## Design

A plan or drawing produced to show the look and function or workings of a building, garment, or other **object** before it is made.

A specification of an object, manifested by an agent, intended to accomplish goals, in a particular environment, using a set of primitive components, satisfying a set of requirements, subject to constraints.

**Computer Aided Design (CAD)**

1. CAD, or computer-aided design and drafting (CADD), is the use of computer technology for design and design documentation.
2. CAD replaces manual drafting with an automated process.
3. Computer-aided drafting (CAD) is the use of computer systems to aid in the creation, modification, analysis, or optimization of a design.
4. CAD/CAM has been utilized in engineering practice in many ways including drafting, design, simulation, analysis and manufacturing.
5. CAD is used to increase the productivity of the designer, improve the quality of design, improve communications through documentation, and to create a database for manufacturing.
6. CAD/CAM includes geometric modeling, computer graphics, design applications, and manufacturing applications.

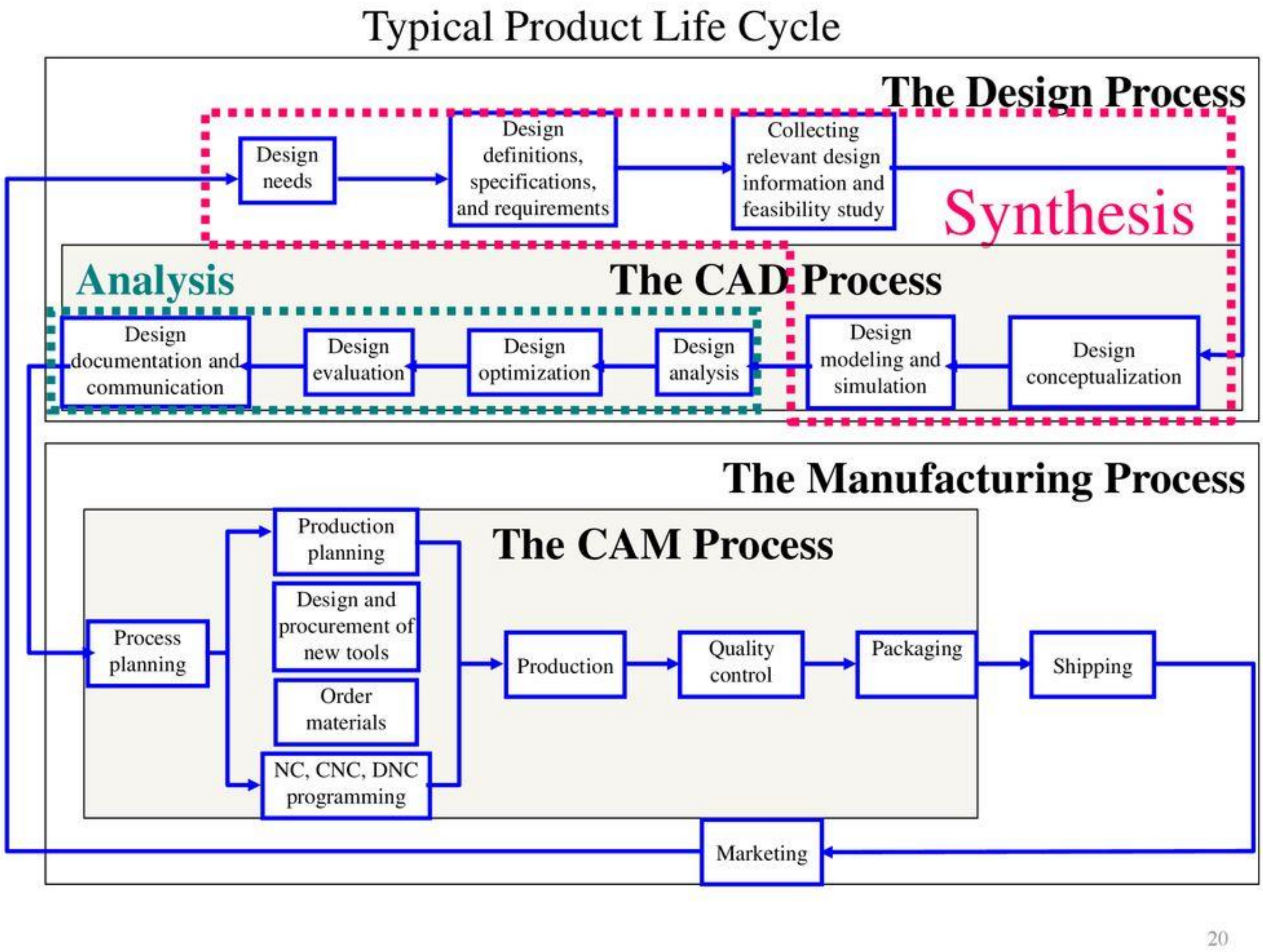
**Need for CAD!!**

1. This century is known for rapid development particularly in the field of computers.
2. Presently industries cannot survive worldwide competition unless they introduce new products with better quality (Q), at lower cost (C), and with shorter lead time (D).
3. Accordingly, many industries have tried to use the computer's huge memory capacity, fast processing speed, and user-friendly interactive graphic capabilities to automate and tie together the cumbersome production tasks, thus reducing the time and cost of product development and production.
4. Computer-aided design (CAD), computer-aided manufacturing (CAM), and computer-aided engineering (CAE) are the technologies used for this purpose during the **product cycle**.

**What is product cycle?**

In the design and manufacture of a product various activities and functions must be accomplished. These activities and functions are referred to as the "**Product Cycle**".

Product cycle What is product cycle? In the design and manufacture of a product various activities and functions must be accomplished. These activities and functions are referred to as the "Product Cycle". A typical product cycle




1. Two main processes: Design process and the manufacturing process.
2. The activities involved in the **design process** can be classified largely as two types: synthesis and analysis.
3. The philosophy, functionality, and uniqueness of the product are all determined during **synthesis**.
4. During synthesis, a design takes the form of sketches and layout drawings that show the relationship among the various product parts.
5. Most of the information generated and handled in the synthesis sub process is qualitative and consequently it is hard to capture in a computer system.
6. The **analysis** sub-process begins with an attempt to put the conceptual design into the context of engineering sciences to evaluate the performance of the expected product.
7. This requires design modeling and simulation. An important aspect of analysis is the "What if" questions that help us to eliminate multiple design choices and find the best solution to each design problem.

8. Bodies with symmetries in their geometry and loading are usually analyzed by considering a portion of the model.
9. Example: stress analysis to verify the strength of the design, interference checking to detect collision between components while they are moving in an assembly, and kinematic analysis to check whether the machine to be used will provide the required motions.
10. The quality of the results obtained from these activities is directly related to and limited by the quality of the analysis model chosen.
11. Prototypes may be built for the design evaluation. Prototypes can be constructed for the given design by using software packages (CAM). The new technology called **rapid prototyping** is become popular now for constructing prototypes. This technology enables the construction of a prototype by depositing layers from the bottom to the top.
12. The outcome of analysis is the design documentation in the form of engineering drawings (blueprints).
13. The **manufacturing process** begins with process planning, using the drawings from the design process, and it ends with the actual products.
14. Process planning is a function that establishes which processes—and the proper parameters for the processes—ends with the actual products.

Process planning is a function that establishes which processes—and the proper parameters for the processes—are to be used. (backbone)

It also selects the most efficient sequence for the production of the product. The outcome of the processplanning is a production plan, tools procurement, materials order, and machine programming. Other special requirements, such as design of jigs and fixtures, are also planned.
The relationship of process planning to the manufacturing process is analogous to that of synthesis to the design process: It involves considerable human experience and qualitative decisions. This description implies that it would be difficult to computerize process planning.
Once process planning has been completed, the actual product is produced and inspected against quality requirements. Parts that pass the quality control inspection are assembled, functionally tested, packaged, labeled, and shipped to customers.
Market feedback is usually incorporated into the design process. With this feedback, a closed-loop product cycle results.

CAD and CAM Disciplines

Design Process
In general it is agreed that the design progresses in a step by step manner from identification of need for the problem, a search for solutions and development of chosen solution to manufacture, test and use.
These descriptions of design are often called models of the design process.

Conceptual design Embodiment design Detail design

Pahl and Beitz model of the design process

Ohsuga and Earle model of the design process

All the above models of the design process follow a traditional view which consists of various steps.

However, the pressure to reduce product design and development time-scales (Lead time) demands for simultaneous/ concurrent/ parallel engineering.

The design, development, analysis and the preparation of manufacturing information are done in parallel then it is termed as Simultaneous
/concurrent engineering.

Sequential Engineering

In the traditional product development process the product design, planning, manufacturing, supporting quality and testing, marketing activities are carried out one after another. All these phases are carried out sequentially hence it is called as sequential engineering.

This situation assumes that there is no interaction among the major departments involved in product manufacturing during the initial development process.
Often the need for engineering changes is discovered during planning or manufacturing or assembly.
In each phase/activities there is no interaction between them. The other
name for sequential approach is 'over wall' approach.
There is no interaction i.e. there is a communication barrier between each department.

Limitations

- Decisions are taken by individuals.
- Product modifications/changes will be slow.
- Each activity is carried out sequentially.
- Because of the above reasons, time taken for product development is more. Lead to inevitable conflicts; each department sticking to their own decisions and may often require intervention of senior management to resolve
- conflicts or differences in opinion.
- Long lead time
- If any modification to be made on the product by stream department, it has to be fed back and this often involves in additional expenditure and also results in unnecessary delay in product cycle.
- Lower quality

## The History of CAD/CAM

The phrase "computer-aided design" was coined by Douglas T. Ross, a researcher at MIT in the early 1950s who saw the potential in military radar technology to create designs on a computer display system. Separate research undertaken by Patrick Hanratty at the General Motors Research Laboratories saw the development of Design Automated by Computer (DAC), regarded as the first CAD system to use interactive graphics.

But the first true 3D CAD/CAM (computer-aided manufacturing) program was created between 1966 and 1968 by Pierre Bézier, an engineer at Renault. His UNISURF CAD system transformed design and manufacturing, moving the vehicle design process from manual drawing boards to computer-aided design. UNISURF is regarded as the original model for many generations of CAD programs.

The automation of design tools did not mean that drafters were replaced by coders, especially after the development of SKETCHPAD, a program written by MIT's Ivan Sutherland in 1963. Instead, this software enabled drafters to feed their design into a computer by drawing with a light pen on a CRT monitor.

As computers became more affordable and shrank to the size of desktop PCs, the use of CAD/CAM spread beyond the automotive, aerospace, and electronics industries to enjoy near-universal usage. The 1970s and 1980s witnessed the emergence of 3D modeling and 3D designs, with programs including Romulus, Uni-Solid, CATIA, and the well-known AutoCAD system.

By this point, CAD/CAM was also being used to design industrial tools. Manufacturers valued CAM for its precision and ability to optimize the manufacturing process, decrease material waste, shorten turnaround times, and provide clear visualizations.

By the 1990s, algorithms had grown increasingly sophisticated, with engines capable of advanced parametric techniques. By 1994 over one million units of AutoCAD had been sold, with 350,000 users of CAD/CAM reported worldwide.

Today, the CAD software market is faced with the emergence of free and open-source CAD software including LibreCAD and FreeCAD. CAD/CAM is used by drafters across dozens of specialties including aeronautical, architectural, civil, electrical, electronics, mechanical, pipeline, and photovoltaic drafting.

The efficiency and user-friendliness of CAD/CAM software have evolved to the point where the career of the trained drafter may one day be threatened by software that anyone can use.

### What's Next for CAD/CAM Technology?

The following trends may show us where the next great leap in CAD/CAM technology will emerge:

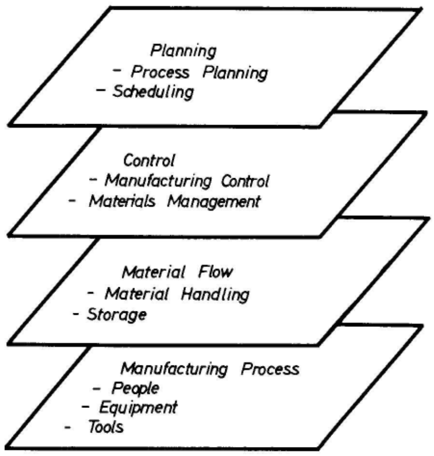
- **Artificial Intelligence:** Incorporating AI into design software allows the automation of design tasks, enhances quality control by anticipating design errors and (with machine learning) paves the way for the creation of unique designs without human input.
- **Cloud collaboration:** Cloud technology allows CAD/CAM to move beyond a single computer at a workplace to universal access through a Saas (Software-as-a-Service) model. This will mean several people can work on the same project at once while sharing across departments and geographies has become much easier.
- **Virtual reality:** VR helmets and VR glasses can be used to take advantage of the life-like visualization offered by sophisticated CAD software. For instance, an architect can now offer a "walkthrough" of a building that exists only as a digital model.
- **Customization:** Software providers are moving away from a one-size-fits-all solution to provide the option of configuring CAD/CAM to suit your work environment, and choose only the tools that will be required for a particular job. This may be a way to offer affordability by cutting out dozens of features that the average user may never need.

For observers in other professions worried about the latest software making their jobs obsolete, the decades-long technological evolution of the drafter's craft shows how smart software can enhance - rather than replace - a profession.

# Introduction to Computer Integrated Manufacturing (CIM)

1. Flexible Manufacturing System (FMS)
2. Variable Mission Mfg. (VMM)
3. Computerized Mfg. System (CMS)

## Four-Plan Concept of Manufacturing




## CIM System discussed:

- Computer Numerical Control (CNC)
- Direct Numerical Control (DNC)
- Computer Process Control
- Computer Integrated Production Management
- Automated Inspection Methods
- Industrial Robots etc.

## A CIM System consists of the following basic components:

I. Machine tools and related equipment
II. Material Handling System (MHS)
III. Computer Control System
IV. Human factor/labor

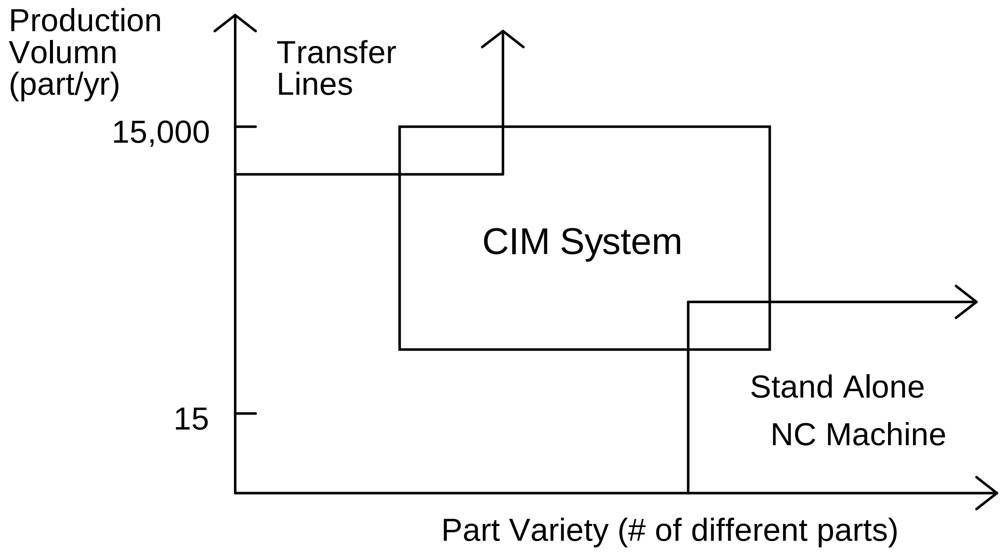
## CIMS Benefits:

1. Increased machine utilization
2. Reduced direct and indirect labor
3. Reduce mfg. lead time
4. Lower in process inventory
5. Scheduling flexibility
6. etc.

CIM refers to a production system that consists of:

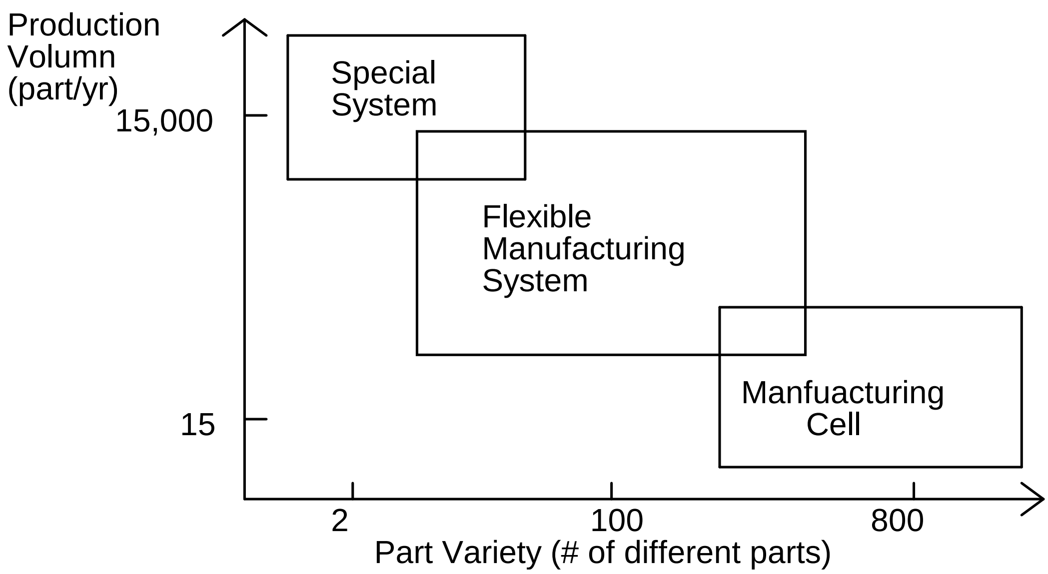
1. A group of NC machines connected together by
2. An automated materials handling system
3. And operating under computer control

Why CIMS?
In Production Systems




1. Transfer Lines: is very efficient when producing "identical" parts in large volumes at high product rates.
2. Stand Alone: NC machine: are ideally suited for variations in work part configuration.

In Manufacturing Systems:




1. Special Mfg. System: the least flexible CIM system. It is designed to produce a

very limited number of different parts (2 - 8).

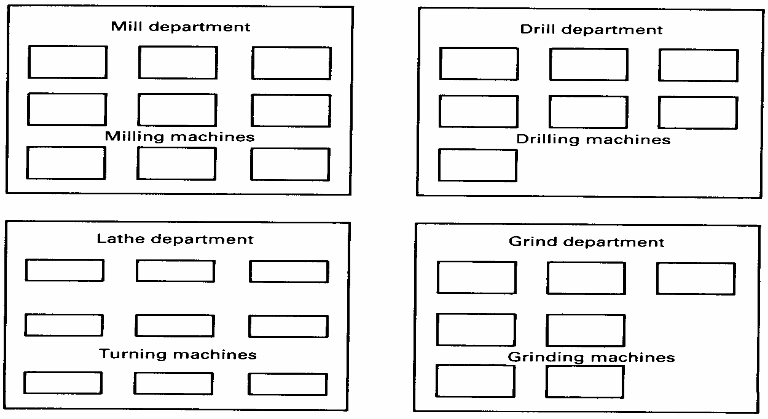
2. Mfg. Cell: the most flexible but generally has the lowest number of different parts manufactured in the cell would be between 40 - 80. Annual production rates rough from 200 - 500.
3. Flexible Mfg. System: A typical FMS will be used to process several part families with 4 to 100 different part numbers being the usual case.

## General FMS

## Conventional Approaches to Manufacturing

*Conventional approaches to manufacturing have generally centered around machines laid out in logical arrangements in a manufacturing facility. These machine layouts are classified by:*

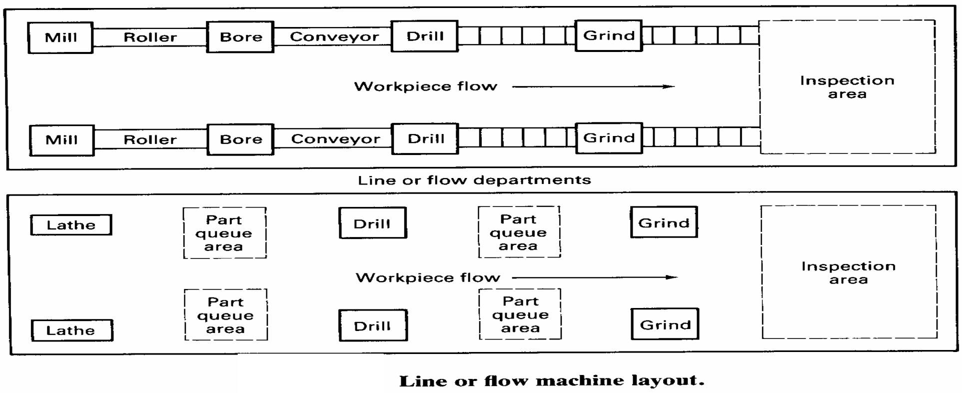
1. Function - Machines organized by function will typically perform the same function, and the location of these departments relative to each other is normally




**Machine layout by function.**

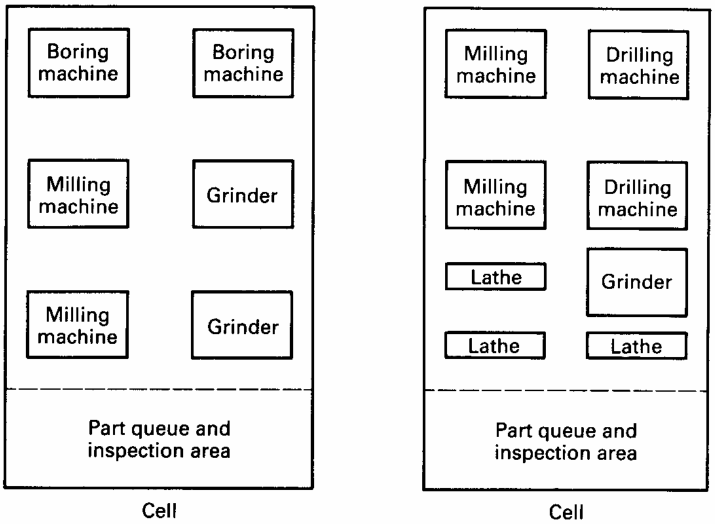
arranged so as to minimize interdepartmental material handling. Workpiece produced in functional layout departments and factories are generally manufactured in small batches up to fifty pieces (a great variety of parts).

2. Line or flow layout - the arrangement of machines in the part processing order or sequence required. A transfer line is an example of a line layout. Parts progressively move from one machine to another in a line or flow layout by means of a roller conveyor or through manual material handling. Typically, one or very few different parts are produced on a line or flow type of layout, as all parts processed require the same processing sequence of operations. All machining is performed in one department, thereby minimizing interdepartmental material handling.




**Line or flow machine layout.**

3. Cell - It combines the efficiencies of both layouts into a single multi-functional unit. It referred to as a group technology cell, each individual cell or department is comprised of different machines that may not be identical or even similar. Each cell is essentially a factory within a factory, and parts are grouped or arranged into families requiring the same type of processes, regardless of processing order. Cellular layouts are highly advantageous over both function and line machine layouts because they can eliminate complex material flow patterns and consolidate material movement from machine to machine within the cell.




**Machine layout by cell based on part families to be processed**

## Manufacturing Cell

## Four general categories:

1. **Traditional stand-alone NC machine tool** - is characterized as a limited-storage, automatic tool changer and is traditionally operated on a one-to-one machine to operator ratio. In many cased, stand-alone NC machine tools have been grouped together in a conventional part family manufacturing cell arrangement and operating on a one-to-one or two-to-one or three-to-one machine to operator ratio.

2. **Single NC machine cell or mini-cell** - is characterized by an automatic work changer with permanently assigned work pallets or a conveyor-robot arm system mounted to the front of the machine, plus the availability of bulk tool storage. There are many machines with a variety of options, such as automatic probing, broken tool detection, and high-pressure coolant control. The single NC machine cell is rapidly gaining in popularity, functionality, and affordability.

3. **Integrated multi-machine cell** - is made up of a multiplicity of metal-cutting machine tools, typically all of the same type, which have a queue of parts, either at the entry of the cell or in front of each machine. Multi-machine cells are either serviced by a material-handling robot or parts are palletized in a two- or three-machine, in-line system for progressive movement from one machining

station to another.

**FMS** - sometimes referred to as a flexible manufacturing cell (FMC), is characterized by multiple machines, automated random movement of palletize parts to and from processing stations, and central computer control with sophisticated command-driven software. The distinguishing characteristics of this cell are the automated flow of raw material to the cell, complete machining of the part, part washing, drying, and inspection with the cell, and removal of the finished part.

## I. Machine Tools & Related Equipment

- Standard CNC machine tools
- Special purpose machine tools
- Tooling for these machines
- Inspection stations or special inspection probes used with the machine tool

**The Selection of Machine Tools**

1. Part size
2. Part shape
3. Part variety
4. Product life cycle
5. Definition of function parts
6. Operations other than machining - assembly, inspection etc.

## II. Material Handling System

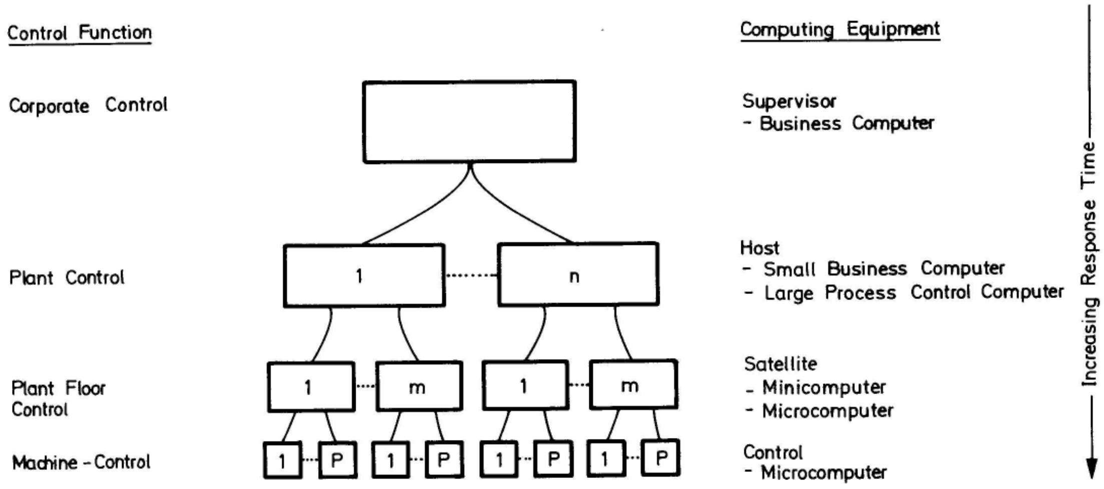
**A. The primary work handling system** - used to move parts between machine tools in the CIMS. It should meet the following requirements.

i). Compatibility with computer control
ii). Provide random, independent movement of palletized work parts between machine tools.
iii). Permit temporary storage or banking of work parts.
iv). Allow access to the machine tools for maintenance tool changing & so on.
v). Interface with the secondary work handling system
vi). etc.

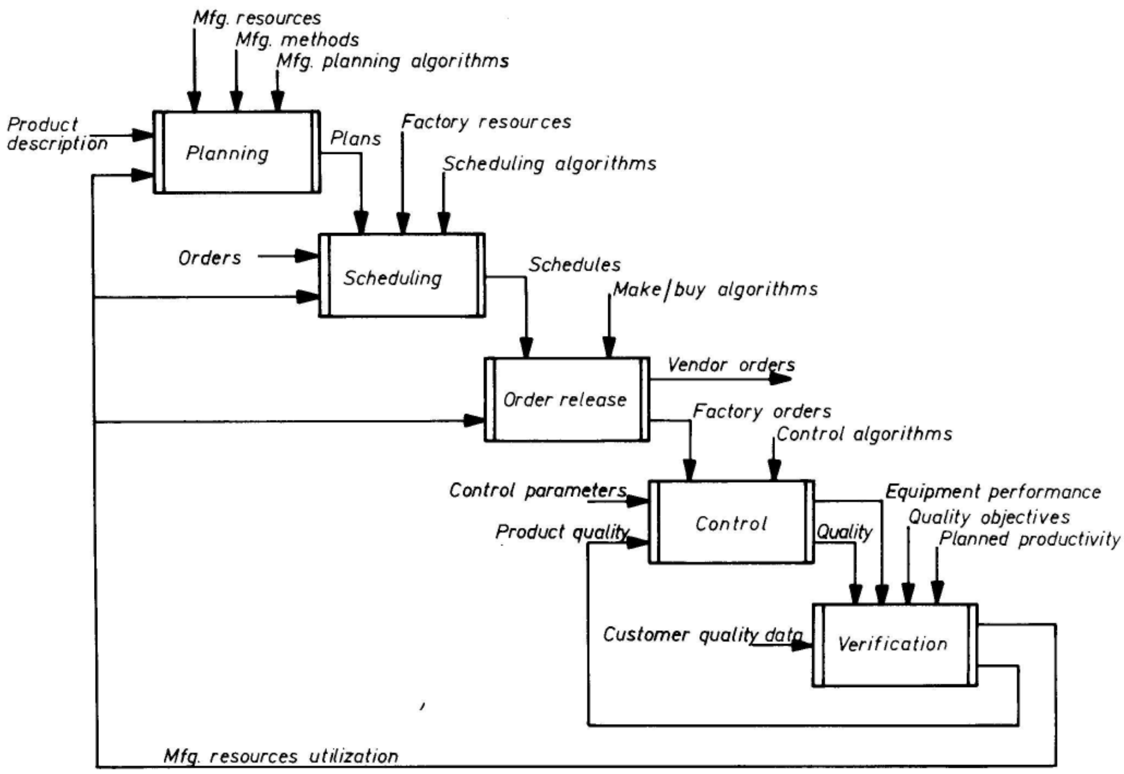
**B. The secondary work handling system** - used to present parts to the individual machine tools in the CIMS.

i). Same as A (i).
ii). Same as A (iii)
iii). Interface with the primary work handling system
iv). Provide for parts orientation & location at each workstation for processing.

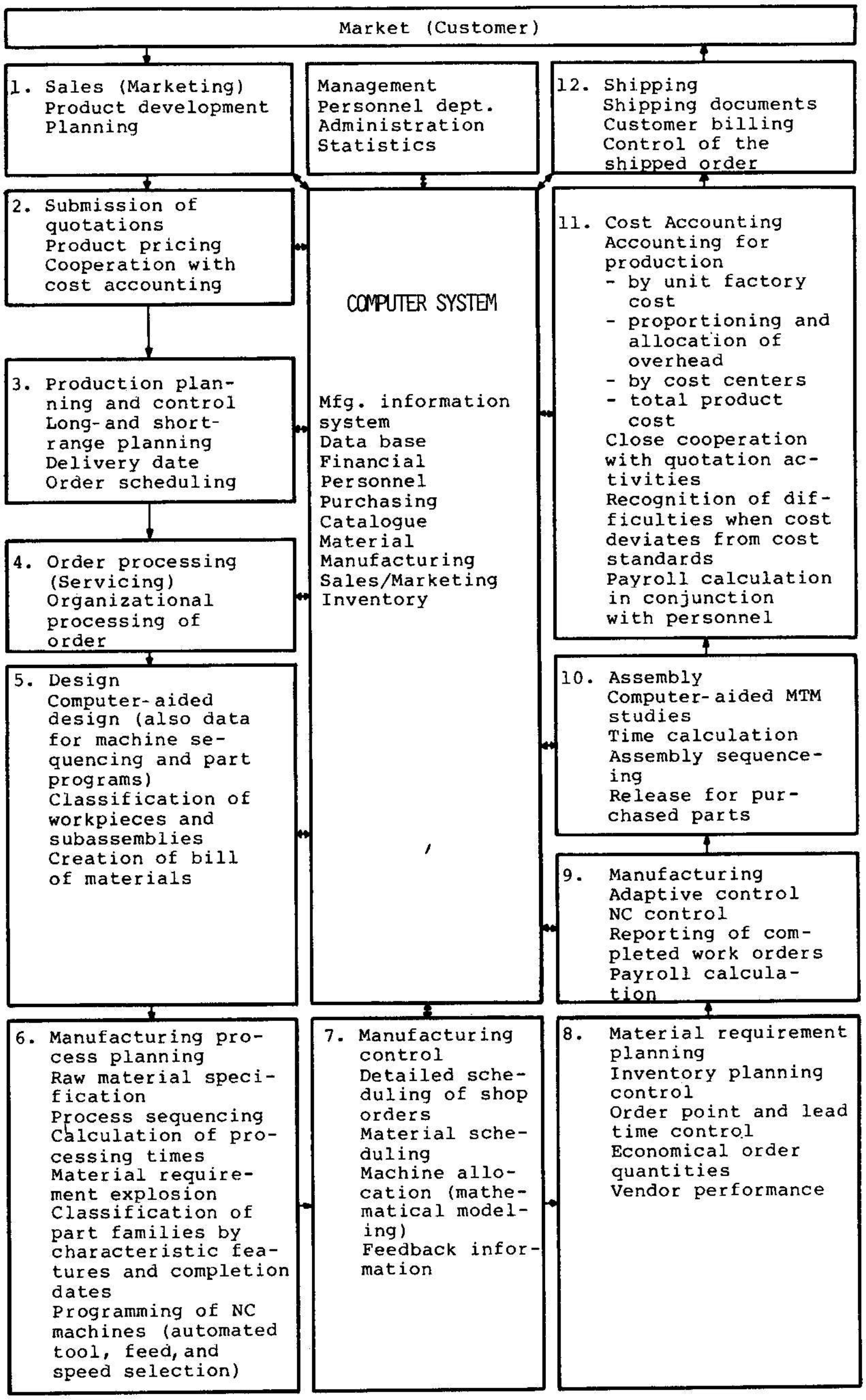
III. Computer Control System - Control functions of a firm and the supporting computing equipment




Control Loop of a Manufacturing System

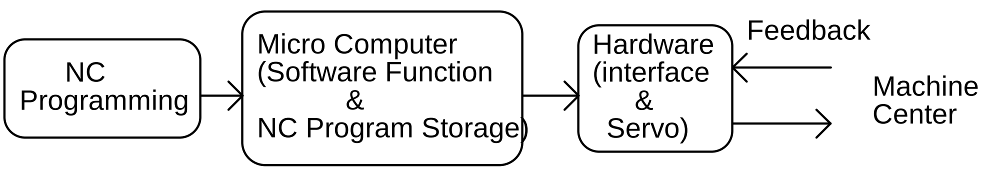


## IV. Functions of the computer in a manufacturing organization




## V. Functions of Computer in CIMS

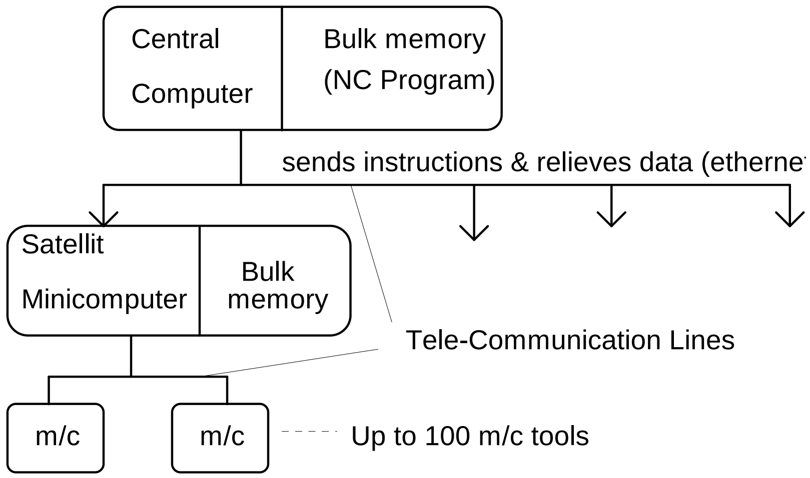
### 1. Machine Control – CNC



**2. Direct Numerical Control (DNC)** - A manufacturing system in which a number of m/c are controlled by a computer through direct connection & in real time.

**Consists of 4 basic elements:**

- Central computer
- Bulk memory (NC program storage)
- Telecommunication line
- Machine tools (up to 100)




**3. Production Control** - This function includes decision on various parts onto the system.

**Decision are based on:**

- red production rate/day for the various parts
- Number of raw work parts available
- Number of available pallets

**4. Traffic & Shuttle Control** - Refers to the regulations of the primary & secondary transportation systems which moves parts between workstation.

**5. Work Handling System Monitoring** - The computer must monitor the status of each cart & /or pallet in the primary & secondary handling system.

**6. Tool Control**

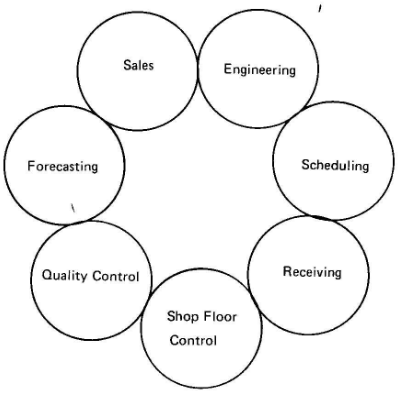
- Keeping track of the tool at each station
- Monitoring of tool life

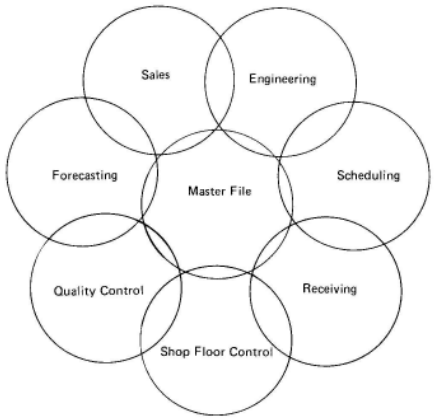
**7. System Performance Monitoring & Reporting** - The system computer can be programmed to generate various reports by the management on system performance.

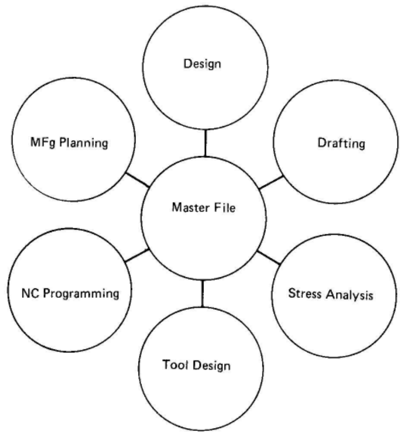
- Utilization reports - summarize the utilization of individual workstation as well as overall average utilization of the system.
- Production reports - summarize weekly/daily quantities of parts produced from a CIMS (comparing scheduled production vs. actual production)
- Status reports - instantaneous report "snapshot" of the present conditions of the CIMS.
- Tool reports - may include a listing of missing tool, tool-life status etc.

## 8. Manufacturing data base

- Collection of independent data bases
- Centralized data base
- Interfaced data base
- Distributed data base




**Production Strategy**

The production strategy used by manufacturers is based on several factors; the two most critical are customer lead time and manufacturing lead time.

*Customer lead time identifies the maximum length of time that a typical customer is willing to wait for the delivery of a product after an order is placed.*

*Manufacturing lead time identifies the maximum length of time between the receipt of an order and the delivery of a finished product.*

Manufacturing lead time and customer lead time must be matched. For example, when a new car with specific options is ordered from a dealer, the customer is willing to wait only a few weeks for delivery of the vehicle. As a result, automotive manufacturers must adopt a production strategy that permits the manufacturing lead-time to match the customer's needs.

The production strategies used to match the customer and manufacturer lead times are grouped into four categories:

1. Engineer to order (ETO)
2. Make to order (MTO)
3. Assemble to order (ATO)
4. Make to stock (MTS)

Engineer to Order

A manufacturer producing in this category has a product that is either in the first stage of the life-cycle curve or a complex product with a unique design produced in single-digit quantities. Examples of ETO include construction industry products (bridges, chemical plants, automotive production lines) and large products with special options that are stationary during production (commercial passenger aircraft, ships, high-voltage switchgear, steam turbines). Due to the nature of the product, the customer is willing to accept a long manufacturing lead time because the engineering design is part of the process.

Make to Order

The MTO technique assumes that all the engineering and design are complete and the production process is proven. Manufacturers use this strategy when the demand is unpredictable and when the customer lead-time permits the production process to start on receipt of an order. New residential homes are examples of this production strategy. Some outline computer companies make personal computer to customer specifications, so they followed MTO specifications.

Assemble to Order

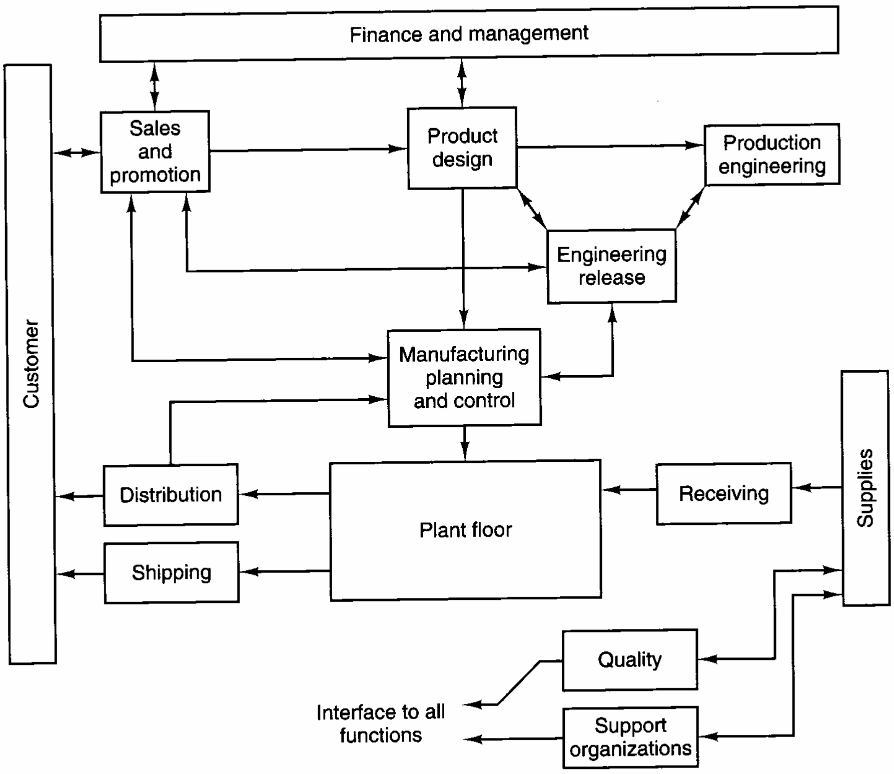
The primary reason that manufacturers adopt the ATO strategy is that customer lead time is less than manufacturing lead time. An example from the automotive industry was used in the preceding section to describe this situation for line manufacturing systems. This strategy is used when the option mix for the products can be forecast statistically: for example, the percentage of four-door versus two-door automobiles assembled per week. In addition, the subassemblies and parts for the final product are carried in a finished components inventory, so the final assembly schedule is determined by the customer order. John Deere and General Motors are examples of companies using this production strategy.

Make to Stock

MTS, is used for two reasons: (1) the customer lead time is less than the manufacturing lead time, (2) the product has a set configuration and few options so that the demand can be forecast accurately. If positive inventory levels (the store shelf is never empty) for a product is an order-winning criterion, this strategy is used. When this order-winning criterion is severe, the products are often stocked in distribution warehouses located in major population centers. This option is often the last phase of a product's life cycle and usually occurs at maximum production volume.

Manufacturing Enterprise (Organization)

- In most manufacturing organizations the functional blocks can be found as:
- A CIM implementation affects every part of an enterprise; as a result, every block in the organizational model is affected.



Sales and Promotion

- The fundamental mission of sales and promotion (SP) is to create customers. To achieve this goal, nine internal functions are found in many companies: sales, customer service, advertising, product research and development, pricing, packaging, public relations, product distribution, and forecasting.

sales and promotion interfaces with several other areas in the business:

- The customer services interface supports three major *customer* functions: order entry, order changes, and order shipping and billing. The order change interface usually involves changes in product specifications, change in product quantity (ordered or available for shipment), and shipment dates and requirements.
- Sales and marketing provide strategic and production planning information to the *finance and management* group, product specification and customer feedback information to *product design*, and information for master production scheduling to the *manufacturing planning and control group*.

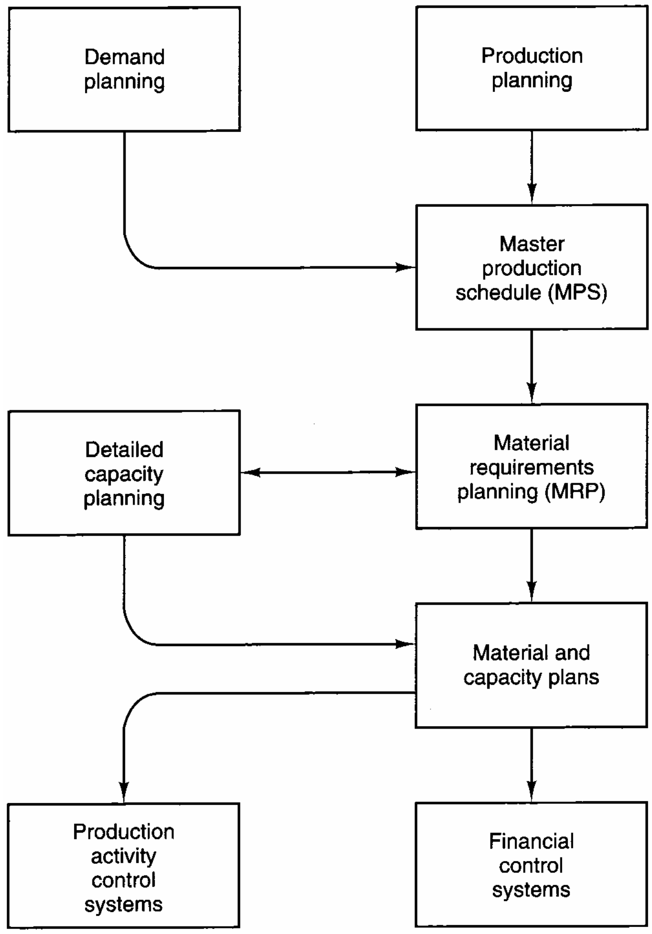
Product/Process Definition Engineering

- The unit includes *product design*, *production engineering*, and *engineering release*.
- The product design provides three primary functions: (1) product design and conceptualization, (2) material selection, and (3) design documentation.
- The production engineering area establishes three sets of standards: work, process, and quality.
- The engineering release area manages engineering change on every production part in the enterprise. Engineering release has the responsibility of securing approvals from departments across the enterprise for changes made in the product or production process.

Manufacturing Planning and Control (MPC)

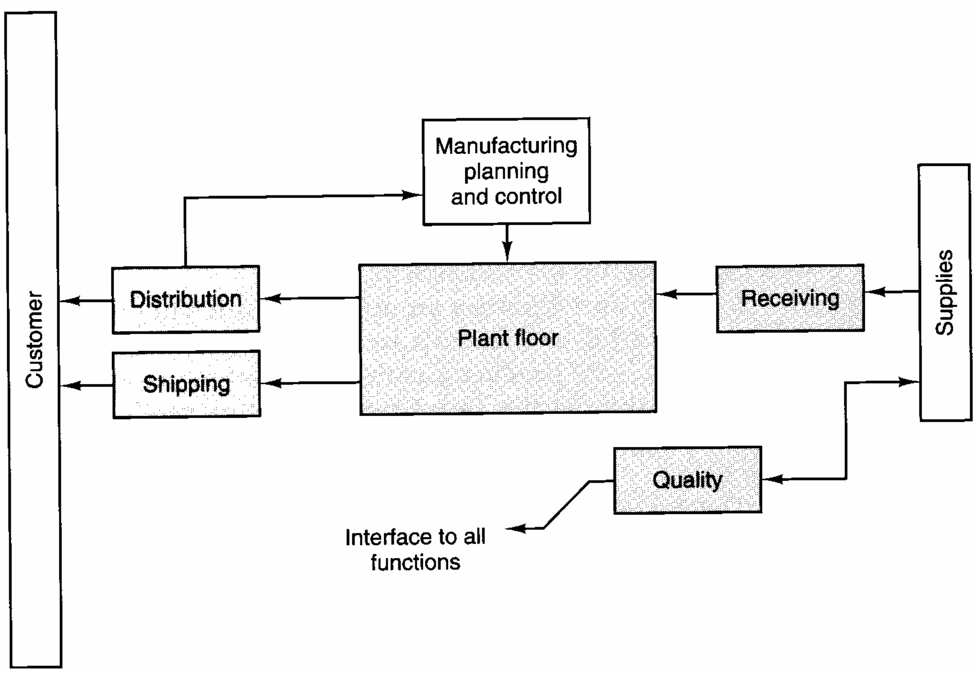
- The manufacturing planning and control unit has a formal data and information interface with several other units and departments in the enterprise.
- The MPC unit has responsibility for:
    1. Setting the direction for the enterprise by translating the management plan into manufacturing terms. The translation is smooth if order-winning criteria were used to develop the management plan.
    2. Providing detailed planning for material flow and capacity to support the overall plan.
    3. Executing these plans through detailed shop scheduling and purchasing action.

MPC Model for Information Flow




Shop Floor

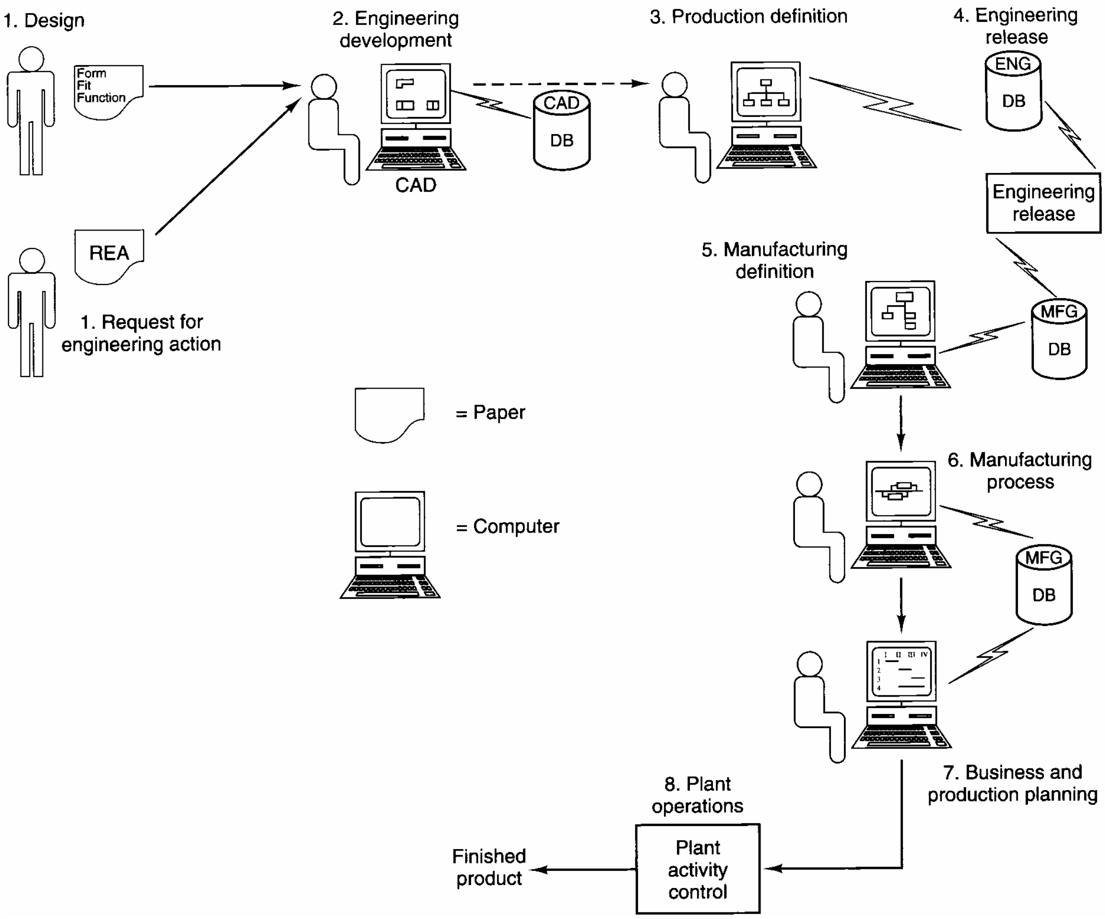
- Shop floor activity often includes job planning and reporting, material movement, manufacturing process, plant floor control, and quality control.
- Interfaces with the shop floor unit are illustrated.



Support Organization

- The support organizations, indicated vary significantly from firm to firm.
- The functions most often included are security, personnel, maintenance, human resource development, and computer services.
- Basically, the support organization is responsible for all of the functions not provided by the other model elements.

Production Sequence :one possibility for the flow required to bring a product to a customer



## Curve Representation

- All forms of geometric modeling require the ability to define curves.

## Curves in Modeling

- Curves produced in modeling systems:
  - Straight lines
  - Conic sections
  - Free-form parametric curves (B-Splines, NURBS)
  - curves of intersection between surfaces defined during model construction

## Curve Use in Design

Engineering design requires ability to express complex curve shapes.

- Some examples
  - turbine blades
  - ship hulls
  - automotive body panels

## Complex Curves

- A curve can be described by a finite number of short straight segments.
  - However, on close inspection this is only an approximation.
  - To get a better approximation we can use more segments per unit length.
  - This increases the amount of data required to store the curve and makes it difficult to manipulate.
  - Need a way of representing these curves in a more mathematical fashion. Ideally, our descriptions will be:

## Curves

Ideally, a curve description should be:

- Reproducible
  - the representation should give the same curve every time;
- Computationally Quick
- Easy to manipulate and edit
  - especially important for design purposes.
- Easy to combine with other curves

## Types of Curve Equations

- explicit (non-parametric)

  $Y = f(X), Z = g(X)$

- implicit (non-parametric)

  $f(X,Y,Z) = 0$

- parametric

  $X = X(t), Y = Y(t), Z = Z(t)$

The explicit and implicit formats have serious disadvantages for use in computer-based modeling

## Curve Equation Examples

- Explicit $y = mx + b$
- Implicit $x^2 + y^2 - R^2 = 0$
- Parametric $x = x_0 + ut$ ; $y = y_0 + ut$

## Explicit Curves

- $y = ax^3 + bx^2 + cx + d$, where a, b, c, d are constants
- For a given range of 'x', will define a curve
- However, cannot turn back upon itself and cannot represent a vertical line

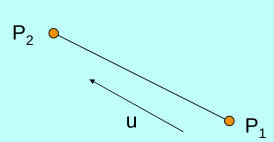
## Implicit Curves

- $ax^2 + by^2 + cxy + d = 0$
- Can produce self intersecting curves
- Easy to determine whether a given point lies on the line (set membership)
- Can be difficult to transform
- Packages may use implicit forms and translate them to parametric when needed.

## Parametric: $X = X(t), Y = Y(t), Z = Z(t);$ $0 \le t \le 1$ (typ)

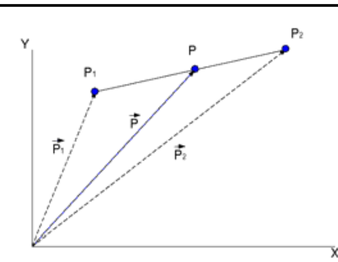
- Substituting a value for *t* gives a corresponding position along curve
- Overcomes problems associated with implicit and explicit methods
- Most commonly used representation scheme in modelers

## Parametric example Linear ($1^{st}$ order) Curve




Parametric representation of a line. The parameter *u*, is varied from 0 to 1 to define all points along the line.

$X = X(u)$ $Y = Y(u)$




$\vec{P} = \vec{P_1} + (\vec{P} - \vec{P_1})$

$\vec{P} - \vec{P_1}$ is proportional to $\vec{P_2} - \vec{P_1}$

such that $\vec{P} - \vec{P_1} = u(\vec{P_2} - \vec{P_1})$

therefore, the parametric equation of line $P_1 P_2$ is

$\vec{P} = \vec{P_1} + u(\vec{P_2} - \vec{P_1})$

where $0 \le u \le 1$

## Parametric Line

- Line in then defined in terms of its endpoints
- Positions along the line are based upon the parameter value
  - For example, the midpoint of a line occurs at u = 0.5

## Parametric Line
### Scalar Formulation

- This means a parametric line can be defined by:

  L(u) = [x(u), y(u), z(u)] = A + (B - A)u

  where A and B and the line endpoints.

  e.g. A line from point A = (2,4,1) to point B = (7,5,5) can be represented as:

  x(u) = 2 + (7-2)u = 2 + 5u

  y(u) = 4 + (5-4)u = 4 + u

  z(u) = 1 + (5-1)u = 1 + 4u

## Another basic example would be that of a conic (circle)

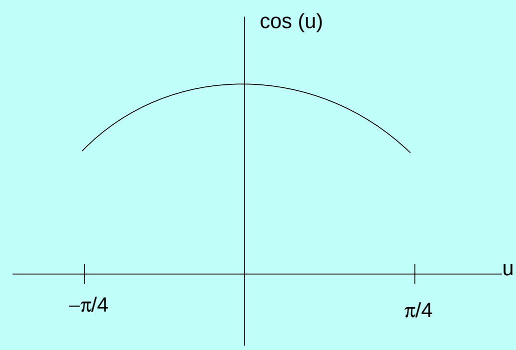
- Two parameter curves are:

  X = cos (u)

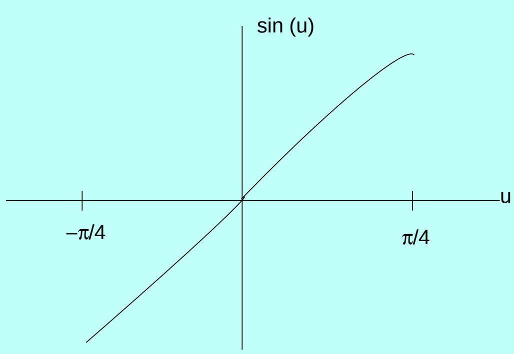
  Y = sin (u)

- with range $-\pi/4 \leq u \leq \pi/4$

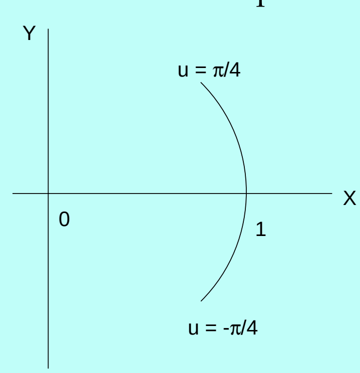
## Graph of X = cos (u)




## Graph of Y = sin (u)




## Combined curve is a quarter circle



## Controls for circular arc

- Shape (based upon parametric equation)
- Location (based upon center point)
- Size
  - arc (based upon parameter range)
  - radius ( a coefficient to unit value)
- Similar list could be formed for other conics

## Curve Tangent Vectors

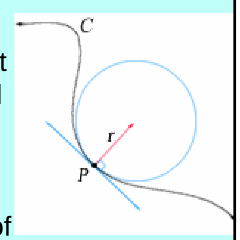
- Curves are defined by parametric equation
- Position along the curve is defined by the equation
- At any point along the curve there exists a vector defining the curve "direction"
- This is the tangent vector. It is defined by the first derivative of the parametric curve equation
- For a straight line this derivative will equal a constant

## Curvature

- Parametric equation defines position along the curve
- The 1st derivative of the parametric equation defines direction or the "tangent vector".
- The 2nd derivative defines the rate of change of direction; this is **curvature**.

## Curvature

For a plane curve (*C*), the curvature at a given point *P* has a magnitude equal to the *reciprocal* of the radius of an osculating circle and is a vector pointing in the direction of that circle's center.




## Curvature

- The osculating circle meets the curve at a tangent point
- An arc or circle has a constant curvature.
- A straight line has a curvature of zero.
  - When a circular fillet is tangent to a straight-line segment, the curvature falls abruptly from a constant value to zero.

## Curvature

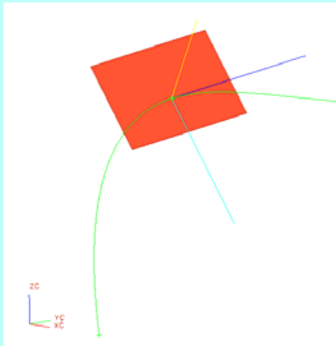
For a plane curve given parametrically as f(*t*) = (*x*(*t*),*y*(*t*)) the curvature is

$$\kappa = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}}$$

where the dots denote differentiation with respect to *t*.

## Osculating Plane

In the case of a space curve, the osculating circle lies in a plane defined by the curve's normal and tangent vectors.



## Moving Triad

- The curve normal vector points in the direction of the radius of curvature.
- The cross product of the normal and tangent vectors yields the bi-normal vector.
- A unique set of these vectors exists at each point along the curve.
- This set is referred to as the "moving triad"

## Moving Triad

- The moving triad provides vital information of the characteristics of a moving object
  - Motion is in the direction of the tangent vector,
  - "up"  is in the direction of the bi-normal vector
  - The rate of turning and turning direction are given by the curvature and the direction of the normal vector, respectively.
  - That is, as you move along the curve this triad always provides you with the forward, up and turning directions.

## Representing Complex Curves

- typically represented as…
  - a series of simpler curves *(each defined by a single equation)* pieced together at their endpoints *(piecewise construction).*
  - These simpler curves may be linear or polynomial
  - simpler curves are based upon ***control points*** (data pts. used to define the curve)

## Use of control points

- General curve shape may be generated using methods of:
  - *Interpolation*

    curve will pass though user defined control points
  - *Approximation*

    curve will pass *near* control points,
    curve may interpolate the start and end points

## Control Points Defining Curves

- The following example shows an:

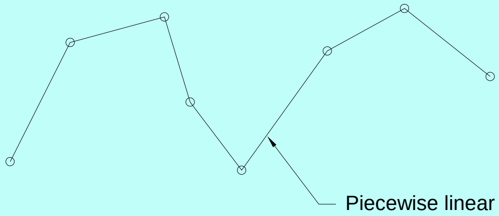
  Interpolating

  (passes through control points)

  Piecewise linear curve

  curve defined by multiple segments, in this case linear

## Interpolating Curve : Piecewise linear

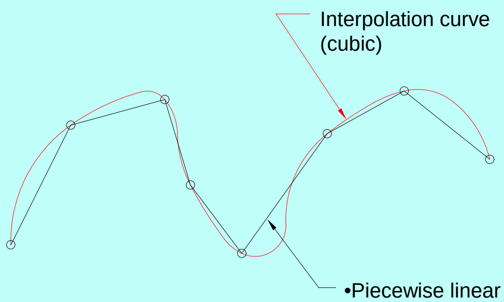



## Interpolating Curve : Piecewise linear

- Linear segments used to approximate smooth shape
- Segments joints known as KNOTS

  (see knot point slide)
- Requires too many datapoints for most shape approximations
- Representation not flexible enough to editing (for example, translating the curve is cumbersome)

## Piecewise polynomial (composite curves)




## Piecewise polynomial (composite curves)

- Segments defined by polynomial functions
- Again, segments join at KNOTS
- Most common polynomial used is cubic (3rd order)*
- Segment shape controlled by two or more adjacent control points.

  * see curve order slide

## Knot points

- Locations where segments join are referred to as knots
- Knots may or may not coincide with control points in interpolating curves, typically they DO NOT coincide.

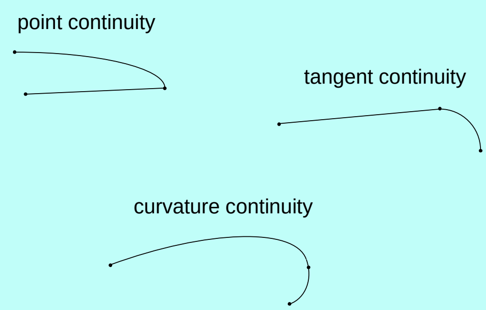
## Curve continuity

- concern is <u>continuity</u> at knots (where curve segments join)
- continuity conditions:
  - point continuity (no slope or curvature restriction)
  - tangent continuity (same slope at knot)
  - curvature continuity (same slope and curvature at knot)

## Curve continuity

- Continuity is symbolically represented by capital "C" with a superscript representing level.
  - $C^0$ continuity is continuity of endpoint only, or continuity of position
  - $C^1$ continuity is tangent continuity or first derivative of position
  - $C^2$ continuity is curvature continuity or second derivative of position

## Composite curves: continuity




## Curve continuity

- Continuity is also an issue at the intersection points of different user defined curves
- When curves are used to define surfaces and solids, poor curve intersections can produce non-smooth geometry (creases, corners)

## Interpolation curves

- Interpolating piecewise polynomial curve
- Typically possess curvature continuity
- Shape defined by:
  - endpoint and control point location
  - tangent vectors at knots*
  - curvature at knots*

*typically calculated internally by default

## Interpolation curves

- User may be able to define:
  - Curve slope
  - Curvature

At the interpolated control points.

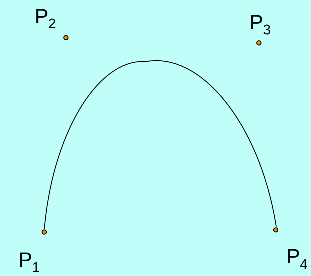
## Approximation techniques:

- Developed to permit greater design flexibility in the generation of freeform curves.
- Two very common methods in modern CAD systems, Bezier and B-Spline.

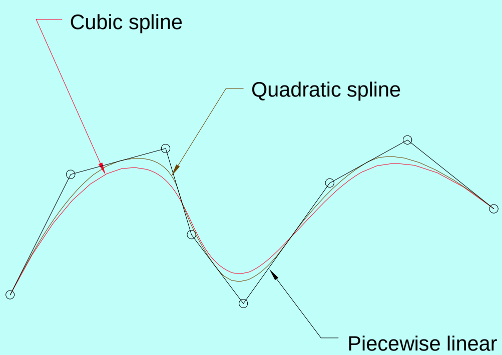
## Approximation techniques

- employ control points (set of vertices that approximate the curve)
- curves do not pass directly through points (except possibly at start and end)
- intermediate points affect shape as if exerting a “pull” on the curve
- allow user to to set shape by “pulling” out curve using control point location

## Bezier curve & control points




## B-Spline curves




## Effect of curve order

- The higher the order of a curve, the “stiffer” the curve (less dramatic curvature changes)
- Maximum curve order dependent upon the number of control points
  order = one less than the number of control pts
- High order curves can exhibit irregularities

## Most modern CAD systems use the NURB curve representation scheme.

- NURB stands for Non-Uniform, Rational, B-spline.
- Uniformity deals with the spacing of control points.
- Rational functions include a weighting value at each control point for effect of control point.

## NURBs

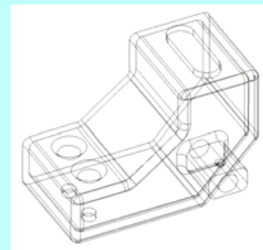
- very popular due to their flexibility in curve generation.
- Permit the mathematical form to be used to represent entire families of curves including:
  - Bezier
  - B-Spline
  - conics

# Wireframe modeling systems

- Definition
- Database information
- Uses/Disadvantages
- Curve representation

# Wireframe

- name is taken from the model appearance
- only the edges of a geometry are displayed.



# Wireframe model construction

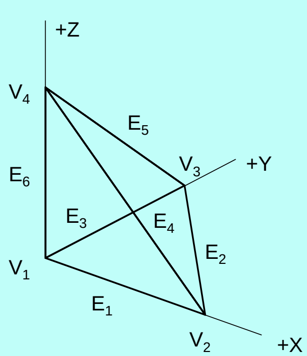
- methods typically very straightforward
- uses the same commands and techniques as 2D construction.
- entities are the same as those for 2D graphics, with inclusion of some extended *database information*, (Z-coordinate data)

# Basic database information

(if we restrict discussion to linear edge models)

- Vertices
  - Coordinate values for vertices
- Edges
  - Vertices associated with edges (endpoints)
- Faces
  - Loops (faces) formed by edges

# Wireframe Tetrahedron



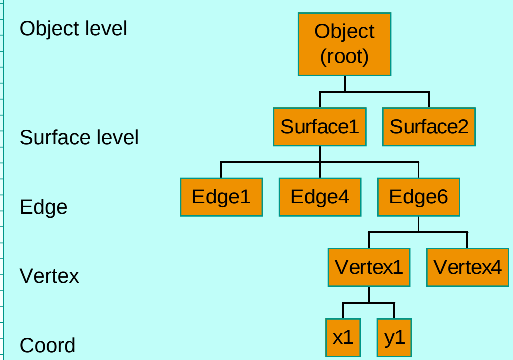

# Brief discussion of basic database structures

- Relational
  - A set of lists, uses arrays for storage
- Hierarchical
  - A trees structure, think of a company's executive structure
- Network
  - Uses data pointers to relate data elements

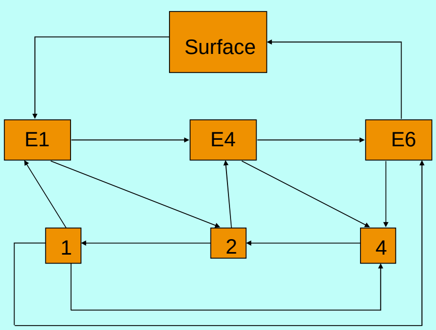
## Relational Database

| Vertex List | Edge List |
|---|---|
| v1 (0,0,0) | E1 [ V1, V2] |
| v2 (1,0,0) | E2 [ V2, V3] |
| v3 (0,1,0) | E3 [ V3, V1] |
| v4 (0,0,1) | E4 [ V2, V4] |
| | E5 [ V4, V3] |
| | E6 [ V1, V4] |

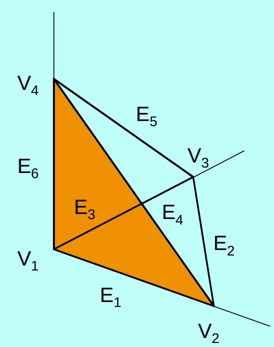
## Hierarchical Database




## "Network" Database




## Bounded face




## Relational Database

| Vertex list | Edge list | Face list |
|---|---|---|
| v1 (0,0,0) | E1 [ V1, V2] | F1 {E1,E4,E6} |
| v2 (1,0,0) | E2 [ V2, V3] | F2 {E4,E2,E5} |
| v3 (0,1,0) | E3 [ V3, V1] | F3 {E3,E6,E5} |
| v4 (0,0,1) | E4 [ V2, V4] | F4 {E1,E3,E2} |
| | E5 [ V4, V3] | |
| | E6 [ V1, V4] | |

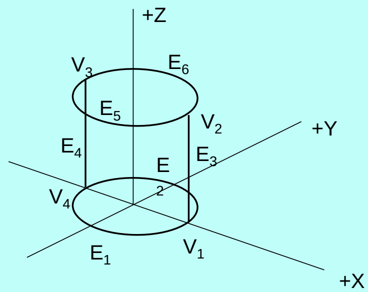
## Introduction to model validity criteria

(example restricted to linear edges)

- Each vertex has 3 coordinate values
- Each edge delimited by two vertices
- Minimum of 3 edges must intersect at each vertex
- Minimum of 3 edges required to define a loop (face)

## Wireframe Cylinder

(structured to conform to validity criteria)




## Wireframe Cylinder

- In the example show, the model definition (edges, vertices) was chosen to meet a validity criteria set.
  - Three edges intersect at a vertex
  - Edges delimited by two vertices

## Model Information Stored vs. Model Usefulness

- The information available from a model database, and hence its usefulness, is dependent upon the data stored.
- User chooses model definition based upon application

## Examples of information available from wireframe database.

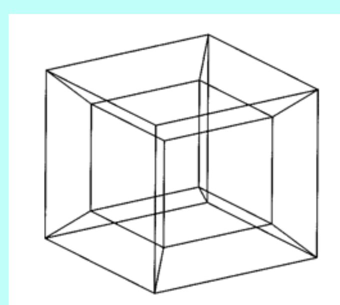
- Distance and length calculations.
  - Distance between vertices, length of edges
- Basic entity connectivity
  - Which vertices delimit an edge.
  - Which edges share vertices.
  - Which edges form a loop (or face).
- A basic visual representation of the object being modeled.

## Visual Deficiencies of Wireframes

- ambiguity
  - complex models difficult to interpret
- does not allow for use of photorealistic rendering tools*

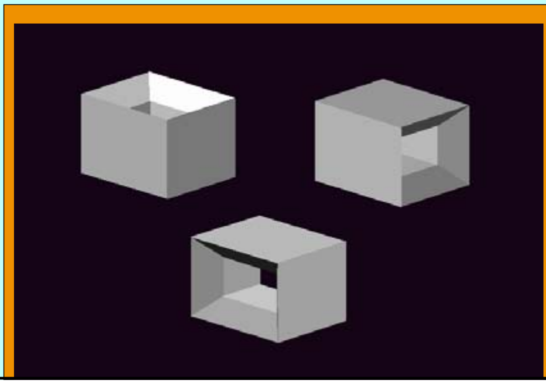
  *some software capable of hidden line removal on limited basis.

## Ambiguous wireframe model



What does this object look like?

Which object does the previous wireframe represent?



## Other Limitations of Wireframe Model Structure

- No ability to determine computationally information on mass properties such as volume, mass, moments of inertia, etc.
- No guarantee that the model definition is correct, complete or manufacturable.

## Other Limitations of Wireframe Model Structure

- No ability to determine computationally information such as the line of intersect between two faces of intersecting models.
- Limited ability for checking interference between mating parts
  (typically visual only)

## Wireframe Uses:

- Geometry display by modeling systems
- Visualization of motion (simple animations)
- Modeling of geometries such as projected profiles and revolutions.
- 2D drafting

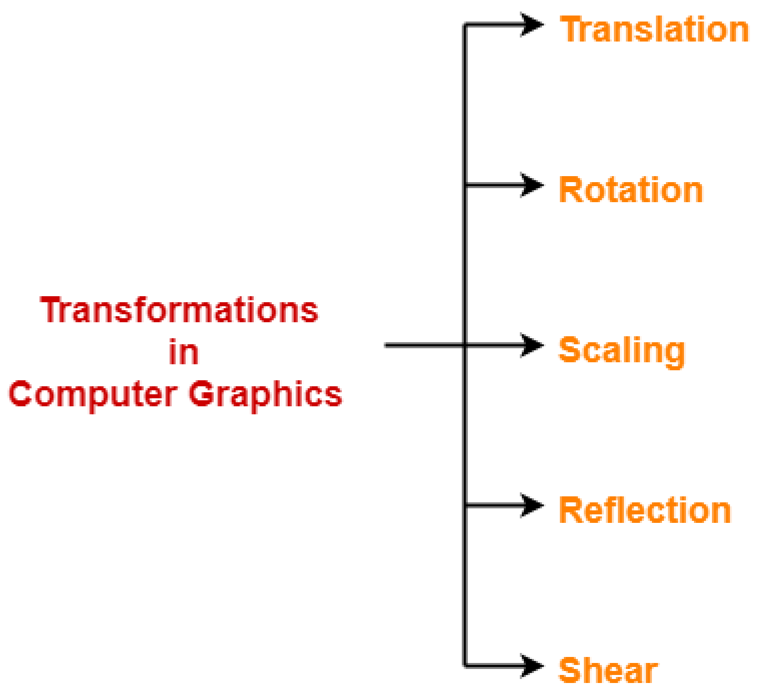
## Wireframe Uses

- Wireframe models of limited value for manufacturing and analysis.
  - Manufacturing: profile geometry only, 2D operations such as turning, profile milling
  - Analysis: can be used as input into FEA packages, especially in the case of truss and frame geometries.

TARNSFORMATION
CHAPTER 2

- Transformation is a process of modifying and re-positioning the existing graphics.
- 3D Transformations take place in a three dimensional plane.

In computer graphics, various transformation techniques are-




1. **Translation**
2. **Rotation**
3. **Scaling**
4. **Reflection**
5. **Shear**

In this article, we will discuss about 3D Shearing in Computer Graphics.

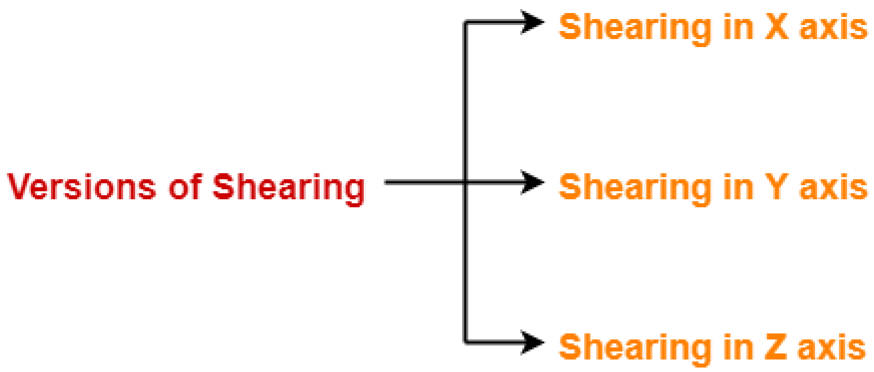
## 3D Shearing in Computer Graphics-

In Computer graphics,

3D Shearing is an ideal technique to change the shape of an existing object in a three dimensional plane.

In a three dimensional plane, the object size can be changed along X direction, Y direction as well as Z direction.

So, there are three versions of shearing-




1. Shearing in X direction
2. Shearing in Y direction
3. Shearing in Z direction

Consider a point object O has to be sheared in a 3D plane.

Let-

- Initial coordinates of the object O = ($X_{old}$, $Y_{old}$, $Z_{old}$)
- Shearing parameter towards X direction = $Sh_x$
- Shearing parameter towards Y direction = $Sh_y$
- Shearing parameter towards Z direction = $Sh_z$
- New coordinates of the object O after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$)

**<u>Shearing in X Axis-</u>**

Shearing in X axis is achieved by using the following shearing equations-

- $X_{new} = X_{old}$
- $Y_{new} = Y_{old} + Sh_y \times X_{old}$
- $Z_{new} = Z_{old} + Sh_z \times X_{old}$

In Matrix form, the above shearing equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ Sh_y & 1 & 0 & 0 \\ Sh_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Shearing Matrix

(In X axis)

**Shearing in Y Axis-**

Shearing in Y axis is achieved by using the following shearing equations-

- $X_{new} = X_{old} + Sh_x \times Y_{old}$
- $Y_{new} = Y_{old}$
- $Z_{new} = Z_{old} + Sh_z \times Y_{old}$

In Matrix form, the above shearing equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & Sh_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & Sh_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Shearing Matrix

(In Y axis)

**Shearing in Z Axis-**

Shearing in Z axis is achieved by using the following shearing equations-

- $X_{new} = X_{old} + Sh_x \times Z_{old}$
- $Y_{new} = Y_{old} + Sh_y \times Z_{old}$
- $Z_{new} = Z_{old}$

In Matrix form, the above shearing equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Sh_x & 0 \\ 0 & 1 & Sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

**3D Shearing Matrix**

**(In Z axis)**

## PRACTICE PROBLEMS BASED ON 3D SHEARING IN COMPUTER GRAPHICS-

### Problem-01:

Given a 3D triangle with points (0, 0, 0), (1, 1, 2) and (1, 1, 3). Apply shear parameter 2 on X axis, 2 on Y axis and 3 on Z axis and find out the new coordinates of the object.

### Solution-

Given-

- Old corner coordinates of the triangle = A (0, 0, 0), B(1, 1, 2), C(1, 1, 3)
- Shearing parameter towards X direction ($Sh_x$) = 2
- Shearing parameter towards Y direction ($Sh_y$) = 2
- Shearing parameter towards Y direction ($Sh_z$) = 3

### Shearing in X Axis-

**For Coordinates A(0, 0, 0)**

Let the new coordinates of corner A after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} = 0$
- $Y_{new} = Y_{old} + Sh_y \times X_{old} = 0 + 2 \times 0 = 0$
- $Z_{new} = Z_{old} + Sh_z \times X_{old} = 0 + 3 \times 0 = 0$

Thus, New coordinates of corner A after shearing = (0, 0, 0).

**For Coordinates B(1, 1, 2)**

Let the new coordinates of corner B after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} = 1$
- $Y_{new} = Y_{old} + Sh_y \times X_{old} = 1 + 2 \times 1 = 3$
- $Z_{new} = Z_{old} + Sh_z \times X_{old} = 2 + 3 \times 1 = 5$

Thus, New coordinates of corner B after shearing = (1, 3, 5).

**For Coordinates C(1, 1, 3)**

Let the new coordinates of corner C after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} = 1$
- $Y_{new} = Y_{old} + Sh_y \times X_{old} = 1 + 2 \times 1 = 3$
- $Z_{new} = Z_{old} + Sh_z \times X_{old} = 3 + 3 \times 1 = 6$

Thus, New coordinates of corner C after shearing = (1, 3, 6).

Thus, New coordinates of the triangle after shearing in X axis = A (0, 0, 0), B(1, 3, 5), C(1, 3, 6).

**Shearing in Y Axis-**

**For Coordinates A(0, 0, 0)**

Let the new coordinates of corner A after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \text{ x } Y_{old} = 0 + 2 \text{ x } 0 = 0$
- $Y_{new} = Y_{old} = 0$
- $Z_{new} = Z_{old} + Sh_z \text{ x } Y_{old} = 0 + 3 \text{ x } 0 = 0$

Thus, New coordinates of corner A after shearing = (0, 0, 0).

**For Coordinates B(1, 1, 2)**

Let the new coordinates of corner B after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \text{ x } Y_{old} = 1 + 2 \text{ x } 1 = 3$
- $Y_{new} = Y_{old} = 1$
- $Z_{new} = Z_{old} + Sh_z \text{ x } Y_{old} = 2 + 3 \text{ x } 1 = 5$

Thus, New coordinates of corner B after shearing = (3, 1, 5).

**For Coordinates C(1, 1, 3)**

Let the new coordinates of corner C after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \text{ x } Y_{old} = 1 + 2 \text{ x } 1 = 3$
- $Y_{new} = Y_{old} = 1$
- $Z_{new} = Z_{old} + Sh_z \text{ x } Y_{old} = 3 + 3 \text{ x } 1 = 6$

Thus, New coordinates of corner C after shearing = (3, 1, 6).

Thus, New coordinates of the triangle after shearing in Y axis = A (0, 0, 0), B(3, 1, 5), C(3, 1, 6).

**Shearing in Z Axis-**

**For Coordinates A(0, 0, 0)**

Let the new coordinates of corner A after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \times Z_{old} = 0 + 2 \times 0 = 0$
- $Y_{new} = Y_{old} + Sh_y \times Z_{old} = 0 + 2 \times 0 = 0$
- $Z_{new} = Z_{old} = 0$

Thus, New coordinates of corner A after shearing = (0, 0, 0).

**For Coordinates B(1, 1, 2)**

Let the new coordinates of corner B after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \times Z_{old} = 1 + 2 \times 2 = 5$
- $Y_{new} = Y_{old} + Sh_y \times Z_{old} = 1 + 2 \times 2 = 5$
- $Z_{new} = Z_{old} = 2$

Thus, New coordinates of corner B after shearing = (5, 5, 2).

**For Coordinates C(1, 1, 3)**

Let the new coordinates of corner C after shearing = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the shearing equations, we have-

- $X_{new} = X_{old} + Sh_x \times Z_{old} = 1 + 2 \times 3 = 7$
- $Y_{new} = Y_{old} + Sh_y \times Z_{old} = 1 + 2 \times 3 = 7$
- $Z_{new} = Z_{old} = 3$

Thus, New coordinates of corner C after shearing = (7, 7, 3).

Thus, New coordinates of the triangle after shearing in Z axis = A (0, 0, 0), B(5, 5, 2), C(7, 7, 3).

To gain better understanding about 3D Shearing in Computer Graphics,

**Watch this Video Lecture**

**Next Article- Bezier Curves**

Get more notes and other study material of **Computer Graphics**.

Watch video lectures by visiting our YouTube channel **LearnVidFun.**

**3D Reflection in Computer Graphics | Definition | Examples**

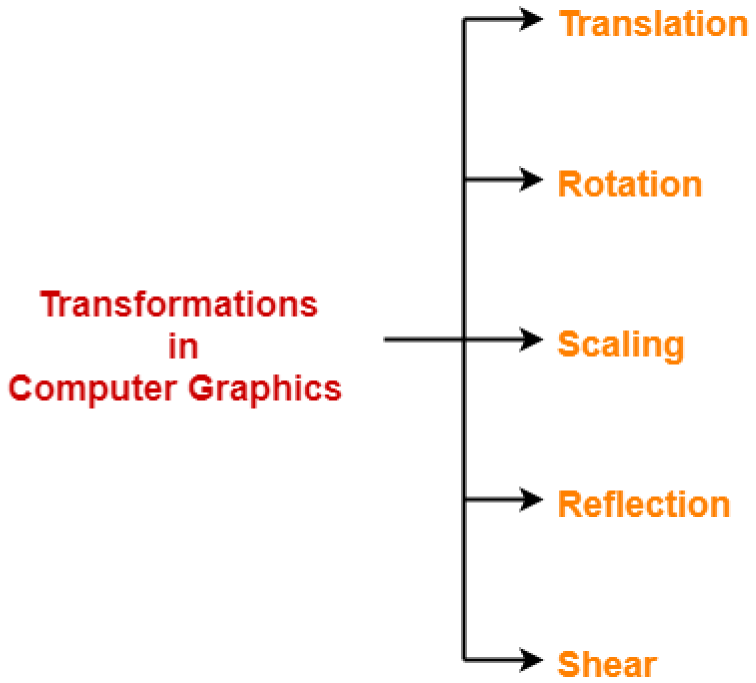
Computer Graphics

**3D Transformations in Computer Graphics-**

We have discussed-

- Transformation is a process of modifying and re-positioning the existing graphics.
- 3D Transformations take place in a three dimensional plane.

In computer graphics, various transformation techniques are-




1. **Translation**
2. **Rotation**
3. **Scaling**

4. **Reflection**
5. **Shear**

In this article, we will discuss about 3D Reflection in Computer Graphics.

## 3D Reflection in Computer Graphics-

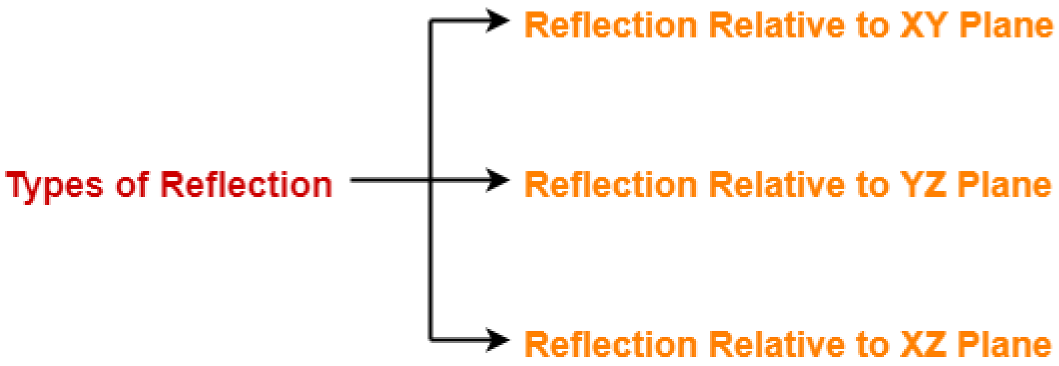
- Reflection is a kind of rotation where the angle of rotation is 180 degree.
- The reflected object is always formed on the other side of mirror.
- The size of reflected object is same as the size of original object.

Consider a point object O has to be reflected in a 3D plane.

Let-

- Initial coordinates of the object O = ($X_{old}$, $Y_{old}$, $Z_{old}$)
- New coordinates of the reflected object O after reflection = ($X_{new}$, $Y_{new}$,$Z_{new}$)

In 3 dimensions, there are 3 possible types of reflection-




- Reflection relative to XY plane
- Reflection relative to YZ plane
- Reflection relative to XZ plane

### Reflection Relative to XY Plane:

This reflection is achieved by using the following reflection equations-

- $X_{new} = X_{old}$
- $Y_{new} = Y_{old}$

- $Z_{new} = -Z_{old}$

In Matrix form, the above reflection equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

**3D Reflection Matrix**

**(Reflection Relative to XY plane)**

**Reflection Relative to YZ Plane:**

This reflection is achieved by using the following reflection equations-

- $X_{new} = -X_{old}$
- $Y_{new} = Y_{old}$
- $Z_{new} = Z_{old}$

In Matrix form, the above reflection equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

**3D Reflection Matrix**

**(Reflection Relative to YZ plane)**

**Reflection Relative to XZ Plane:**

This reflection is achieved by using the following reflection equations-

- $X_{new} = X_{old}$
- $Y_{new} = -Y_{old}$
- $Z_{new} = Z_{old}$

In Matrix form, the above reflection equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

**3D Reflection Matrix**

**(Reflection Relative to XZ plane)**

**PRACTICE PROBLEMS BASED ON 3D REFLECTION IN COMPUTER GRAPHICS-**

**Problem-01:**

Given a 3D triangle with coordinate points A(3, 4, 1), B(6, 4, 2), C(5, 6, 3). Apply the reflection on the XY plane and find out the new coordinates of the object.

**Solution-**

Given-

- Old corner coordinates of the triangle = A (3, 4, 1), B(6, 4, 2), C(5, 6, 3)
- Reflection has to be taken on the XY plane

**For Coordinates A(3, 4, 1)**

Let the new coordinates of corner A after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 3$
- $Y_{new} = Y_{old} = 4$
- $Z_{new} = -Z_{old} = -1$

Thus, New coordinates of corner A after reflection = (3, 4, -1).

**For Coordinates B(6, 4, 2)**

Let the new coordinates of corner B after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 6$
- $Y_{new} = Y_{old} = 4$
- $Z_{new} = -Z_{old} = -2$

Thus, New coordinates of corner B after reflection = (6, 4, -2).

**For Coordinates C(5, 6, 3)**

Let the new coordinates of corner C after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 5$
- $Y_{new} = Y_{old} = 6$
- $Z_{new} = -Z_{old} = -3$

Thus, New coordinates of corner C after reflection = (5, 6, -3).

Thus, New coordinates of the triangle after reflection = A (3, 4, -1), B(6, 4, -2), C(5, 6, -3).

**Problem-02:**

Given a 3D triangle with coordinate points A(3, 4, 1), B(6, 4, 2), C(5, 6, 3). Apply the reflection on the XZ plane and find out the new coordinates of the object.

**Solution-**

Given-

- Old corner coordinates of the triangle = A (3, 4, 1), B(6, 4, 2), C(5, 6, 3)
- Reflection has to be taken on the XZ plane

**For Coordinates A(3, 4, 1)**

Let the new coordinates of corner A after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 3$
- $Y_{new} = -Y_{old} = -4$
- $Z_{new} = Z_{old} = 1$

Thus, New coordinates of corner A after reflection = (3, -4, 1).

**For Coordinates B(6, 4, 2)**

Let the new coordinates of corner B after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 6$
- $Y_{new} = -Y_{old} = -4$
- $Z_{new} = Z_{old} = 2$

Thus, New coordinates of corner B after reflection = (6, -4, 2).

**For Coordinates C(5, 6, 3)**

Let the new coordinates of corner C after reflection = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the reflection equations, we have-

- $X_{new} = X_{old} = 5$
- $Y_{new} = -Y_{old} = -6$
- $Z_{new} = Z_{old} = 3$

Thus, New coordinates of corner C after reflection = (5, -6, 3).

Thus, New coordinates of the triangle after reflection = A (3, -4, 1), B(6, -4, 2), C(5, -6, 3).

To gain better understanding about 3D Reflection in Computer Graphics,

**Watch this Video Lecture**

**Next Article- 3D Shearing in Computer Graphics**

Get more notes and other study material of **Computer Graphics**.

Watch video lectures by visiting our YouTube channel **LearnVidFun**.

**3D Scaling in Computer Graphics | Definition | Examples**

Computer Graphics

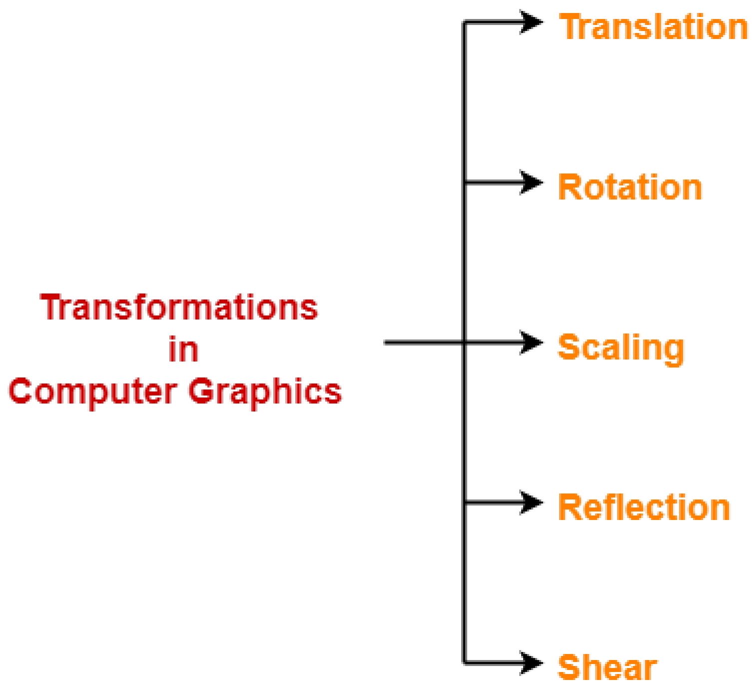
**3D Transformations in Computer Graphics-**

We have discussed-

- Transformation is a process of modifying and re-positioning the existing graphics.
- 3D Transformations take place in a three dimensional plane.

INTERNATIONAL INSTITUTE OF TECHNOLOGY & MANAGEMENT, MURTHAL SONEPAT
E-NOTES , Subject : CAD, Subject Code: ME 402B Course: B.tech , Branch : Mechanical Engineering , Sem.-8th , UNIT: $1^{ST}$ Transformation
( Prepared By: Ms. Promila , Assistant Professor , MED)

In computer graphics, various transformation techniques are-




1. **Translation**
2. **Rotation**
3. **Scaling**
4. **Reflection**
5. **Shear**

In this article, we will discuss about 3D Scaling in Computer Graphics.

## 3D Scaling in Computer Graphics-

In computer graphics, scaling is a process of modifying or altering the size of objects.

- Scaling may be used to increase or reduce the size of object.
- Scaling subjects the coordinate points of the original object to change.
- Scaling factor determines whether the object size is to be increased or reduced.
- If scaling factor > 1, then the object size is increased.
- If scaling factor < 1, then the object size is reduced.

Consider a point object O has to be scaled in a 3D plane.

Let-

- Initial coordinates of the object O = ($X_{old}$, $Y_{old}$,$Z_{old}$)
- Scaling factor for X-axis = $S_x$
- Scaling factor for Y-axis = $S_y$
- Scaling factor for Z-axis = $S_z$
- New coordinates of the object O after scaling = ($X_{new}$, $Y_{new}$, $Z_{new}$)

This scaling is achieved by using the following scaling equations-

- $X_{new} = X_{old} \times S_x$
- $Y_{new} = Y_{old} \times S_y$
- $Z_{new} = Z_{old} \times S_z$

In Matrix form, the above scaling equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Scaling Matrix

**PRACTICE PROBLEMS BASED ON 3D SCALING IN COMPUTER GRAPHICS-**

**Problem-01:**

Given a 3D object with coordinate points A(0, 3, 3), B(3, 3, 6), C(3, 0, 1), D(0, 0, 0). Apply the scaling parameter 2 towards X axis, 3 towards Y axis and 3 towards Z axis and obtain the new coordinates of the object.

**Solution-**

Given-

- Old coordinates of the object = A (0, 3, 3), B(3, 3, 6), C(3, 0, 1), D(0, 0, 0)
- Scaling factor along X axis = 2
- Scaling factor along Y axis = 3
- Scaling factor along Z axis = 3

**For Coordinates A(0, 3, 3)**

Let the new coordinates of A after scaling = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the scaling equations, we have-

- $X_{new} = X_{old} \times S_x = 0 \times 2 = 0$
- $Y_{new} = Y_{old} \times S_y = 3 \times 3 = 9$
- $Z_{new} = Z_{old} \times S_z = 3 \times 3 = 9$

Thus, New coordinates of corner A after scaling = (0, 9, 9).

**For Coordinates B(3, 3, 6)**

Let the new coordinates of B after scaling = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the scaling equations, we have-

- $X_{new} = X_{old} \times S_x = 3 \times 2 = 6$
- $Y_{new} = Y_{old} \times S_y = 3 \times 3 = 9$
- $Z_{new} = Z_{old} \times S_z = 6 \times 3 = 18$

Thus, New coordinates of corner B after scaling = (6, 9, 18).

**For Coordinates C(3, 0, 1)**

Let the new coordinates of C after scaling = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the scaling equations, we have-

- $X_{new} = X_{old} \times S_x = 3 \times 2 = 6$
- $Y_{new} = Y_{old} \times S_y = 0 \times 3 = 0$
- $Z_{new} = Z_{old} \times S_z = 1 \times 3 = 3$

Thus, New coordinates of corner C after scaling = (6, 0, 3).

**For Coordinates D(0, 0, 0)**

Let the new coordinates of D after scaling = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the scaling equations, we have-

- $X_{new} = X_{old} \times S_x = 0 \times 2 = 0$
- $Y_{new} = Y_{old} \times S_y = 0 \times 3 = 0$
- $Z_{new} = Z_{old} \times S_z = 0 \times 3 = 0$

Thus, New coordinates of corner D after scaling = (0, 0, 0).

To gain better understanding about 3D Scaling in Computer Graphics,

**Watch this Video Lecture**

**Next Article- 3D Reflection in Computer Graphics**

Get more notes and other study material of **Computer Graphics**.

Watch video lectures by visiting our YouTube channel **LearnVidFun**.

**3D Rotation in Computer Graphics | Definition | Examples**

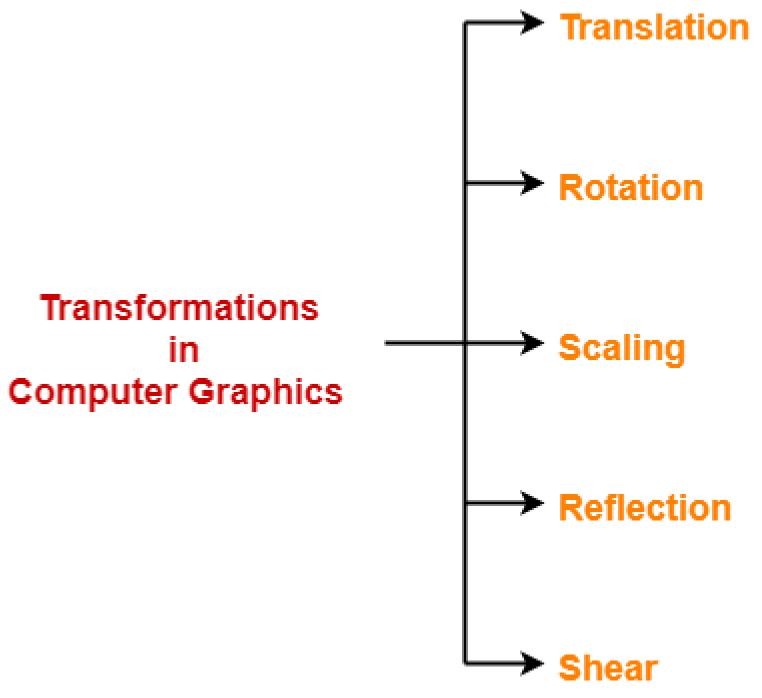
Computer Graphics

**3D Transformations in Computer Graphics-**

We have discussed-

- Transformation is a process of modifying and re-positioning the existing graphics.
- 3D Transformations take place in a three dimensional plane.

In computer graphics, various transformation techniques are-




1. **Translation**
2. **Rotation**
3. **Scaling**
4. **Reflection**
5. **Shear**

In this article, we will discuss about 3D Rotation in Computer Graphics.

## 3D Rotation in Computer Graphics-

In Computer graphics,

3D Rotation is a process of rotating an object with respect to an angle in a three dimensional plane.

Consider a point object O has to be rotated from one angle to another in a 3D plane.

Let-

- Initial coordinates of the object O = $(X_{old}, Y_{old}, Z_{old})$
- Initial angle of the object O with respect to origin = $\Phi$
- Rotation angle = $\theta$
- New coordinates of the object O after rotation = $(X_{new}, Y_{new}, Z_{new})$

In 3 dimensions, there are 3 possible types of rotation-

- X-axis Rotation
- Y-axis Rotation
- Z-axis Rotation

**For X-Axis Rotation-**

This rotation is achieved by using the following rotation equations-

- $X_{new} = X_{old}$
- $Y_{new} = Y_{old} \times \cos\theta - Z_{old} \times \sin\theta$
- $Z_{new} = Y_{old} \times \sin\theta + Z_{old} \times \cos\theta$

In Matrix form, the above rotation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Rotation Matrix

(For X-Axis Rotation)

**For Y-Axis Rotation-**

This rotation is achieved by using the following rotation equations-

- $X_{new} = Z_{old} \times \sin\theta + X_{old} \times \cos\theta$
- $Y_{new} = Y_{old}$
- $Z_{new} = Y_{old} \times \cos\theta - X_{old} \times \sin\theta$

In Matrix form, the above rotation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{x} \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Rotation Matrix

(For Y-Axis Rotation)

**For Z-Axis Rotation-**

This rotation is achieved by using the following rotation equations-

- $X_{new} = X_{old} \text{ x } \cos\theta - Y_{old} \text{ x } \sin\theta$
- $Y_{new} = X_{old} \text{ x } \sin\theta + Y_{old} \text{ x } \cos\theta$
- $Z_{new} = Z_{old}$

In Matrix form, the above rotation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{x} \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Rotation Matrix

(For Z-Axis Rotation)

**PRACTICE PROBLEMS BASED ON 3D ROTATION IN COMPUTER GRAPHICS-**

**Problem-01:**

Given a homogeneous point (1, 2, 3). Apply rotation 90 degree towards X, Y and Z axis and find out the new coordinate points.

**Solution-**

Given-

- Old coordinates = ($X_{old}$, $Y_{old}$, $Z_{old}$) = (1, 2, 3)
- Rotation angle = $\theta$ = 90º

**For X-Axis Rotation-**

Let the new coordinates after rotation = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the rotation equations, we have-

- $X_{new} = X_{old} = 1$
- $Y_{new} = Y_{old} \times \cos\theta - Z_{old} \times \sin\theta = 2 \times \cos 90° - 3 \times \sin 90° = 2 \times 0 - 3 \times 1 = -3$
- $Z_{new} = Y_{old} \times \sin\theta + Z_{old} \times \cos\theta = 2 \times \sin 90° + 3 \times \cos 90° = 2 \times 1 + 3 \times 0 = 2$

Thus, New coordinates after rotation = (1, -3, 2).

**For Y-Axis Rotation-**

Let the new coordinates after rotation = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the rotation equations, we have-

- $X_{new} = Z_{old} \times \sin\theta + X_{old} \times \cos\theta = 3 \times \sin 90° + 1 \times \cos 90° = 3 \times 1 + 1 \times 0 = 3$
- $Y_{new} = Y_{old} = 2$
- $Z_{new} = Y_{old} \times \cos\theta - X_{old} \times \sin\theta = 2 \times \cos 90° - 1 \times \sin 90° = 2 \times 0 - 1 \times 1 = -1$

Thus, New coordinates after rotation = (3, 2, -1).

**For Z-Axis Rotation-**

Let the new coordinates after rotation = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the rotation equations, we have-

- $X_{new} = X_{old} \times \cos\theta - Y_{old} \times \sin\theta = 1 \times \cos 90° - 2 \times \sin 90° = 1 \times 0 - 2 \times 1 = -2$
- $Y_{new} = X_{old} \times \sin\theta + Y_{old} \times \cos\theta = 1 \times \sin 90° + 2 \times \cos 90° = 1 \times 1 + 2 \times 0 = 1$
- $Z_{new} = Z_{old} = 3$

Thus, New coordinates after rotation = (-2, 1, 3).

To gain better understanding about 3D Rotation in Computer Graphics,

**Watch this Video Lecture**

**Next Article- 3D Scaling in Computer Graphics**

Get more notes and other study material of **Computer Graphics**.

Watch video lectures by visiting our YouTube channel **LearnVidFun.**

**3D Translation in Computer Graphics | Definition | Examples**

Computer Graphics

**3D Transformation in Computer Graphics-**

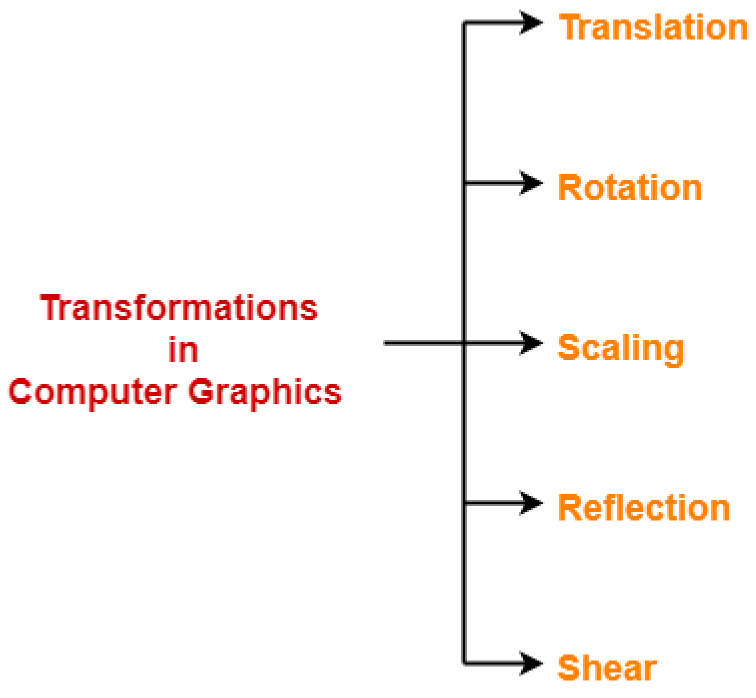
In Computer graphics,

Transformation is a process of modifying and re-positioning the existing graphics.

- 3D Transformations take place in a three dimensional plane.
- 3D Transformations are important and a bit more complex than 2D Transformations.
- Transformations are helpful in changing the position, size, orientation, shape etc of the object.

## Transformation Techniques-

In computer graphics, various transformation techniques are-




1. **Translation**
2. **Rotation**
3. **Scaling**
4. **Reflection**
5. **Shear**

In this article, we will discuss about 3D Translation in Computer Graphics.

## 3D Translation in Computer Graphics-

In Computer graphics,

3D Translation is a process of moving an object from one position to another in a three dimensional plane.

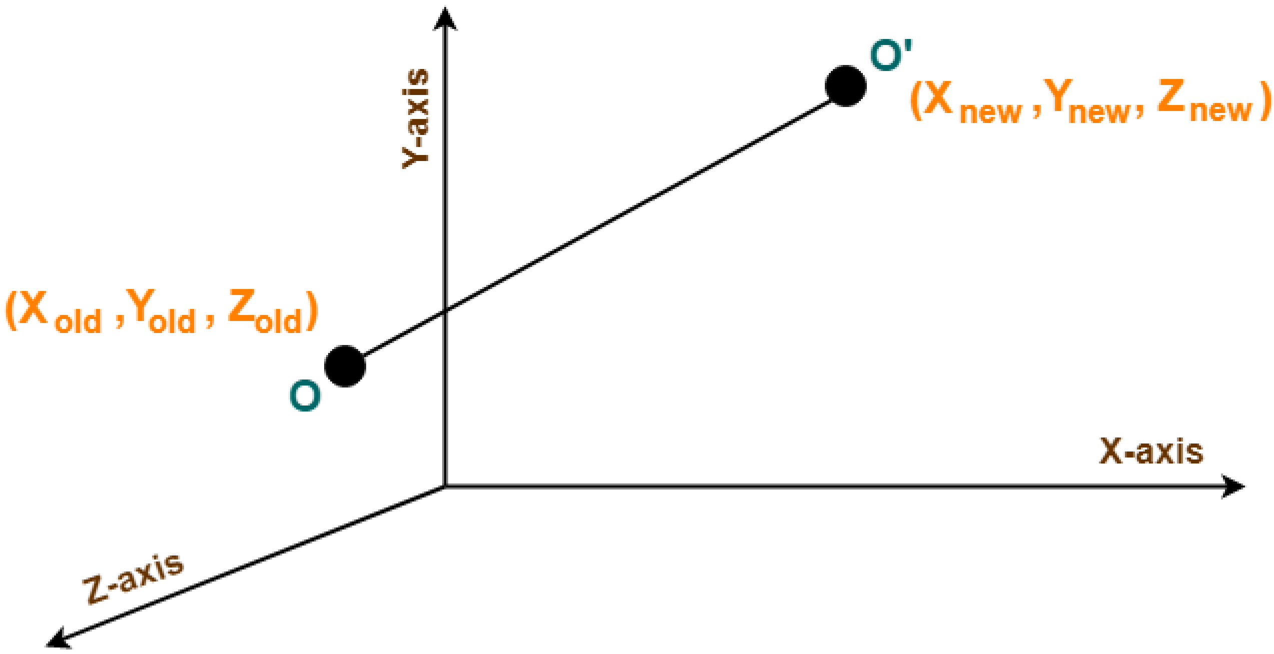
Consider a point object O has to be moved from one position to another in a 3D plane.

Let-

- Initial coordinates of the object O = ($X_{old}$, $Y_{old}$, $Z_{old}$)
- New coordinates of the object O after translation = ($X_{new}$, $Y_{new}$, $Z_{old}$)
- Translation vector or Shift vector = ($T_x$, $T_y$, $T_z$)

Given a Translation vector ($T_x$, $T_y$, $T_z$)-

- $T_x$ defines the distance the $X_{old}$ coordinate has to be moved.
- $T_y$ defines the distance the $Y_{old}$ coordinate has to be moved.
- $T_z$ defines the distance the $Z_{old}$ coordinate has to be moved.




3D Translation in Computer Graphics

This translation is achieved by adding the translation coordinates to the old coordinates of the object as-

- $X_{new} = X_{old} + T_x$ (This denotes translation towards X axis)
- $Y_{new} = Y_{old} + T_y$ (This denotes translation towards Y axis)
- $Z_{new} = Z_{old} + T_z$ (This denotes translation towards Z axis)

In Matrix form, the above translation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Translation Matrix

**Also Read- 2D Translation in Computer Graphics**

## PRACTICE PROBLEM BASED ON 3D TRANSLATION IN COMPUTER GRAPHICS-

### Problem-

Given a 3D object with coordinate points A(0, 3, 1), B(3, 3, 2), C(3, 0, 0), D(0, 0, 0). Apply the translation with the distance 1 towards X axis, 1 towards Y axis and 2 towards Z axis and obtain the new coordinates of the object.

### Solution-

Given-

- Old coordinates of the object = A (0, 3, 1), B(3, 3, 2), C(3, 0, 0), D(0, 0, 0)
- Translation vector = $(T_x, T_y, T_z) = (1, 1, 2)$

### For Coordinates A(0, 3, 1)

Let the new coordinates of A = $(X_{new}, Y_{new}, Z_{new})$.

Applying the translation equations, we have-

- $X_{new} = X_{old} + T_x = 0 + 1 = 1$
- $Y_{new} = Y_{old} + T_y = 3 + 1 = 4$
- $Z_{new} = Z_{old} + T_z = 1 + 2 = 3$

Thus, New coordinates of A = (1, 4, 3).

**For Coordinates B(3, 3, 2)**

Let the new coordinates of B = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the translation equations, we have-

- $X_{new} = X_{old} + T_x = 3 + 1 = 4$
- $Y_{new} = Y_{old} + T_y = 3 + 1 = 4$
- $Z_{new} = Z_{old} + T_z = 2 + 2 = 4$

Thus, New coordinates of B = (4, 4, 4).

**For Coordinates C(3, 0, 0)**

Let the new coordinates of C = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the translation equations, we have-

- $X_{new} = X_{old} + T_x = 3 + 1 = 4$
- $Y_{new} = Y_{old} + T_y = 0 + 1 = 1$
- $Z_{new} = Z_{old} + T_z = 0 + 2 = 2$

Thus, New coordinates of C = (4, 1, 2).

**For Coordinates D(0, 0, 0)**

Let the new coordinates of D = ($X_{new}$, $Y_{new}$, $Z_{new}$).

Applying the translation equations, we have-

- $X_{new} = X_{old} + T_x = 0 + 1 = 1$
- $Y_{new} = Y_{old} + T_y = 0 + 1 = 1$
- $Z_{new} = Z_{old} + T_z = 0 + 2 = 2$

Thus, New coordinates of D = (1, 1, 2).

Thus, New coordinates of the object = A (1, 4, 3), B(4, 4, 4), C(4, 1, 2), D(1, 1, 2).a

# UNIT -2

Wireframe Modeling
Surface Modeling

## Why draw 3D Models?

- 3D models are easier to interpret.
- Simulation under real-life conditions.
- Less expensive than building a physical model.
- 3D models can be used to perform finite element analysis (stress, deflection, thermal…..).
- 3D models can be used directly in manufacturing, Computer Numerical Control (CNC).
- Can be used for presentations and marketing.

## History of Geometric Modeling

| Year | Modeler | Developer |
|---|---|---|
| 1972 | PAP, PADL-I, PADL2 | Univ. of Rochester, Voelcker & Requicha |
| 1973 | Build-I<br>Build-II | Braid's CAD Group in Cambridge, UK |
| 1973 | TIPS-I | Hokkaido University, Japan |
| 1975 | GLIDE-I | Eastman's Group in CMU, USA |
| 1975 | Euler Ops<br>Winged Edge, B-rep | Baumgart, Stanford Univ., USA |
| 1981 | Romulus | Evans and Sutherland, 1st commercial |

## 3D Modeling

There are three basic types of three-dimensional computer geometric modeling methods:

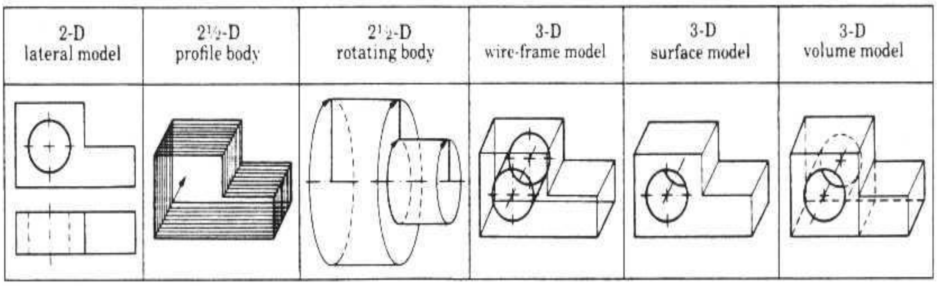
- Wireframe modeling
- Surface modeling
- Solid modeling

## Types of modeling for CAD

Axisymmetric Model

Surface Model

Solid Model

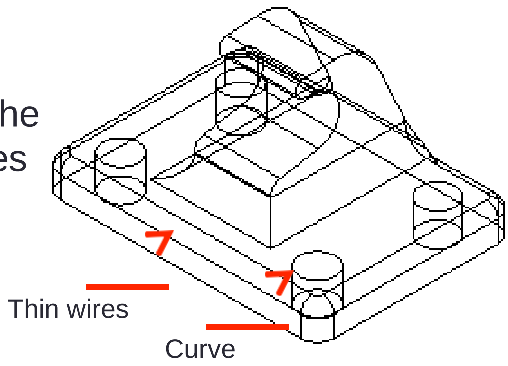



## WIREFRAME MODELING

## Wireframe Modeling

A wireframe representation is a 3-D line drawing of an object showing only the edges without any side surface in between.

A frame constructed from thin wires representing the edges and projected lines and curves.




## Wireframe Modeling

A ***wireframe model*** of an object is the simplest and represents mathematically in the computers.

It is most commonly used technique and all commercial CAD/CAM systems are wire-frame based.

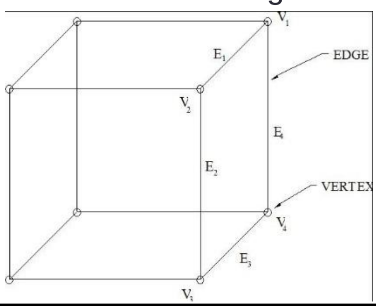
Basic wire-frame entities can be divided into analytic and synthetic entities.

## Need ?

- Geometry display by modeling systems
- Visualization of motion (simple animations)
- Modeling of geometries such as projected profiles and revolutions.
- 2D drafting

## Wireframe Modeling

- Contains information about the locations of all the points (vertices) and edges in space coordinates.
- Each vertex is defined by x, y, z coordinate.
- Edges are defined by a pair of vertices.
- Faces are defined as three or more edges.
- Wireframe is a collection of edges, there is no skin defining the area between the edges.



## Advantages

- Can quickly and efficiently convey information than multiview drawings.
- The only lines seen are the intersections of surfaces.
- Can be used for finite element analysis.
- Can be used as input for CNC machines to generate simple parts.
- Contain most of the information needed to create surface, solid and higher order models

## Disadvantages

- Geometric entities are lines and curves in 3D
- Volume or surfaces of object not defined
- Easy to store and display
- Hard to interpret – ambiguous
- Hidden lines are not removed
- For complex items, the result can be a jumble of lines that is impossible to determine.
- No ability to determine computationally information such as the line of intersect between two faces of intersecting models.

## Disadvantages

- Subject human interpretation
- Complex objects with many edges become confusing
- Lengthy and verbose to define
- Do not represent an actual solids (no surface and volume).
- Cannot model complex curved surfaces.
- Cannot be used to calculate dynamic properties.
- Limited ability for checking interference between mating parts (typically visual only)
- No guarantee that the model definition is correct, complete or manufacturable

## Wireframe Modeling

There are two important aspects to the use of wire-frame models in CAD.

The first is the computer representation of an object, and this is concerned with the structure needed to encode a wire-frame model.

The second is concerned with the computational procedures needed to produce and manipulate the viewing or visualization of this representation.

## Wireframe Modeling

A computer representation of a wire-frame structure consists essentially of two types of information.

The first is termed metric or geometric data which relate to the 3D coordinate positions of the wire-frame node′ points in space.

The second is concerned with the connectivity or topological data, which relate pairs of points together as edges.

## Real objects & wire-frame models

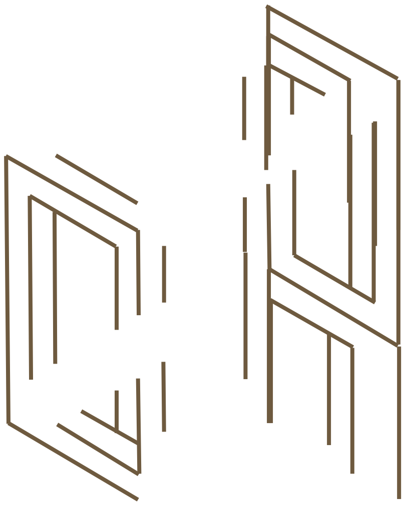
However, in most cases 3D wire-frames are used to model objects in the real world, providing (amongst other things) a tool to aid object visualization.

Even supposing that all of these requirements are satisfied, three-of a sphere is dimensional wire-frames still have failings in two major areas: *lack of validity* and *ambiguity.*

## Non-sense object

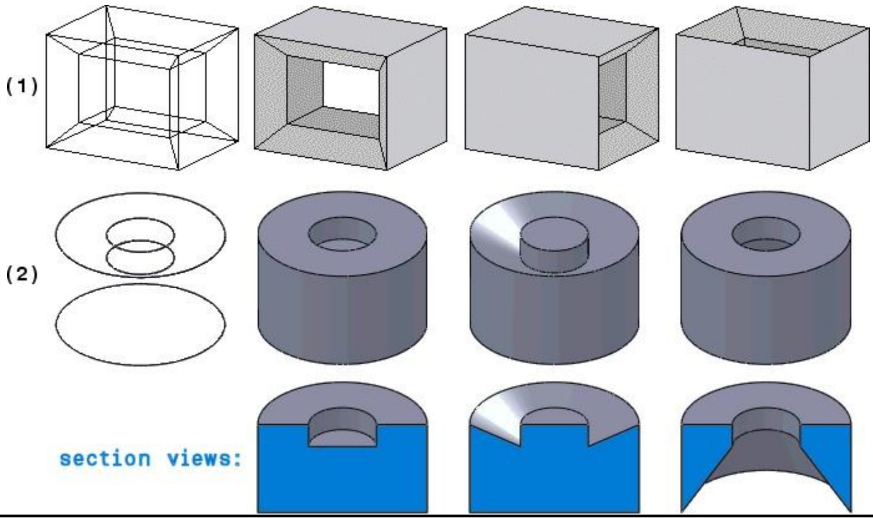
This can best is to select a represented by the type of structure, consisting of simple (vertex) points and linear edges.

Each point is well defined in 3D space; each edge is associated with just two end-points; the face edges all form closed loops; and no faces are self-intersecting.



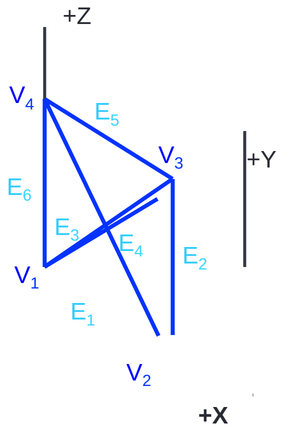
## Ambiguous wireframe model

The wire-frame can represent any of the other real objects shown in the figure.



# Basic database information

- Vertices
  - Coordinate values for vertices
- Edges
  - Vertices associated with edges (endpoints)
- Faces
  - Loops (faces) formed by edges




*Wireframe tetrehedron*

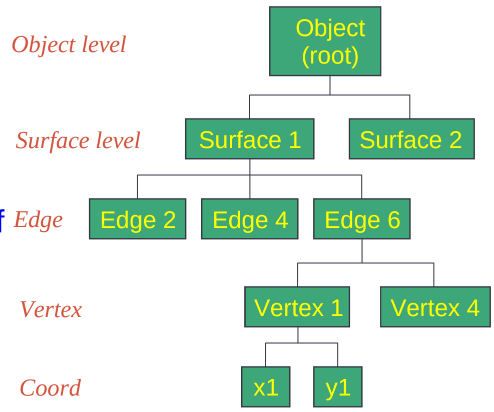
# Basic database structures

- Relational
  - A set of lists, uses arrays for storage
- Hierarchical
  - A trees structure, think of a company's executive structure
- Network
  - Use data pointers

| Vertex list | Edge list |
|---|---|
| $V_1(0,0,0)$ | $E_1[V_1,V_2]$ |
| $V_2(1,0,0)$ | $E_2[V_2,V_3]$ |
| $V_3(0,1,0)$ | $E_3[V_3,V_1]$ |
| $V_4(0,0,1)$ | $E_4[V_2,V_4]$ |
| | $E_5[V_4,V_3]$ |
| | $E_6[V_1,V_4]$ |

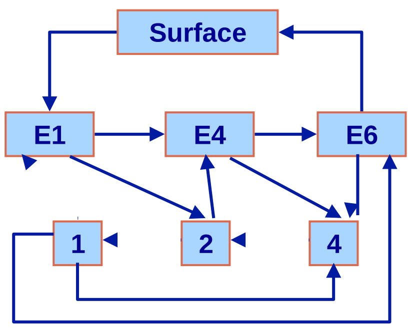
# Basic database structures

- Relational
  - A set of lists, uses arrays for storage
- Hierarchical
  - A trees structure, think of a company's executive structure
- Network
  - Use data pointers




# Basic database structures

- Relational
  - A set of lists, uses arrays for storage
- Hierarchical
  - A trees structure, think of a company's executive structure
- Network
  - Use data pointers



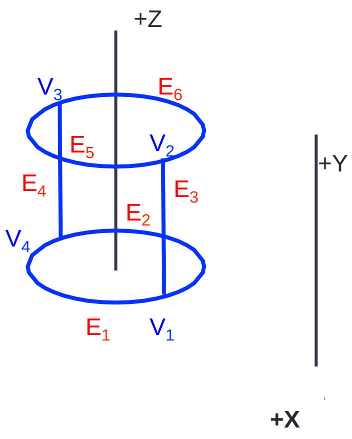
## Model validity criteria (restrict to linear edges)

- Each vertex has 3 coordinate values
- Each edge delimited by two vertices
- Minimum of 3 edges must intersect at each vertex
- Minimum of 3 edges required to define a loop (face)

## Wireframe Cylinder

In the example show, the model definition (edges, vertices) was chosen to meet a validity criteria set.

- Three edges intersect at a vertex
- Edges delimited by two vertices



# SURFACE MODELING

## Surface Modeling

*A surface model represents the skin of an object, these skins have no thickness or material type.*

- Surface models define the surface features, as well as the edges, of objects.
- A mathematical function describes the path of a curve (parametric techniques).
- Surfaces are edited as single entities.

## Surface Modeling

***Advantages:***

- Eliminates ambiguity and non-uniqueness present in wireframe models by hiding lines not seen.
- Renders the model for better visualization and presentation, objects appear more realistic.
- Provides the surface geometry for CNC machining.
- Provides the geometry needed for mold and die design.
- Can be used to design and analyze complex free-formed surfaces *(ship hulls, airplane fuselages, car bodies, …).*
- Surface properties such as roughness, color and reflectivity can be assigned and demonstrated.

## Surface Modeling

***Disadvantages:***

- Surface models provide no information about the inside of an object.
- Complicated computation, depending on the number of surfaces

## Surface Modeling

Surface modeling was essentially the situation in the early 1940s.

The pressures of wartime production, particularly in the aircraft industry, led to changes in the way the geometry was represented.

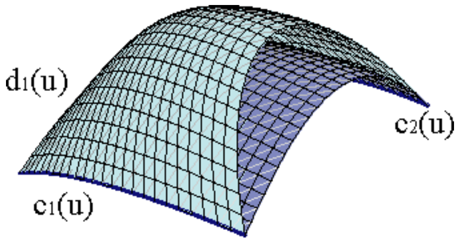
## Surface Modeling

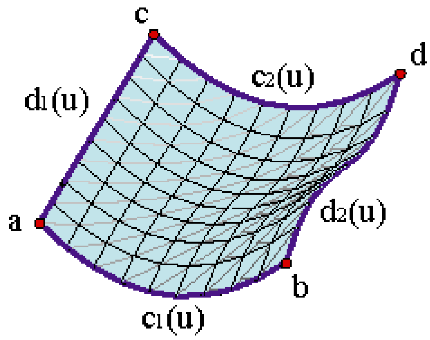
N. Lidbro [1956] describes a system used by Saab-Scania in Sweden in the 1950s.

N. Lidbro, (1956) "Modern Aircraft Geometry: A Description of the Mathematical Method used at SAAB, Sweden, for Aircraft Dimensioning and Shape Determination", Aircraft Engineering and Aerospace Technology, Vol. 28 Iss: 11, pp. 388 - 394

## Surface Modeling

The use of parametric techniques became popular in the 1960s, largely due to the pioneering work of Coons [1964].

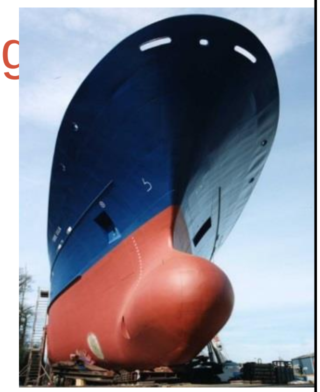



## Surface Modeling

<u>Widespread in</u>

- ➢➢shipbuilding
- ➢➢automotive manufacture
- ➢➢shoe industry
- ➢➢companies manufacturing
  - forgings,
  - castings.



## Surface Modeling

It is particularly useful for modeling objects, which can be modeled as

- shells,
- car body panels,
- aircraft fuselages
- fan blades.
- wind turbine



## Surface Modeling

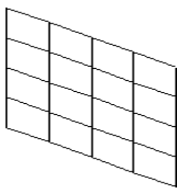
CAD software packages use two basic methods for the creation of surfaces.

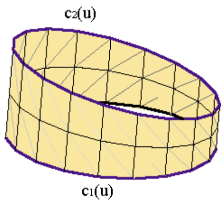
| | |
|---|---|
| The first begins with construction curves (splines) from which the 3D surface is then swept (section along guide rail) or meshed (lofted) through. | The second method is direct creation of the surface with manipulation of the surface poles/control points. |

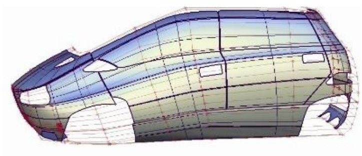
## Surface: definition

- An area bounded by an identifiable perimeter.
- In Computer Graphics, is an area within which every position is defined by mathematical methods.

| Planar surface | Cylindrical/conic | Sculptured |
|---|---|---|





## Surface modeling

As a surface model defines adequate data on a component's surface geometry hidden lines and surfaces are readily and automatically removed as required.

This gives rise to non-ambiguous visualization of the object when viewed from any direction.

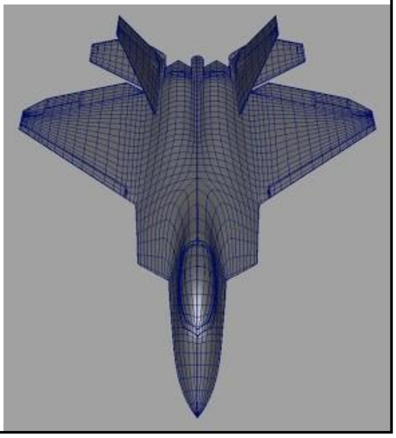
The software calculates the amount of light reflected back to the user from different areas on the surface and each area is color filled with varying shades accordingly.

# Surface Modeling

Complex objects such as car or airplane body can not be achieved utilizing wireframe modeling.

Surface modeling are used in

- ✓✓calculating mass porperties
- ✓✓checking for interference
- ✓✓between mating parts
- ✓✓generating cross-section views
- ✓✓generating finite elements meshes
- ✓✓generating NC tool paths for
- ✓✓continuous path machining

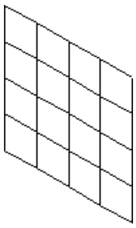


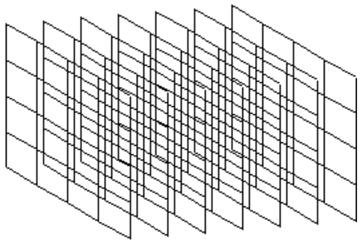
# Advantages of Solid & Surface Modelling

| Solid Modelling | Surface Modelling |
|---|---|
| Easy to learn/use | More flexible in modelling complex geometry |
| Parametric/associative capabilities | Interactive modelling capabilities |
| Quicker creation and updating of assemblies | Quicker creation and updating of complex components and tooling |
| Excellent for creating functional models | Excellent for creating aesthetic or ergonomic free-form models |

## Plane surface

This is the simplest surface. It requires three non-coincident points to define an infinite plane. The plane surface can be used to generate cross-sectional views by intersecting a surface model with it, generate cross sections for mass property calculations, or other similar applications where a plane is needed.





## Plane surface

A plane surface that passes through three points, $P_0$, $P_1$ and $P_2$ is given by

$$\mathbf{P}(u,v) = \mathbf{P}_0 + u(\mathbf{P}_1 - \mathbf{P}_0) + v(\mathbf{P}_2 - \mathbf{P}_0), \quad 0 \le u \le 1, \ 0 \le v \le 1$$

The surface normal vector then is

$$\mathbf{n}(u,v) = \frac{(\mathbf{P}_1 - \mathbf{P}_0) \times (\mathbf{P}_2 - \mathbf{P}_0)}{|(\mathbf{P}_1 - \mathbf{P}_0) \times (\mathbf{P}_2 - \mathbf{P}_0)|}, \quad 0 \le u \le 1, \ 0 \le v \le 1$$

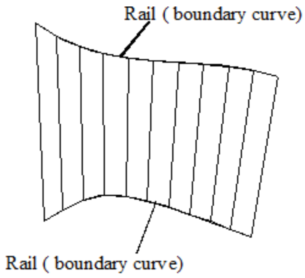
Once the normal unit vector is known, the surface can be also expressed in nonparametric form as

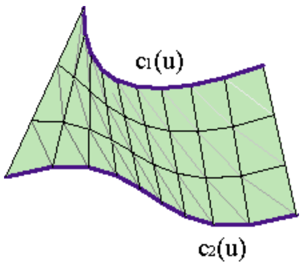
$$(\mathbf{P} - \mathbf{P}_0) \cdot \mathbf{n} = 0$$

## Ruled Surface (lofted surface)

A ruled surface is generated by joining two space curves (rails) with a straight line (ruling or generator). If two curves are denoted by F(u) and G(u) respectively, for a value of u, then the parametric equation is given by

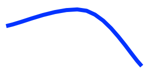
$$\mathbf{P}(u,v) = (1-v)\mathbf{G}(u) + v\mathbf{F}(u), \quad 0 \le u \le 1,\ 0 \le v \le 1$$

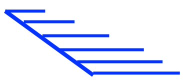


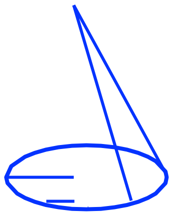

## Ruled surface

- Linear interpolation between two bounding geometric elements (curves).
- Elemental division the same for each curve.
- Bounding curves must both be either geometrically open (line, arc) or closed (circle, ellipse).
- Curvature in one direction only.





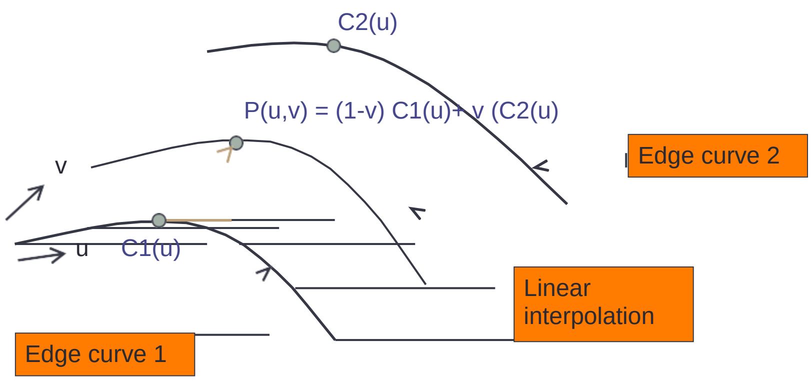
Both geometries open (line & arc)



Both geometries closed (circle & point *)

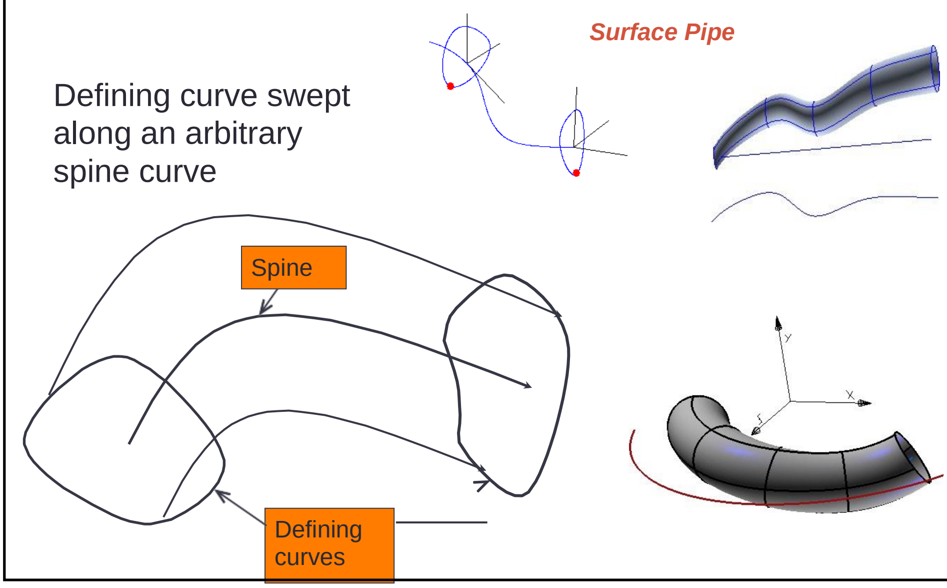
# Ruled Surface

- Linear interpolation between two edge curves
- Created by lofting through cross sections




# Lofted Surface

Defining curve swept along an arbitrary spine curve

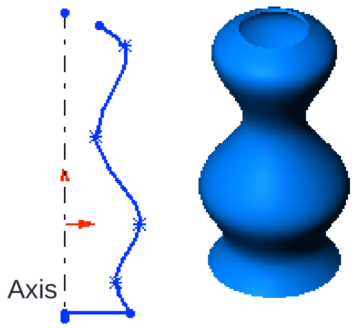


# Revolved surface

A revolved surface is generated a space curve about an axis of rotation. The parametric representation of the curve in the working coordinate system that has x- and y-axes on the perpendicular plane to the axis of rotation that is z-axis can be expressed by

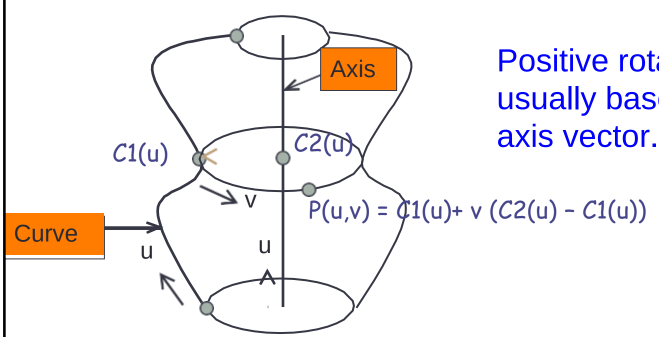
$$\mathbf{P}(u,v) = r_z(u)\cos v\mathbf{n}_x + r_z(u)\sin v\mathbf{n}_y + z(u)\mathbf{n}_z, \quad 0 \le u \le 1, \; 0 \le v \le 2\pi$$

This is an axisymmetric surface that can model axisymmetric objects.
It is generated by rotating a planar wireframe entity in space about the axis of symmetry a certain angle.




# Revolution of Surface

Surface of revolution requires:
– a shape curve (must be continuous)
– a specified angle
– an axis defined in 3D modelspace.




Positive rotation direction usually based upon direction of axis vector.

## Tabulated cylinder

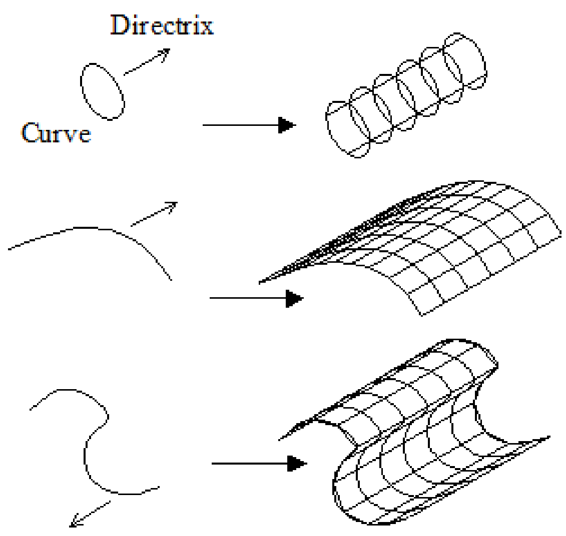
A tabulated cylinder is generated by moving a straight line along a space curve. The parametric representation of the curve can be expressed by

$$\mathbf{P}(u,v) = \mathbf{F}(u) + v\mathbf{n}_v, \quad 0 \le u \le u_{max}, \; 0 \le v \le v_{max}$$

where $n_v$ is the unit vector in the direction of generatix.

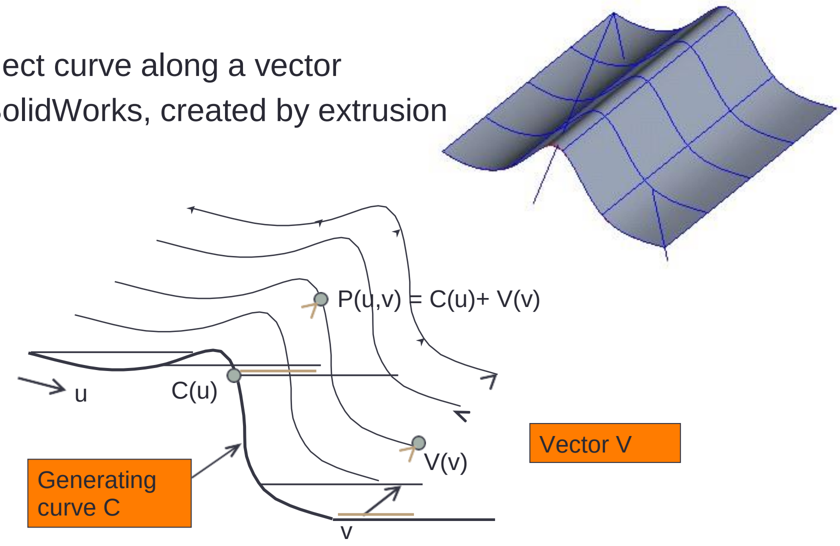
- Defined by projecting a shape curve (or profile) along a direction vector.
- Curvature in one direction only (along shape curve), linear in other direction.

## Tabulated Cylinder



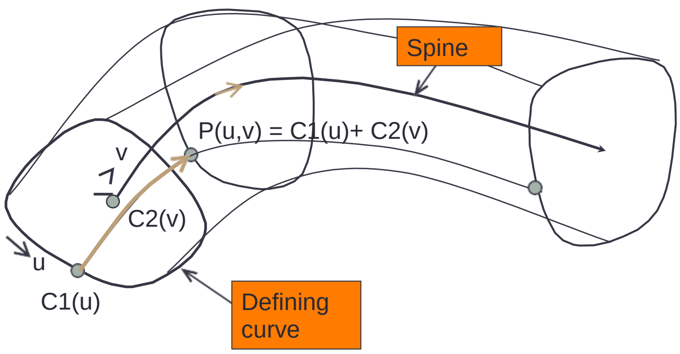
# Tabulated Cylinder

- Project curve along a vector
- In SolidWorks, created by extrusion




# Swept Surface

- Defining curve swept along an arbitrary spine curve



## Hermite Bicubic Surface

This surface is formed by Hermite cubic splines running in two different directions. It interpolates to a finite number of data points to form the surface. The bicubic interpolation is an invaluable tool used in image processing.

$$\mathbf{P}(u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}C_{ij}u^i v^j, \quad 0\le u,v\le 1$$

In a matrix form it can be expressed.

$$\mathbf{P}(u,v)=\mathbf{U}^T[\mathbf{C}]\mathbf{V}\equiv\mathbf{U}^T[\mathbf{M}_H][\mathbf{B}][\mathbf{M}_H]^T\mathbf{V}$$

where

$$\mathbf{U}^T=\{u^3,u^2,u,1\},\ \mathbf{V}=\{v^3,v^2,v,1\}^T$$

$[\mathbf{B}]$ has 4 x 4 = 16

## Hermite Bicubic Surface

Applying the boundary conditions (continuity and tangency) at data points determines all coefficients. Here

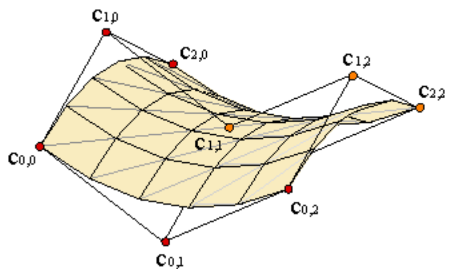
$$[\mathbf{B}]=\begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} & \left.\frac{\partial \mathbf{P}}{\partial v}\right|_{00} & \left.\frac{\partial \mathbf{P}}{\partial v}\right|_{01} \\ \mathbf{P}_{10} & \mathbf{P}_{11} & \left.\frac{\partial \mathbf{P}}{\partial v}\right|_{10} & \left.\frac{\partial \mathbf{P}}{\partial v}\right|_{11} \\ \left.\frac{\partial \mathbf{P}}{\partial u}\right|_{00} & \left.\frac{\partial \mathbf{P}}{\partial u}\right|_{01} & \left.\frac{\partial^2 \mathbf{P}}{\partial u\partial v}\right|_{00} & \left.\frac{\partial^2 \mathbf{P}}{\partial u\partial v}\right|_{01} \\ \left.\frac{\partial \mathbf{P}}{\partial u}\right|_{10} & \left.\frac{\partial \mathbf{P}}{\partial u}\right|_{11} & \left.\frac{\partial^2 \mathbf{P}}{\partial u\partial v}\right|_{10} & \left.\frac{\partial^2 \mathbf{P}}{\partial u\partial v}\right|_{11} \end{bmatrix}$$

This matrix can be determined by imposing the smoothness conditions at data points joining two adjacent panels.

# Bezier Surface

This is a surface that approximates given input data.

It is different from the previous surfaces in that it is a synthetic surface, it does not pass through all given data points.

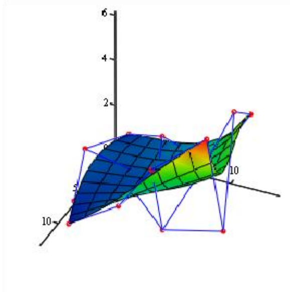



Bezier surface is an extension of Bezier curve and interpolates to a finite number of data point. It can be expressed as

$$\mathbf{P}(u,v)=\sum_{j=0}^{n}\sum_{i=0}^{m}\mathbf{P}_{i,j}B_{i,m}(u)B_{j,n}(v),\quad 0\le u,v\le 1$$

# Bezier Surfaces

- Bezier curves can be extended to surfaces
- Same problems as for Bezier curves:
  - no local modification possible
  - smooth transition between adjacent patches difficult to achieve

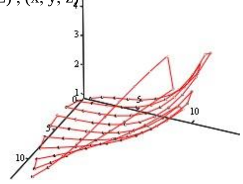


$$C_{n,i} := \frac{n!}{i!\cdot n-i!} \qquad B(u,i) := C_{n,i}\cdot u^{i}\cdot(1-u)^{n-i}$$

$$P(u,v,r) := \sum_{i=0}^{m}\sum_{j=0}^{n}\left(p_{i,j}\right)_r\cdot B(u,i)\cdot B(v,j)$$

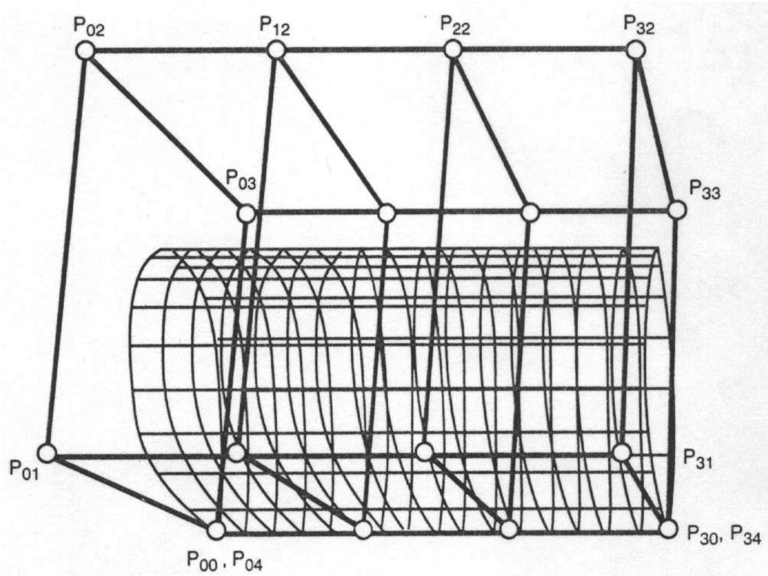
Isoparametric curves used for tool path generation.

(X, Y, Z) , (x, y, z)



(x,y, z)

## Closed Bezier patch

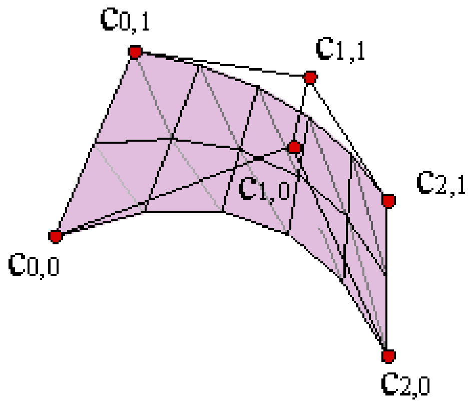


# B-Spline Surface

This is a surface that can approximate or interpolate given input data.

It is a synthetic surface.

It is a general surface like the Bezier surface but with the advantage of permitting local control of the surface.

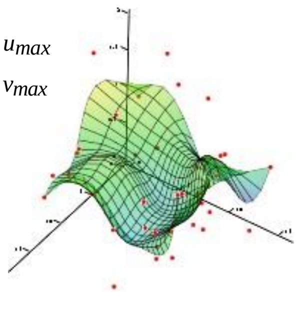



# B-Spline Surfaces

- As with curves, B-spline surfaces are a generalization of Bezier surfaces
- The surface approximates a control polygon
- Open and closed surfaces can be represented

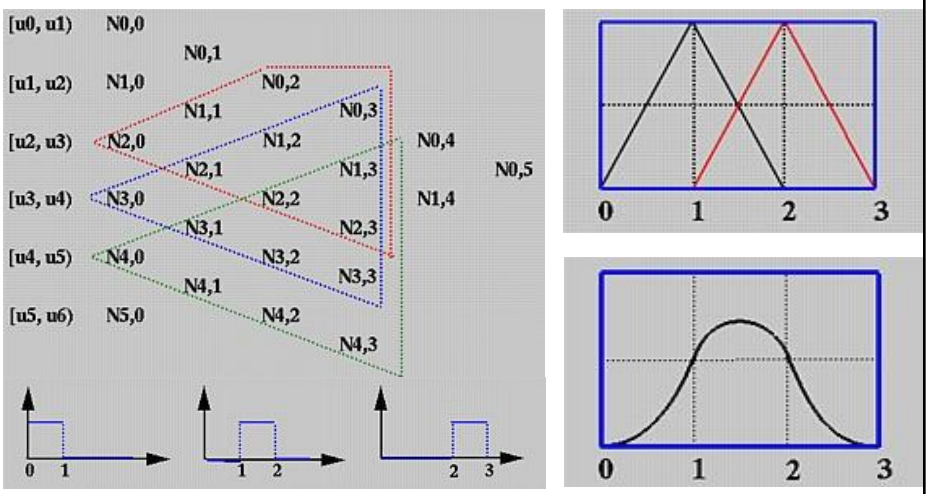
$$P(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} P_{ij} \cdot N_{i,k}(u) \cdot N_{j,l}(v) \qquad 0 \le u \le u_{max} \quad 0 \le v \le v_{max}$$

$$N_{i,k}(u) = (u - u_i) \cdot \frac{N_{i,k-1}}{u_{i+k-1} - u_i} + (u_{i+k} - u) \cdot \frac{N_{i+1,k-1}}{u_{i+k} - u_{i+1}}$$



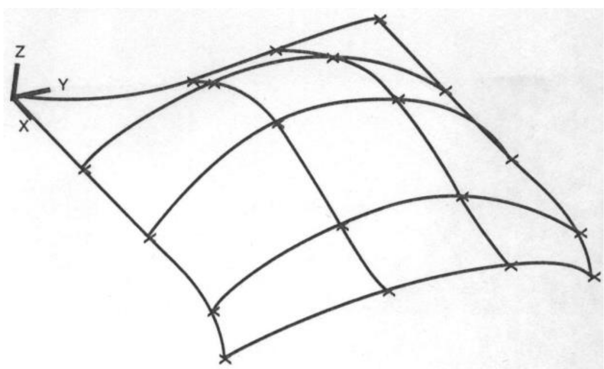
B-Spline Surface

## B-Spline Surfaces




## B-spline surfaces

- Rational parametric surfaces
- Sphere, cylinders, cones

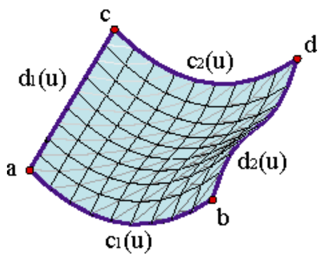


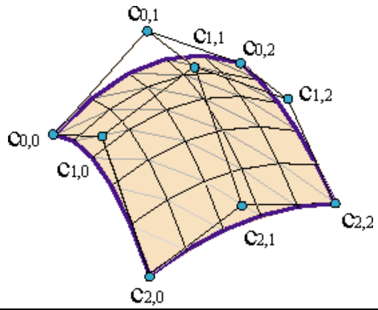

Interpolated B-spline surface patch

# Coons Surface

The above surfaces are used with either open boundaries or given data points.

The **Coons patch** is used to create a surface using curves that form closed boundaries .

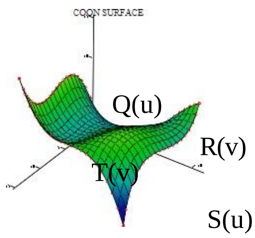



**Steven Anson Coons** was an early pioneer in the field of computer graphical methods. He was a professor at the MIT. He had a vision of interactive computer graphics as a design tool to aid the engineer.

# Linearly Blended Coons Patch

- Surface is defined by linearly interpolating between the boundary curves
- Simple, but doesn't allow adjacent patches to be joined smoothly.



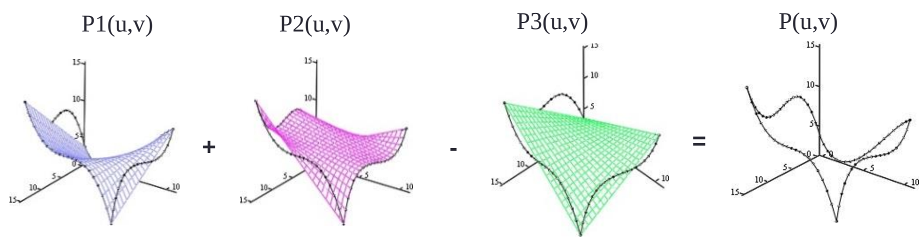

$$P1(u, v) := (1-u)\cdot T(v) + u\cdot R(v)$$

$$P2(u, v) := (1-v)\cdot Q(u) + v\cdot S(u)$$

$$P3(u, v) := (1-v)\cdot[(1-u)\cdot T(0) + u\cdot R(0)] + v\cdot[(1-u)\cdot T(1) + u\cdot R(1)]$$

$$P(u, v) := P1(u, v) + P2(u, v) - P3(u, v)$$

.



# Gorden Surface

A spline - blended surface interpolating the network of curves, known also as Gordon surface

$$G(u,v) = G_1(u,v) + G_2(u,v) - G_{12}(u,v)$$

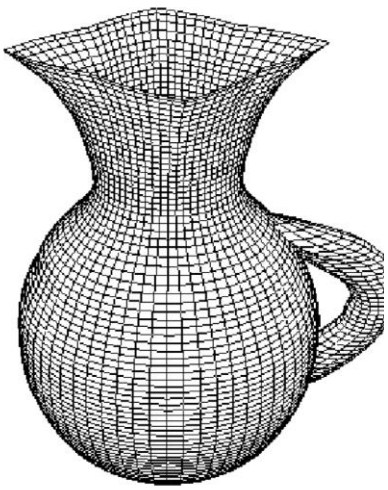
where

$$\begin{aligned} G_1(u,v) &= \sum_{i=1}^{m} G(u_i,v) L_i^m(u) \\ G_2(u,v) &= \sum_{j=1}^{n} G(u,v_j) L_j^n(v) \\ G_{12}(u,v) &= \sum_{i=1}^{m}\sum_{j=1}^{n} G(u_i,v_j) L_i^m(u) L_j^n(v) \end{aligned}$$

and $L_i^m(u)$ are blending functions satisfying:

$$L_i^m(u_i) = 1, \; L_i^m(u_k) = 0, i \neq k$$

# Gorden Surface



Wireframe Model

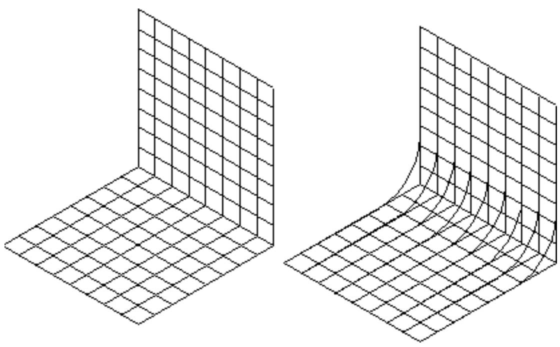
Shaded Model

## Other Surfaces

In addition to surfaces mentioned, other surfaces are blending surface, offset surface, triangular patches, sculptured (or free-form) surface that is a collection of interconnected and bounded parametric patches together with blending and interpolation formulas, and rational parametric surface.

## Fillet Surface

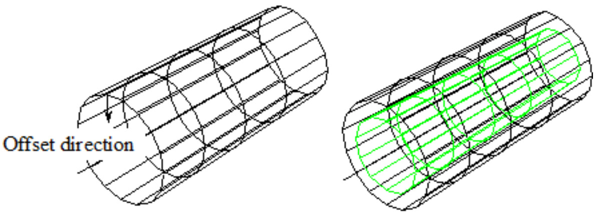
This is a B-spline surface that blends two surfaces together.



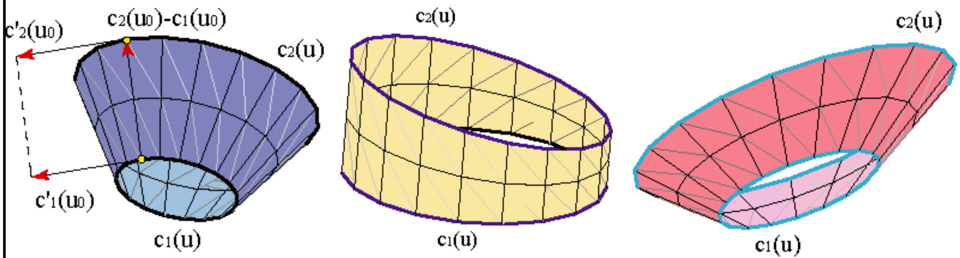
## Offset Surface

Existing surfaces can be offset to create new ones identical in shape but may have different dimensions.

To create a hollow cylinder, the outer or inner cylinder can be created using a cylinder command and the other one can be created by an offset command.




## Other surface generation methods




**Developed Surface** **Cylindrical Surface** **Conical Surface**

# Finite Element Analysis

David Roylance
Department of Materials Science and Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139

February 28, 2001

## Introduction

Finite element analysis (FEA) has become commonplace in recent years, and is now the basis of a multibillion dollar per year industry. Numerical solutions to even very complicated stress problems can now be obtained routinely using FEA, and the method is so important that even introductory treatments of Mechanics of Materials – such as these modules – should outline its principal features.

In spite of the great power of FEA, the disadvantages of computer solutions must be kept in mind when using this and similar methods: they do not necessarily reveal how the stresses are influenced by important problem variables such as materials properties and geometrical features, and errors in input data can produce wildly incorrect results that may be overlooked by the analyst. Perhaps the most important function of theoretical modeling is that of sharpening the designer's intuition; users of finite element codes should plan their strategy toward this end, supplementing the computer simulation with as much closed-form and experimental analysis as possible.

Finite element codes are less complicated than many of the word processing and spreadsheet packages found on modern microcomputers. Nevertheless, they are complex enough that most users do not find it effective to program their own code. A number of prewritten commercial codes are available, representing a broad price range and compatible with machines from microcomputers to supercomputers[1]. However, users with specialized needs should not necessarily shy away from code development, and may find the code sources available in such texts as that by Zienkiewicz[2] to be a useful starting point. Most finite element software is written in Fortran, but some newer codes such as `felt` are in C or other more modern programming languages.

In practice, a finite element analysis usually consists of three principal steps:

1. *Preprocessing:* The user constructs a *model* of the part to be analyzed in which the geometry is divided into a number of discrete subregions, or "elements," connected at discrete points called "nodes." Certain of these nodes will have fixed displacements, and others will have prescribed loads. These models can be extremely time consuming to prepare, and commercial codes vie with one another to have the most user-friendly graphical "preprocessor" to assist in this rather tedious chore. Some of these preprocessors can overlay a mesh on a preexisting CAD file, so that finite element analysis can be done conveniently as part of the computerized drafting-and-design process.

[1] C.A. Brebbia, ed., *Finite Element Systems, A Handbook,* Springer-Verlag, Berlin, 1982.
[2] O.C. Zienkiewicz and R.L. Taylor, *The Finite Element Method,* McGraw-Hill Co., London, 1989.

2. *Analysis:* The dataset prepared by the preprocessor is used as input to the finite element code itself, which constructs and solves a system of linear or nonlinear algebraic equations

$$\mathbf{K}_{ij}\mathbf{u}_j = \mathbf{f}_i$$

   where $\mathbf{u}$ and $\mathbf{f}$ are the displacements and externally applied forces at the nodal points. The formation of the $\mathbf{K}$ matrix is dependent on the type of problem being attacked, and this module will outline the approach for truss and linear elastic stress analyses. Commercial codes may have very large element libraries, with elements appropriate to a wide range of problem types. One of FEA's principal advantages is that many problem types can be addressed with the same code, merely by specifying the appropriate element types from the library.

3. *Postprocessing:* In the earlier days of finite element analysis, the user would pore through reams of numbers generated by the code, listing displacements and stresses at discrete positions within the model. It is easy to miss important trends and hot spots this way, and modern codes use graphical displays to assist in visualizing the results. A typical postprocessor display overlays colored contours representing stress levels on the model, showing a full-field picture similar to that of photoelastic or moire experimental results.

The operation of a specific code is usually detailed in the documentation accompanying the software, and vendors of the more expensive codes will often offer workshops or training sessions as well to help users learn the intricacies of code operation. One problem users may have even after this training is that the code tends to be a "black box" whose inner workings are not understood. In this module we will outline the principles underlying most current finite element stress analysis codes, limiting the discussion to linear elastic analysis for now. Understanding this theory helps dissipate the black-box syndrome, and also serves to summarize the analytical foundations of solid mechanics.

## Matrix analysis of trusses

Pin-jointed trusses, discussed more fully in Module 5, provide a good way to introduce FEA concepts. The static analysis of trusses can be carried out exactly, and the equations of even complicated trusses can be assembled in a matrix form amenable to numerical solution. This approach, sometimes called "matrix analysis," provided the foundation of early FEA development.

Matrix analysis of trusses operates by considering the stiffness of each truss element one at a time, and then using these stiffnesses to determine the forces that are set up in the truss elements by the displacements of the joints, usually called "nodes" in finite element analysis. Then noting that the sum of the forces contributed by each element to a node must equal the force that is externally applied to that node, we can assemble a sequence of linear algebraic equations in which the nodal displacements are the unknowns and the applied nodal forces are known quantities. These equations are conveniently written in matrix form, which gives the method its name:

$$\begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1n} \\ K_{21} & K_{22} & \cdots & K_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ K_{n1} & K_{n2} & \cdots & K_{nn} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{Bmatrix}$$

Here $u_i$ and $f_j$ indicate the deflection at the $i^{th}$ node and the force at the $j^{th}$ node (these would actually be vector quantities, with subcomponents along each coordinate axis). The $K_{ij}$ coefficient array is called the *global stiffness matrix*, with the $ij$ component being physically the influence of the $j^{th}$ displacement on the $i^{th}$ force. The matrix equations can be abbreviated as

$$K_{ij} u_j = f_i \quad \text{or} \quad \mathbf{K}\mathbf{u} = \mathbf{f} \tag{1}$$

using either subscripts or boldface to indicate vector and matrix quantities.

Either the force externally applied or the displacement is known at the outset for each node, and it is impossible to specify simultaneously both an arbitrary displacement *and* a force on a given node. These prescribed nodal forces and displacements are the boundary conditions of the problem. It is the task of analysis to determine the forces that accompany the imposed displacements, and the displacements at the nodes where known external forces are applied.

## Stiffness matrix for a single truss element

As a first step in developing a set of matrix equations that describe truss systems, we need a relationship between the forces and displacements at each end of a single truss element. Consider such an element in the $x - y$ plane as shown in Fig. 1, attached to nodes numbered $i$ and $j$ and inclined at an angle $\theta$ from the horizontal.

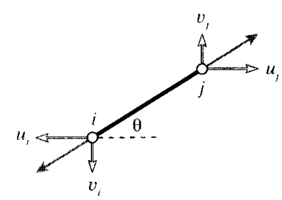



Figure 1: Individual truss element.

Considering the elongation vector $\delta$ to be resolved in directions along and transverse to the element, the elongation in the truss element can be written in terms of the differences in the displacements of its end points:

$$\delta = (u_j \cos\theta + v_j \sin\theta) - (u_i \cos\theta + v_i \sin\theta)$$

where $u$ and $v$ are the horizontal and vertical components of the deflections, respectively. (The displacements at node $i$ drawn in Fig. 1 are negative.) This relation can be written in matrix form as:

$$\delta = \begin{bmatrix} -c & -s & c & s \end{bmatrix} \begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \end{Bmatrix}$$

Here $c = \cos\theta$ and $s = \sin\theta$.

The axial force $P$ that accompanies the elongation $\delta$ is given by Hooke's law for linear elastic bodies as $P = (AE/L)\delta$. The horizontal and vertical nodal forces are shown in Fig. 2; these can be written in terms of the total axial force as:

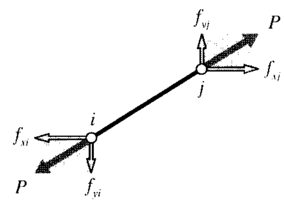



Figure 2: Components of nodal force.

$$\begin{Bmatrix} f_{xi} \\ f_{yi} \\ f_{xj} \\ f_{yj} \end{Bmatrix} = \begin{Bmatrix} -c \\ -s \\ c \\ s \end{Bmatrix} P = \begin{Bmatrix} -c \\ -s \\ c \\ s \end{Bmatrix} \frac{AE}{L}\delta$$

$$= \begin{Bmatrix} -c \\ -s \\ c \\ s \end{Bmatrix} \frac{AE}{L} \begin{bmatrix} -c & -s & c & s \end{bmatrix} \begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \end{Bmatrix}$$

Carrying out the matrix multiplication:

$$\boxed{\begin{Bmatrix} f_{xi} \\ f_{yi} \\ f_{xj} \\ f_{yj} \end{Bmatrix} = \frac{AE}{L} \begin{bmatrix} c^2 & cs & -c^2 & -cs \\ cs & s^2 & -cs & -s^2 \\ -c^2 & -cs & c^2 & cs \\ -cs & -s^2 & cs & s^2 \end{bmatrix} \begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \end{Bmatrix}} \tag{2}$$

The quantity in brackets, multiplied by $AE/L$, is known as the "element stiffness matrix" $k_{ij}$. Each of its terms has a physical significance, representing the contribution of one of the displacements to one of the forces. The global system of equations is formed by combining the element stiffness matrices from each truss element in turn, so their computation is central to the method of matrix structural analysis. The principal difference between the matrix truss method and the general finite element method is in how the element stiffness matrices are formed; most of the other computer operations are the same.

### Assembly of multiple element contributions

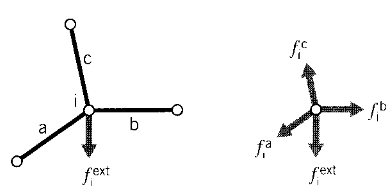



Figure 3: Element contributions to total nodal force.

The next step is to consider an assemblage of many truss elements connected by pin joints. Each element meeting at a joint, or node, will contribute a force there as dictated by the displacements of both that element's nodes (see Fig. 3). To maintain static equilibrium, all element force contributions $f_i^{elem}$ at a given node must sum to the force $f_i^{ext}$ that is externally applied at that node:

$$f_i^{ext} = \sum_{elem} f_i^{elem} = (\sum_{elem} k_{ij}^{elem} u_j) = (\sum_{elem} k_{ij}^{elem}) u_j = K_{ij} u_j$$

Each element stiffness matrix $k_{ij}^{elem}$ is added to the appropriate location of the overall, or "global" stiffness matrix $K_{ij}$ that relates all of the truss displacements and forces. This process is called "assembly." The index numbers in the above relation must be the "global" numbers assigned to the truss structure as a whole. However, it is generally convenient to compute the individual element stiffness matrices using a local scheme, and then to have the computer convert to global numbers when assembling the individual matrices.

---

**Example 1**

The assembly process is at the heart of the finite element method, and it is worthwhile to do a simple case by hand to see how it really works. Consider the two-element truss problem of Fig. 4, with the nodes being assigned arbitrary "global" numbers from 1 to 3. Since each node can in general move in two directions, there are $3 \times 2 = 6$ total degrees of freedom in the problem. The global stiffness matrix will then be a $6 \times 6$ array relating the six displacements to the six externally applied forces. Only one of the displacements is unknown in this case, since all but the vertical displacement of node 2 (degree of freedom number 4) is constrained to be zero. Figure 4 shows a workable listing of the global numbers, and also "local" numbers for each individual element.

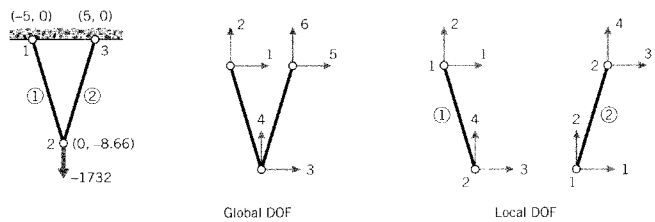



Figure 4: Global and local numbering for the two-element truss.

Using the local numbers, the 4×4 element stiffness matrix of each of the two elements can be evaluated according to Eqn. 2. The inclination angle is calculated from the nodal coordinates as

$$\theta = \tan^{-1} \frac{y_2 - y_1}{x_2 - x_1}$$

The resulting matrix for element 1 is:

$$k^{(1)} = \begin{bmatrix} 25.00 & -43.30 & -25.00 & 43.30 \\ -43.30 & 75.00 & 43.30 & -75.00 \\ -25.00 & 43.30 & 25.00 & -43.30 \\ 43.30 & -75.00 & -43.30 & 75.00 \end{bmatrix} \times 10^3$$

and for element 2:

$$k^{(2)} = \begin{bmatrix} 25.00 & 43.30 & -25.00 & -43.30 \\ 43.30 & 75.00 & -43.30 & -75.00 \\ -25.00 & -43.30 & 25.00 & 43.30 \\ -43.30 & -75.00 & 43.30 & 75.00 \end{bmatrix} \times 10^3$$

(It is important the units be consistent; here lengths are in inches, forces in pounds, and moduli in psi. The modulus of both elements is $E = 10$ Mpsi and both have area $A = 0.1$ in$^2$.) These matrices have rows and columns numbered from 1 to 4, corresponding to the local degrees of freedom of the element.

However, each of the local degrees of freedom can be matched to one of the global degrees of the overall problem. By inspection of Fig. 4, we can form the following table that maps local to global numbers:

| local | global, element 1 | global, element 2 |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 2 | 4 |
| 3 | 3 | 5 |
| 4 | 4 | 6 |

Using this table, we see for instance that the second degree of freedom for element 2 is the fourth degree of freedom in the global numbering system, and the third local degree of freedom corresponds to the fifth global degree of freedom. Hence the value in the second row and third column of the element stiffness matrix of element 2, denoted $k_{23}^{(2)}$, should be added into the position in the fourth row and fifth column of the $6 \times 6$ global stiffness matrix. We write this as

$$k_{23}^{(2)} \longrightarrow K_{4,5}$$

Each of the sixteen positions in the stiffness matrix of each of the two elements must be added into the global matrix according to the mapping given by the table. This gives the result

$$K = \begin{bmatrix} k_{11}^{(1)} & k_{12}^{(1)} & k_{13}^{(1)} & k_{14}^{(1)} & 0 & 0 \\ k_{21}^{(1)} & k_{22}^{(1)} & k_{23}^{(1)} & k_{24}^{(1)} & 0 & 0 \\ k_{31}^{(1)} & k_{32}^{(1)} & k_{33}^{(1)} + k_{11}^{(2)} & k_{34}^{(1)} + k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} \\ k_{41}^{(1)} & k_{42}^{(1)} & k_{43}^{(1)} + k_{21}^{(2)} & k_{44}^{(1)} + k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} \\ 0 & 0 & k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} \\ 0 & 0 & k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} \end{bmatrix}$$

This matrix premultiplies the vector of nodal displacements according to Eqn. 1 to yield the vector of externally applied nodal forces. The full system equations, taking into account the known forces and displacements, are then

$$10^3 \begin{bmatrix} 25.0 & -43.3 & -25.0 & 43.3 & 0.0 & 0.00 \\ -43.3 & 75.0 & 43.3 & -75.0 & 0.0 & 0.00 \\ -25.0 & 43.3 & 50.0 & 0.0 & -25.0 & -43.30 \\ 43.3 & -75.0 & 0.0 & 150.0 & -43.3 & -75.00 \\ 0.0 & 0.0 & -25.0 & -43.3 & 25.0 & 43.30 \\ 0.0 & 0.0 & -43.3 & -75.0 & 43.3 & 75.00 \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ 0 \\ u_4 \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ -1732 \\ f_5 \\ f_5 \end{Bmatrix}$$

Note that either the force or the displacement for each degree of freedom is known, with the accompanying displacement or force being unknown. Here only one of the displacements ($u_4$) is unknown, but in most problems the unknown displacements far outnumber the unknown forces. Note also that only those elements that are physically connected to a given node can contribute a force to that node. In most cases, this results in the global stiffness matrix containing many zeroes corresponding to nodal pairs that are not spanned by an element. Effective computer implementations will take advantage of the matrix sparseness to conserve memory and reduce execution time.

In larger problems the matrix equations are solved for the unknown displacements and forces by Gaussian reduction or other techniques. In this two-element problem, the solution for the single unknown displacement can be written down almost from inspection. Multiplying out the fourth row of the system, we have

$$0 + 0 + 0 + 150 \times 10^3 u_4 + 0 + 0 = -1732$$

$$u_4 = -1732/150 \times 10^3 = -0.01155 \text{ in}$$

Now any of the unknown forces can be obtained directly. Multiplying out the first row for instance gives

$$0 + 0 + 0 + (43.4)(-0.0115) \times 10^3 + 0 + 0 = f_1$$

$$f_1 = -500 \text{ lb}$$

The negative sign here indicates the horizontal force on global node #1 is to the left, opposite the direction assumed in Fig. 4.

---

The process of cycling through each element to form the element stiffness matrix, assembling the element matrix into the correct positions in the global matrix, solving the equations for displacements and then back-multiplying to compute the forces, and printing the results can be automated to make a very versatile computer code.

Larger-scale truss (and other) finite element analysis are best done with a dedicated computer code, and an excellent one for learning the method is available from the web at `http://felt.sourceforge.net/`. This code, named `felt`, was authored by Jason Gobat and Darren Atkinson for educational use, and incorporates a number of novel features to promote user-friendliness. Complete information describing this code, as well as the C-language source and a number of trial runs and auxiliary code modules is available via their web pages. If you have access to X-window workstations, a graphical shell named `velvet` is available as well.

---

**Example 2**

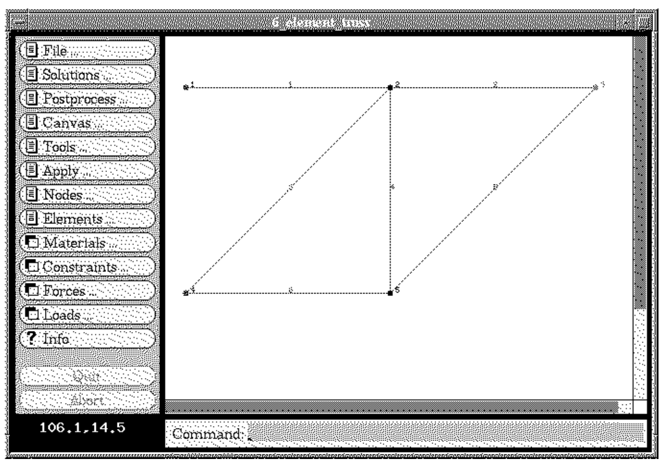



Figure 5: The six-element truss, as developed in the `velvet/felt` FEA graphical interface.

To illustrate how this code operates for a somewhat larger problem, consider the six-element truss of Fig. 5, which was analyzed in Module 5 both by the joint-at-a-time free body analysis approach and by Castigliano's method.

The input dataset, which can be written manually or developed graphically in `velvet`, employs parsing techniques to simplify what can be a very tedious and error-prone step in finite element analysis. The dataset for this 6-element truss is:

```
problem description
nodes=5 elements=6

nodes
1  x=0 y=100 z=0 constraint=pin
```

```
2  x=100 y=100 z=0 constraint=planar
3  x=200 y=100 z=0 force=P
4  x=0 y=0 z=0 constraint=pin
5  x=100 y=0 z=0 constraint=planar

truss elements
1  nodes=[1,2] material=steel
2  nodes=[2,3]
3  nodes=[4,2]
4  nodes=[2,5]
5  nodes=[5,3]
6  nodes=[4,5]

material properties
steel E=3e+07 A=0.5

distributed loads

constraints
free Tx=u Ty=u Tz=u Rx=u Ry=u Rz=u
pin Tx=c Ty=c Tz=c Rx=u Ry=u Rz=u
planar Tx=u Ty=u Tz=c Rx=u Ry=u Rz=u

forces
P Fy=-1000

end
```

The meaning of these lines should be fairly evident on inspection, although the `felt` documentation should be consulted for more detail. The output produced by `felt` for these data is:

```
**  **

Nodal Displacements
-----------------------------------------------------------------------------
Node #      DOF 1       DOF 2       DOF 3       DOF 4       DOF 5       DOF 6
-----------------------------------------------------------------------------
  1             0           0           0           0           0           0
  2      0.013333    -0.03219           0           0           0           0
  3          0.02   -0.084379           0           0           0           0
  4             0           0           0           0           0           0
  5    -0.0066667   -0.038856           0           0           0           0

Element Stresses
-------------------------------------------------------------------------------
  1:         4000
  2:         2000
  3:      -2828.4
  4:         2000
  5:      -2828.4
  6:        -2000


Reaction Forces
-----------------------------------
Node #     DOF     Reaction Force
-----------------------------------
```

```
  1        Tx              -2000
  1        Ty                  0
  1        Tz                  0
  2        Tz                  0
  3        Tz                  0
  4        Tx               2000
  4        Ty               1000
  4        Tz                  0
  5        Tz                  0

Material Usage Summary
--------------------------
Material: steel
Number:   6
Length:   682.8427
Mass:       0.0000

Total mass:     0.0000
```

The vertical displacement of node 3 (the `DOF 2` value) is -0.0844, the same value obtained by the closed-form methods of Module 5. Figure 6 shows the `velvet` graphical output for the truss deflections (greatly magnified).

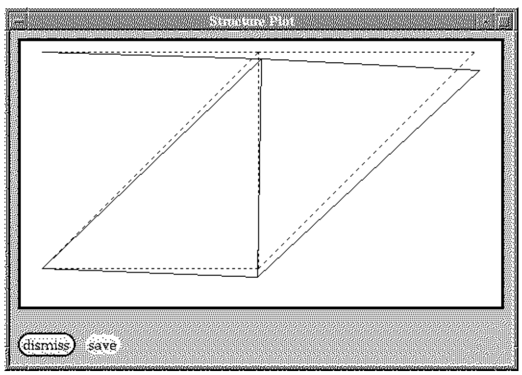



Figure 6: The 6-element truss in its original and deformed shape.

---

## General Stress Analysis

The element stiffness matrix could be formed exactly for truss elements, but this is not the case for general stress analysis situations. The relation between nodal forces and displacements are not known in advance for general two- or three-dimensional stress analysis problems, and an approximate relation must be used in order to write out an element stiffness matrix.

In the usual "displacement formulation" of the finite element method, the governing equations are combined so as to have only displacements appearing as unknowns; this can be done by using the Hookean constitutive equations to replace the stresses in the equilibrium equations by the strains, and then using the kinematic equations to replace the strains by the displacements. This gives

$$\mathbf{L}^{\mathrm{T}}\sigma = \mathbf{L}^{\mathrm{T}}\mathbf{D}\epsilon = \mathbf{L}^{\mathrm{T}}\mathbf{D}\mathbf{L}\mathbf{u} = \mathbf{0} \tag{3}$$

Of course, it is often impossible to solve these equations in closed form for the irregular boundary conditions encountered in practical problems. However, the equations are amenable to discretization and solution by numerical techniques such as finite differences or finite elements.

Finite element methods are one of several approximate numerical techniques available for the solution of engineering boundary value problems. Problems in the mechanics of materials often lead to equations of this type, and finite element methods have a number of advantages in handling them. The method is particularly well suited to problems with irregular geometries and boundary conditions, and it can be implemented in general computer codes that can be used for many different problems.

To obtain a numerical solution for the stress analysis problem, let us postulate a function $\tilde{\mathbf{u}}(x, y)$ as an approximation to $\mathbf{u}$:

$$\tilde{\mathbf{u}}(x, y) \approx \mathbf{u}(x, y) \tag{4}$$

Many different forms might be adopted for the approximation $\tilde{\mathbf{u}}$. The finite element method discretizes the solution domain into an assemblage of subregions, or "elements," each of which has its own approximating functions. Specifically, the approximation for the displacement $\tilde{\mathbf{u}}(x, y)$ within an element is written as a combination of the (as yet unknown) displacements at the nodes belonging to that element:

$$\tilde{\mathbf{u}}(x, y) = N_j(x, y)\mathbf{u}_j \tag{5}$$

Here the index $j$ ranges over the element's nodes, $\mathbf{u}_j$ are the nodal displacements, and the $N_j$ are "interpolation functions." These interpolation functions are usually simple polynomials (generally linear, quadratic, or occasionally cubic polynomials) that are chosen to become unity at node $j$ and zero at the other element nodes. The interpolation functions can be evaluated at any position within the element by means of standard subroutines, so the approximate displacement at any position within the element can be obtained in terms of the nodal displacements directly from Eqn. 5.

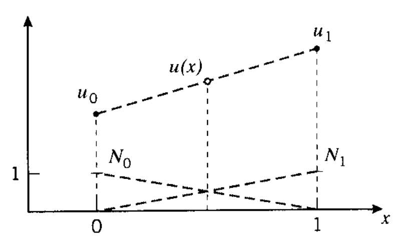



Figure 7: Interpolation in one dimension.

The interpolation concept can be illustrated by asking how we might guess the value of a function $u(x)$ at an arbitrary point $x$ located between two nodes at $x = 0$ and $x = 1$, assuming we know somehow the nodal values $u(0)$ and $u(1)$. We might assume that as a reasonable approximation $u(x)$ simply varies linearly between these two values as shown in Fig. 7, and write

$$u(x) \approx \tilde{u}(x) = u_0\,(1 - x) + u_1\,(x)$$

or

$$\tilde{u}(x) = u_0\, N_0(x) + u_1\, N_1(x), \quad \begin{cases} N_0(x) = (1-x) \\ N_1(x) = x \end{cases}$$

Here the $N_0$ and $N_1$ are the linear interpolation functions for this one-dimensional approximation. Finite element codes have subroutines that extend this interpolation concept to two and three dimensions.

Approximations for the strain and stress follow directly from the displacements:

$$\tilde{\boldsymbol{\epsilon}} = \mathbf{L}\tilde{\mathbf{u}} = \mathbf{L}N_j\mathbf{u}_j \equiv \mathbf{B}_j\mathbf{u}_j \tag{6}$$

$$\tilde{\boldsymbol{\sigma}} = \mathbf{D}\tilde{\boldsymbol{\epsilon}} = \mathbf{D}\mathbf{B}_j\mathbf{u}_j \tag{7}$$

where $\mathbf{B}_j(x,y) = \mathbf{L}N_j(x,y)$ is an array of derivatives of the interpolation functions:

$$\mathbf{B}_j = \begin{bmatrix} N_{j,x} & 0 \\ 0 & N_{j,y} \\ N_{j,y} & N_{j,x} \end{bmatrix} \tag{8}$$

A "virtual work" argument can now be invoked to relate the nodal displacement $\mathbf{u}_j$ appearing at node $j$ to the forces applied externally at node $i$: if a small, or "virtual," displacement $\delta\mathbf{u}_i$ is superimposed on node $i$, the increase in strain energy $\delta U$ within an element connected to that node is given by:

$$\delta U = \int_V \delta\boldsymbol{\epsilon}^T\,\boldsymbol{\sigma}\,dV \tag{9}$$

where $V$ is the volume of the element. Using the approximate strain obtained from the interpolated displacements, $\delta\tilde{\boldsymbol{\epsilon}} = \mathbf{B}_i\delta\mathbf{u}_i$ is the approximate virtual increase in strain induced by the virtual nodal displacement. Using Eqn. 7 and the matrix identity $(\mathbf{AB})^{\mathrm{T}} = \mathbf{B}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}$, we have:

$$\delta U = \delta\mathbf{u}_i^T \int_V \mathbf{B}_i^T\mathbf{D}\mathbf{B}_j\,dV\,\mathbf{u}_j \tag{10}$$

(The nodal displacements $\delta\mathbf{u}_i^T$ and $\mathbf{u}_j$ are not functions of $x$ and $y$, and so can be brought from inside the integral.) The increase in strain energy $\delta U$ must equal the work done by the nodal forces; this is:

$$\delta W = \delta\mathbf{u}_i^T\mathbf{f}_i \tag{11}$$

Equating Eqns. 10 and 11 and canceling the common factor $\delta\mathbf{u}_i^T$, we have:

$$\boxed{\left[\int_V \mathbf{B}_i^T\mathbf{D}\mathbf{B}_j\,dV\right]\,\mathbf{u}_j = \mathbf{f}_i} \tag{12}$$

This is of the desired form $\mathbf{k}_{ij}\mathbf{u}_j = \mathbf{f}_i$, where $\mathbf{k}_{ij} = \int_V \mathbf{B}_i^T\mathbf{D}\mathbf{B}_j\,dV$ is the element stiffness.

Finally, the integral in Eqn. 12 must be replaced by a numerical equivalent acceptable to the computer. Gauss-Legendre numerical integration is commonly used in finite element codes for this purpose, since that technique provides a high ratio of accuracy to computing effort. Stated briefly, the integration consists of evaluating the integrand at optimally selected integration points within the element, and forming a weighted summation of the integrand values at these points. In the case of integration over two-dimensional element areas, this can be written:

$$\int_A f(x,y)\,dA \approx \sum_l f(x_l,y_l)w_l \tag{13}$$

The location of the sampling points $x_l, y_l$ and the associated weights $w_l$ are provided by standard subroutines. In most modern codes, these routines map the element into a convenient shape, determine the integration points and weights in the transformed coordinate frame, and then map the results back to the original frame. The functions $N_j$ used earlier for interpolation can be used for the mapping as well, achieving a significant economy in coding. This yields what are known as "numerically integrated isoparametric elements," and these are a mainstay of the finite element industry.

Equation 12, with the integral replaced by numerical integrations of the form in Eqn. 13, is the finite element counterpart of Eqn. 3, the differential governing equation. The computer will carry out the analysis by looping over each element, and within each element looping over the individual integration points. At each integration point the components of the element stiffness matrix $\mathbf{k}_{ij}$ are computed according to Eqn. 12, and added into the appropriate positions of the $\mathbf{K}_{ij}$ global stiffness matrix as was done in the assembly step of matrix truss method described in the previous section. It can be appreciated that a good deal of computation is involved just in forming the terms of the stiffness matrix, and that the finite element method could never have been developed without convenient and inexpensive access to a computer.

## Stresses around a circular hole

We have considered the problem of a uniaxially loaded plate containing a circular hole in previous modules, including the theoretical Kirsch solution (Module 16) and experimental determinations using both photoelastic and moire methods (Module 17). This problem is of practical importance —- for instance, we have noted the dangerous stress concentration that appears near rivet holes — and it is also quite demanding in both theoretical and numerical analyses. Since the stresses rise sharply near the hole, a finite element grid must be refined there in order to produce acceptable results.

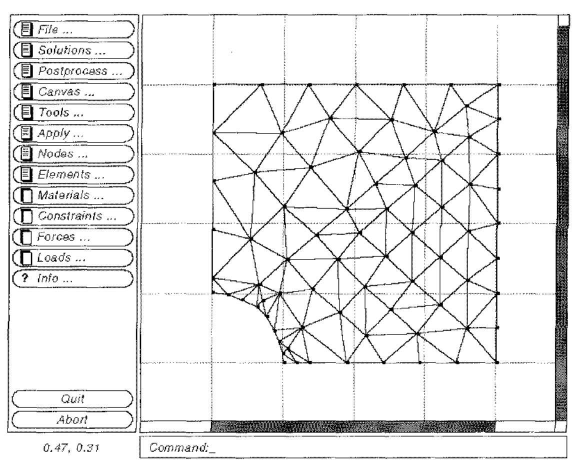



Figure 8: Mesh for circular-hole problem.

Figure 8 shows a mesh of three-noded triangular elements developed by the `felt-velvet` graphical FEA package that can be used to approximate the displacements and stresses around a uniaxially loaded plate containing a circular hole. Since both theoretical and experimental results for this stress field are available as mentioned above, the circular-hole problem is a good one for becoming familiar with code operation.

The user should take advantage of symmetry to reduce problem size whenever possible, and in this case only one quadrant of the problem need be meshed. The center of the hole is kept fixed, so the symmetry requires that nodes along the left edge be allowed to move vertically but not horizontally. Similarly, nodes along the lower edge are constrained vertically but left free to move horizontally. Loads are applied to the nodes along the upper edge, with each load being the resultant of the far-field stress acting along half of the element boundaries between the given node and its neighbors. (The far-field stress is taken as unity.) Portions of the `felt` input dataset for this problem are:

```
problem description
nodes=76 elements=116

nodes
1  x=1 y=-0 z=0 constraint=slide_x
2  x=1.19644 y=-0 z=0
3  x=0.984562 y=0.167939 z=0 constraint=free
4  x=0.940634 y=0.335841 z=0
5  x=1.07888 y=0.235833 z=0
   .
   .
   .
72 x=3.99602 y=3.01892 z=0
73 x=3.99602 y=3.51942 z=0
74 x=3.33267 y=4 z=0
75 x=3.57706 y=3.65664 z=0
76 x=4 y=4 z=0

CSTPlaneStress elements
1  nodes=[13,12,23] material=steel
2  nodes=[67,58,55]
6  nodes=[50,41,40]
   .
   .
   .
7  nodes=[68,67,69] load=load_case_1
8  nodes=[68,58,67]
9  nodes=[57,58,68] load=load_case_1
10 nodes=[57,51,58]
11 nodes=[52,51,57] load=load_case_1
12 nodes=[37,39,52] load=load_case_1
13 nodes=[39,51,52]
   .
   .
   .
116 nodes=[2,3,1]

material properties
steel E=2.05e+11 nu=0.33 t=1

distributed loads
load_case_1 color=red direction=GlobalY values=(1,1) (3,1)
```

```
constraints
free Tx=u Ty=u Tz=u Rx=u Ry=u Rz=u
slide_x color=red Tx=u Ty=c Tz=c Rx=u Ry=u Rz=u
slide_y color=red Tx=c Ty=u Tz=c Rx=u Ry=u Rz=u

end
```

The $y$-displacements and vertical stresses $\sigma_y$ are contoured in Fig. 9(a) and (b) respectively; these should be compared with the photoelastic and moire analyses given in Module 17, Figs. 8 and 10(a). The stress at the integration point closest to the $x$-axis at the hole is computed to be $\sigma_{y,max} = 3.26$, 9% larger than the theoretical value of 3.00. In drawing the contours of Fig. 9b, the postprocessor extrapolated the stresses to the nodes by fitting a least-squares plane through the stresses at all four integration points within the element. This produces an even higher value for the stress concentration factor, 3.593. The user must be aware that graphical postprocessors smooth results that are themselves only approximations, so numerical inaccuracy is a real possibility. Refining the mesh, especially near the region of highest stress gradient at the hole meridian, would reduce this error.

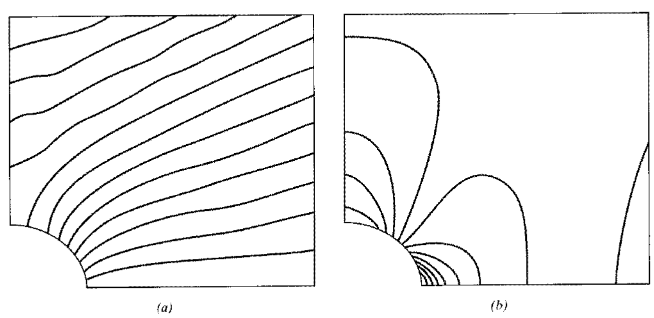



Figure 9: Vertical displacements (a) and stresses (b) as computed for the mesh of Fig. 8.

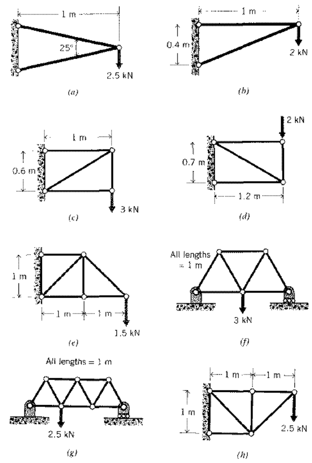
## Problems

1. (a) – (h) Use FEA to determine the force in each element of the trusses drawn below.

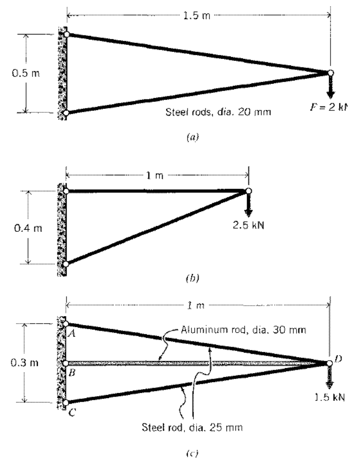
2. (a) – (c) Write out the global stiffness matrices for the trusses listed below, and solve for the unknown forces and displacements. For each element assume $E = 30$ Mpsi and $A = 0.1$ in$^2$.

3. Obtain a plane-stress finite element solution for a cantilevered beam with a single load at the free end. Use arbitrarily chosen (but reasonable) dimensions and material properties. Plot the stresses $\sigma_x$ and $\tau_{xy}$ as functions of $y$ at an arbitrary station along the span; also plot the stresses given by the elementary theory of beam bending (c.f. Module 13) and assess the magnitude of the numerical error.

4. Repeat the previous problem, but with a symmetrically-loaded beam in three-point bending.

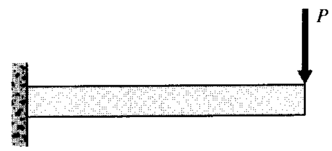



*Prob. 1*

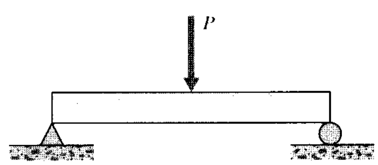



*Prob. 2*

5. Use axisymmetric elements to obtain a finite element solution for the radial stress in a thick-walled pressure vessel (using arbitrary geometry and material parameters). Compare the results with the theoretical solution (c.f. Prob. 2 in Module 16).




*Prob. 3*




*Prob. 4*

# UNIT 3
# Solid Modeling

- ## Wireframe, surface, solid modeling

Solid modeling gives a complete and unambiguous definition of an object, describing not only the shape of the boundaries but also the object's interior and exterior regions.

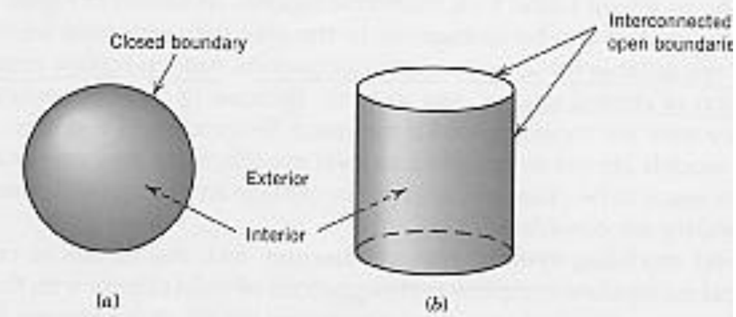



FIGURE 12.2 Examples of (a) closed boundary surface, and (b) open boundary surface.

- ## Disadvantages of wireframe representations

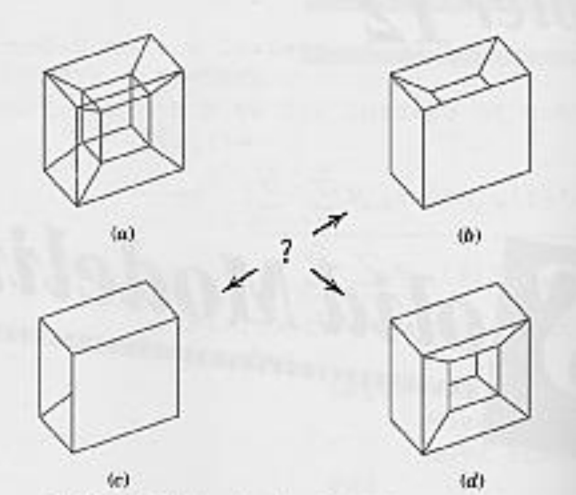



FIGURE 12.1 Ambiguity of wireframe models.

1. Ambiguous in the way to represent an object.
2. Not suitable for
   a. Mass property calculations
   b. Hidden surface removal
   c. Shaded images generation

- ## Solid Representation

- Real world object satisfy specific properties causing them to be

1. **Bounded** – limited boundary, contain interior of the solid
2. **Homogeneously three-dimensional** – no dangling edge or faces, the boundary is always in contact with the interior of the solid.
3. **Finite** – finite in size and limited amount of information (area, mass and volume determinations).

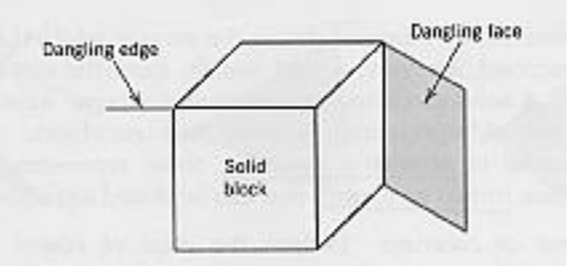



FIGURE 12.3 Homogeneous three-dimensionality avoids dangling edges or faces.

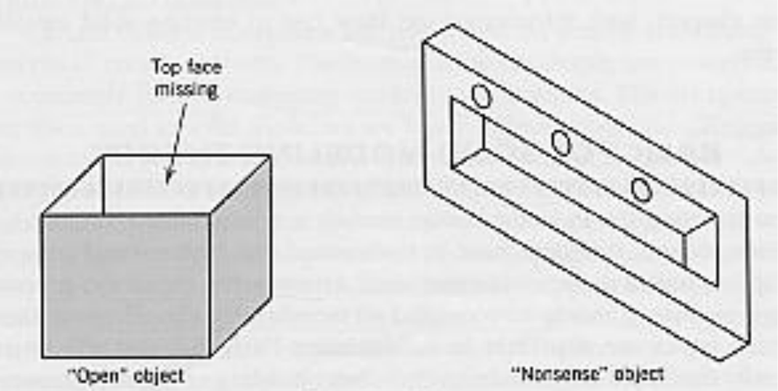



FIGURE 12.4 Examples of invalid objects.

- Formal properties of geometric modeling

1. Domain or coverage – define object classes
2. Validity – legal model
3. Completeness – complete solid with enough data for geometric calculation performed
4. Uniqueness

- Solid models representation schemes

1. CSG (Constructive Solid Geometry)
2. B-Rep (Boundary Representation)
3. Sweeping
4. Spatial Enumeration

- Fundamental geometric principles

1. Geometry
2. Topology
3. Geometric closure
4. Set theory and operations
5. Set membership classification

- Basics of Solid Modeling Theory

The fundamental geometric principles

1. **Geometry and topology** –

**Definition:** Geometry relates to the information containing shape-defining parameters, such as the coordinates of the vertices in a polyhedral object.

**Definition:** Topology describes the connectivity among the various geometric components, i.e. the relational information between the different parts of an object.

(x,y,z) coordinates of vertices → geometry

connectivity matrix → topology

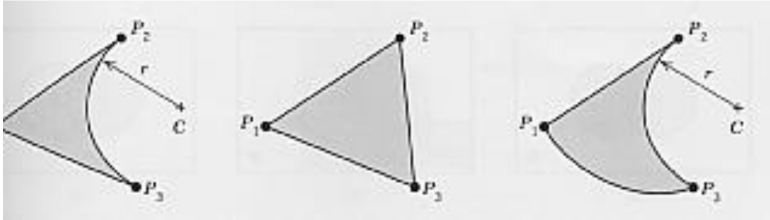



FIGURE 12.5 Variations in the geometric information for the same topology.

2. **Geometric closure** – Bounded, Finite, No dangling
3. **Set Theory** –

   A set is defined as any collection of objects, called “elements” or “members.” Universal set *W*, containing all points in $E^3$ space, and the null set, Ø, no elements.

   Set operations: union (∪), intersection (∩), difference (–).
4. **Regularized set operations** – Boolean operations ensure the validity of geometric models, avoiding the creation of nonsense objects.

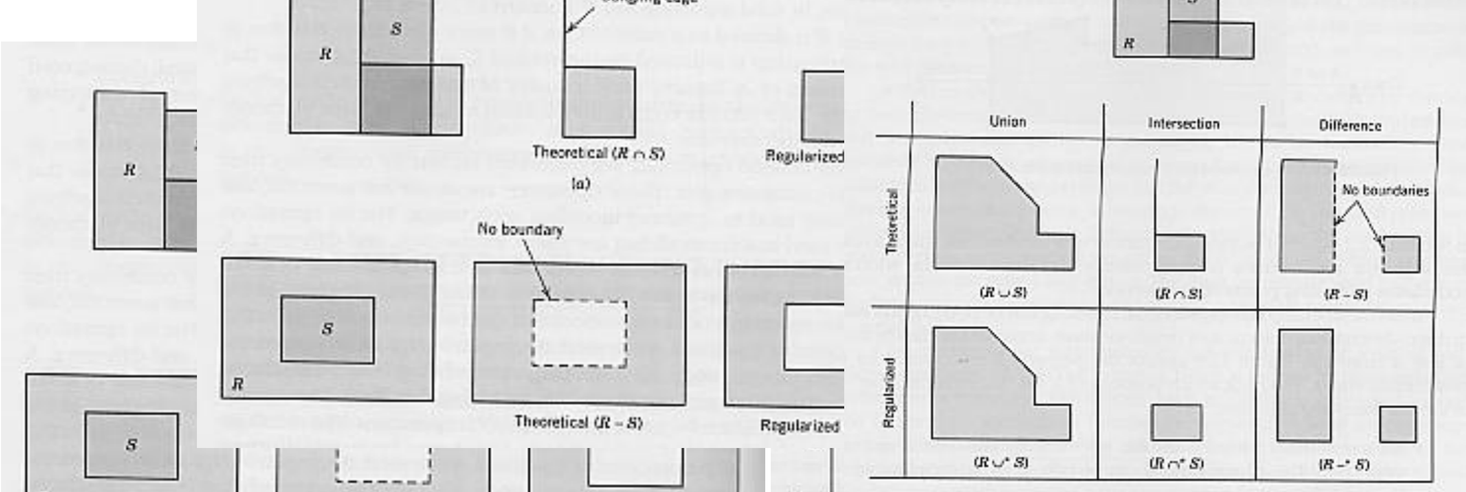



FIGURE 12.8 Conventional and theoretical set operations. (Two-dimensional objects used for simplicity).

5. **Set membership classification** – two sets X and S, check how various parts of X can be assigned to S as being on its interior, exterior, or on its boundaries. X is partitioned into subsets XinS, XonS, XoutS.

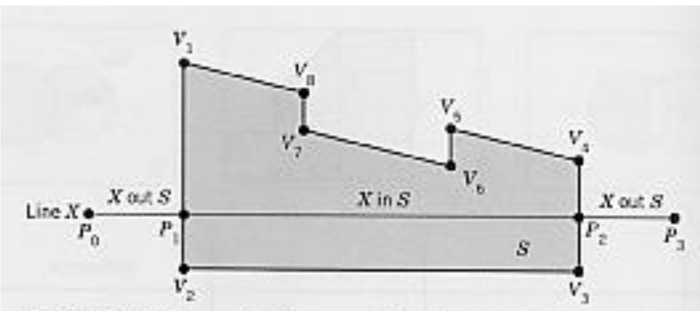



FIGURE 12.9 Example of set membership classification.

- Constructive Solid Geometry (CSG)

A CSG model assumes that physical objects can be created by combining basic elementary shapes (primitives) through specific rules.

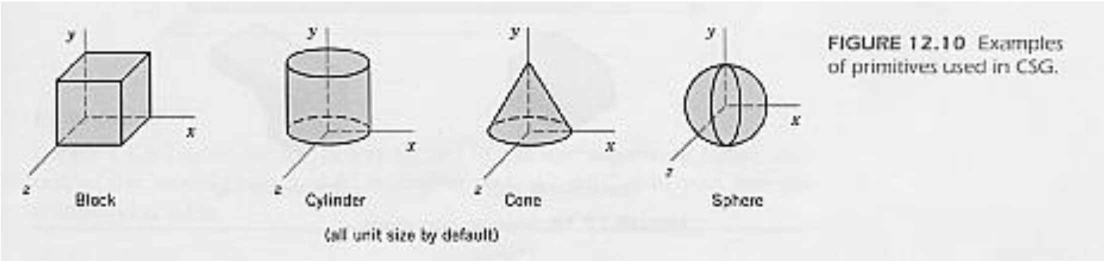


FIGURE 12.10 Examples of primitives used in CSG.

CSG primitives are represented by the intersection of a set of half-spaces, as shown in Figure 12.10.

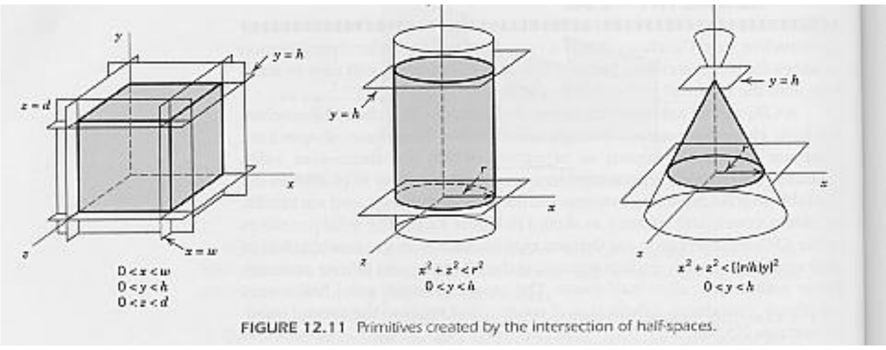


FIGURE 12.11 Primitives created by the intersection of half-spaces.

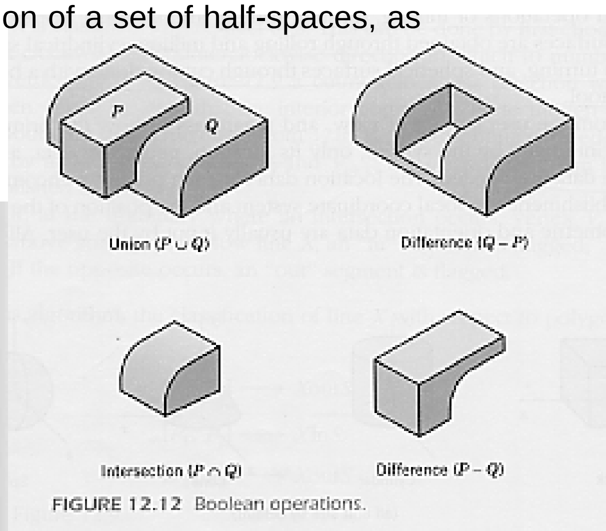


FIGURE 12.12 Boolean operations.

Quadric surfaces are commonly used in CSG because they represent the most commonly used surface in mechanical design produced by the stand operations of milling, turning, rolling. E.g. planar surfaces are obtained through rolling and milling, cylindrical surface through turning, spherical surfaces through cutting done with a ball-end cutting tool.

Data structure for the CSG representation is based on the binary tree structure.

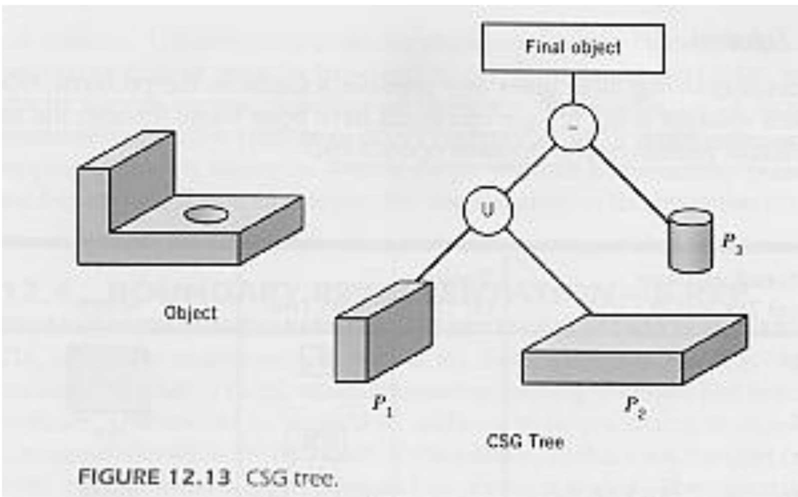
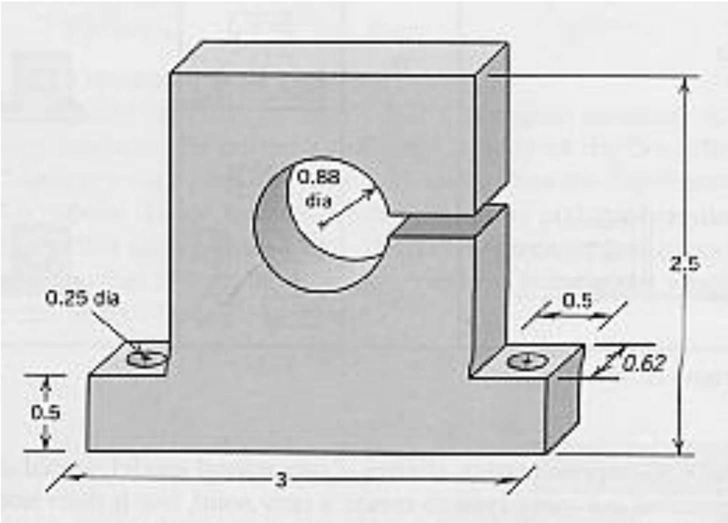


FIGURE 12.13 CSG tree.




## CSG example

CSG Table

| Primitive No. | Primitive Type | Transformations $S(x,y,z)$ $T(x,y,z)$ $R(x,y,z)$ | Boolean ($U$, $D$, $I$) | CSG Tree | Sketch |
|---|---|---|---|---|---|
| 1 | Block | $S(3.0,2.5,.62)$ | | $P_1$ | 2.5 (0,0,0) 3 0.62 |
| 2/3 | Block | $S(0.5,2.0,.62)$ $T(2.5,0.5,0.0)$ ‖ $S(0.5,2.0,.62)$ $T(0.0,0.5,0.0)$ | D/D | $SOL_2$ = $SOL_1$ D $P_3$; $SOL_1$ = $P_1$ D $P_2$ | |
| 4 | Cylinder | $S(r = 0.44,\ h = 0.62)$ $T(1.5,1.5,0.0)$ | D | $SOL_3$ = $SOL_2$ D $P_4$ | |
| 5 | Block | $S(0.56,0.12,0.62)$ $T(1.94,1.44,0.0)$ | D | $SOL_4$ = $SOL_3$ D $P_5$ | |
| 6/7 | Cylinder | $S(r = 0.125,\ h = 0.5)$ $T(0.25,0.0,0.31)$ $R(90.0,0.0,0.0)$ ‖ $S$(same) $T(2.75,0.0,0.31)$ $R$(same) | D/D | $SOL_6$ = $SOL_5$ D $P_7$; $SOL_5$ = $SOL_4$ D $P_6$ | |

*Default primitives have origin as shown in Figure 12.10, and unit size.

## Disadvantages:

1. The way of primitive combinations for the CSG representation is not unique. It has been found through the use of different primitives and Boolean operations.
2. The CSG doesn't specify quantitative values for the new solid (unevaluated model). The new model must be checked through a boundary evaluation routine with will supply quantitative information about its vertices, edges, faces.
3. It shorts of intersection calculation in the form of curve/curve, curve/surface, or surface/surface intersections.

- Boundary Representation – B-rep

The B-rep is built on the idea that a physical object is enclosed by a set of faces, which themselves belong to closed and orientable surfaces.

**Geometric and topology entities**

| | |
|---|---|
| **Point** | **Vertex** |
| **Curve** | **Edge** |
| **Surface** | **Face** |

The Eular-Poincaré law gives a quantitative relationship among faces, edges, vertices, faces' inner loops, bodies or through holes (genus) in solids.

**The Eular-Poincaré law**

$$F-E+V-L = 2(B-G)$$

A loop represents a connected portion of the boundary of a face. The face's inner loop represents the connected portion of the boundary of two faces. Eular law is not only suit for solids with planar faces, but also for curved objects with closed curved faces or edges.

**Simplest form : F-E+V=2**

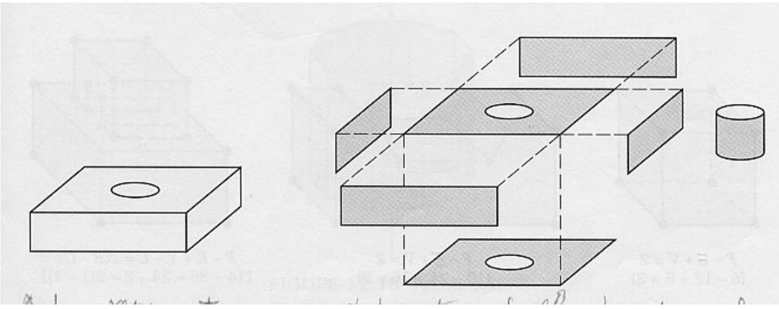


FIGURE 12.14 Boundary representation (B-rep).

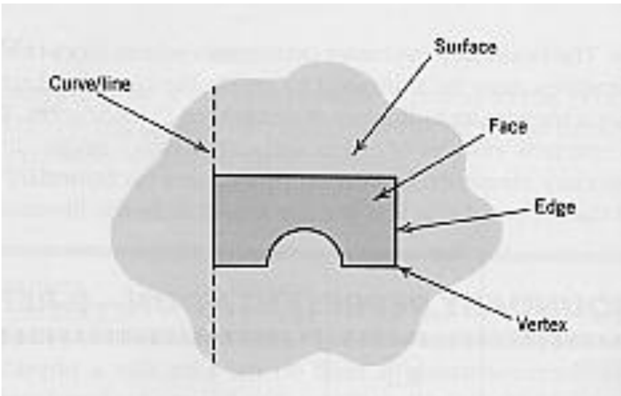



FIGURE 12.15 Difference between geometric and topological entities.

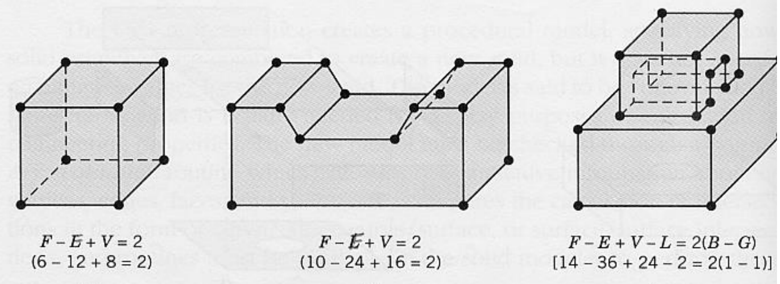


FIGURE 12.16 Euler's law applied to polyhedral objects.

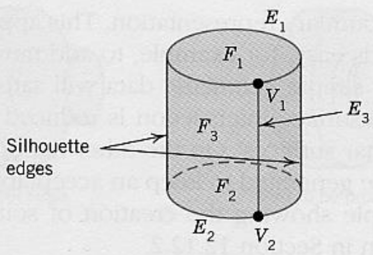


FIGURE 12.17 Euler's law for objects with curved faces or edges.

- Faceted representation

**Advantages:** easy to add new surface types and a small amount of vary simple geometric data will satisfy all the needs.

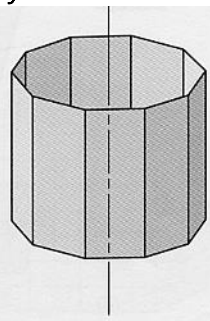


FIGURE 12.18 Faceted

- Sweep representation

Translation and rotational sweeping are used to create the sweep solid. In engineering applications sweeping can be used to detect possible interference between moving parts, or simulate and analyze material removal operations in manufacturing (tool moving along a predefined path intersects the raw stock of the part).

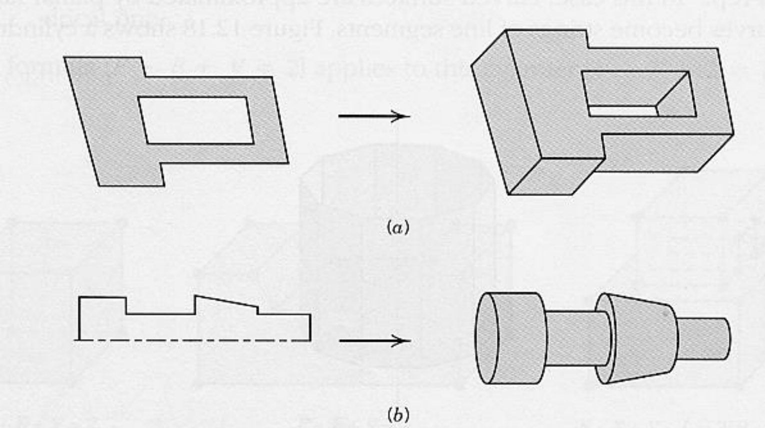



FIGURE 12.19 Sweeping (a) Translational. (b) Rotational.

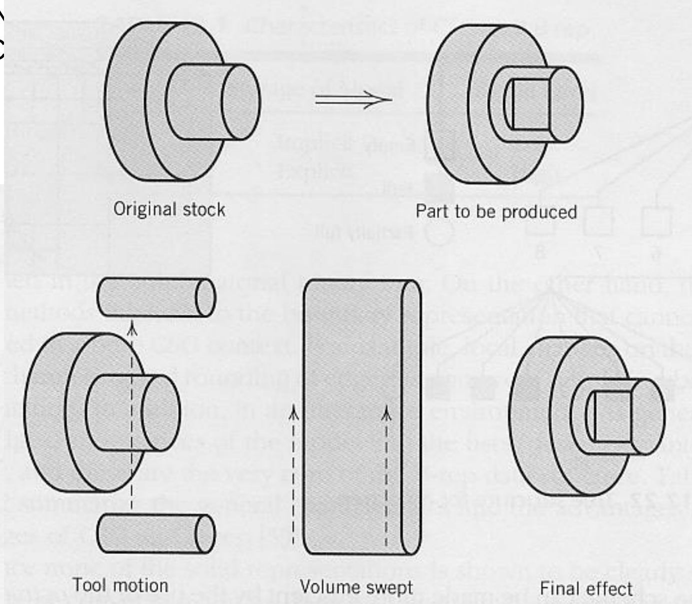



FIGURE 12.20 Use of sweeping to simulate material removal operations in engineering manufacturing.

- Spatial enumeration schemes

The smaller the cell is, the more accurate the model is.

Spatial enumeration schemes have the advantage of easy access to any part of the model and the assurance of spatial uniqueness.

2D – quadtree

3D – octree

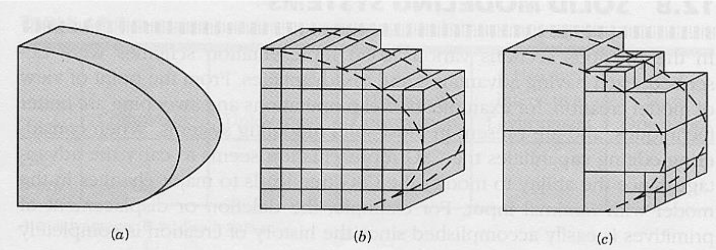



FIGURE 12.21 (a) Solid. (b) Spatial enumeration scheme. (c) Octree.

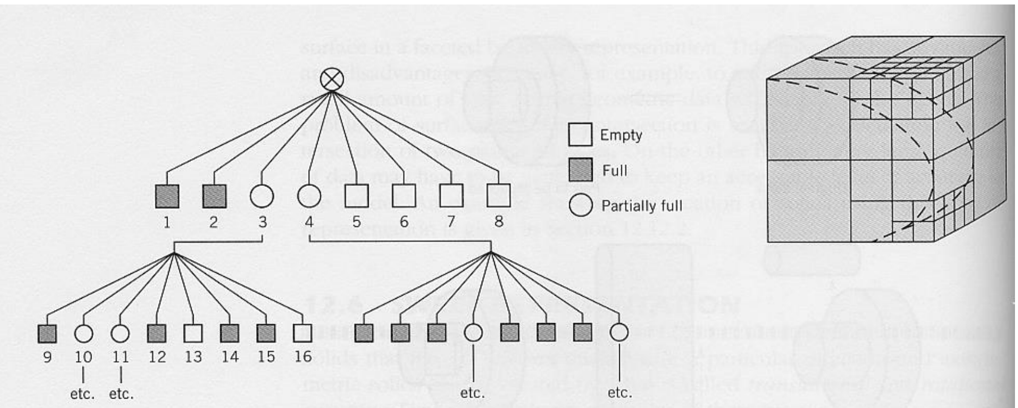



FIGURE 12.22 Tree structure for an octree.

- Solid modeling systems

**Comparison between CSG and B-rep representations.**

| | Storage of Model | Detail Level |
|---|---|---|
| CSG | Implicit | Low |
| B-rep | Explicit | High |

**Advantages (A) and Disadvantages (D) comparisons.**

| | Complexity | Uniqueness | History of Construction | Use in Interactive Environment | Local Operations |
|---|---|---|---|---|---|
| CSG | A | D | A | D | D |
| B-rep | D | A | D | A | A |

Conversion among representations.

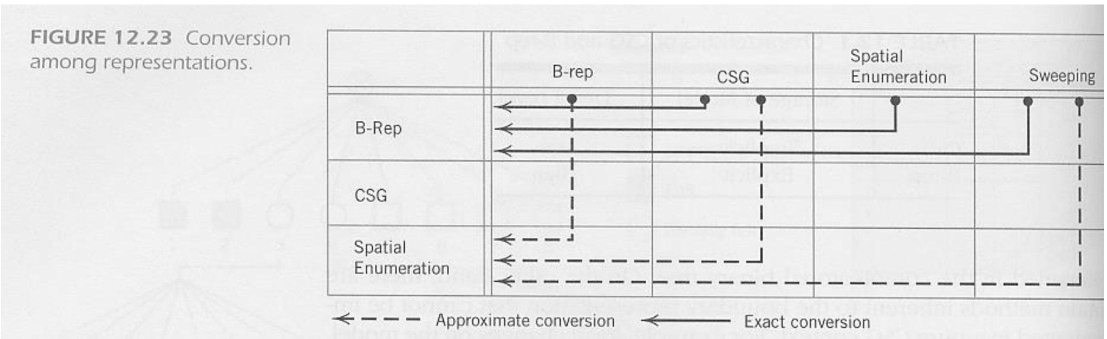


FIGURE 12.23 Conversion among representations.

Solid modeling systems are characterized as "mostly CSG" or "mostly B-rep". (Dual representation of a solid model)

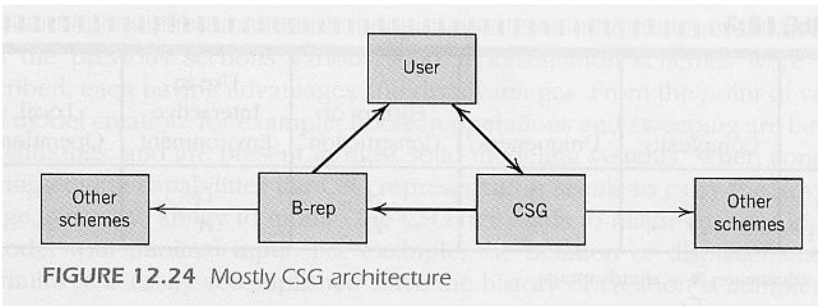


FIGURE 12.24 Mostly CSG architecture

Mostly CGS – model creation and editing is done only in the CSG form; once the model is created, a boundary evaluator algorithm is used to obtain a boundary representation (store internally along with the CSG tree).

Mostly B-rep – the user can create the model in either CSG or B-rep, but the CSG representation is discarded by the system.

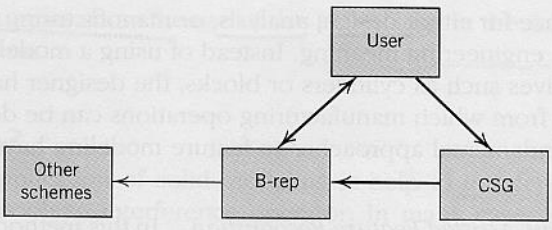


FIGURE 12.25 Mostly B-rep architecture

- NURBS can exactly represent quadric surfaces, so internal operations in the modeler, such as the calculation of surface/surface intersection are accomplished with a single algorithm. (minimize the amount of geometric software required in the modeler)

- **Feature modeling**

Features can link CAD and CAM in an efficient way.

The feature modeler contains not only a geometric and topological structure but also support geometric characteristics of a part. (shapes of holes, cutouts, slots, chamfers, ribs, etc.)

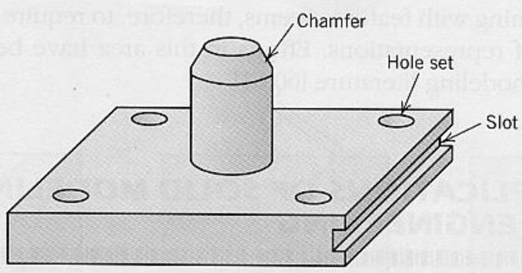


FIGURE 12.26 The shape of holes, slots, and so on is part of a feature modeler.

## Three fundamental approaches to feature modeling

1. Human-assisted feature recognition

E.g. Tolerance or surface of model are created and stored in the database and later used by process planning systems.

2. Automatic feature recognition

Find and extract form features the correspond to some predefined geometric pattern. (very difficult)

3. Design by feature

- Applications of Solid Modeling to Engineering

Solid modeling is commonly used in engineering to aid visual analysis of a design idea, mass property calculations, and static interference analysis.

- A simple Solid Modeling System (sample)

Four steps for developing the solid modeling system

1. CSG representation for user input (create quadric primitives)
2. Conversion to a faceted representation
3. Intersection calculation – Boolean operations
4. Rendering