1.1 Introduction

In general, a Computer Aided Design (CAD) package has three components: a) Design, b) Analysis, and c) Visualization, as shown in the sketch. A brief description of these components follows.

a) **Design**: Design refers to geometric modeling, i.e., 2-D and 3-D modeling, including, drafting, part creation, creation of drawings with various views of the part, assemblies of the parts, etc.

b) **Analysis**: Analysis refers to finite element analysis, optimization, and other number crunching engineering analyses. In general, a geometric model is first created and then the model is analyzed for loads, stresses, moment of inertia, and volume, etc.

c) **Visualization**: Visualization refers to computer graphics, which includes: rendering a model, creation of pie charts, contour plots, shading a model, sizing, animation, etc.
Each of these three areas has been extensively developed in the last 30 years. Several books are written on each of these subjects and courses are available through the academic institutions and the industry.

Most commercial CAD packages (software) consist of only a single component: design or analysis or visualization. However, a few of the vendors have developed an integrated package that includes not only these three areas, but also includes the manufacturing software (CAM). Due to the large storage requirement, integrated packages use either an UNIX workstation or a mainframe platform, and not the popular PC platform. With the improvement in PC computing speed, it’s only a matter of time before we see an integrated package run on a PC. CAD has revolutionized the modern engineering practice; small and large companies use it alike, spending several billion dollars for the initial purchase or lease alone. CAD related jobs are high in demand and the new graduates have advantage over their senior colleagues, as they are more up to date and more productive.

In this course, we will limit our coverage to the design only. Those of you interested in analysis area, look into the course ME 160 – Introduction to Finite Element Analysis.

**Concurrent Engineering**

Concurrent Engineering is another powerful CAD concept that has evolved in the 90’s. According to this concept, there is an instantaneous communication between the designer, analyst, and manufacturing. Changes made at any of these work centers are immediately passed on to the others and the product is modified without delay. Often, the customer, management, and the marketing people join in and become part of the process. Concurrent engineering saves the valuable time and helps get the product out in the market quicker. Products that use to take years from the date of its concept to the actual production now take only a few weeks, and the final product is better and cost-effective.
Some large organizations have invested in Rapid Prototyping process. In this process, the part is created by a CAD package and downloaded into the rapid prototyping machine; the machine immediately manufactures the part, using a plastic material. This is a good example of concurrent engineering, sometimes referred as Art to Part concept.

**CAD Hardware**

There are basically two types of devices that constitute CAD hardware: a) Input devices, and b) Output devices. A brief description follows.

**Input Devices**

These are the devices that we use for communicating with computer, and providing our input in the form of text and graphics. The text input is mainly provided through keyboard. For graphic input, there are several devices available and used according to the work environment. A brief description of these devices is given here.

**Mouse:** This is a potentiometric device, which contains several variable resistors that send signals to the computer. The functions of a mouse include locating a point on the screen, sketching, dragging an object, entering values, accepting a software command, etc. Joystick and trackballs are analogous to a mouse device, and operate on the same principle.

**Digitizers:** Digitizers are used to trace a sketch or other 2-D entities by moving a cursor over a flat surface (which contains the sketch). The position of the cursor provides a feedback to the computer connected with the device. There are electrical wires embedded in orthogonal directions that receive and pass signals between the device and the computer. The device is basically a free moving puck or pen shaped stylus, connected to a tablet.

**Light Pens:** Lockheed’s CADAM software utilized this device to carry out the graphic input. A light pen looks like a pen and contains a photocell, which emits an electronic signal. When the pen is pointed at the monitor screen, it senses light, which is
converted to a signal. The signal is sent to the computer for determination of the exact location of the pen on the monitor screen.

**Touch Sensitive Screens:** This device is embedded in the monitor screens, usually, in the form of an overlay. The screen senses the physical contact of the user. The new generation of the Laptop computers is a good example of this device.

**Other Graphic Input Devices:** In addition to the devices described above, some CAD software will accept input via Image Scanners, which can copy a drawing or schematic with a camera and light beam assembly and convert it into a pictorial database.

The devices just described are, in general, independent of the CAD package being used. All commercial CAD software packages contain the device drivers for the most commonly used input devices. The device drivers facilitate a smooth interaction between our input, the software, and the computer. An input device is evaluated on the basis of the following factors:

- Resolution
- Accuracy
- Repeatability
- Linearity

**Output Devices**

After creating a CAD model, we often need a hard copy, using an output device. Plotters and printers are used for this purpose. A plotter is often used to produce large size drawings and assemblies, where as, a laser jet printer is adequate to provide a 3-D view of a model. Most CAD software require a plotter for producing a shaded or a rendered view.

**CAD Software**

CAD software is written in FORTRAN and C languages. FORTRAN provides the number crunching, where as, C language provides the visual images. Early CAD packages were turnkey systems, i.e., the CAD packages were sold as an integrated
software and hardware package, with no flexibility for using second vendor hardware (1970s and 80s). These systems were based on 16-bit word, and were incapable of networking. The modern CAD software utilizes the open architecture system, i.e., software vendors do not design and manufacture their own hardware. Third party software can be used to augment the basic CAD package. Most popular CAD package will facilitate integration of the Finite Element Analysis and other CAD software from more than one vendor. For example, IDEAS preprocessor can work with almost all the FEA packages for pre and post analyses.

Networking is an important consideration in applications of CAD software. A model created by one engineer must be readily accessible to others in an organization, which is linked by a LAN or other means. The designer, analyst, management, marketing, vendor, and others generally share a model. This is the concurrent engineering in action, mentioned earlier.

**CAD Platform**

In general, we can run CAD software on three different CAD platforms: Mainframe, Workstation, and PC. When the CAD programs first became available, they could only be run on a mainframe computer. However, as the PCs have become faster and cheaper, almost all the CAD vendors have introduced a version of their CAD software that will effectively run on a Pentium or higher computer. Currently, the most popular platforms are PCs and Workstations. Popularity of Workstations stems from their ability to network easily with other computers, and also, due to their large memory storage capability. However, PC platform is still the most preferred medium for most engineers. Increasing popularity of the PC platform can be attributed to several factors, including, total user control, the speed, capability of storing large memory, ease of hardware upgrading and maintenance, and the overall reasonable cost.
## CAD PLATFORMS

<table>
<thead>
<tr>
<th>MAINFRAME</th>
<th>WORKSTATIONS</th>
<th>PCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large Data storage</td>
<td>Medium size data storage</td>
<td>Limited data storage</td>
</tr>
<tr>
<td>Networked</td>
<td>Networked</td>
<td>Can be networked</td>
</tr>
<tr>
<td>Expensive</td>
<td>Relatively inexpensive</td>
<td>Inexpensive</td>
</tr>
<tr>
<td>Need interface language</td>
<td>Runs on Unix</td>
<td>Run on MS-Windows</td>
</tr>
<tr>
<td>No user control</td>
<td>No user control</td>
<td>User controlled</td>
</tr>
<tr>
<td>Good for Large corporations</td>
<td>Good for small companies</td>
<td>Good for all users</td>
</tr>
</tbody>
</table>
In the current CAD market, ProE and AutoCAD are arguably the most dominating CAD software. AutoCAD is basically a 2-D program, with some capability to create 3-D models, whereas ProE is a truly 3-D CAD package. Besides this software, there is several other CAD software, listed in the previous section (Sec 1.3), that has sales exceeding $100 millions. No one CAD package is suitable for all the CAD users in the world. The product we are designing dictates the type of CAD package we need. A good CAD package includes good software, as well as, a compatible hardware. Following is a brief description of the general criteria for evaluating a CAD package.

**Hardware:** Most desirable features in a good hardware are:

- Open architecture
- High speed, large storage
- Compact size
- Inexpensive components
- Inexpensive upgrading

**Software:** In general, the most comprehensive software are written to satisfy almost all the modeling needs of a modeler, consequently, the software tend to be very complex and hard to learn. To create a simple model, we go through several unnecessary steps, and lack the intuitiveness of a simple, straightforward program. ProE is a good example, where we have to go through several layers of menus to create a simple solid. On the other hand, if we were to use a simpler CAD program, the same solid can be created by only a few simple commands. There are several other factors that we should consider when evaluating software. Following is a brief description of these factors.

- **Operating System:** Unix or Windows/NT. PCs in general use Microsoft Windows, whereas, operating system for Workstations is Unix. For a large organization, Workstations are preferable.
• **User Interface**: Most popular CAD software have menu driven commands, which is preferable to the old system of non-menu driven, where user interface was completely by responding to software commands. The most popular CAD programs work with menu driven interface, with some input/action required through command prompts.

• **Documentation and Support**: Learning a software can be very difficult if the software lacks good documentation. Documentation usually comes in the form of a user’s manual, a tutorial book, commands manual, and on-line help. The recent trend is to provide access to the above-mentioned documentation through the Internet, or provide the manuals on a CD ROM. Some CAD vendors provide additional technical support help through phone – ProE is a very good example of this type of support.

• **Maintenance**: Cost of the hardware and software upgrades can significantly impact the small and medium size companies’ decision to choose one software over the others. Most CAD vendors go through an upgrade, on the average, every two years. Usually, hardware upgrade is not as frequent.

• **Modeling Capabilities**: In, general, a CAD software can be classified as either a 2-D or a 3-D program. If we were basically involved in 2-D drawings, any well established 2-D software, similar to AutoCAD would suffice our needs. On the other hand, if we need to create 3-D models and assemblies, we will be better off with a 3-D molder – ProE, SOLIDWORKS, etc.

• **Ease of Modeling**: As a rule-of-thumb, a general, all-purpose type CAD software is much more complex and difficult to learn than a special purpose CAD package.

• **Interface with other CAD Packages and Data Transferability**: A CAD package is used to create models that will be used for analysis, manufacturing, or some other applications. Therefore, a CAD software should be capable of transferring and accepting files from other CAD or CAM programs, without this provision, the CAD program has only a very limited use.

• **Design Documentation**: Besides creating a model, the software should be capable of creating drawings, assemblies, dimensioning, various views (isometric, orthogonal, etc.), labels and attributes, etc.
Following is a brief description of the applications of CAD in mechanical engineering.

- **Two Dimensional Drafting**: This is the most common use of a CAD package. 2-D drawings are used for manufacturing a product.

- **Report Generating**: To generate reports and bill of materials. Spreadsheets and word-processors can be linked to provide a report writing facility.

- **3-D Modeling**: To create the wireframe, surface and solid models. The 3-D models are for concept verification, manufacturing, FEA, etc.

- **Finite Element Analysis**: FEA package is used for pre-processing, analysis, and post-analysis of structures. For this application, a CAD package contains both the modeling and analysis modules.

- **Manufacturing**: manufacturing software is usually called CAM, and contains CAD software as one of the components. CAM software provides capabilities of carrying out 2 and 3-axes machining.
UNIT-II (GEOMETRIC MODELING)

Introduction

In order to understand the significance of curves, we should look into the types of model representations that are used in geometric modeling. Curves play a very significant role in CAD modeling, especially, for generating a wireframe model, which is the simplest form for representing a model.

We can display an object on a monitor screen in three different computer-model forms:

d) Wireframe model

e) Surface Model

f) Solid model

**Wireframe model**: A wireframe model consist of points and curves only, and looks as if its made up with a bunch of wires. This is the simplest CAD model of an object. Advantages of this type of model include ease of creation and low level hardware and software requirements. Additionally, the data storage requirement is low. The main disadvantage of a wireframe model is that it can be very confusing to visualize. For example, a blind hole in a box may look like a solid cylinder, as shown in the figure.

![A wireframe model – Model of a Solid object with a blind hol](image-url)
In spite of its ambiguity, a wireframe model is still the most preferred form, because it can be created quickly and easily to verify a concept of an object. The wireframe model creation is somewhat similar to drawing a sketch by hand to communicate or conceptualize an object. As stated earlier, a wireframe model is created using points and curves only.

**Surface Model:** sweeping a curve around or along an axis can create a surface model. The figures below show two instances of generating a surface model.

Generating a cylinder by sweeping a circle in the direction of an axis

Generating a donut by sweeping a circle around an axis
The appearance or resolution of a surface model depends on the number of sweeping instances we select. For a realistic looking model, we need to select a large number of instances, requiring a large computer memory, or, opt for a not-so realistic model by selecting a small number of instances, and save memory. In some commercial CAD packages we have the option of selecting the resolution of a model; other packages have a fixed value for resolution that cannot be changed by users.

Surface models are useful for representing surfaces such as a soft-drink bottle, automobile fender, aircraft wing, and in general, any complicated curved surface. One of the limitations of a surface model is that there is no geometric definition of points that lie inside or outside the surface.

**Solid Model**: Representation of an object by a solid model is relatively a new concept. There were only a couple of solids modeling CAD programs available in late 1980s, and they required mainframe computers to run on. However, in 1990s, due to the low cost and high speed, PCs have become the most popular solid modeling software platform, prompting almost all the CAD vendors to introduce their 3-D solid modeling software that will run on a PC.

Solid models represent objects in a very realistic and unambiguous form; however, they require a large amount of storage memory and high-end computer hardware. A solid model can be shaded and rendered in desired colors to give it a more realist appearance.

**Role of Curves in Geometric Modeling**

Curves are used to draw a wireframe model, which consists of points and curves; the curves are...
utilized to generate surfaces by performing parametric transformations on them. A curve can be as simple as a line or as complex as a B-spline. In general, curves can be classified as follows:

- **Analytical Curves**: This type of curve can be represented by a simple mathematical equation, such as, a circle or an ellipse. They have a fixed form and cannot be modified to achieve a shape that violates the mathematical equations.

- **Interpolated curves**: An interpolated curve is drawn by interpolating the given data points and has a fixed form, dictated by the given data points. These curves have some limited flexibility in shape creation, dictated by the data points.

- **Approximated Curves**: These curves provide the most flexibility in drawing curves of very complex shapes. The model of a curved automobile fender can be easily created with the help of approximated curves and surfaces.

In general, sweeping a curve along or around an axis creates a surface, and the generated surface will be of the same type as the generating curve, e.g., a fixed form curve will generate a fixed form surface.

As stated earlier, curves are used to generate surfaces. To facilitate the computer-language algorithm, curves are represented by parametric equations. Non-parametric equations are used only to locate a point of intersection on the curve, and not for generating them. Let us briefly discuss the parametric and non-parametric form of a curve.

**Conic Sections or Conic Curves**

A conic curve is generated when a plane intersects a cone, as shown.

![Conic Curve Diagram](image)
The intersection of the plane PQ and the cone is a circle, where as, the intersection created by the plane AB is an ellipse. Other curves that can be created are parabola and hyperbola.

Conic curves are used to create simple wireframe models of objects, which have edges that can be represented by these analytical curves. The fixed-form or analytical curves do not have inflection points, i.e., curves have slopes that are either positive or negative and do not change their sign (positive slope will remain positive and negative slope will remain negative). All conic curves can be represented by a quadratic equation, for example, circular and elliptical curves have quadratic polynomial equations.

Cubic Spline – Synthetic Curves

The analytical and interpolated curves, discussed in the previous section (4.4) and (4.5) are insufficient to meet the requirements of mechanical parts that have complex curved shapes, such as, propeller blades, aircraft fuselage, automobile body, etc. These components contain non-analytical, synthetic curves. Design of curved boundaries and surfaces require curve representations that can be manipulated by changing data points, which will create bends and sharp turns in the shape of the curve. The curves are called synthetic curves, and the data points are called vertices or control points. If the curve passes through all the data points, it is called an interpolant (interpolated). Smoothness of the curve is the most important requirement of a synthetic curve.

Various continuity requirements at the data points can be specified to impose various degrees of smoothness of the curve. A complex curve may consist of several curve segments joined together. Smoothness of the resulting curve is assured by imposing one of the continuity requirements. A zero order continuity ($C^0$) assures a continuous curve, first order continuity ($C^1$) assures a continuous slope, and a second order continuity ($C^2$) assures a continuous curvature, as shown below.
A cubic polynomial is the lowest degree polynomial that can guarantee a $C^2$ curve. Higher order polynomials are not used in CAD, because they tend to oscillate about the control points and require large data storage. Major CAD/CAM systems provide three types of synthetic curves: Hermite Cubic Spline, Bezier Curves, and B-Spline Curves.

Hermite Cubic Spline

Hermite cubic curve is also known as parametric cubic curve, and cubic spline. This curve is used to interpolate given data points that result in a synthetic curve, but not a free form, unlike the Bezier and B-spline curves. The most commonly used cubic spline is a three-dimensional planar curve (not twisted). The curve is defined by two data points that lie at the beginning and at the end of the curve, along with the slopes at these points. It is represented by a cubic polynomial. When two end points and their slopes define a curve, the curve is called a Hermite cubic curve. Several cubic splines can be joined together by imposing the slope continuity at the common points. In design applications, cubic splines
are not as popular as the Bezier and B-spline curves. There are two reasons for this:

- The curve cannot be modified locally, i.e., when a data point is moved, the entire curve is affected, resulting in a global control, as shown in the figure.

- The order of the curve is always constant (cubic), regardless of the number of data points. Increase in the number of data points increases shape flexibility, however, this requires more data points, creating more splines, that are joined together (only two data points and slopes are utilized for each spline).

![Effect of Moving the Data Point](image1.png) ![Effect of Change in slope](image2.png)

**Bezier Curves**

Equation of the Bezier curve provides an approximate polynomial that passes near the given control points and through the first and last points. In 1960s, the French engineer P. Bezier, while working for the Renault automobile manufacturer, developed a system of curves that combine the features of both interpolating and approximating polynomials. In this curve, the control points influence the path of the curve and the first two and last two control points define lines which are tangent to the beginning and the end of the curve. Several curves can be combined and blended together. In engineering, only the quadratic, cubic and quartic curves are frequently used.

**Bezier’s Polynomial Equation**

The curve is defined by the equation
\[ P(t) = \sum V_i B_{i,n}(t) \quad \text{where}, \quad 0 \leq t \leq 1 \quad \text{and} \quad i = 0, 1, 2, \ldots, n \]

Here, \( V_i \) represents the \( n+1 \) control points, and \( B_{i,n}(t) \) is the blending function for the Bezier representation and is given as

\[ B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i} \]

Where \( n \) is the degree of the polynomial and

\[ \binom{n}{i} = \frac{n!}{i!(n-i)!} \quad i = 0, 1, 2, \ldots, n \]
B-Spline Curve

B-spline curves use a blending function, which generates a smooth, single parametric polynomial curve through any number of points. To generate a Bezier curve of the same quality of smoothness, we will have to use several pieces of Bezier curves. Unlike the Bezier curve, the degree of the polynomial can be selected independently of the number of control points. The degree of the blending function controls the degree of the resulting B-spline curve. The curve has good local control, i.e., if one vertex is moved, only some curve segments are affected, and the rest of the curve remains unchanged.

The mathematical derivation of the B-spline curve is complex and beyond the scope of this course. The equation is of the form:

\[ P(t) = \sum N_{i,k}(t) V_i \]

Where,

- \( P(t) \) is a point on the curve.
- \( i \) indicates the position of control point \( i \)
- \( k \) is order of curve
- \( N_{i,k}(t) \) are blending functions
- \( V_i \) are control points

The matrix form of the uniform cubic B-spline curve is

\[
P_i(t) = \frac{1}{6} \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \end{bmatrix} \begin{bmatrix} V_{i-1} \\ V_i \\ V_{i+1} \\ V_{i+2} \end{bmatrix}
\]
INTRODUCTION

- **Computer Graphics:**
  
  **As objects:** images generated and/or displayed by computers.

  **As a subject:** the science of studying how to generate and display images using computers.

- **Why:** a picture is worth ten thousand words

- **Classifications:**
  
  **Dimensions:** 2 or 3

  **Types:** *vector*, *raster*

  **Vector Graphics:** deals with points, lines and shapes.

\[
\begin{align*}
\text{p1}(x_1, y_1) & \quad \text{---} \quad \text{p2}(x_2, y_2)
\end{align*}
\]
**Raster Graphics**: deals with raster images. A raster image is a 2D grid of colored dots. Each dot is called a **pixel** (picture element). Each horizontal line of pixels is called a **scan line**.

![Raster Image Example](image_url)

- **Display Hardware**:
  - **CRT Display**: 1950, MIT, for the Whirlwind computer
    
    (a vector display device).

    All modern computers use rater display devices.

- **Visual Realism**: to make computer generated images look as real as possible.
  
  Techniques for generating realistic perceptions:

  - **Depth**: hidden surface removal, foreshortening, and depth cueing.
**Direction:** lighting, shading (constant, Gourand, Phong).

**Material:** shininess, translucency, reflection, shadow, (ray tracing), texture

**Ray-tracing:** simulate how rays of light reflect from the surfaces of objects.
Motion: animation

Stereo: two images: left and right

- polarized glass, LCD glass, R-B glass,
- hologram, out-of-focus

- Beyond Realism: new technologies, 90s~
  Virtual Reality: interactive animated 3D stereo
  Internet: HTML, GUI, Java 2D and 3D, VRML (virtual reality markup language), X3D.

- New Trends:

Applications:

- from CAD and flight simulation to data and information visualization;
- from workstations to PCs: Windows: GDI, WinG, OpenGL.
Research:

more realism (radiosity, physically based rendering, ...);

more speed (parallel rendering, GPU, 3D display device, ...);

more data types (geometrical, attributed, scattered, ...)

Course to be taught in five segments:

2. Understand graphics using VRML.
3. Look inside graphics using OpenGL.
4. Learn how to design and implement graphics engines.
5. Write graphics applications for New Media Arts.
SHADING

In computer graphics, shading refers to the process of altering the color of an object/surface/polygon in the 3D scene, based on its angle to lights and its distance from lights to create a photorealistic effect. Shading is performed during the rendering process by a program called a shader.

**Angle to light source**

Shading alters the colors of faces in a 3D model based on the angle of the surface to a light source or light sources.

The first image below has the faces of the box rendered, but all in the same color. Edge lines have been rendered here as well which makes the image easier to see.

The second image is the same model rendered without edge lines. It is difficult to tell where one face of the box ends and the next begins.

The third image has shading enabled, which makes the image more realistic and makes it easier to see which face is which.

Rendered image of a box. This image has no shading on its faces, but uses edge lines to separate the faces.

This is the same image with the edge lines removed.

This is the same image rendered with shading of the faces to alter the colors of the 3 faces based on their angle to the light sources.

www.studentsfocus.com

Fatima Michael College of Engineering & Technology
Shading effects from floodlight.

Shading is also dependent on the lighting used. Usually, upon rendering a scene a number of different lighting techniques will be used to make the rendering look more realistic. Different types of light sources are used to give different effects.

**Ambient lighting**

An ambient light source represents a fixed-intensity and fixed-color light source that affects all objects in the scene equally. Upon rendering, all objects in the scene are brightened with the specified intensity and color. This type of light source is mainly used to provide the scene with a basic view of the different objects in it. This is the simplest type of lighting to implement and models how light can be scattered or reflected many times producing a uniform effect.

Ambient lighting can be combined with ambient occlusion to represent how exposed each point of the scene is, affecting the amount of ambient light it can reflect. This produces diffuse, non-directional lighting throughout the scene, casting no clear shadows, but with enclosed and sheltered areas darkened. The result is usually visually similar to an overcast day.

**Directional lighting**

A directional light source illuminates all objects equally from a given direction, like an area light of infinite size and infinite distance from the scene; there is shading, but cannot be any distance falloff.
Point lighting

Light originates from a single point, and spreads outward in all directions.

Spotlight lighting

Models a Spotlight. Light originates from a single point, and spreads outward in a cone.

Area lighting

Light originates from a small area on a single plane. A more accurate model than a point light source.

Volumetric lighting

Light originating from a small volume, an enclosed space lighting objects within that space.

Shading is interpolated based on how the angle of these light sources reach the objects within a scene. Of course, these light sources can be and often are combined in a scene. The renderer then interpolates how these lights must be combined, and produces a 2d image to be displayed on the screen accordingly.

Distance falloff

Theoretically, two surfaces which are parallel, are illuminated the same amount from a distant light source, such as the sun. Even though one surface is further away, your eye sees more of it in the same space, so the illumination appears the same.

Notice in the first image that the color on the front faces of the two boxes is exactly the same. It appears that there is a slight difference where the two faces meet, but this is an optical illusion because of the vertical edge below where the two faces meet.

Notice in the second image that the surfaces on the boxes are bright on the front box and darker on the back box. Also the floor goes from light to dark as it gets farther away.

This distance falloff effect produces images which appear more realistic without having to add additional lights to achieve the same effect.
Two boxes rendered with an OpenGL renderer. Note that the colors of the two front faces are the same even though one box is farther away.

The same model rendered using ARRIS CAD which implements "Distance Falloff" to make surfaces that are closer to the eye appear brighter.

Distance falloff can be calculated in a number of ways:

- **None** - The light intensity received is the same regardless of the distance between the point and the light source.
- **Linear** - For a given point at a distance \( x \) from the light source, the light intensity received is proportional to \( 1/x \).
- **Quadratic** - This is how light intensity decreases in reality if the light has a free path (i.e. no fog or any other thing in the air that can absorb or scatter the light). For a given point at a distance \( x \) from the light source, the light intensity received is proportional to \( 1/x^2 \).
- **Factor of \( n \)** - For a given point at a distance \( x \) from the light source, the light intensity received is proportional to \( 1/x^n \).
- Any number of other mathematical functions may also be used.

**Flat shading**

Flat shading is a lighting technique used in 3D computer graphics to shade each polygon of an object based on the angle between the polygon's surface normal and the direction of the light source, their respective colors and the intensity of the light source. It is usually used for high speed rendering where more advanced shading techniques are too
computationally expensive. As a result of flat shading all of the polygon's vertices are colored with one color, allowing differentiation between adjacent polygons. Specular highlights are rendered poorly with flat shading: If there happens to be a large specular component at the representative vertex, that brightness is drawn uniformly over the entire face. If a specular highlight doesn’t fall on the representative point, it is missed entirely. Consequently, the specular reflection component is usually not included in flat shading computation.

**Smooth shading**

In contrast to flat shading with smooth shading the color changes from pixel to pixel. It assumes that the surfaces are curved and uses interpolation techniques to calculate the values of pixels between the vertices of the polygons.

Types of smooth shading include:

- **Gouraud shading**
  1. Determine the normal at each polygon vertex
  2. Apply an illumination model to each vertex to calculate the vertex intensity
  3. Interpolate the vertex intensities using **bilinear interpolation** over the surface polygon

- **Phong shading**

**Gouraud shading**

1. Determine the normal at each polygon vertex
2. Apply an illumination model to each vertex to calculate the vertex intensity
3. Interpolate the vertex intensities using bilinear interpolation over the surface polygon

**Data structures**

- Sometimes vertex normals can be computed directly (e.g. height field with uniform mesh)
- More generally, need data structure for mesh
- Key: which polygons meet at each vertex

**Advantages**

- Polygons, more complex than triangles, can also have different colors specified for each vertex. In these instances, the underlying logic for shading can become more intricate.
Problems

- Even the smoothness introduced by Gouraud shading may not prevent the appearance of the shading differences between adjacent polygons.
- Gouraud shading is more CPU intensive and can become a problem when rendering real time environments with many polygons.
- T-Junctions with adjoining polygons can sometimes result in visual anomalies. In general, T-Junctions should be avoided.

Phong shading

Phong shading is similar to Gouraud shading, except that the Normals are interpolated. Thus, the specular highlights are computed much more precisely than in the Gouraud shading model:

1. Compute a normal $N$ for each vertex of the polygon.
2. From bilinear interpolation compute a normal, $N_i$ for each pixel. (This must be renormalized each time)
3. From $N_i$ compute an intensity $I_i$ for each pixel of the polygon.
4. Paint pixel to shade corresponding to $I_i$.

Other Approaches

Both Gouraud shading and Phong shading can be implemented using bilinear interpolation. Bishop and Weimer proposed to use a Taylor series expansion of the resulting expression from applying an illumination model and bilinear interpolation of the normals. Hence, second degree polynomial interpolation was used. This type of biquadratic interpolation was further elaborated by Barrera et al., where one second order polynomial was used to interpolate the diffuse light of the Phong reflection model and another second order polynomial was used for the specular light.

Spherical Linear Interpolation (Slerp) was used by Kuij and Blake for computing both the normal over the polygon as well as the vector in the direction to the light source. A similar approach was proposed by Hast, which uses Quaternion interpolation of the normals with the advantage that the normal will always have unit length and the computationally heavy normalization is avoided.
Computer animation, or CGI animation, is the process used for generating animated images by using computer graphics. The more general term computer-generated imagery encompasses both static scenes and dynamic images while computer animation only refers to moving images.

Modern computer animation usually uses 3D computer graphics, although 2D computer graphics are still used for stylistic, low bandwidth, and faster real-time renderings. Sometimes, the target of the animation is the computer itself, but sometimes the target is another medium, such as film.

Computer animation is essentially a digital successor to the stop motion techniques used in traditional animation with 3D models and frame-by-frame animation of 2D illustrations. Computer-generated animations are more controllable than other more physically based processes, such as constructing miniatures for effects shots or hiring extras for crowd scenes, and because it allows the creation of images that would not be feasible using any other technology. It can also allow a single graphic artist to produce such content without the use of actors, expensive set pieces, or props.

To create the illusion of movement, an image is displayed on the computer monitor and repeatedly replaced by a new image that is similar to it, but advanced slightly in time (usually at a rate of 24 or 30 frames/second). This technique is identical to how the illusion of movement is achieved with television and motion pictures.

For 3D animations, all frames must be rendered after the modeling is complete. For 2D vector animations, the rendering process is the key frame illustration process, while tweened frames are rendered as needed. For pre-recorded presentations, the rendered frames are transferred to a different format or medium, such as film or digital video. The frames may also be rendered in real time as they are presented to the end-user audience. Low bandwidth animations transmitted via the internet (e.g. 2D Flash, X3D) often use software on the end-users computer to render in real time as an alternative to streaming or pre-loaded high bandwidth animations.
UNIT IV ASSEMBLY OF PARTS

TOLERANCE ANALYSIS

Tolerance analysis is the general term for activities related to the study of potential accumulated variation in mechanical parts and assemblies. Its methods may be used on other types of systems subject to accumulated variation, such as mechanical and electrical systems. Engineers analyze tolerances for the purpose of evaluating geometric dimensioning and tolerancing (GD&T). Methods include 2D tolerance stacks, 3D Monte Carlo simulations, and datum conversions.

Tolerance stackups or tolerance stacks are terms used to describe the problem-solving process in mechanical engineering of calculating the effects of the accumulated variation that is allowed by specified dimensions and tolerances. Typically these dimensions and tolerances are specified on an engineering drawing. Arithmetic tolerance stackups use the worst-case maximum or minimum values of dimensions and tolerances to calculate the maximum and minimum distance (clearance or interference) between two features or parts. Statistical tolerance stackups evaluate the maximum and minimum values based on the absolute arithmetic calculation combined with some method for establishing likelihood of obtaining the maximum and minimum values, such as Root Sum Square (RSS) or Monte-Carlo methods.

Modeling

In performing a tolerance analysis, there are two fundamentally different analysis tools for predicting stackup variation: worst-case analysis and statistical analysis.

Worst-case

Worst-case tolerance analysis is the traditional type of tolerance stackup calculation. The individual variables are placed at their tolerance limits in order to make the measurement as large or as small as possible. The worst-case model does not consider the distribution of the individual variables, but rather that those variables do not exceed their respective specified limits. This model predicts the maximum expected variation of the measurement. Designing to worst-case tolerance requirements guarantees 100 percent of
the parts will assemble and function properly, regardless of the actual component variation. The major drawback is that the worst-case model often requires very tight individual component tolerances. The obvious result is expensive manufacturing and inspection processes and/or high scrap rates. Worst-case tolerancing is often required by the customer for critical mechanical interfaces and spare part replacement interfaces. When worst-case tolerancing is not a contract requirement, properly applied statistical tolerancing can ensure acceptable assembly yields with increased component tolerances and lower fabrication costs.

**Statistical variation**

The statistical variation analysis model takes advantage of the principles of statistics to relax the component tolerances without sacrificing quality. Each component’s variation is modeled as a statistical distribution and these distributions are summed to predict the distribution of the assembly measurement. Thus, statistical variation analysis predicts a distribution that describes the assembly variation, not the extreme values of that variation. This analysis model provides increased design flexibility by allowing the designer to design to any quality level, not just 100 percent.

There are two chief methods for performing the statistical analysis. In one, the expected distributions are modified in accordance with the relevant geometric multipliers within tolerance limits and then combined using mathematical operations to provide a composite of the distributions. The geometric multipliers are generated by making small deltas to the nominal dimensions. The immediate value to this method is that the output is smooth, but it fails to account for geometric misalignment allowed for by the tolerances; if a size dimension is placed between two parallel surfaces, it is assumed the surfaces will remain parallel, even though the tolerance does not require this. Because the CAD engine performs the variation sensitivity analysis, there is no output available to drive secondary programs such as stress analysis. This software approach is typified in CE-TOL aka Ti-TOL, originally from ADCATS at BYU.

In the other, the variations are simulated by allowing random changes to geometry, constrained by expected distributions within allowed tolerances with the resulting parts assembled, and then measurements of critical places are recorded as if in an actual manufacturing environment. The collected data is analyzed to find a fit with a known distribution and mean and standard deviations derived from them. The immediate value to this method is that the output represents what is acceptable, even when that is from
imperfect geometry and, because it uses recorded data to perform its analysis, it is possible to include actual factory inspection data into the analysis to see the effect of proposed changes on real data. In addition, because the engine for the analysis is performing the variation internally, not based on CAD regeneration, it is possible to link the variation engine output to another program. For example, a rectangular bar may vary in width and thickness; the variation engine could output those numbers to a stress program which passes back peak stress as a result and the dimensional variation be used to determine likely stress variations. The disadvantage is that each run is unique, so there will be variation from analysis to analysis for the output distribution and mean, just like would come from a factory. This approach was used by Variation Systems Analysis from VSA, now owned by Siemens.

While no official engineering standard covers the process or format of tolerance analysis and stackups, these are essential components of good product design. Tolerance stackups should be used as part of the mechanical design process, both as a predictive and a problem-solving tool. The methods used to conduct a tolerance stackup depend somewhat upon the engineering dimensioning and tolerancing standards that are referenced in the engineering documentation, such as American Society of Mechanical Engineers (ASME) Y14.5, ASME Y14.41, or the relevant ISO dimensioning and tolerancing standards. Understanding the tolerances, concepts and boundaries created by these standards is vital to performing accurate calculations.

Tolerance stackups serve engineers by:

- helping them study dimensional relationships within an assembly.
- giving designers a means of calculating part tolerances.
- helping engineers compare design proposals.
- helping designers produce complete drawings.

_Concerns with tolerance stackups_

A safety factor is often included in designs because of concerns about:

- Operational temperature and pressure of the parts or assembly.
- Wear.
- Deflection of components after assembly.
The possibility or probability that the parts are slightly out of specification (but passed inspection).

The sensitivity or importance of the stack (what happens if the design conditions are not met).

MASS PROPERTY CALCULATION

Calculation of Mass Properties using Traditional Methods

Choosing the Reference Axes

The first step in calculating mass properties of an object is to assign the location of the reference axes. The center of gravity and the product of inertia of an object can have any numerical value or polarity, depending on the choice of axes that are used as a reference for the calculation. Stating that a CG coordinate is "0.050 inches" means nothing unless the position of the reference axis is also precisely defined. Any reference axes may be chosen. For example, the center of gravity of a cylinder may be 4.050 inches from one end, 0.050 inches from its midpoint, and 3.950 inches from the other end. Furthermore, each end of the cylinder may not be perpendicular to the central axis, so that a means of determining the "end" of the cylinder would have to be further defined.

Three mutually perpendicular reference axes are required to define the location of the center of gravity of an object. These axes are usually selected to coincide with edges of the object, accurately located details, or the geometric center of the object. It is not sufficient to state that an axis is the centerline of the object. You must also specify which surfaces on the object define this centerline. Moment of inertia is a rotational quantity and requires only one axis for its reference. Although this can theoretically be any axis in the vicinity of the object, this axis usually is the geometric center, the rotational center (if the object revolves on bearings), or a principal axis (axis passing through the center of gravity which is chosen so the products of inertia are zero). Product of inertia requires three mutually perpendicular reference axes. One of these axes may be a rotational axis or a geometric centerline. For maximum accuracy, it is important to use reference axes that can be located with a high degree of precision. If the object is an aerospace item, then we recommend that this object be designed with two reference datum rings per section, which can be used to define the reference axes. These rings can be precision attachment points that are used to interface the object with another section of a spacecraft or rocket, or they can be rings that were provided solely for the purpose of alignment.
and/or measurement of mass properties. The accuracy of calculation (and the subsequent accuracy of measurement of an actual piece of hardware) is only as good as the accuracy of the means of locating the reference axes. We have found that the single largest source of error in mass properties calculations is the uncertainty of the reference. The dimensional data provided to the mass properties engineer must be sufficiently accurate to permit mass properties tolerances to be met. For example, if you are asked to make precise calculations of mass properties of a projectile, you should establish the error due to reference misalignment as the first step in your calculations. If you are required to calculate CG within an accuracy of 0.001 inch and the reference datum is not round within 0.003 inch, then you cannot accomplish your task. There is no sense in making a detailed analysis of the components of an object when the reference error prevents accurate calculations. Furthermore, it will be impossible to accurately measure such a part after it is fabricated and verify the accuracy of your calculations. The location and accuracy of the reference axes must be of the highest precision.

If your task is to calculate the mass properties of a vehicle that is assembled in sections, then serious thought should be given to the accuracy of alignment of the sections when they are assembled. Often this can be the biggest single factor in limiting the degree of balance (if the vehicle was balanced in sections because the total vehicle is too big for the balancing machine). Alignment error is amplified for long rockets... a 0.001 inch lean introduced by alignment error on a 12 inch diameter can result in a 0.007 inch CG error on a 15 foot long rocket section. This is discussed in detail in the sections of this paper that present the math for combining the mass properties of subassemblies.

The accuracy required for various types of calculations is summarized in later sections of this paper.

**Choosing the Location of the Axes**

The first step in calculating mass properties is to establish the location of the X, Y, and Z axes. The accuracy of the calculations (and later on the accuracy of the measurements to verify the calculations) will depend entirely on the wisdom used in choosing the axes. Theoretically, these axes can be at any location relative to the object being considered, provided the axes are mutually perpendicular. However, in real life, unless the axes are chosen to be at a...
Reference axes must be located at physical points on the object that can be accurately measured. Although the center line of a ring may exist in midair, it can be accurately measured and is therefore a good reference location as can the center of a close tolerance hole which could be identified as the zero degree reference to identify the X axis (Fig. 4). An axis should always pass through a surface that is rigidly associated with the bulk of the object. In Figure 5 it would be better to locate the origin at the end of the object rather than the fitting that is loosely dimensioned relative to the end.
GRAPHICAL KERNEL SYSTEM

The Graphical Kernel System (GKS) was the first ISO standard for low-level computer graphics, introduced in 1977. GKS provides a set of drawing features for two-dimensional vector graphics suitable for charting and similar duties. The calls are designed to be portable across different programming languages, graphics devices and hardware, so that applications written to use GKS will be readily portable to many platforms and devices.

GKS was fairly common on computer workstations in the 1980s and early 1990s, and formed the basis of Digital Research's GSX and GEM products; the latter was common on the Atari ST and was occasionally seen on PCs particularly in conjunction with Ventura Publisher. It was little used outside these markets and is essentially obsolete today except insofar as it is the underlying API defining the Computer Graphics Metafile.

A descendant of GKS was PHIGS.

A main developer and promoter of the GKS was José Luis Encarnação, formerly director of the Fraunhofer Institute for Computer Graphics (IGD) in Darmstadt, Germany.

GKS has been standardized in the following documents:[1]

- The language bindings are ISO standard ISO 8651.
- GKS-3D (Graphical Kernel System for Three Dimensions) functional definition is ISO standard ISO 8805, and the corresponding C bindings are ISO 8806.

The functionality of GKS is wrapped up as a data model standard in the STEP standard, section ISO 10303-46.

OPEN GRAPHICS LIBRARY

OpenGL (Open Graphics Library)[1][4][5] is a cross-language, multi-platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware accelerated rendering.

**Design**

An illustration of the graphics pipeline process

The OpenGL specification describes an abstract API for drawing 2D and 3D graphics. Although it is possible for the API to be implemented entirely in software, it is designed to be implemented mostly or entirely in hardware.

The API is defined as a number of functions which may be called by the client program, alongside a number of named integer constants (for example, the constant GL_TEXTURE_2D, which corresponds to the decimal number 3553). Although the function definitions are superficially similar to those of the C programming language, they are language-independent. As such, OpenGL has many language bindings, some of the most noteworthy being the JavaScript binding WebGL (API, based on OpenGL ES 2.0, for 3D rendering from within a web browser); the C bindings WGL, GLX and CGL; the C binding provided by iOS; and the Java and C bindings provided by Android.

In addition to being language-independent, OpenGL is also platform-independent. The specification says nothing on the subject of obtaining, and managing, an OpenGL context, leaving this as a detail of the underlying windowing system. For the same reason, OpenGL is purely concerned with rendering, providing no APIs related to input, audio, or windowing.
OpenGL is an evolving API. New versions of the OpenGL specifications are regularly released by the Khronos Group, each of which extends the API to support various new features. The details of each version are decided by consensus between the Group's members, including graphics card manufacturers, operating system designers, and general technology companies such as Mozilla and Google.[7]

In addition to the features required by the core API, GPU vendors may provide additional functionality in the form of *extensions*. Extensions may introduce new functions and new constants, and may relax or remove restrictions on existing OpenGL functions. Vendors can use extensions to expose custom APIs without needing support from other vendors or the Khronos Group as a whole, which greatly increases the flexibility of OpenGL. All extensions are collected in, and defined by, the OpenGL Registry.[8]

Each extension is associated with a short identifier, based on the name of the company which developed it. For example, Nvidia's identifier is NV, which is part of the extension name GL_NV_half_float, the constant GL_HALF_FLOAT_NV, and the function glVertex2hNV().[9] If multiple vendors agree to implement the same functionality using the same API, a shared extension may be released, using the identifier EXT. In such cases, it could also happen that the Khronos Group's Architecture Review Board gives the extension their explicit approval, in which case the identifier ARB is used.[10]

The features introduced by each new version of OpenGL are typically formed from the combined features of several widely-implemented extensions, especially extensions of type ARB or EXT.

**Documentation**

OpenGL's popularity is partially due to the quality of its official documentation. The OpenGL Architecture Review Board released a series of manuals along with the specification which have been updated to track changes in the API. These are almost universally known by the colors of their covers:

**The Red Book**

The Initial Graphics Exchange Specification (IGES) (pronounced eye-jess) is a vendor-neutral file format that allows the digital exchange of information among computer-aided design (CAD) systems.


Using IGES, a CAD user can exchange product data models in the form of circuit diagrams, wireframe, freeform surface or solid modeling representations. Applications supported by IGES include traditional engineering drawings, models for analysis, and other manufacturing functions.

**CAD DATA EXCHANGE**

**CAD data exchange** involves a number of software technologies and methods to translate data from one computer-aided design system to another CAD file format.
This PLM technology is required to facilitate collaborative work (CPD) between OEMs and their suppliers.

The main topic is with the translation of geometry (wireframe, surface and solid) but also of importance is other data such as attributes; metadata, assembly structure and feature data.

Methods of translation

There are basically three methods of transferring data from one CAD system to another.

- Direct CAD system export/import
- Direct 3rd party translators.
- Intermediate data exchange formats

Direct internal

Some CAD systems can directly read and/or write other CAD formats, simply by using file open and file save as options. As most CAD file formats are not open, this option is limited to either systems owned by the same company or via hacking of competitor's file format.

Direct external

There are a number of companies that specialize in CAD data translation software, providing software that can read one system and write the information in another CAD system format. These systems have their own proprietary intermediate format some of which will allow reviewing the data during translation. Some of these translators work stand-alone while others require one or both of the CAD packages installed on the translation machine as they use code (APIs) from these systems to read/write the data.

Data translation formats

A common method of translation is via an intermediary format. The sending CAD system exports out to this format and the receiving CAD system reads in this format. Some formats are independent of the CAD vendors being defined by standards organisations while others, although owned by a company, are widely used and are regarded as quasi industry standards. It is becoming increasingly common for companies owning these
quasi industry standards to further the use of their formats by openly publishing these data formats.

Example formats:

- **STEP – ISO 10303**, a replacement for IGES and VDA-FS with the CAD specific parts:
  - STEP AP203 and AP214: Mechanical CAD systems
  - STEP AP210: CAD systems for printed circuit board
  - STEP AP212: CAD systems for electrical installation and cable harness
  - STEP-NC AP238: CAD, CAM, and CNC machining process information
  - STEP AP 242, Managed Model-Based 3D Engineering – the merging of the two leading STEP application protocols, AP 203 and AP 214
- IGES
- VDA-FS
- DXF
- Parasolid XT
- JT Open
- DRG

A number of CAD data exchange methods are described by recent academic studies. The neutral modeling command (NMC) method, proposed by Zhejiang University, is an example of these methods.

*Level of information detail translated.*

As each CAD system has its own method of describing geometry, both mathematically and structurally, there is always some loss of information when translating data from one CAD data format to another. The intermediate file formats are also limited in what they can describe, and they can be interpreted differently by both the sending and receiving systems.

It is therefore important when transferring data between systems to identify what needs to be translated.
If only the 3D model is required for the downstream process, then only the model description needs to be transferred. However, there are levels of detail. For example: is the data wireframe, surface, or solid; is the topology (BREP) information required; must the face and edge identifications be preserved on subsequent modification; must the feature information and history be preserved between systems; and is PMI annotation to be transferred.

With product models, retaining the assembly structure may be required.

If drawings need to be translated, the wireframe geometry is normally not an issue; however text, dimensions and other annotation can be an issue, particularly fonts and formats.

No matter what data is to be translated, there is also a need to preserve attributes (such as color and layer of graphical objects) and text information stored within the files.

Sometimes, however, there is a problem caused by too much information being preserved. An example are the constraints placed on designers arising out of the design intent-history captured in parametric design systems. The receiving system must provide designers with the design freedom to modify geometry without having to understand the history of, or undo, the design tree.

Some translation methods are more successful than others at translating data between CAD systems.