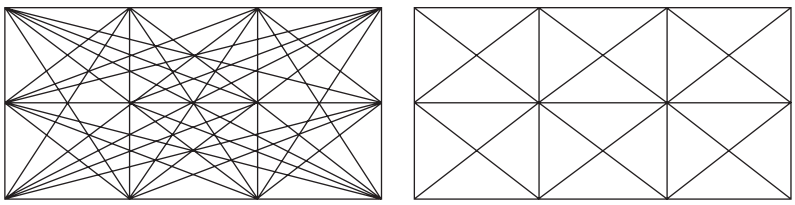
Appendix

Mesh Generation

A1.1 Mesh Generation with Discrete Elements

Discrete elements like the truss, beam or frame elements are simple in topology (two vertices and an edge) and finite element implementation and perhaps yield faster results compared to their continuum counterparts, that is, the triangular and quadrilateral elements in two dimensions, and tetrahedral and octahedral elements in three dimensions. Two kinds of mesh implementations are employed with discrete finite elements. The first is the *full ground structure* wherein each node is connected to every other node in the region by means of truss, beam or frame elements. Full ground structures are used to capture as much of the region as possible (Figure A1.1 a). However, numerous elements intersect or overlap which may be avoided by using a *partial* or *super ground structure* (Figure A1.1 b) that uses an array of elements arranged in a cell (square or cube). In full ground structures, node placement can either be uniform or random though in partial ground structures, it is usually uniform. The two ground structures can also be generated in three-dimensional solids.



(a) (b)

Figure A1.1 Full and partial ground structures

A1.2 Mesh Generation with Continuum Elements

Two of the simplest and most used finite elements in two dimensions are the triangular and quadrilateral elements. Triangular meshes can be mapped or freely generated though in the former, the region to be discretized needs to be triangular. The mapping involves blending of a mesh generated in a *parametric* domain into the real domain, as the parametric boundary blends into the real boundary. Thus, a region $Q_1Q_2Q_3$ may be mapped to a parametric equilateral triangle of unit edge $P_1P_2P_3$ as shown in Figure A1.2. If P is a point in the interior of $P_1P_2P_3$, then

$$P = (A_1/A)P_1 + (A_2/A)P_2 + (A_3/A)P_3 \tag{A1.1}$$

where A_1, A_2 and A_3 are the triangular areas as shown and $A = A_1 + A_2 + A_3$. For a one-to-one map between triangles $P_1P_2P_3$ and $Q_1Q_2Q_3$ if Q is an interior point in the latter, then

$$Q = (A_1/A)Q_1 + (A_2/A)Q_2 + (A_3/A)Q_3 \tag{A1.2}$$

Thus, if P coincides with P_1, $A_1 = A$, and $A_2 = A_3 = 0$ in which case Q overlaps with Q_1. Likewise, one can argue for P_2 coinciding with Q_2 and P_3 coinciding with Q_3. In other words, the boundary of the parametric triangle blends with that of the real one. The notion is to mesh the region $P_1P_2P_3$ using equilateral (regular) triangles of predetermined size and transport the mesh (node points and element connectivity) back to the original domain. For this reason, the method is termed as the *transport mapping method*.

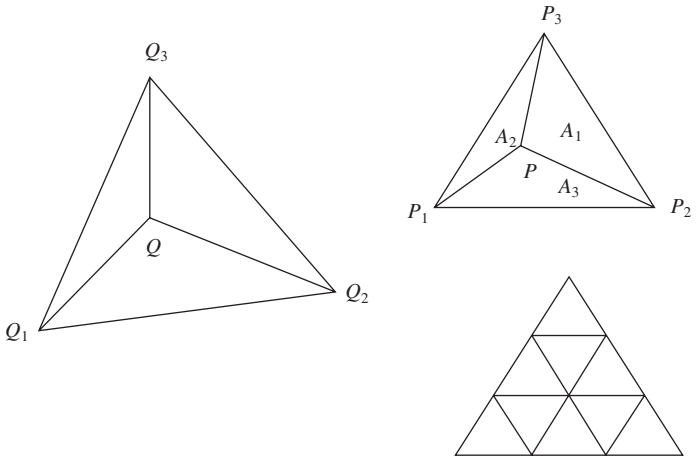


Figure A1.2 Structured triangular meshes

A1.2.1 Unstructured Meshes with Triangular Elements

Any arbitrary complex geometry can be more flexibly filled with unstructured meshes of triangular elements. There exist numerous methods in use for triangulation of generic domains. Most primitive is the *manual mesh generation* wherein the user defines each element by the vertices. The approach is infeasible and time consuming if the number of elements is large. Among the automatic ones are the (a) *advancing front method*, (b) *Delaunay-Voronoi triangulation* and (c) *sweepline method* which are discussed below.

A1.2.2 Triangulation with Advancing Fronts

The advancing front method is used to generate grids having triangular or/and quadrilateral elements. The domain boundaries are initialized as piecewise linear curves with nodes and edges which forms the *front*. As the algorithm progresses, new internal nodes are generated, and triangular and quadrilateral elements are formed at the contour. The front is initialized to the new internal boundary and the algorithm continues until the front is empty, that is, when there exists no internal boundary to be advanced further. A stepwise implementation of the algorithm is provided as follows:

1. The domain boundary is discretised using piecewise linear curves which is initialized as the *front*.
2. The front is updated (edges are deleted and added in the front) as triangulation proceeds.

3. Two consecutive edges are considered from the front for triangulation. For angle α between the two edges, three possibilities may be identified

   (i) $\alpha < \frac{1}{2}\pi$: the edges (*bc* and *cd*) form the part of the single triangle created (Figure A1.3a).
   (ii) $\frac{1}{2}\pi \leq \alpha \leq \frac{2}{3}\pi$: an internal point *i* and two triangles are generated (Figure A1.3b).
   (iii) $\alpha > \frac{2}{3}\pi$: a triangle is created with edge *ab* and an internal point *i*

4. The internal nodes are positioned to be *optimal* in that the element with such a node is as regular as possible. For case in Figure A1.3 (b), the internal node *i* is generated on the angular bisector at a distance dependent on the lengths of edges *bc* and *cd*, that is

$$| d_{ic} | = (1/6)(2 |d_{bc} | + 2| d_{cd} | + | d_{ab} | + | d_{de} | \tag{A1.3}$$

   for angles β and γ between $(1/5)\pi$ and $2\pi - (1/5)\pi$ with $(1/5)\pi$ an empirically chosen value. Otherwise, construction in Step 1 may be incorporated. For case in Figure A1.3 (c), a triangle as equilateral as possible may be formed.

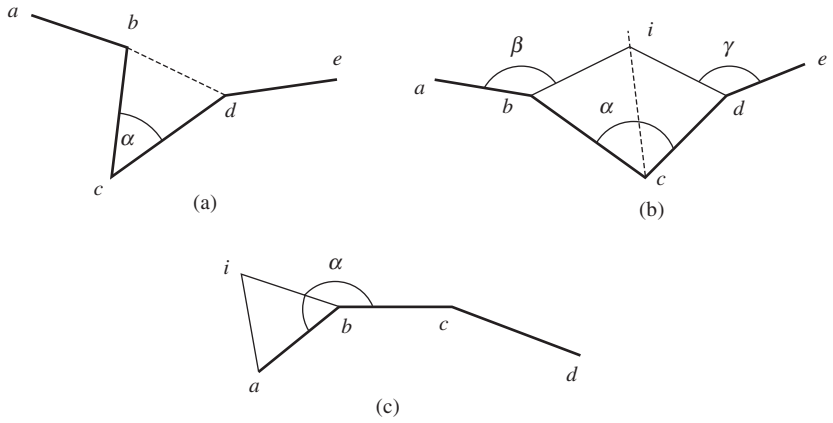


Figure A1.3 Advancing front method with various internal angles between consecutive edges of the front

5. It must be ascertained for an internal point that

   (i) it must be a part of the domain, that is, it must be inside the primary contour of the domain and outside the contour of holes that may be present
   (ii) there should not be any existing node within a certain proximity of the internal point. If there is, the node becomes the internal point
   (iii) the internal point should not be contained within the existing triangular element(s).

6. The node and element connectivity lists are updated
7. A new front is formed by

   - deleting the edges from the present front belonging to a triangle created from the present front
   - adding those edges of triangle(s) created which are not common to two elements, and to the two fronts.

8. The procedure is continued unless the front is empty. Figure A1.4 shows a few steps of the advancing front method.

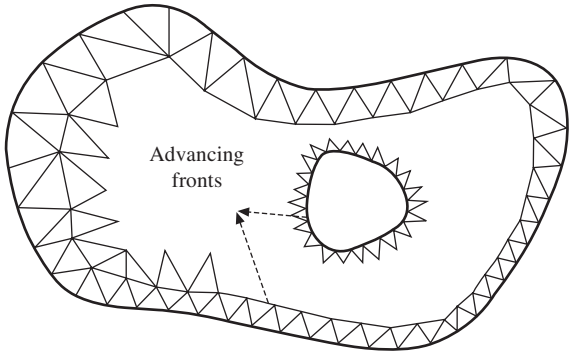


Figure A1.4 Schematic of the advancing front method

The advancing front method can be extended to three dimensions in that the front would comprise a set of triangles at the surface of the solid to start with and interior points would be generated to create almost regular tetrahedral elements.

A1.2.3 Delaunay Triangulation

The method is one of the most widely used as it yields efficient triangulation with relatively easy implementation and provides better results for most applications. Delaunay triangulation is the geometric *dual* of Voronoi tessellation also known as Theissen or Dirichlet tessellation in that one can be derived from the other. For *N* points in a plane, Voronoi tessellation divides the domain into a set of polygonal regions, the boundaries of which are perpendicular bisectors of the lines joining the nodes (Figure A1.5 a). Each polygonal region contains only one of the *N* points.

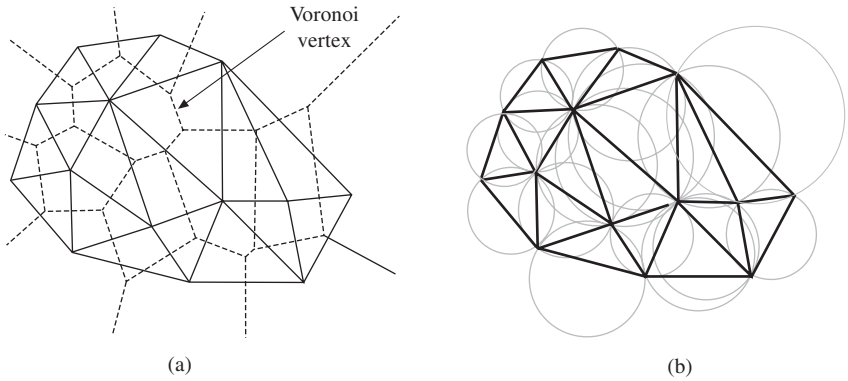
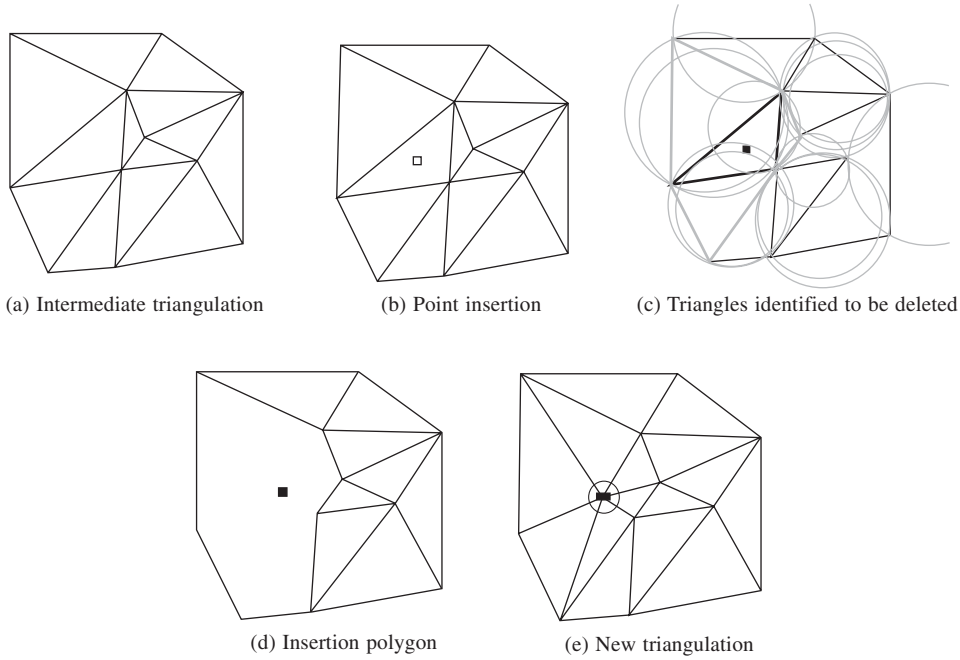


Figure A1.5 (a) Delaunary triangulation (solid) and Dirichlet tessellation (dashed) and (b) Delaunay triangles with circumcircles

As per the Delaunay criterion, in a valid triangulation, the circumcircle of each triangle does not contain any node of the mesh. By construction, each Voronoi vertex is the circumcenter of a Delaunay triangle. Delaunay triangulation being the geometric dual of Voronoi tessellation, many methods have been developed to arrive at the former using the latter, though many methods for direct triangulation are also in use. A simple and widely used Watson's algorithm for Delaunay triangulation is briefed

here. The algorithm starts by forming a super-triangle that encompasses all the given (boundary) points of the domain. Initially, the super-triangle is flagged as incomplete and the algorithm proceeds by incrementally inserting new points[1] in the existing triangulation. A search is made for all triangles whose circumcircles contain the new point. Such triangles are deleted to give an *insertion polygon* and a new set of triangles is formed locally with the inserted point. This process is continued until point insertion is accomplished and thereafter all triangles having the vertices of the super-triangle are deleted. Figures A1.6 depict the implementation of the Watson's algorithm.



(a) Intermediate triangulation (b) Point insertion (c) Triangles identified to be deleted

(d) Insertion polygon (e) New triangulation

Figure A1.6 Watson's algorithm: intermediate steps of point insertion, local deletion of triangles to form the insertion polygon and new local triangulation

In three dimensions, a super tetrahedron may initially be formed to encompass all boundary nodes. At an intermediate step, a point may be generated in space and checked whether it lies within the circumspheres of the existing tetrahedra. Such tetrahedra may be deleted to result in an insertion polyhedron within which new tetrahedral elements may be formed.

A1.2.4 Quadtree Approach

For a two dimensional domain approximated using polygonal discretization, a *quadtree* grid is constructed as shown in Figure A1.7 (a) in the following manner.

1. A square cell of minimum size is formed to contain all contour points
2. The quadtree approach consists of cell *splitting* involving each cell (parent) to be divided into four (children) cells (Section 9.6.1).

[1]Point insertion itself may be iterative to ensure that the resultant triangles formed are almost regular.

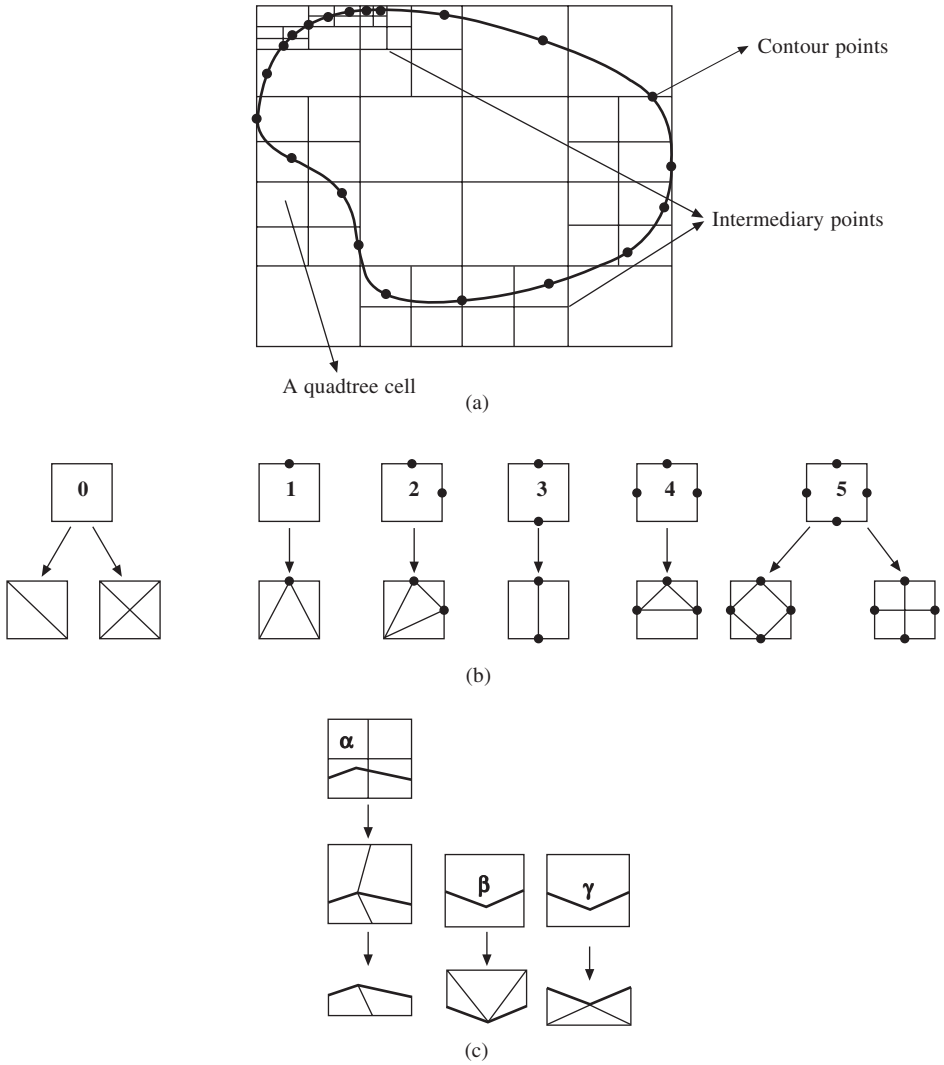


Figure A1.7 Triangulation using the quadtree approach (a) a quadtree grid including the domain such that at most one contour point lies on or within a cell (b) patterns 0, . . . ,5 for cells with the intermediary points and their discretization schemes (c) internal cells intersecting with contour (dark lines) and discretization based on how the contour point is located and which region within the cell is inside the domain

3. In a recursive process, cell splitting is performed such that each cell contains at most one contour point. Thus, for a fine mesh around a contour, the latter should be approximated using more nodes to have many children cells encompassing the contour. This, in a way, serves to control the element size around a contour.
4. Once splitting is over, each cell can be analyzed as follows:

   (a) External cell: Any cell not within the domain and also not containing any segment of the contour is not of interest and is eliminated

(b) Internal cell not intersecting with the contour can be of six kinds:

(i) those without any intermediary point (Quadtree node where cells from two different levels of decomposition meet) on its sides (pattern 0)
(ii) those with one intermediary point on one side (pattern 1)
(iii) those with an intermediary point each on two consecutive sides (pattern2)
(iv) those with an intermediary point each on non consecutive sides (pattern 3)
(v) those with an intermediary point each on three sides (pattern 4)
(vi) those with an intermediary point on each of their sides (pattern 5)

Such cells can be discretized as shown in Figure A1.7 (b). An internal cell not including any contour point produces a quadrilateral that can be split into triangles if its sides do not include any intermediary point (pattern 0), or is split into triangles or possibly quadrilaterals for patterns 1, . . . 5.

(c) Cells intersecting with the contour can be as follows:

(i) contour point included in a cell is close to a vertex of the cell (pattern α in Figure A1.7c).
(ii) contour point is close to the mid point (or *clearly* internal) of the cell (patterns β, γ, etc. in Figure A1.7c).

The intersection of the contour and the sides of the cell are created. A partitioning of the cell is defined where only the part internal to the domain is retained. The final contour of the mesh is created at this stage.

Element formation in the final mesh is done as a consequence of the enumeration of different patterns possible. Once the final mesh is obtained, the *regularization* of the internal points is then performed. Internal points are the vertices of the cells excluding those on the contour. Regularization or *mesh smoothing* may be performed such that for an internal point, its neighboring points are determined and their *barycenter* or the geometric center is computed, and that internal point is repositioned to this geometric center. Further, to avoid *flat elements*, diagonal swapping between two neighboring triangles can be applied iteratively. For three-dimensional mesh generation, an *octree* type cell decomposition may be incorporated. Cell patterns like those in Figure A1.7 may be categorized and identified, and tetrahedral elements may be generated.

A1.2.5 Meshes with Quadrilateral Elements

Noting that two neighboring triangular elements may be combined to form a quadrilateral element, the result of the triangulation algorithms may be used to generate grids exclusively comprising of the four-noded elements. A goal of triangular-to-quadrilateral mesh conversion is to maximize the number of adjacent triangular pairs and minimize the number of triangular elements in the process. The adjacent triangles may be selected based on how best (close to a square) a quadrilateral element may be formed, and then *fused* at their common diagonal. With such algorithms, not all triangles may be able to participate in quadrilateral formation and as a result, some isolated triangles may appear in the mesh. As the goal is to have a mesh exclusively of quadrilateral elements, a *swapping* scheme may be employed to swap the edges of quadrilaterals lying between two isolated triangles until the triangles become adjacent. Another way may be to subdivide or swap the edges of isolated triangles until they locally get converted to all-quadrilateral elements like in Figure A1.8.

Of the non-conversion or direct approaches for quadrilateral mesh generation is a semi-automatic approach called the *multi-block* method which is based on mapped meshing. The domain to be

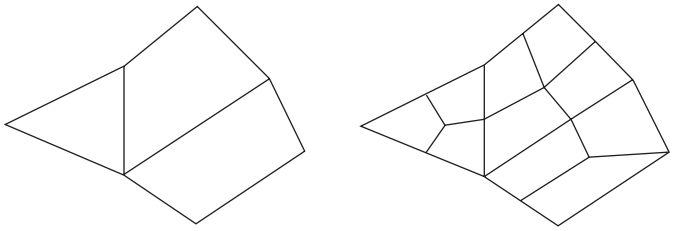


Figure A1.8 Conversion of a quad-dominant mesh to all-quadrilateral mesh

discretized is divided into blocks which are then meshed separately using parametric mapping (Figure A1.9). Though mesh of individual blocks may be regarded as structured, the overall mesh is unstructured because of the likewise decomposition of the block. Human intervention is involved in manually decomposing the geometry into blocks though some algorithms attempt to automate the geometry decomposition using medial-axes and medial-surface techniques with some heuristics. However, automatic decomposition of a complex domain into mappable regions seems non-trivial. Significant disadvantage of the mapped meshing algorithms is the limited flexibility of the mesh size control. To ensure mesh conformity at the common block boundaries, the same division must be used in neighboring blocks.

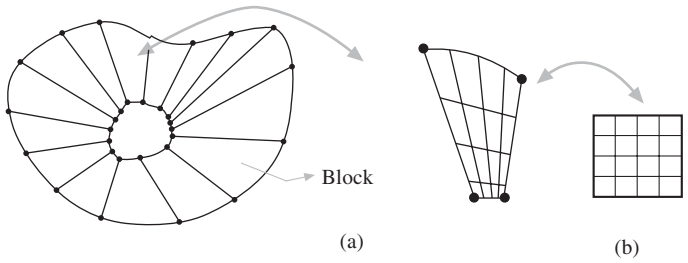


Figure A1.9 Semiautomatic quadrilateral mesh generation: (a) decomposition into blocks and (b) mesh generation in a block using mapping

A1.3 Mesh Evaluation

We note that the applicability and accuracy of the finite element analysis is dependent to a large extent on the validity and quality of the elements generated in a mesh. Of the various criteria in use for mesh quality, some for planar/surface meshes are

1. The variation in the element area should not be large. That is, the ratio of the area of the largest/smallest element to all the immediate neighbours should not be drastically low/high.
2. Elements, especially the continuum type, should be as regular in shape as possible. It is desired for triangular elements to be possibly close to equilateral triangles, and for quadrilateral elements to be of square shape. A measure called the *aspect ratio* for elements should be as close to 1 as possible. For triangular elements, the aspect ratio is defined as the ratio of the circumradius of the triangle to twice its inradius. Note that the aspect ratio of an equilateral triangle is 1.
3. The ratio of the largest to the smallest edge/angle (may also be regarded as the aspect ratio) of the element should be close to 1.
4. No two elements in a mesh should intersect. (Full ground structure with discrete elements is an exception).

Suggested Projects

Project 1

Find the configuration of a set of rigid bodies that satisfy a set of geometric constraints. Take an example of a four-bar mechanism and simulate graphically the system based on the vector loop method. Also plot the locus, velocity and acceleration of the point on the coupler.

Project 2

An object B of drop shape is tied to a rope fixed at O (0, 0, 10) and revolved around the Z-axis such that the apex A traces a circle of radius 5 on Z = 0 plane counterclockwise as shown in Figure P1. A point P initially located at (5.9688, 0, – 1.9377) also traces counterclockwise a circular path of radius 5.9688 about Z-axis on the plane Z = –1.9377. At time $t = 0$ the point P and the apex A of the object B lie on the XZ plane. Refer Figure P1 for all dimensions related to the configuration. Assume that the object B revolves with a constant angular velocity $\overline{\omega}_b$ and point P is revolving with an initial angular velocity $\overline{\omega}_p$ and with a constant angular acceleration α_0. The point P which is inside the object B at time $t = 0$ will exit the object B's volume after some time instance t_s and then reenter the drop after time instance t_e.

A 3D object's surface can be approximated by a set of triangular patches. The first order representation of this kind is a very useful tool for computer display as well as computer aided manufacturing. An industry standard for such a representation is STL. An STL file contains a series of triangular patch data consisting of normal information and coordinates for the three vertices. A typical ascii STL file appears as:

```
solid Sample
    facet normal –0.36970 0.27086 –0.88880
    outer loop
      vertex 122.91010 27.04771 182.30157
      vertex 101.78409 25.09600 190.49422
      vertex 101.80428 38.64565 194.61511
    endloop
    endfacet
    facet normal –0.30950 0.36777 –0.87690
    outer loop
      vertex 122.91010 27.04771 182.30157
      vertex 101.80428 38.64565 194.61511
```

```
            vertex 119.99031 40.82108 189.10866
        endloop
        endfacet
        :
        :
        :
        :
endsolid Sample
```

Create an STL file of the object B of drop shape (use any of the solid modeling packages to create a drop (unit radius half sphere and a cone of height 3 units mounted on the circular face, export the geometry in STL file format) with apex A at origin and the major axis aligned to Z-axis. Code an algorithm in MATLAB to do the following. Import the STL file and display the same (help: use *trisurf* function and set 100% transparency for the faces). Apply the necessary initial transformation to the given object so that the drop assumes the orientation as depicted in Figure P1 for time $t = 0$. Apply then the necessary transformation for each time step so as to simulate the motion of object B as well as point P and display the motion as an animated sequence from time $t = 0$ to $t = t_e$. During the animated sequence show the point P with different colors depending on the status whether the point is inside/on or outside the object B. Do not analytically compute t_s and t_e but simulate the motion and find position and orientation of object B and point P for each unit time step and perform a PMC (point membership classification) query for point P in object B.

A small note on how to evaluate whether a point is inside/on an object or not. This is also referred to as PMC (point membership classification) and is one of the most important computations in geometry. The logic described here holds well for this problem only. The object under consideration is convex and also the relative motion between the object and the point is simple. Let P_{xy}, P_{yz} and P_{zx} be the orthographic projections of the point P on to the principal planes. B_{xy}, B_{yz} and B_{zx} be the projections of the object B on to principal planes. The projections B_{xy}, B_{yz} and B_{zx} will be polygons with triangular mesh for a triangulated object as is the present case. Find the bounding polygon for each of the projections as BP_{xy}, BP_{yz} and BP_{zx}. The bounding polygon is the convex hull of the projected vertices in this case since the object drop is convex (help: use *convhull function*). Now perform a 2D PMC for point P_{xy} in BP_{xy}, P_{yz} in BP_{yz} and P_{zx} in BP_{zx}. (help: use *inpolygon* function). The point P is outside the object B if for any one of the three 2D PMC the answer is "out". Else the point is inside/on the object.

Interested readers may refer to text on computational geometry for the *ray-tracing* algorithm, which is a generic PMC but is algorithmically as well as computationally more involved.

The program is to be designed so that the user imports the data, selects the constant angular velocity for the object B, initial angular velocity as well as constant angular acceleration for point P. Use Graphical User Interface (GUI). Report also the time values t_s and t_e in a textbox in the GUI. The program should be self-explanatory (use adequate comments so as to follow the code). Perform two simulation runs, Case 1: $\overline{\omega}_b$, $\overline{\omega}_p$ and $\alpha_0 = 0$, Case 2: $\overline{\omega}_b$, $\overline{\omega}_p$ with positive α_0.

Project 3

Develop a program to generate automatic cutout for tailoring a simple men's shirt. The program shall input the basic feature dimensions such as chest diameter, arm length, wrist diameter, shoulder width, collar diameter and produce the required cut plans as drawings constituting of lines and Bézier curves. The first step is to develop a feature graph constituting the various feature elements (in this

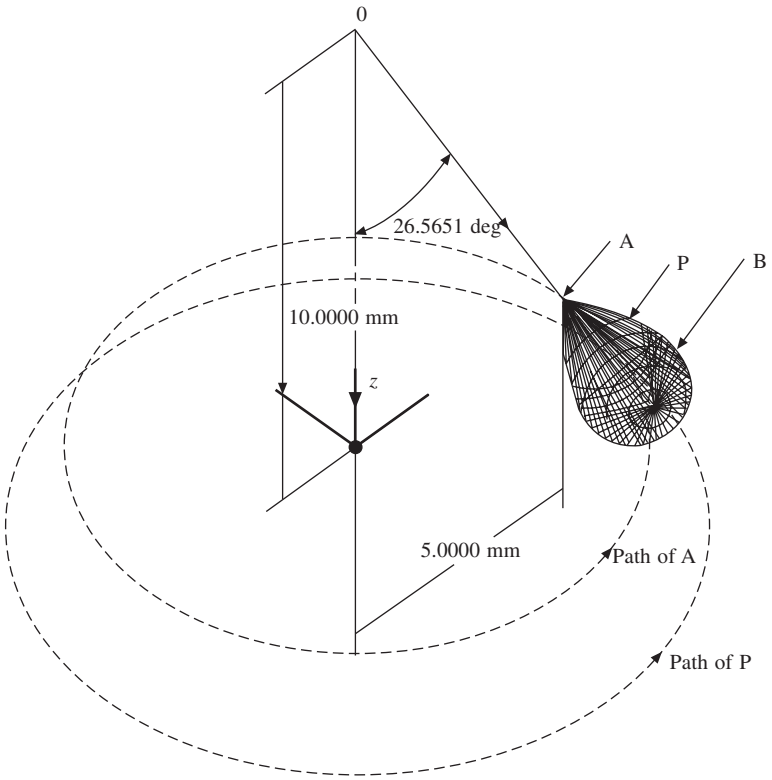


Figure P1. The configuration of the object "Drop" and point P at time $t = 0$

case the cloth cuttings required for stitching the shirt) and then developing the relation between the geometric parameters required to draw these cutting in terms of the user specified feature dimensions.

Project 4

Develop an algorithm to design generic B-spline curves and surfaces. The Input parameters shall be the order, number of control points and their co-ordinates. Write functions to calculate the knot vectors, basis functions and finally the function, for calculating points on a B-spline to display. Also for a parameter value u on the curve, calculate the position vector, tangent and curvature.

Project 5

Develop algorithms for designing surfaces of revolution and sweep surfaces. The Input parameters shall be the order, number of control points, their coordinates for the B-spline curve and a vector defining the axis of revolution or the direction of sweep as the case may be. Use the code developed in Project 4 to design a B-spline curve to be used as curve to be revolved or swept. For simplicity, assume the curve to be lying on the XY plane. In case of surface of revolution, the axis of revolution is the Y-axis and in case of a sweep surface, the direction of sweep is the Z-axis. Also for parameter values 'u' and 'v' on the surface, calculate the position vector, tangents and curvatures.

Project 6

Design a wine glass and a speed breaker bump surface as shown in Figures P2 and P3 using the code developed in Project 5 and show 3D display.

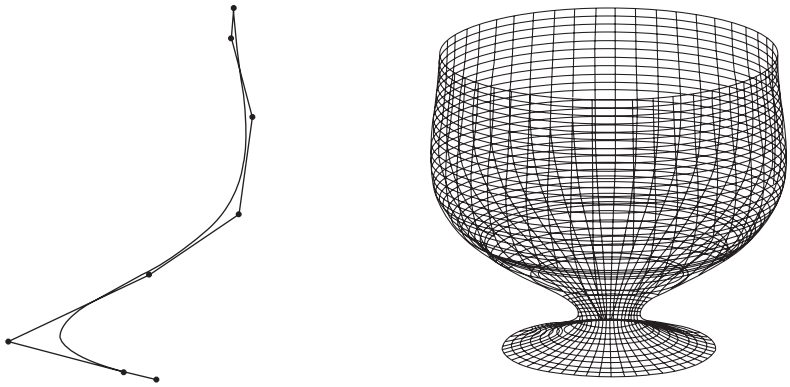


Figure P2 **A B-spline and the wine glass generated by revolving the same about Y-axis. The curve is a cubic uniform B-spline with 8 control points. Assume suitable locations for them.**

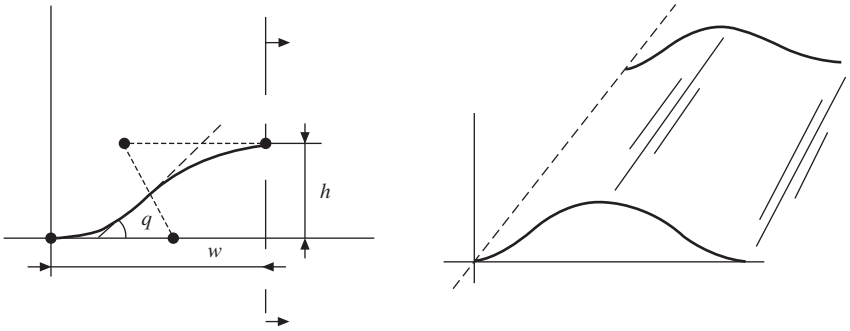
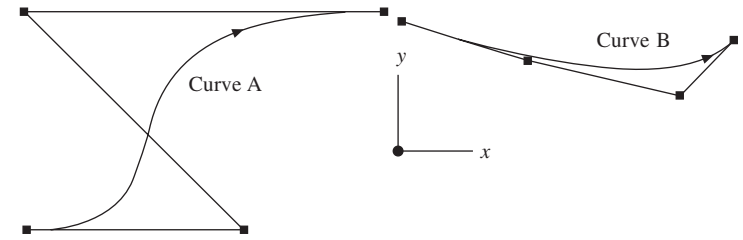


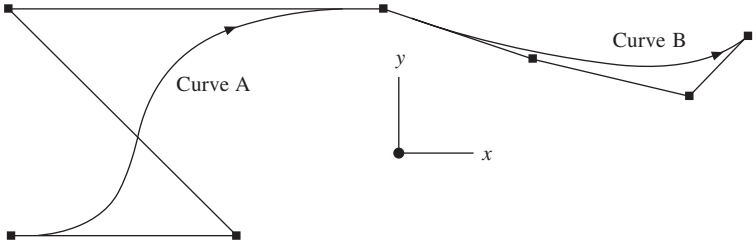
Figure P3 **Cross section profile and the corresponding sweep surface. Input parameters for the cross section profile are (w, h and q). First find the position vectors r_0, r_1, r_2 and r_3 and then proceed.**

Project 7

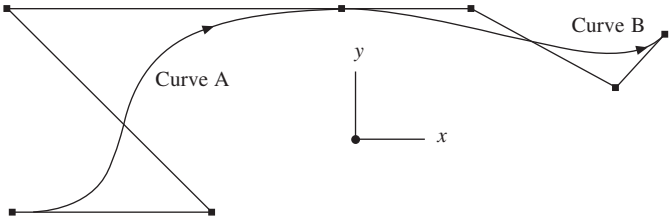
Create a software to design two cubic Bézier curves and later specify the required continuity to get the final blended shape. The user should be able to design the curves by providing the location of control points (control polyline) preferably by mouse and not by entering the values through key. Once the curves are created, the user specifies the required continuity for blending [C^0 (position continuity), C^1 (tangent continuity) or C^2 (curvature continuity)]. Let the curves be A and B. Keeping the curve A as the reference, modify curve B such that curve B blends with curve A with the required continuity. Now measure the positional difference on curve B before and after blending and plot the difference as a function of the parameter u. An illustrative example on Bézier curves is shown in Figure P4 to give an idea of what may be expected as an out come.



(a) Curves before blending



(b) Curves after Curve B has been blended with curve A with C^0 continuity



(c) Curves after Curve B has been blended with Curve A with C^1 continuity

Figure P4 Illustrative example on Bézier curves.

Project 8

Develop a surface modeler to design an automobile hood. The feature representing the automobile hood can be approximated to a patch layout as illustrated in Figure P5. The feature constitutes of three primary surfaces, three quadratic fillet surfaces and one triangular Bézier patch. The whole set when mirrored about the symmetry plane produces the complete hood. The primary surfaces are to be modeled as bicubic hermite/Bézier/B-spline surface patches. In your software include all options for specifying the type of surface and then the required parameters. The secondary surfaces should be computed based on the primary surfaces. Note that they can be modeled as quadratic fillet surfaces and they need not be of constant radius as is occurring between primary surface one and two. Model the corner formed by fillet surfaces by a triangular quadratic Bézier patch. On the mirror plane at least first order continuity is required on the hood surfaces.

The software should be interactive so that the user can dynamically modify the surfaces till satisfactory results are obtained.

Project 9

Develop a software to model a 2D polygonal object in terms of its polytree. Use quadtree decomposition. The software may require to input the vertex list defining the object boundary and the number of

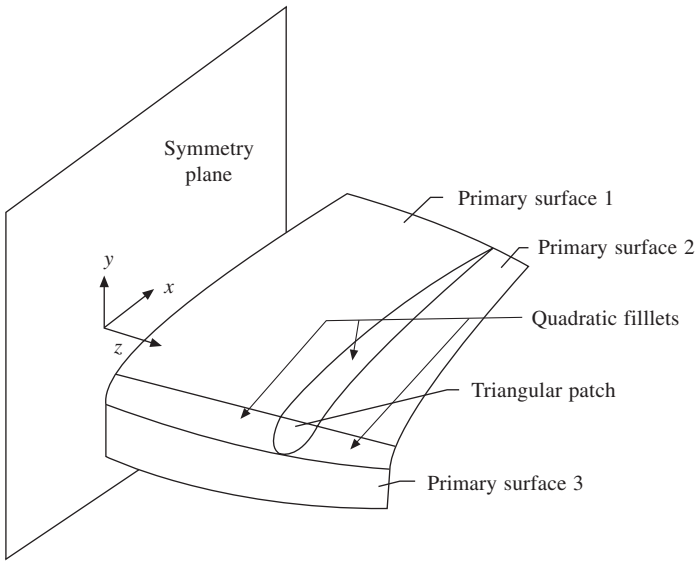


Figure P5 A scheme of surface patches for the automobile hood

levels of decomposition. The output shall be the quadtree data structure to the required levels. One would need a robust point membership classification algorithm for 2D as a pre-requisite.

Project 10

Develop a modeling package incorporating various manufacturing operations required for sheet metal applications. The software should model various operations like stamping, bending, welding and sweeping. Develop a proper data structure, graphical user interface and rendering.

Project 11

Blank nesting is an important job in the tool and die industry. Any effort design stage to minimize scrap be selecting interlocking figures or by using careful layouts may result in substantial material savings. Develop a two-dimensional nesting algorithm for nesting 2D polygonal objects. Refer to the following literature:

Nee A.Y.C., Computer aided layout of metal stamping blanks, Proceedings of Institution of Mechanical Engineers, 1984, Vol. 1, 98B, No. 10.

Prasad, Y.K.D.V. and Somasundaram S., CASNS: A heuristic algorithm for the nesting of irregular-shaped sheet-metal blanks, Computer-Aided Engineering Journal, 1991.

Project 12

This project requires the study of how an expert taylor, with minimum number of measurements on the human torso, creates the shirts, trousers and other apparels. The cut on the cloth-piece are free-form curves optimally made so that various pieces can be accomodated on the standard width of the cloth price.

Create a software to help the taylor in preparing a layout for various types of apparels.

Project 13

Many engineering structures are made up of simple elements like beams, trusses and plates. Create a finite element software to help in the static analysis (stress, deflection) of simple structures.

Project 14

Vibration analysis of beams, shafts and uniform plates and simple spring/dashpot type machine parts are often desirable. Finite element analysis can help the designer in carrying out a simulation. Create software modules for some simple elements with material properties, boundary conditions, force and frequency magnitudes etc. as user defined inputs.

Bibliography

1. Baer, A., Eastman, C. and Henrion, M. *Geometric Modeling: a Survey*", Computer Aided Design, 1979, **11**, 5, pp. 253–272.
2. Barthold, Lichtenbelt, Randy, Crane, Shaz Naqvi. Introduction to Volume Rendering, Prentice-Hall, New Jersey, 1998.
3. bø K., Lillehange F.M. CAD Systems Frame Work, North-Holland Publishing Company, Amsterdam, 1983.
4. Brunet, P., Hoffman, C. and Roller, D. New concepts and approaches in various topics of computer-aided design. CAD tools and algorithms for product design, Springer-Verlag, Berlin Heidelberg, 2000.
5. Charles, E. Knight. A Finite Element Method Primer for Mechanical Design, PWS Publishing Company, Boston, 1994.
6. Charles, M. Foundyller, CAD/CAM, CAE, The Contemporary Technology, Daratech Associates, USA, 1984.
7. Charles, S. Knox, Engineering Documentation for CAD-CAM Applications Marcel Dekker Inc., New York, 1984.
8. Choi, B.K., Surface Modeling for CAD/CAM, Elsevier, Amsterdam, Netherlands, 1991.
9. Cohen, E., Riesenfeld, R.F. and Elber, G., Geometric Modeling with Splines A.K. Peters Ltd., MA, 2001.
10. Cook, R.D., Malkus, D.S. and Plesha, M.E. Concepts and Applications of Finite Element Analysis, John Wiley & Sons (ASIA) Pte Ltd, 1989.
11. Cormen, T.H., Leiserson, C.E. and Rivest, R.L. Introduction to Algorithms, MIT Press, Cambridge, MA, 1990.
12. Cox, M.G. Practical Spline Approximation. In: Turner, P.R. (Ed.) Topics in Numerical Analysis, Springer, USA, 1981, pp. 79–112.
13. Coxeter, Harold S. Macdonald, Regular Polytopes, Macmillan, 1963.
14. Daniel, L. Ryan, Computer Aided Graphics and Design, Marcel Dekker Inc., USA, 1979.
15. David, H. von Seggern, CRC Handbook of Mathematical Curves and Surfaces, CRC press Inc., Florida, 1990.
16. David, Salomon, Computer Graphics & Geometric Modeling, Springer-Verlag, New York, 1999.
17. de Boor C. 1972, On Calculating with B-splines, Journal of Approximation Theory, Vol 6, pp 50–62.
18. de Boor C. A Practical Guide to Splines, Springer, 1978.
19. Deb, K. Optimization for Engineering Design-Algorithms and Examples, Prentice-Hall of India Private Limited, New Delhi, 1996.
20. Dierck'x P., Curve and Surface Fitting with Splines, Oxford University Press, Oxford, 1993.
21. Ellis, T.M.R. and Sewenkov, O.L., Advances in CAD/CAM, North-Holland Publishing Company, Amsterdam, 1983.
22. Encarnacao, J., Schuster, R. and Voge, E. Product Data Interfaces in CAD/CAM Applications, Springer-Verlag, Berlin, 1986.

23. Eric, Teicholz, CAD/CAM Hand Book, McGraw-Hill Book Company, USA, 1985.
24. Eric, W. Weisstein. Homeomorphism. From *MathwWorld*-A wolfram Web Resource. http://mathworld.wolfram.com/Homeomorphism.html.
25. Eric, W. Weisstein. Topology. From *MathWorld*-A wolfram Web Resource. http://mathworld.wolfram.com/Topology.html.
26. Farin, G. Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide, Academic press, San Diego, 1990.
27. Farin, G., Hansford, D., Peters, A.K. and Natick, M.A. (Eds.) The geometry toolbox for graphics and modeling, ISBN 1-56881-074-1.
28. Faux, I.D. and Pratt, M.J., Computational Geometry for Design and Manufacture, Ellis Harwood Limited, West Sussex, England, 1979.
29. Foley, J.D., Van Dam, A., Feiner, S.K. and Hughes, J.F., Computer Graphics: Principles and Practice, Addison Wesley Publishing, Massachusets, 1996.
30. Franco, P. Preparata and Michael Ian Shamos, Computational Geometry, an introduction, Springer-Verlag, New York, 1985.
31. Gallier, J. Curves and Surfaces in Geometric Modeling: Theory and Algorithms, Morgan Kaufmann Publishers, CA, 2000.
32. George, P.L. Automatic Mesh Generation, John Wiley, Chichester, 1991.
33. Gero, J.S. Expert Systems in Computer Aided Design, Elsevier Science Publisher, North-Holland, 1987.
34. Glen, Mullineux. CAD: Computational Concepts and Methods, Kogan Page Ltd., London, 1986.
35. Goetz, A. Introduction to Differential Geometry, Addison Wesley Publishing Co., New York, 1970.
36. Goldberg, D.E. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.
37. Gouri Dhatt, Gilbert and Touzot. The Finite Element Method Displayed, John Wiley & Sons, New York, 1984.
38. Gray, A. Modern Differential Geometry of Curves and Surfaces with Mathematica,®, CRC Press, Washington, DC, 1998.
39. Gregory, J.A. The Mathematics of Surfaces, Clarandon Press, Oxford, 1986.
40. Hetem, V. Communication: Computer Aided Engineering in the Next Millennium, Computer Aided Design, 2000, Vol. 32, pp. 389–394.
41. Hill Jr., F.S. Computer Graphics using OpenGL, Pearson Education, Singapore, 2003.
42. Hoggar, S.G. Mathematics for Com puter Graphics, Cambridge University Press, 1992.
43. http://www-gap.dcs.st-and.ac.uk/~history/HistTopics/Topology_in_mathematics.html: A history of topology.
44. Jeong, J. Kim K., Park, H., Cho, H., Jung, M., "B-Spline Surface Approximation to Cross-Sections Using Distance Maps", Int. J. Adv. Manuf. Technol, 1999, 15: pp 876–885.
45. John Stark. What every engineer should know about practical CAD/CAM Applications, Marcel Dekker Inc., New York, 1986.
46. Juneja, B.L. Pujara K.K. and Sagar, R. CAD, CAM, Robotics and Factories of the Future, Tata McGraw-Hill Publishing Company Ltd., New Delhi, 1989.
47. Kapur J.N. Mathematical Modeling, Wiley Eastern Limited, New Delhi, 1998.
48. Kelley, J.L. General Topology, Van Nostrand Reinhold Company, 1955.
49. Kochan, D. Integration of CAD/CAM, North-Holland publishing company, Amsterdam, 1984.
50. Krause, F.L. and Jansen, H. Advanced Geometric Modeling for Engineering Applications, Elsevier Science Publishers, North-Holland, 1990.
51. Kunii, T.L. Shinagama Y. and Gotoda H. Intelligent Design and Manufacturing, New York, Wiley, 1992.
52. Kunwoo Lee. Principles of CAD/CAM/CAE Systems, Addison-Wesley, Reading, MA, 1999.
53. Laszlo, M.J. Computational Geometry and Computer Graphics in C++, Prentice Hall, 1996.
54. Loney, S.L. The Elements of Coordinate Geometry, Macmillan and Company Ltd., London, 1953.
55. Lord, E.A. and Wilson C.B. The Mathematical Description of Shape and Form, Ellis Horwood Ltd., England, 1984.
56. M.A. Armstrong, Basic Topology, Springer, 1980.

57. Mansfield, M.J. Introduction to Topology, D. Van Nostrand Company, 1963.
58. Márta, Szilvási-Nagy. *"Almost Curvature Continuous Fitting of B-Spline Surfaces"*, J. for Geometry and Graphics, Vol. 2, 1998, No. 1, pp. 33–43.
59. Medland, A.J. and Piers Burnett. CAD CAM in Practice – a manager's guide to understanding and using CAD CAM, Kogan Page Ltd., London, 1986.
60. Michael, J. French. Conceptual Design for Engineers (second edition), The Design Council, London, 1985.
61. Mortenson, M.E. Geometric Modeling, John Wiley and Sons, New York, 1985.
62. Mortenson, M.E. Mathematics for Computer Graphics Applications (second edition), Industrial Press Inc., New York, 1999.
63. Parviz, E. Nikravesh. Computer Aided Analysis of Mechanical Systems, Prentice-hall International Edition, USA, 1988.
64. Paul, de Faget and de Casteljau. Mathematics and CAD – Shape Mathematics and CAD, Hermes Publishing, France, 1985.
65. Penna, M.A. and Patterson, R.R. Projective Geometry and its Applications to Computer Graphics, Prentice Hall, NJ, 1986.
66. Piegl, L. On NURBS: a survey, IEEE 1991, pp. 55–71.
67. Prakash, N. Differential Geometry, An Integratad Approach, Tata McGraw-Hill Co. Ltd., New Delhi, 1981.
68. Rao, S.S. Optimization: Theory and Applications, Wiley Eastern Limited, New Delhi, 1984.
69. Rao, P.N. CAD/CAM Principles and Applications, Tata McGraw-Hill Co. Ltd., New Delhi, 2002.
70. Reddy, J.N. Applied Functional Analysis and Variational Methods in Engineering, McGraw-Hill Book Company, Singapore, 1986.
71. Robert, F. Steidel and Jr. Jerald. M. Henderson, The Graphic Languages of Engineering, John Wiley & Sons Inc, New york, 1963.
72. Rogers, D.F. and Adams, J. A. Mathematical Elements of Computer graphics, 2nd edition, McGraw-Hill Publishing, N.Y., 1990.
73. Rogers, D.F. Procedural Elements for Computer Graphics, McGraw-Hill Book Company, USA, 1985.
74. Rooney, J. and Steadman, P. Principles of Computer Aided Design, Prentice Hall, NJ, 1987.
75. Schumaker, L.L. Reconstructing 3D Objects from Cross-sections, Computation of Curves and Surfaces, W. Dahmen et al. (Eds), Amsterdam: Kluwer Academic Publishers, pp. 275–309, 1990.
76. Shene, C.K., CS3621 Introduction to Computing with Geometry—Notes http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/notes.html
77. Singer, I. M. and Thorpe, J. A. Lecture Notes On elementary Topology and Geometry, Springer-Verlag, 1967.
78. Struik, D.J., Lectures on Classical Differential Geometry, Addison Wesley Publishing Co., MA, 1961.
79. Ten Hagen P.J. W., Tomiyama T. Intelligent CAD Systems – theoretical and methodological aspects, Springer-verlag Berlin, 1987.
80. Thomas, E. French, Charles, J. Vierck and Robert, J. Foster, Graphic Science and Design, International Student Edition, McGraw-Hill Book Company, New York, 1984.
81. Tuckey, C.O. and Armistead, W. Coordinate Geometry Longmans, Green and Co. Ltd., London, 1953.
82. Vanderplaats, G.N. Numerical Optimization Techniques for Engineering Design with Applications, McGraw-Hill Book Company, USA, 1984.
83. Wade, T.L. and Taylor, H.E. contemporary Analytic Geometry, McGraw-Hill, New York, 1969.
84. Walter, J. Meyer. Concepts of Mathematical Modeling, McGraw-Hill Book Company, Singapore, 1985.
85. William, M. Newman and Robert, F. Sproull, Principles of Interactive Computer Graphics, McGraw-Hill Book Company, Japan, 1979.
86. Willmore, T.J. and Gerald Farin, Geometric curve approximation, Computer Aided Geometric Design, Volume 14, 1997, pp. 499–513.
87. Wozny, M.J. Turner, J.U. and Preiss, K. Geometric Modeling for Product Engineering, elsevier Science Publishers, North Holland, 1990.
88. Yamaguchi, F. Curves and Surfaces in Computer Aided Geometric Design, Springer-Verlag, Berlin, 1988.
89. Yvon Gardan, Mathematics and CAD-Numerical Methods for CAD, Hermes Publishing, France, 1985.
90. Zeid, I. CAD/CAM Theory and Practice, McGraw-Hill Publishing, N.Y., 1991.

Index

Active constraints 354, 355, 357, 364
Advancing front method 371, 372, 373
Affine 84, 106, 107
Affine transformation 106
Analysis situs 249
Analytical surfaces 201
Angular accelerations 14
Angular velocities 14
Aspect ratio 377
Auxiliary edge 261
Axonometric 49
Axonometric projections 55, 60

Barycenter 104
Barycentric 314
Barycentric coordinates 104, 105
Barycentric property 157
Basic solution 360
Basic surfaces 252
Basic variables 360
Basis functions 87, 132
Baumgart's winged-edge 259
Beam 310
Beam elements 318, 319, 320, 323
Bell-shaped 132
Bernstein polynomials 100, 103, 104, 105, 106, 107, 109, 110, 122, 132, 153, 155
Bézier 10, 85
Bézier and Ferguson Segments 117
Bézier conics 100
Bézier curves 103, 107, 108, 110, 111, 114, 115, 118, 122, 144
Bézier segments 107, 108, 109, 111, 112, 113, 114, 115, 116, 118, 123, 130, 151
Bézier surface patches 211
Bi-cublic 202
Bilinear elements 333
Binormal 77, 78
Bi-quadratic 202
Bisection methods 341, 342
Blending functions 87
Boolean operations 44, 265, 267, 269, 290, 291
Boundary and interior 247
Boundary determinism 248, 249
Boundary Interpolation surfaces 218
Boundary representation (B-rep) 11, 247
Boundary representation scheme 259
Bounded 251
Boundedness 247
Bracketing methods 340
Brent's algorithm 345
B-rep 301
B-Spines (Basis splines) 10, 136, 137, 141, 142, 302, 305, 307, 314
B-Spline basis 143, 144, 145
B-Spline curves 151, 152, 153, 154, 155, 157, 160, 161
B-Spline Surface patch 241
Built-in (clamped) end 135

C^0 Continuity 119
C^0 Continuous 90
C^1 continuity 90, 91, 96, 97
C^1 continuous 93, 94, 119, 120
Cabinet 60, 61
Cavalier 60, 61, 62
Centripetal method 158
Chord length method 158
Circle 67
Clamped 152
Classification 302
Closed 251
Closed B-spline surface 243
Closed-up surfaces 252

Closure 248
Compliant mechanisms 367
Composite Bézier surface 229
Composite curve 90
Composite Ferguson curves 89, 130
Composite Ferguson's surface 226
Composite Surfaces 226
Computational geometry 275
Computed Tomography (CT) 297, 298, 299
Computer graphics 2, 3
Cone 168
Conics 66, 67, 97, 98, 99
Connected 251
Connected sum 252
Connectivity 247
Connectivity number of a surface 255
Constructive Solid Geometry (CSG) 11, 247, 265, 266
Control polyline 102
Convex hull property 110, 153
Convex hulls 104, 106, 110
Coon's patches 219, 240, 306
Corner points 204
Cross caps 253, 254
Crossings check 281
CSG Tree 267, 268
Curvature 80
Curve blending 96
Curve fitting 70, 72
Curve interpolation 70
Curve trimming 94
Curves of Intersection 74
Curvilinear 165
Cut 44

de Casteljau's Algorithm 101, 102, 105, 144
de Casteljau's points 102, 103
Delaunay triangulation 373
Delaunay-Voronoi triangulation 371
Developibility 207
Developable surface 176, 181
Difference 265
Dimetric 49, 56, 58
Direct substitution 350
Directrix 269
Discrete elements 370
Divided differences 138, 139, 140, 141, 142, 144
Doubly curved surfaces 183
Ducks 130

Edge detection 300
Edge table 258, 260
Edge-based 303, 304, 305
Effect of tangent magnitudes 89
Element displacement vector 311
Element force vector 311
Element stiffness matrix 311
Ellipse 67
Ellipsoid 168
Elliptic 207
Elliptic Hyperboloid 168
Elliptic point 176, 207
Engineering design 1
Euler characteristics 255
Euler-Poncaré Formula 261
Euler-Poncaré Operators 263
Explicit 73

Face-based 303, 304, 305
False positioning 342
Feasible solutions 359
Ferguson curve 9
Ferguson's Bi-cubic patch 208
Ferguson's Bi-cubic surface patch 203
Ferguson's or Hermite cubic segments 85, 87
Fibonacci search 343, 344
Finite elements 309, 310, 314, 321, 336, 370, 377
Finite Element Analysis 247
Finite element method (FEM) 12
First fundamental form co-efficients 172
First fundamental matrix 172
First order necessary conditions 348
Fitting 302
Flat elements 376
Flat or planar point 180
Flat point 177
Foreshortening 56
Foreshortening factors 56, 57
Foreshortening ratios 56
Four-node elements 331, 332
Frame 310
Frame elements 322, 323, 324, 325, 370
Free end 134
Frenet-Serret formulae 80
Front 371
Full ground structure 370
Fundamental spline 137

Gauss elimination 360
Gauss points 334, 335
Gaussian 166
Gaussian and mean curvatures 178, 180, 181

Gaussian Curvature 181, 182, 183, 184, 207, 209, 214
Genetic Algorithms 365
Genus 255
Geodesic curvature 178
Geographical information system (GIS) 298
Geometric Matrix 88, 96, 98
Geometric reflections 41
Geometric transformations 23, 24, 36
Geometry Invariance 110
GKS 6
Global frame 24
Golden mean 343
Golden section search 343, 344
Graphic standards 6

Half spaces 256
Handles 254
Hardware 4
Helical compression springs 16, 17
Hessian 349, 351, 352, 353, 364, 365
Holes 247
Homeomorphism 249, 252
Homogeneous Co-ordinates 25, 26, 121
Homogeneous three-dimensionality 248
Hybrid sweep 269
Hyperbola 67, 74
Hyperbolic 207
Hyperbolic paraboloid 168
Hyperbolic point 177, 207
Hyperboloid of one sheet 168
Hyperboloid of two sheets 168
Hyperpatch 270

Implicit 73
Implicit representation 201
Inactive constraints 354
Inequality constraints 353, 358
Input devices 4
Interferometry 297
Interior 247
Interpolating surfaces 202
Interpolation 160
Intersect 44
Intersection 248, 265
Isometric 49, 56, 58
Isometric scale 57
Iso-parametric curves 173
Isoparametric elements 333

Join 44

Jordan's curve 280
Jordan's theorem 248

Karush-Kuhn-Tucker necessary condition for optimality (KKT) 355, 356, 357, 358, 364
Kinematic co-efficients (KC) 14, 15
Klein bottle 252, 254
Knot sequence 132
Knot spans 132
Knot vector generation 159
Knot-multiplicity 155, 156
Knots 132, 134, 136, 137, 146, 149, 150, 152, 158, 161

Lagrange multipliers 350, 354, 355, 359, 364, 365
Lagrangian interpolation co-efficients 68
Langrangian multipliers 351, 357
Least square 72
Leibnitz result on divided differences 143
Line of sight 49
Linear interpolation 342
Linear Programming 359
Local support 147
Lines 97
Local changes 130
Local frame 24
Local modification 154
Local stiffness matrix 315
Local stress constraints 367

Magnetic resonance imaging (MRI) 297, 298, 299
Make and Kill groups 263
Manifolds 255
Mean curvatures 181, 182, 183, 186, 207
Membership classification 286
Meridians 189, 190
Mesh Evaluation 377
Mesh smoothing 376
Mesh/faceted models 298
Method of successive substitution 345
Minimal support 137
Möbius strip 252, 253
Multi-block 376
Multiple knots 150
Multi-view projections 49, 54

Natural parameter 76
Natural splines 134
Negative definite 349
Newton's divided differences 68, 69, 138
Newton-Raphson method 346, 347, 348

Nodes 309
Nonlinear sweep 369
Nonself-intersecting 251
Non-uniform 132
Non-uniform rational B-splines (NURBS) 161, 162
Normal curvatures 178, 179, 180, 185
Normal plane 77
Normalized B-spline 145, 146, 147, 148

Object coherence 248
Oblique Projections 60, 61
Octrees 287
One-manifold 256
One-to-one 24
Open methods 345
Optimality criteria 339
Optimization 12
Ordinary surface 168
Orientable 251
Orthogonal 34
Orthogonal Transformation matrices 32
Orthographic projections 49, 54, 55, 58
Osculating circle 77, 79
Osculating plane 77, 78, 79
Output devices 5

Parabola 67, 74, 101
Parabolic 207
Parabolic interpolation 345
Parabolic point 176, 207
Parallel lines 67
Parallel surfaces 185
Parallels 189, 190
Parameterization 158
Parametric 73
Parametric description 84
Parametric design 273
Parametric forms 201
Parametric velocity 76
Partial ground structure 370
Partition of Unity 104, 148
Peano's theorem 142
Perspective projection 49
PHIGS 6
Photogrametry 297
Piecewise fitting 72
Placement ratio 343
Plane stress 328
Point 207
Point cloud 201, 295, 296, 297, 298, 301, 302, 304, 306
Point membership classification 249
Polyhedral representation 248
Polylines 102, 111, 115, 151, 158
Polynomial splines 132, 133, 135, 136
Position continuous 90
Positive definite 349, 352
Primitives 265, 266, 271, 272
Principal normal 77
Profile curve 189
Projections 48
Projective plane 253
Prosthetic design 295

Quadratic end spans 135
Quadrees 287
Quadric Circular Cylinder 168
Quadric Parabolic Cylinder 168
Quadric surface patches 202
Quadrilateral 310
Quadtree 374, 375

Radius of Curvature 79, 80
Ranging methods 297
Rapid prototyping (RP) 247
Rational Bézier curves 121, 124, 161
Rational Bézier segment 123
Rectifying plane 77, 78
Recursion relation 143, 144
Reflection 23, 29, 30, 31, 32, 36, 42
Region elimination methods 341
Regula falsi 342, 348
Regular points 76, 168
Regular surfaces 168, 169
Regularized Boolean operations 268
Rendering 275
Re-parameterization 94
Representing Curves 73
Reverse engineering 295, 296
Rigid-body transformations 9, 23, 24, 26
Rigidity 248
Rotation 23, 24, 25, 26, 29, 36, 37
Rotational sweep 269
Rubber sheet geometry 249
Ruled surfaces 181, 183

Saddle point 349
Scaling 23, 34, 36, 40
Scaling Matrix 34
Secant method 347
Second fundamental form co-efficients 175
Second fundamental matrix 173, 174, 175

Segmentation 302
Sequential Linear Programming (SLP) 359, 363, 367
Sequential Quadratic Programming (SQP) 359, 364, 365, 367, 368
Shape functions 326
Shape manipulation 155
Shape or interpolating functions 314
Shear 23, 34, 35, 36, 41
Simple Sheet 168
Simplex method 360, 363, 364
Simulated Annealing 365
Singly curved surfaces 181
Singular 76
Singular points 168, 169
Sixteen point form surface patch 210
Skinning 300, 301
Slope 90
Software 6, 7, 8
Solid Modeling 247
Spatial addressing 248
Spatial occupancy 247
Spatial uniqueness 248
Splines 130, 135
Splitting 374, 375
Standardization condition 144
Stationary points 349
Strain-displacement matrix 315, 328
Structured lightling 297
Subdivision 113
Subtraction 248
Sufficiency condition 359
Surface patches 201
Surfaces of revolution 188
Swapping 376
Sweep surfaces 190
Sweepline method 371

Tangent 209
Tangent plane 170, 207
Tangents 204, 205, 206
Tensor product surface 204
Theissen or Dirichlet tessellation 373
Three-tangent theorem 101
Tiling 300, 301
Topology 249, 255, 368
Topology of Surfaces 251
Torsion 80
Torus 168
Translation 23, 24, 25, 27, 29, 31, 36
Translational sweep 269
Triangular 310
Triangular elements 325, 326, 328, 331
Triangulation 297
Trimetric 49, 56, 58
Truncated functions 142
Truncated power functions 141, 143
Truss 310
Truss elements 313, 315, 316, 323, 324
Twist vectors 204, 205, 206
Two-manifold 256

Umbilical point 180
Unclamped 152
Uniform 132
Uniform cublic B-spline surface 241
Uniform quadratic B-spline surface 241
Uniform scaling 34, 35, 40
Uniformly spaced method 158
Union 248, 265
Unit tangent 76

Variation diminishing property 111, 154
Vertex table 258
Vivani's curve 74, 75, 196
Voronoi tessellation 373

Watson's Algoithm 374
Weights 87
Winged-edge 260
Winged-edge data structure 259, 261
Wireframe 247
Wireframe modeling 257