, orthogonality) can be improved by application of Gridgen's elliptic PDE techniques. Specifically, Gridgen solves Poisson's equation using an explicit, point-wise, successive over relaxation (SOR) algorithm with optimal relaxation factors ([4]) subject to user selected control functions. The elliptic PDE method is quite flexible and allows the user to apply a wide variety of attributes, some of which are listed below.

[3] Gridgen can also create a domain by applying a hyperbolic PDE technique to a connector.

- Control Functions
  - LaPlace (smoothness)
  - Fixed Grid (smoothness)
  - Thomas-Middlecoff (clustering) [5]
  - Sorenson (orthogonality) [6]
  - Hilgenstock-White (orthogonality) [7] [8]
- Boundary Conditions
  - Fixed
  - Float
  - Ortho
- Surface Shape
  - free
  - fixed
  - database
- Numerical Solution
  - relaxation factor

One of the most important elliptic PDE attributes is the surface shape. This attribute in combination with the control functions allows the grid points to redistribute themselves in order to obtain the desired grid qualities while also adhering to the desired shape. Of course, the most important shape that users need to maintain is the shape of surface grids constrained to the database. There are two techniques with which Gridgen can make grid points adhere to the database. As was the case for TFI, if Gridgen determines that the surface grid points lie on the database, the elliptic PDEs will be solved in the parametric space (u, v) of the database and the model space coordinates will be obtained from the known surface shape $\vec{r} = f(u, v)$ (parametric technique). The second technique for keeping grid points on the database is a simple projection (conventional technique). Each domain being run in the elliptic PDE solver maintains the database entities to project onto, and the projection orientation (which is computed automatically by Gridgen). The advantage of the parametric elliptic PDE technique is that it's much easy to set-up (no additional attributes) and it's much faster (no projections). Fortunately, Gridgen has *automated* the application of the database surface technique. Specifically, when Gridgen detects that all of a grid point's neighbors (those points used in the finite differences for solution of the elliptic PDEs) lie on the same database surface the code will *automatically* apply the parametric technique, thus making the calculation much more efficient. When domains span multiple database surfaces, only those grid points that lie on or near the seam between the surfaces will require projection. Of course, grid points may migrate across the seam from one surface to another. Figure 11 shows a surface grid that has been created on the surface of the bluff body database spanning two forebody surfaces with notations as to which grid points are solved parametrically and which are solved conventionally.

It is often the case that the precise shape of a surface grid's perimeter (the connector) is not known by the user. This is especially true for connectors shared between two domains. Even when shape isn't important the requirement for smooth variation of grid lines across the connector is required. Rather than forcing the user to choose between 1) manually fine-tuning the connector's shape or 2) being stuck with something inadequate, Gridgen's elliptic PDE techniques provide a boundary condition that allows the connector shape to float subject to the elliptic PDE solution. The float boundary condition treats the grid points on a connector shared between two domains in the same manner as interior domain points; they move subject to the elliptic PDE solution. When the solver is done the original connector shape is replaced by a new shape defined by the refined grid points. An example of this feature is shown in Figure 12.

The user may also defined sub-domains which, as the name implies, are subsets of a domain. Subdomains may then be used with the TFI and elliptic PDE methods to restrict the effects of the methods to a small region of the grid. Subdomains may be thought of as templates that fit over the domain grid and define the region to which changes should be made. Each subdomain saves its own unique combination of elliptic PDE attributes. When grid modifications require re-application of the PDE methods (see previous sections) the user may simply invoke the PDE method without resetting attributes.

BLOCKS

Volume grids (blocks) are created by the user's selection of the domains that comprise the block's perimeter. [4] Any number of domains may be used on the block's perimeter but they must be arranged into six faces such that opposite faces have the same number of grid points (ie, the block must map into an $I \times J \times K$ computational parallelepiped). Once the block perimeter has been defined Gridgen will *automatically* create the volume grid points by applying TFI.

Block modifications, including copy, translate, scale, and rotate, will result in *automatic* propagation of changes down Gridgen's data hierarchy (backward editing) to the domains and connectors.

Gridgen has also *automated* one of the most cumbersome aspects of multiple block grid generation: interblock connectivity detection. As a result of Gridgen's data hierarchy, the code *automatically* detects and maintains both full- and partial-face interblock connections simply by noting that the same domain is used in two blocks. Changes to a domain's grid points by the PDE solver, for example, are *automatically* transferred to both blocks that use that domain. Changes to a domain on a block's perimeter will propagate up Gridgen's data hierarchy and cause *automatic* re-TFI of the volume grid points. Also, when the boundary conditions are exported for use with the analysis software the connectivity data is formatted and exported specifically for the chosen analysis software.

Gridgen also offers an interactive tool for re-dimensioning (changing the total number of grid points) the entire blocking system. This feature allows the user to select any connector in the blocking system and change its number of grid points. That change is propagated *automatically* throughout the system to maintain dimensional consistency (ie, maintain $I \times J$ domains and $I \times J \times K$ blocks).

MISCELLANEOUS

One final *automation* feature deserves mention. Gridgen can import "raw" grid points (from a PLOT3D grid file, for example). When this is done, Gridgen *automatically* creates connectors, domains, and blocks, from the raw grid data. Therefore, the user may then avail him/herself of all of Gridgen's commands, including analysis boundary condition setting and file export. Conversely, the Gridgen user may export the grid points from any connector, domain(s), or block(s) to a PLOT3D file for use in other applications.

SUMMARY

Various features within Gridgen have been described and their unique *automatic* aspects have been emphasized. These features contrast sharply with the fact that Gridgen is not an *automatic* grid generator.[5] It is not the intent of this paper to diminish the many published and unpublished works on automatic grid generation. Indeed, the successful application of CFD in a design environment often depends on configuration - limited procedures that may be applied rapidly and easily by the CFD novice. Instead, this paper and Gridgen itself have been motivated by the desire to emphasize grid quality and software

[4] Gridgen can also create a block by applying a hyperbolic PDE technique to a domain.

[5] We define an automatic grid generator as one which will create a suitable grid given a geometry model and little or no user input.

flexibility. These emphases have led to the development of a visually based, 3D, interactive application environment that *automates* much of the low level grid maintenance tasks, freeing the user to apply his/her judgement as to grid suitability. Gridgen's ability to *automate* much of the process is directly attributable to its data hierarchy in which the curve, surface, and volume grids are maintained as secondary components of connector, domain, and block topology components.

ACKNOWLEDGEMENTS

Gridgen is a product of Pointwise, Inc. GRIDGEN Version 9 was sponsored by NASA Ames Research Center through Computer Sciences Corp. Thanks to Mr. Matthew Blake of NASA and Dr. Jin Chou of CSC. GRIDGEN Version 8 was sponsored by NASA Langley Research Center through Computer Sciences Corp. Thanks to Dr. Robert Smith of NASA and Dr. Jamshid Abolhassani of CSC. GRIDGEN Version 6 was sponsored by the United States Air Force Wright Research and Development Center. Thanks to Dr. Donald Kinsey and Lt. John Seo. All trademarks used herein are the property of their respective owner.

REFERENCES

[1] Steinbrenner, J.P., and Chawner, J.R., "The Gridgen User Manual: Version 10", available from Pointwise, Inc., Jan. 1995.

[2] Ahmed, S.R., Ramm, G., and Faltin, G., "Some Salient Features of the Time - Averaged Ground Vehicle Wake," SAE Technical Paper Series No. 840300, SAE International Congress and Exposition, Detroit, MI, March 1984.

[3] Misegades, K.P., "1988 Automobile Aerodynamics Workshop," Second International Conference on Supercomputing in the Automotive Industry, Seville, Spain, October 1988.

[4] Ehrlich, L.W., "An Ad Hoc SOR Method", **Journal of Computational Physics**, Vol. 44, pp. 31-45, 1981.

[5] Thomas, P.D., and Middlecoff, J.F., "Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations", **AIAA Journal**, Vol. 18, pp. 652-656, 1979.

[6] Sorenson, R.L, "The 3DGRAPE Book: Theory, Users' Manual, Examples", NASA TM-102224, July 1989.

[7] Hilgenstock, A., "A Fast Method For The Elliptic Generation of Three-Dimensional Grids With Full Boundary Control", **Numerical Grid Generation in Computational Fluid Mechanics '88**, ed. by Sengupta, S., et al, Pineridge Press Ltd., Swansea, UK, 1988, pp. 137-146.

[8] White, J.A., "Elliptic Grid Generation With Orthogonality and Spacing Control On An Arbitrary Number of Boundaries", AIAA paper no. 90-1568, June 1990.

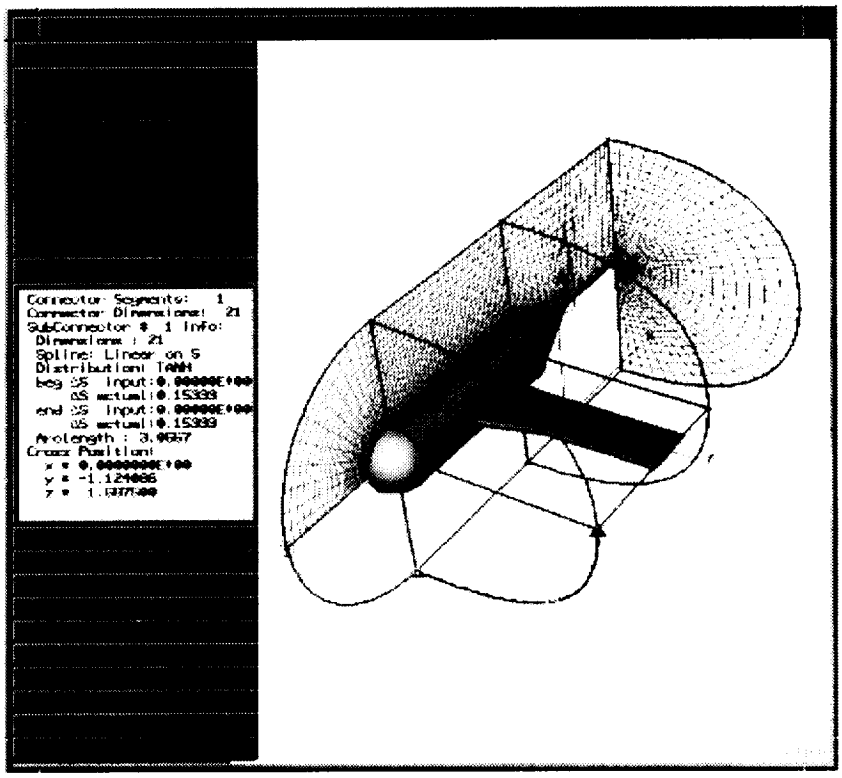

Figure 1: The Gridgen screen.

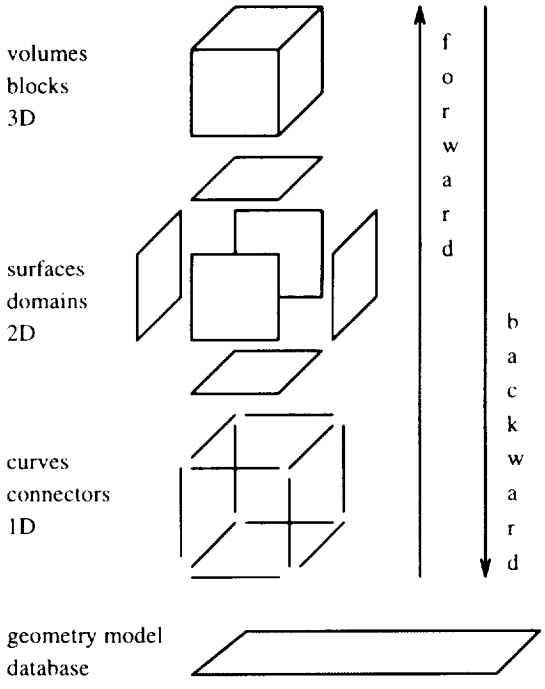

Figure 2: Gridgen's data hierarchy.

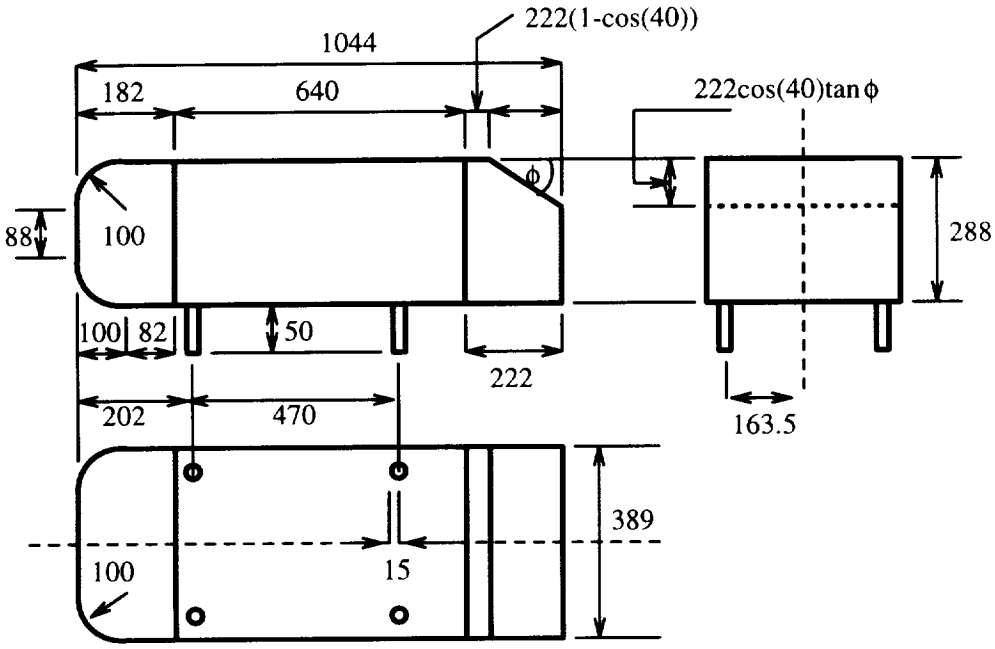

Figure 3: Schematic of a bluff-body, external automotive shape used for demonstration of Gridgen's automation features. All dimensions in millimeters. Adapted from Reference [2].

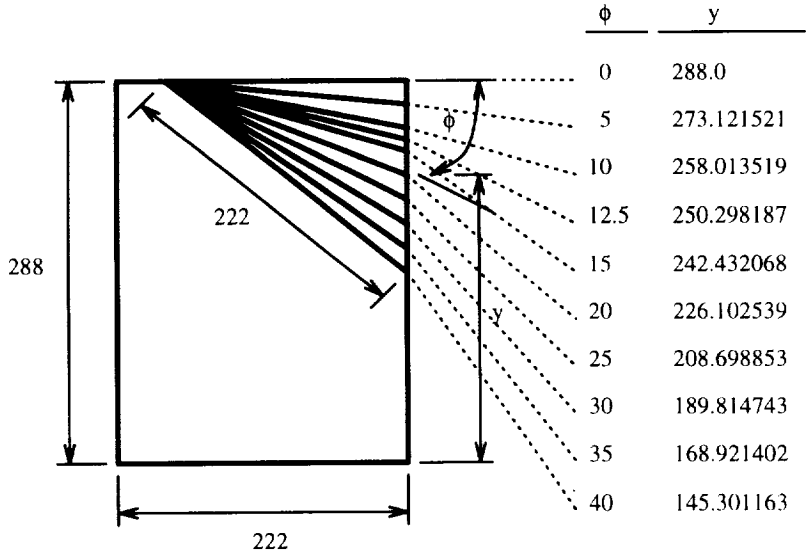

Figure 4: Detailed schematic of the various base angles for the bluff body in Figure 3. All dimensions in millimeters. Adapted from Reference [2].

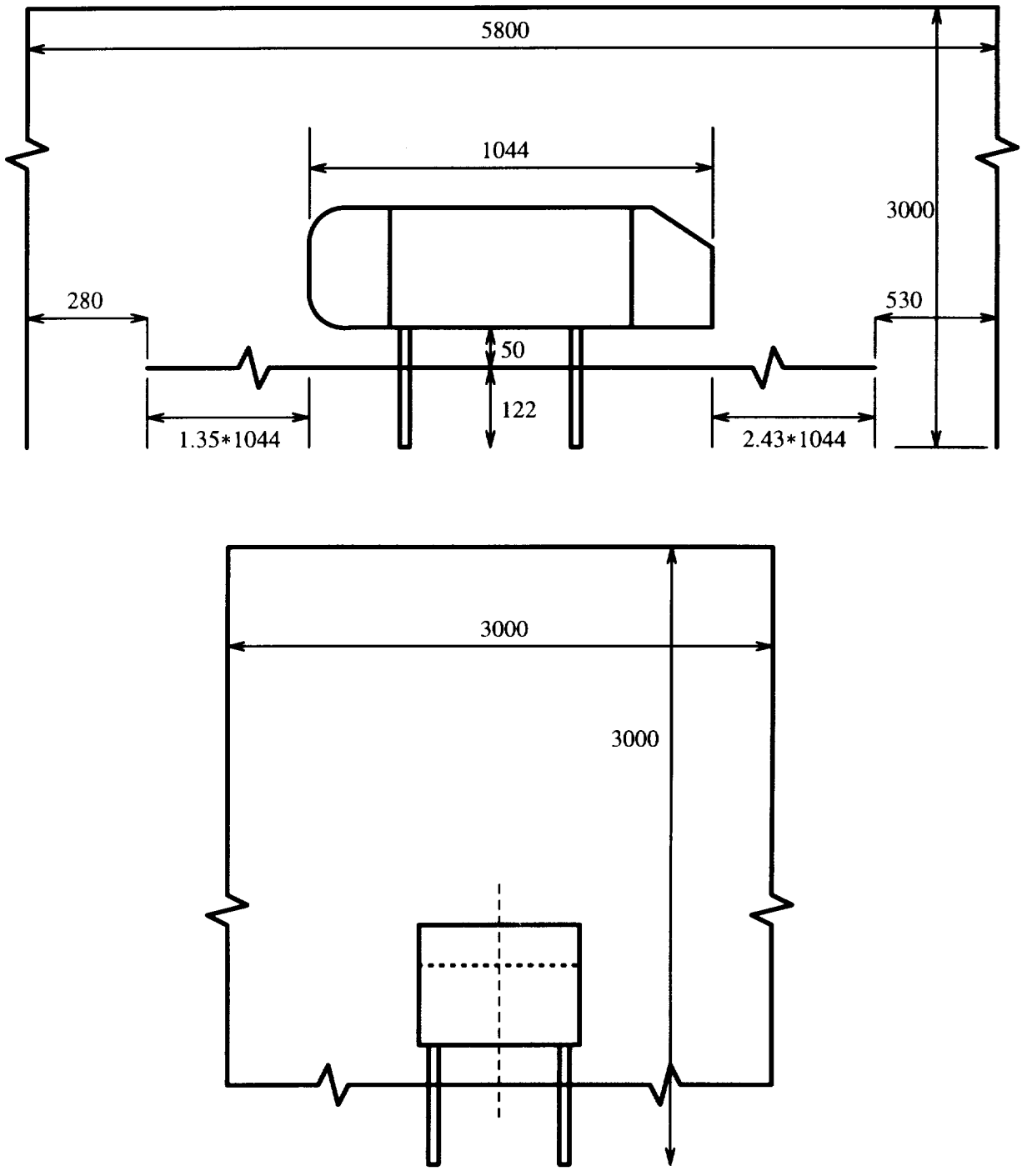

Figure 5: Schematic of the bluff body in Figure 3 installed in the wind tunnel. All dimensions in millimeters. Adapted from Reference [2].

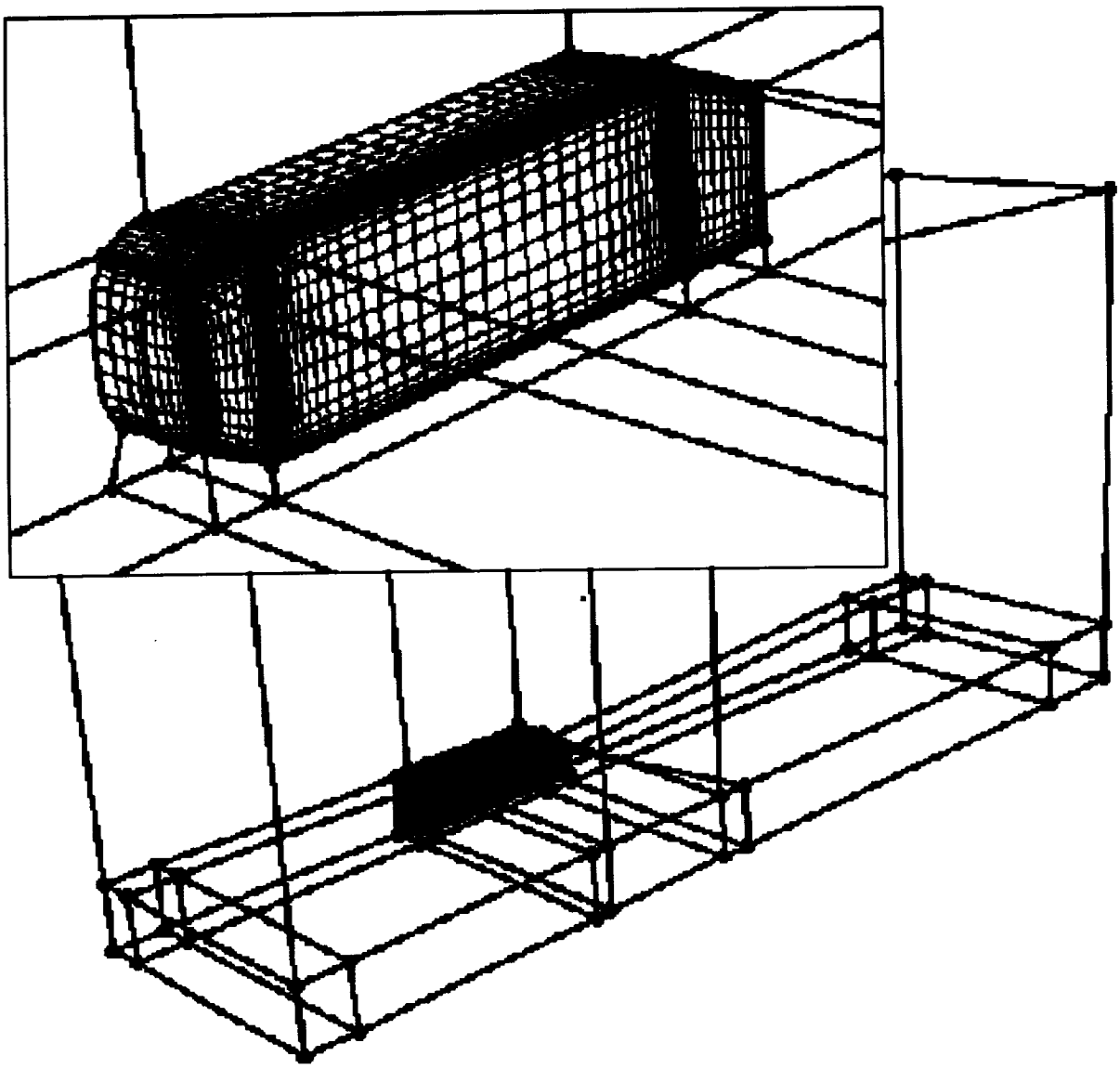

Figure 6: A 15 block grid containing 520,000 grid points was created for the bluff body. A close-up of grids on the surface of the body is also shown.

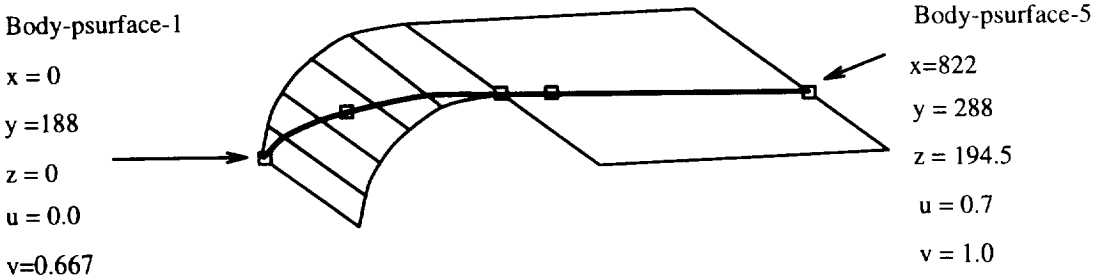

Figure 7: Gridgen maps grid elements to the database via the entity name allowing the underlying shape to be easily changed.

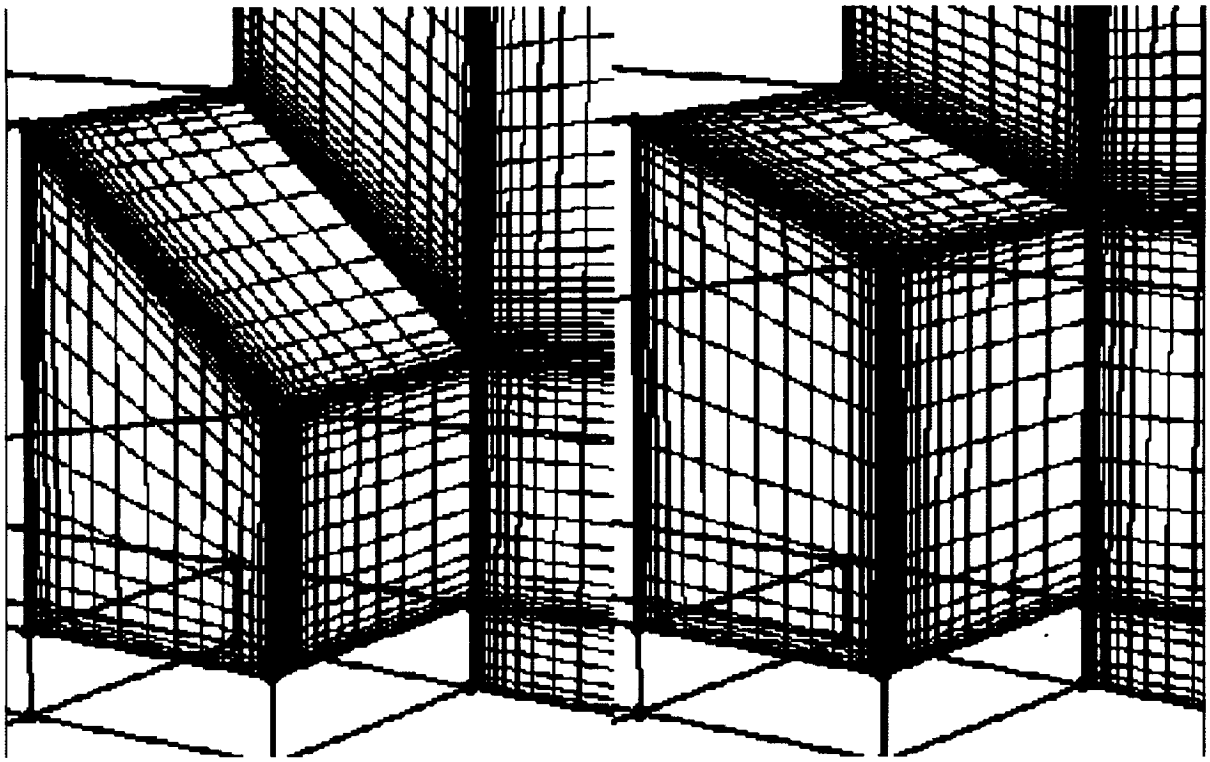

Figure 8: The grid for the 40° base slant angle (left) was changed to the 20° base slant (right) simply by importing a different database file.

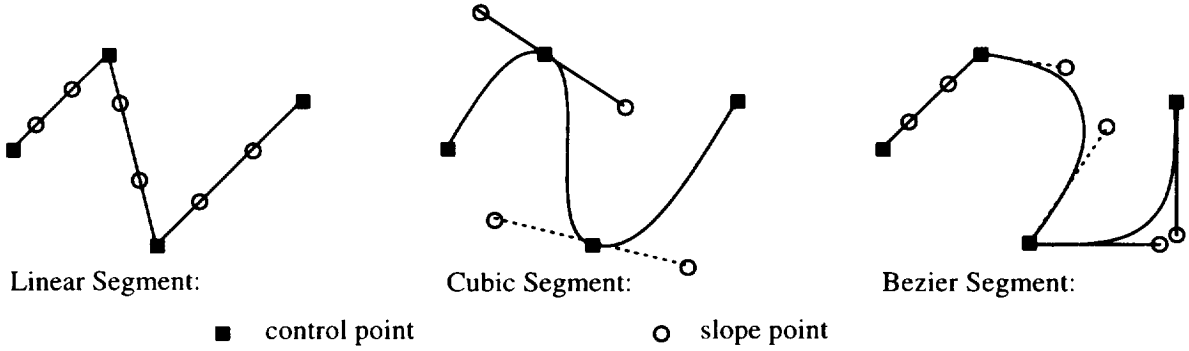

Figure 9: Gridgen's unified geometric foundation allows segments to be easily converted from one type to another. Note that the control points for all three segments are the same.

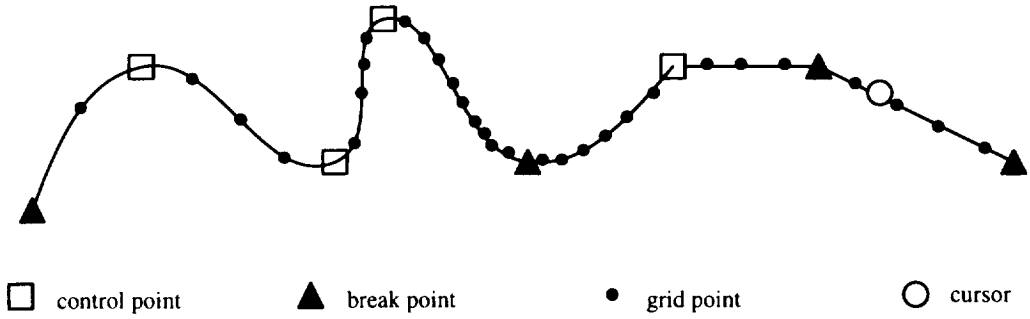

Figure 10: Terminology used in the distribution of grid points along a connector.

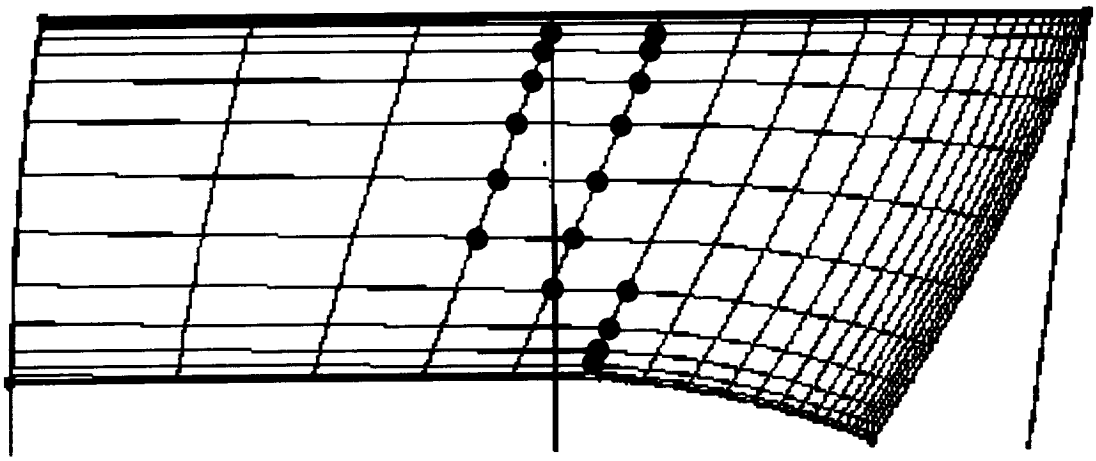

Figure 11: The surface grid on the upper forebody spans two database surfaces (the vertical lines extending beyond the grid are the boundaries of the surfaces). The filled circles denote grid points that are projected onto the surfaces. All other points are maintained on the surfaces parametrically.

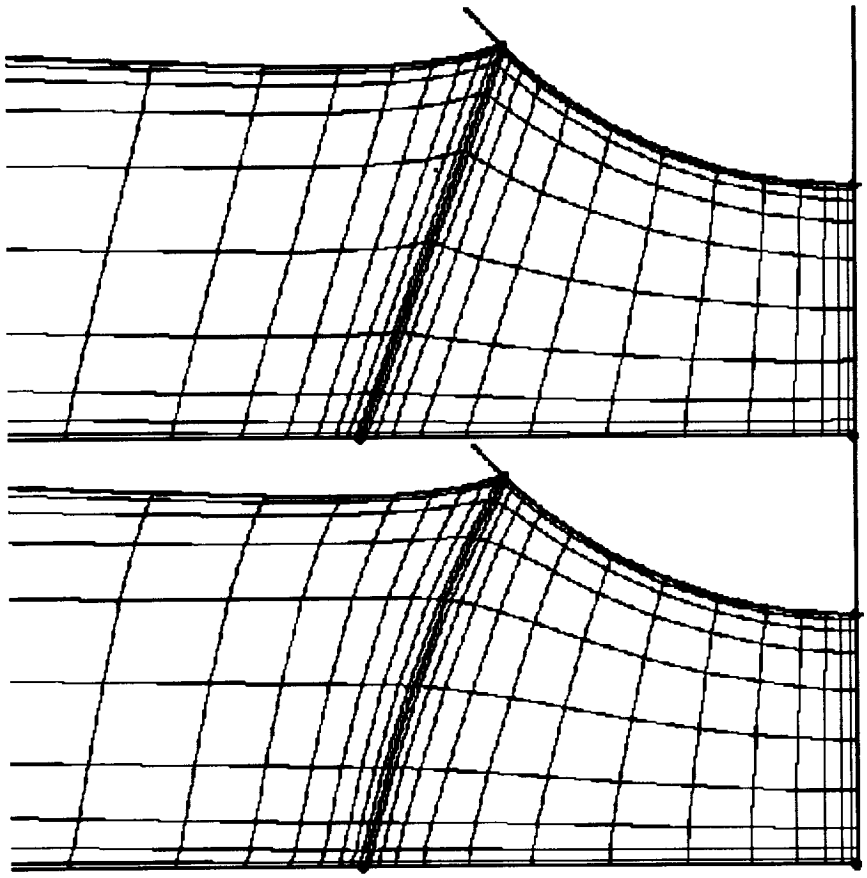

Figure 12: The connector between two surface grids may change shape according to the elliptic PDE solution.