

**AT73.01 CAD/CAM**  
**COMPUTER AIDED DESIGN**

**ERIK L.J. BOHEZ**  
**ASSOCIATE PROFESSOR**  
**DESIGN & MANUFACTURING ENGINEERING**

August 2005

**Lesson I: Relation Between Degree  $n$  and Shape**

**Lesson II: Any Analytic Function Can Be Written As Polynomial**

**Lesson III: Parametric Expression of Curve, Surface and Solid**

**Lesson IV: Representations of Geometry**

**Lesson V: Identifying Unknown Coefficients in Parametric Polynomial Equations of a Curve**

**Lesson VI: Basic 3D Geometry**

**Lesson VII: Perspective and Parallel Projections & Corresponding Clipping Volumes**

**Lesson VIII: CONVEX HULL DEFINITION**

**Lesson IX: BEZIER CURVES**

**Lesson X: B-SPLINE CURVES**

**Lesson XI: NURBS**

**Lesson XII: SURFACES**

**Lesson XIII: SOLID MODELING**

**Lesson XIV: CAD SYSTEM ARCHITECTURE**

**Lesson XV: IGES – STEP**

**Lesson XVI: CAD HARDWARE**

## REFERENCE BOOKS:

D.F. Rogers, and J.A. Adams:

Mathematical Elements for Computer Graphics,  
2<sup>nd</sup> edition, McGraw-Hill, 1990.

G. Farin:

Curves and Surfaces for Computer Aided Geometric Design,  
Academic Press, 1993.

Kunwoo Lee:

Principles of CAD/CAM/CAE Systems,  
Addison-Wesley, 1999

I.D. Faux, and M.J. Pratt:

Computational Geometry for Design and Manufacture,  
Ellis Horwood Limited, 1979.

## REFERENCE BOOKS:

V.B. Anand:

Computer Graphics and Geometric Modeling for Engineers,  
John Wiley & Sons, 1993.

W.K. Giloi:

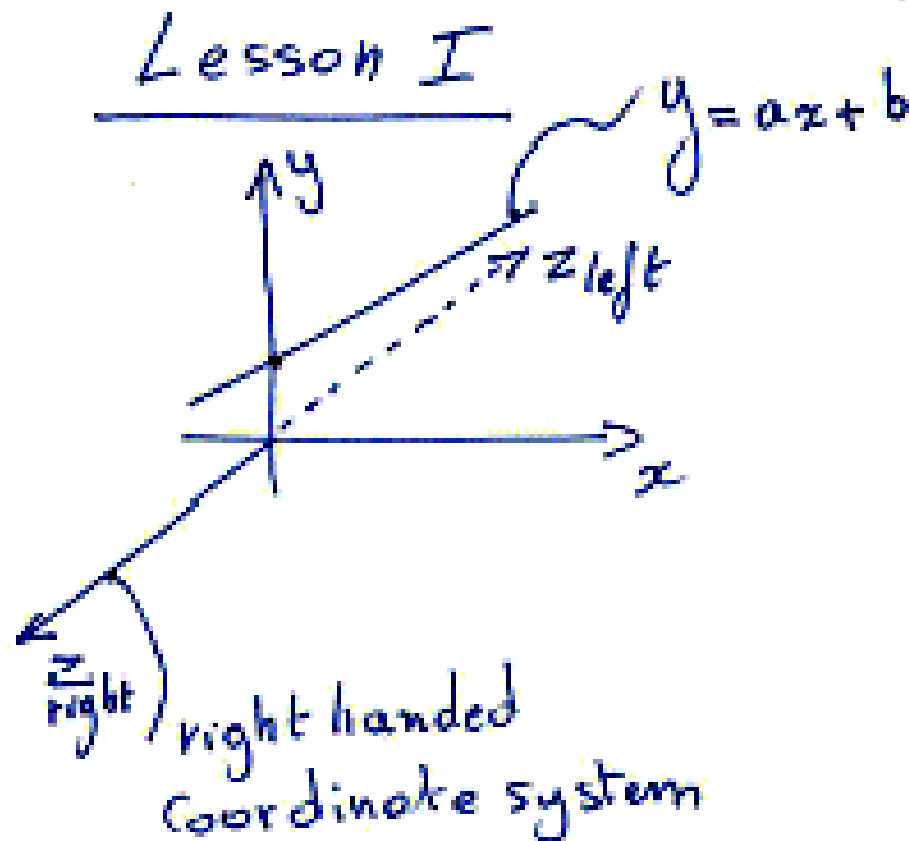
An Introduction to Solid Modeling,  
Computer Science Press, 1998.

M.P. Groover, and E.W. Zimmer:

CAD/CAM: Computer-Aided Design and Manufacturing,  
Prentice Hall, 2000

Zeid: CAD/CAM Theory and Practice,  
McGraw-Hill, 1991.

# Lesson I: Relation Between Degree n and Shape



in CAD/CAM

We always use a right handed coordinate system

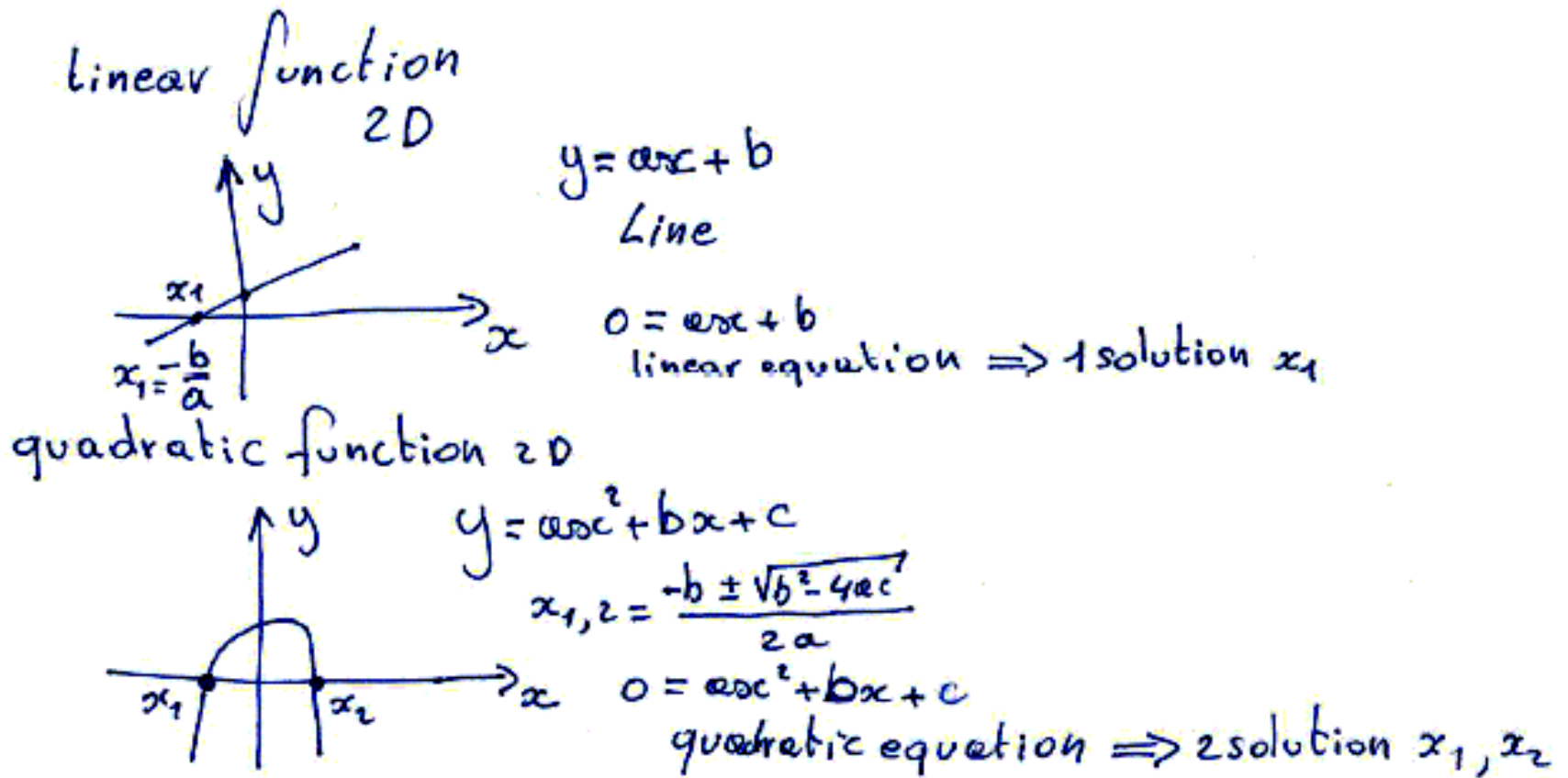
transform from right handed to left handed

$$x_l = x_r$$

$$y_l = y_r$$

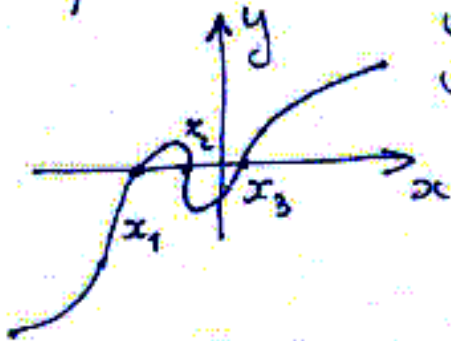
$$z_l = -z_r$$

# Lesson I: Relation Between Degree n and Shape



# Lesson I: Relation Between Degree n and Shape

Cubic function 2D

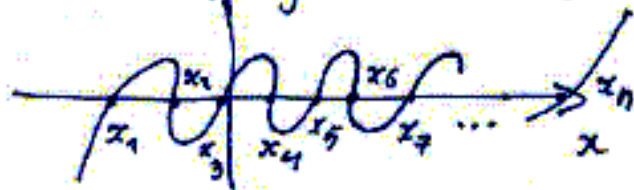


$$y = ax^3 + bx^2 + cx + d$$

$$0 = ax^3 + bx^2 + cx + d$$

Cubic equation  $\Rightarrow$  3 solutions  $x_1, x_2, x_3$

function of degree n



$$y = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \dots + a_0$$

equation of degree n  $\Rightarrow$  n solutions or roots  
 $x_1, x_2, x_3, \dots, x_n$

## Lesson II: Any Analytic Function Can Be Written As Polynomial

any geometrical shape or function can be approximated by a polynomial example  $e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \dots + \frac{x^n}{n!} + \dots$

See Taylor expansion equation

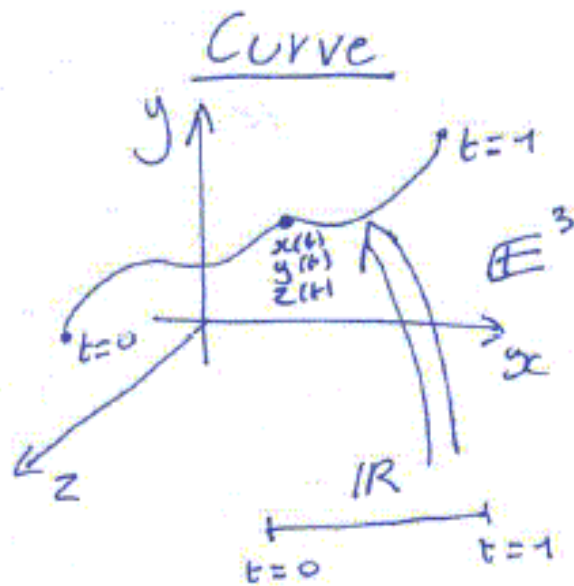
$f(x)$  &  $f^{(n-1)}(x)$  exists for  $[a, b]$   $0 \leq h \leq b-a$

$$f(a+h) = f(a) + \sum_{i=1}^n \frac{f^{(i)}(a)}{i!} h^i + \frac{f^{(n+1)}(p)}{(n+1)!} h^{n+1}$$

with  $p$  some number between  $a$  and  $a+h$



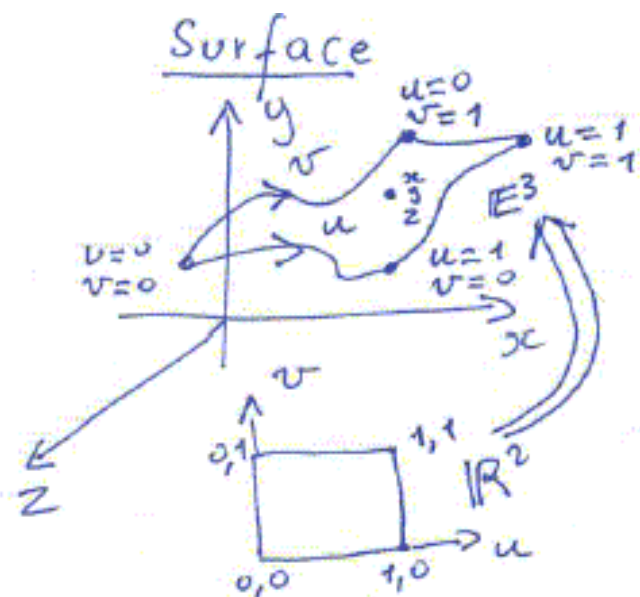
# Lesson III: Parametric Expression of Curve, Surface and Solid



$$\begin{aligned}x(t) &= f_x(t) \\y(t) &= f_y(t) \\z(t) &= f_z(t)\end{aligned}$$

functions of 1 parameter  $t$

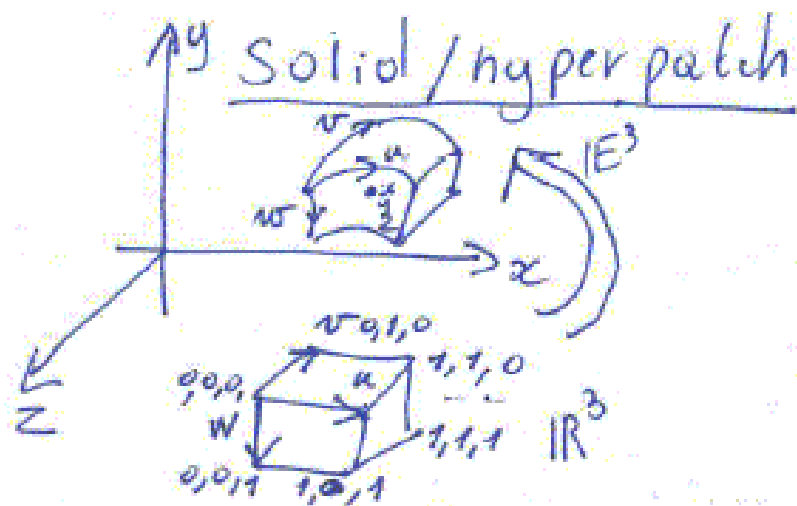
# Lesson III: Parametric Expression of Curve, Surface and Solid



$$\begin{aligned} x(u,v) &= f_x(u,v) \\ y(u,v) &= f_y(u,v) \\ z(u,v) &= f_z(u,v) \end{aligned}$$

functions of 2 parameters  $u, v$

# Lesson III: Parametric Expression of Curve, Surface and Solid

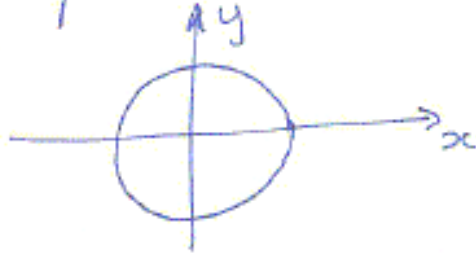


$$\begin{aligned} x(u, v, w) &= f_x(u, v, w) \\ y(u, v, w) &= f_y(u, v, w) \\ z(u, v, w) &= f_z(u, v, w) \end{aligned}$$

functions of  
3 parameters

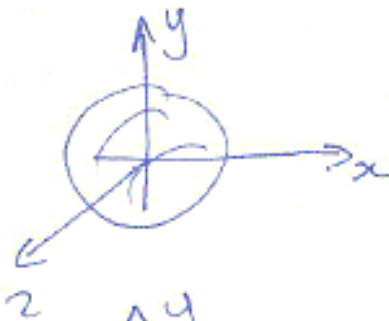
## Lesson IV: Representations of Geometry

Representations 1) Non Parametric representations

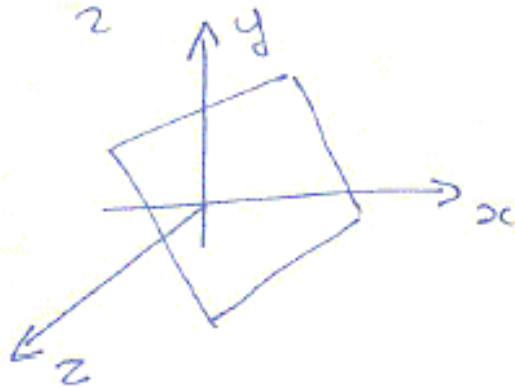


Circle  $x^2 + y^2 = R^2$  implicit

$$y = \pm \sqrt{R^2 - x^2} \text{ explicit}$$



Sphere  $x^2 + y^2 + z^2 = R^2$



plane  $Ax + By + Cz = D$

# Lesson IV: Representations of Geometry

## 2. Parametric representations



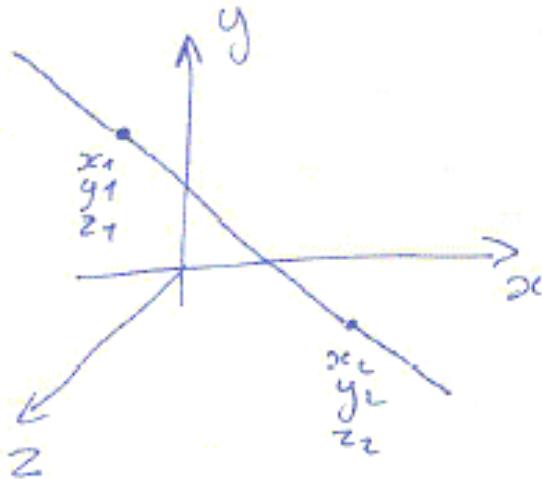
Circle

$$\begin{aligned} x &= R \cos \theta \\ y &= R \sin \theta \end{aligned} \quad 0 \leq \theta \leq 2\pi$$



Sphere

$$\begin{aligned} x &= r \cos \phi \cos \theta \\ y &= r \cos \phi \sin \theta \\ z &= r \cos \phi \end{aligned} \quad \begin{aligned} -\frac{\pi}{2} &\leq \phi \leq \frac{\pi}{2} \\ 0 &\leq \theta \leq 2\pi \end{aligned}$$



Line

$$\begin{aligned} x(t) &= x_1 + (x_2 - x_1)t \\ y(t) &= y_1 + (y_2 - y_1)t \\ z(t) &= z_1 + (z_2 - z_1)t \end{aligned}$$

$$-1 < t < 1$$

today most CAD/CAM systems  
use parametric representations

# Lesson IV: Representations of Geometry

## CURVE REPRESENTATION



- Points

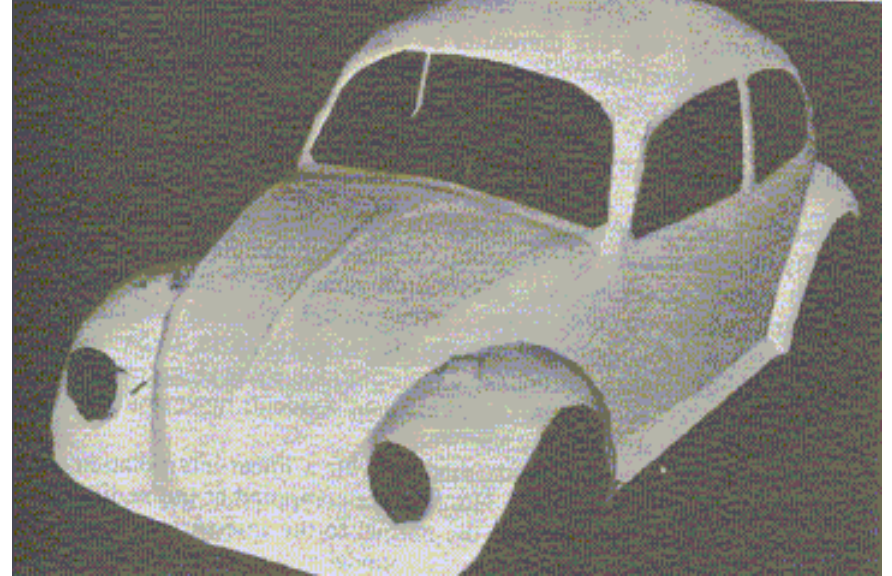
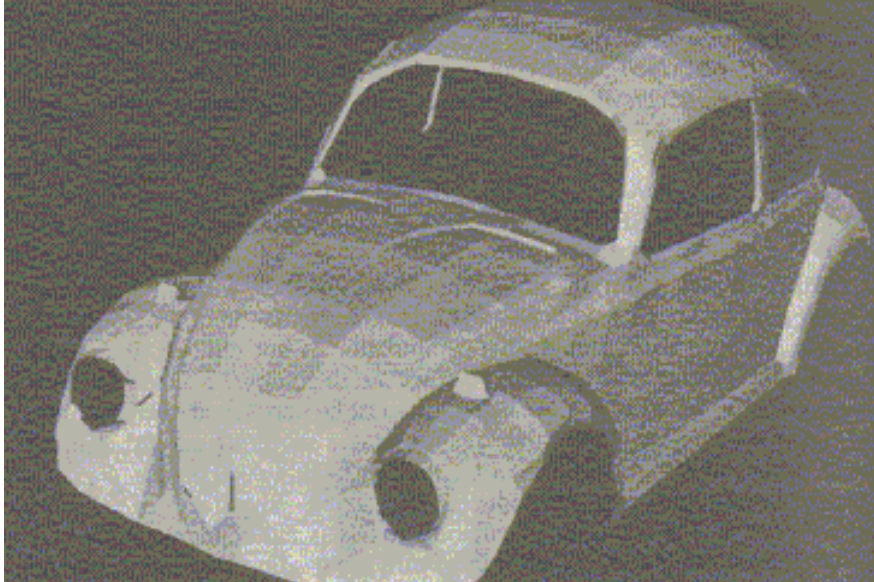
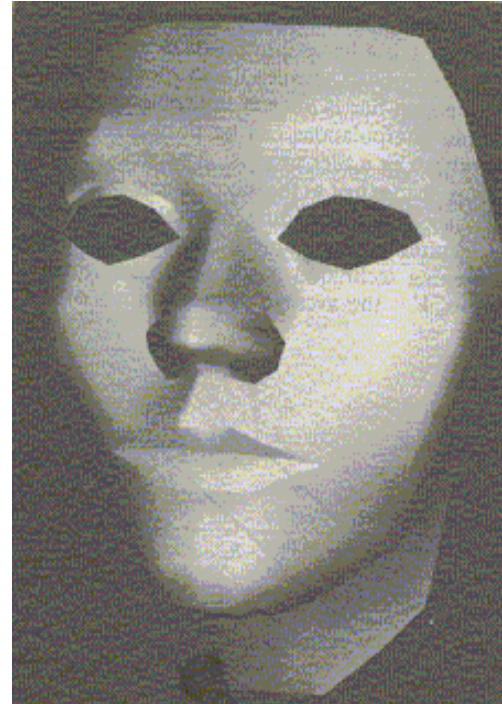
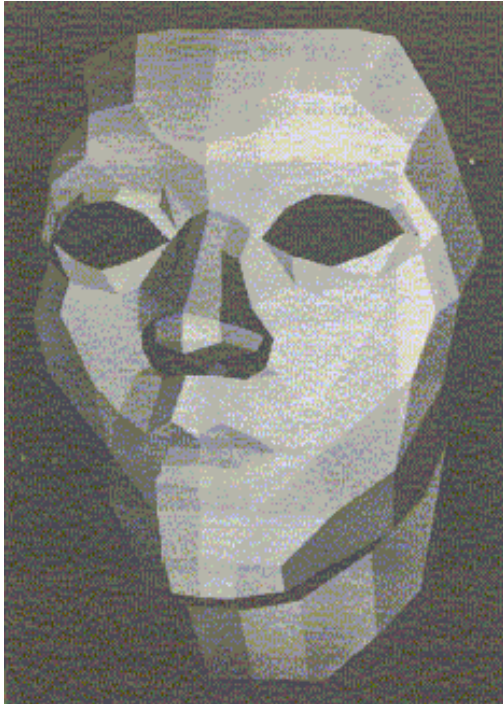
- straight line segments

- curve



gives poor representation where radius of curvature is small





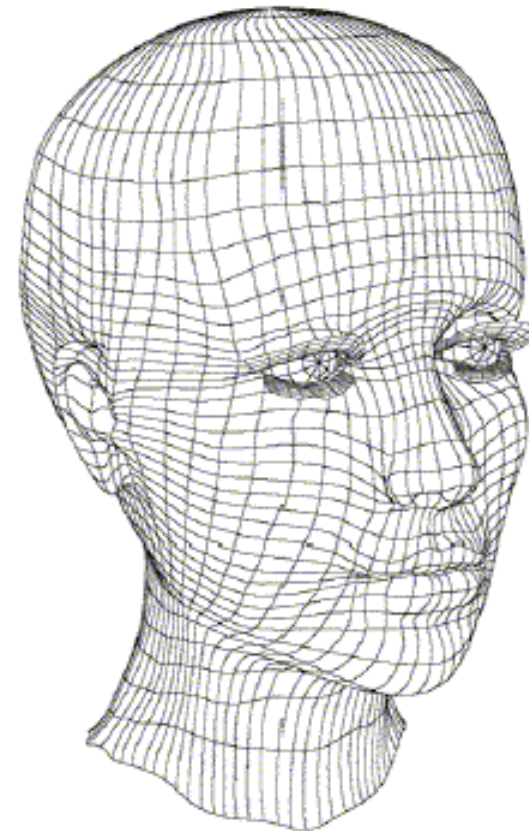
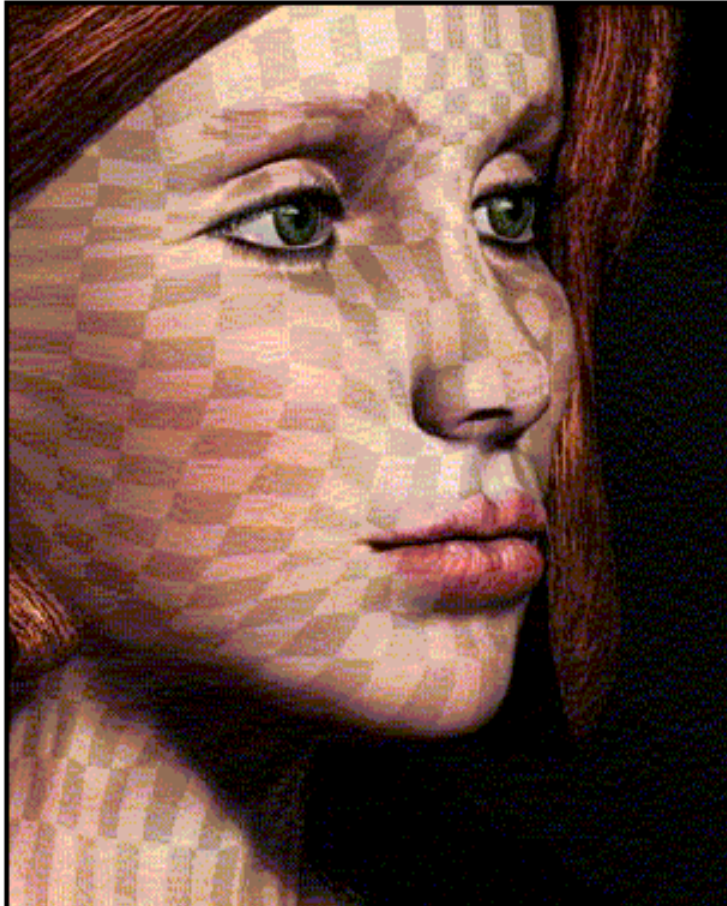
POLYGON MESH

SMOOTH CURVES

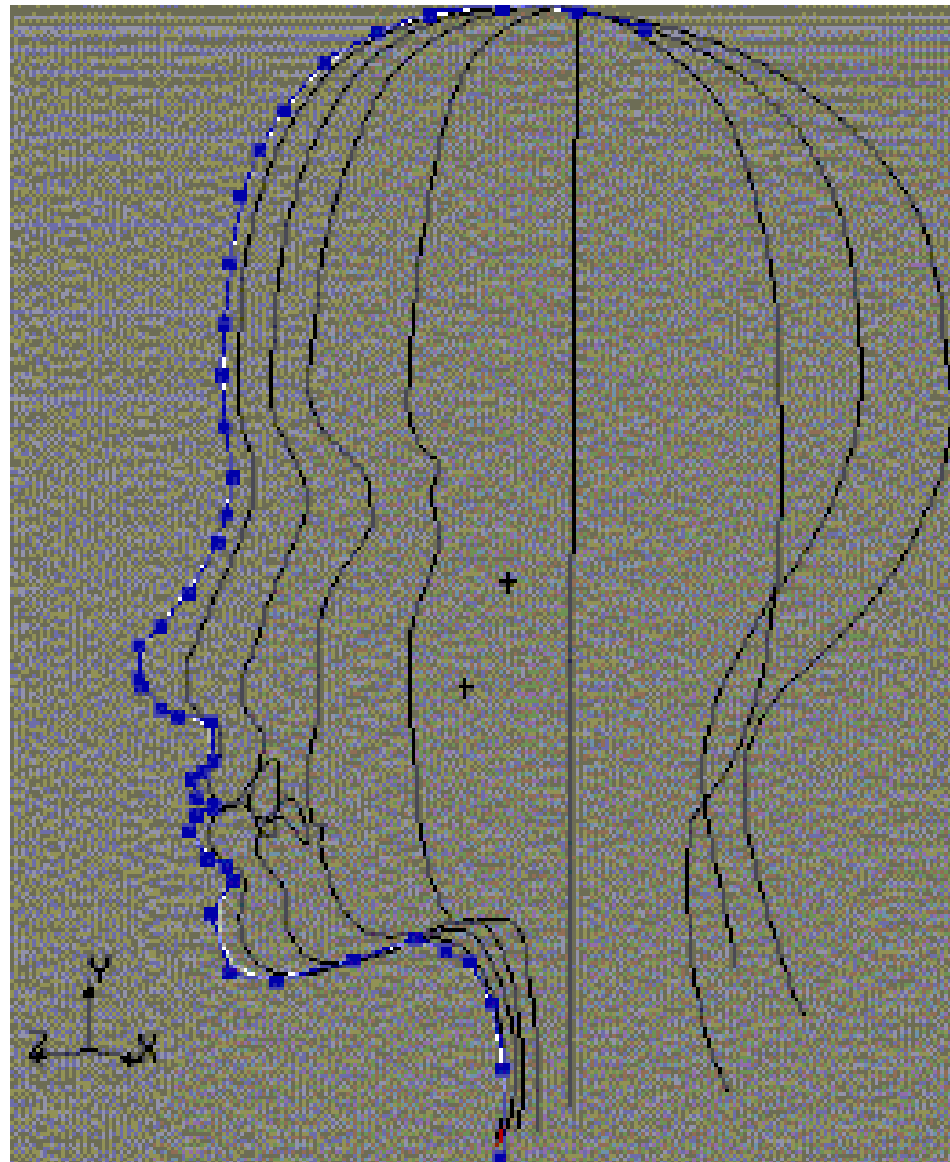
**TUTORIAL: NURBS Head Modeling**  
An (old but still working) Tutorial by Jeremy Birn

<http://www.3drender.com/jbirn/ea/HeadModel.html>

NURBS Head Tutorial







*Start simple, and build in more detail after you like the basic shape.*

## Lesson IV: Representations of Geometry

- ⇒ Curves can be adequately represented as a collection of points
- ⇒ analytical representation has several advantages:
  - Precision
  - Compact storage
  - ease of calculation of intermediate points
  - Slope & radius of curvature are easily determined

Various techniques for analytically representing curves Exist!

- The problem of analytically defining a curve from a known set of data points is INTERPOLATION. A curve that passes through all the data points is said to FIT the data. One common technique for curve fitting is piece wise polynomial approximation

## Lesson IV: Representations of Geometry

- If the datapoints are only approximations to some unknown true values, then a curve that shows the correct trend of the data is required. The curve may not actually pass through any of the data points. The curve is said to FAIR the data (e.g. Least Square)



## NONPARAMETRIC CURVES

Mathematically, either a parametric or a non parametric form is used to represent a curve!

Nonparametric : - explicit  $y = f(x)$   
- implicit  $f(x, y) = 0$

Closed or multiple-value curves cannot be represented explicitly.

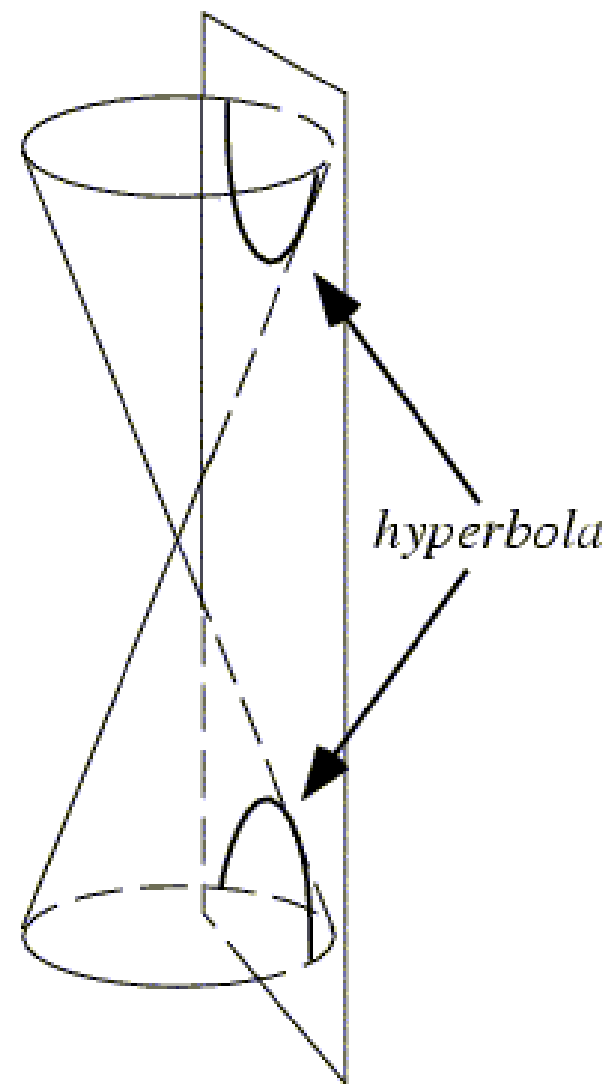
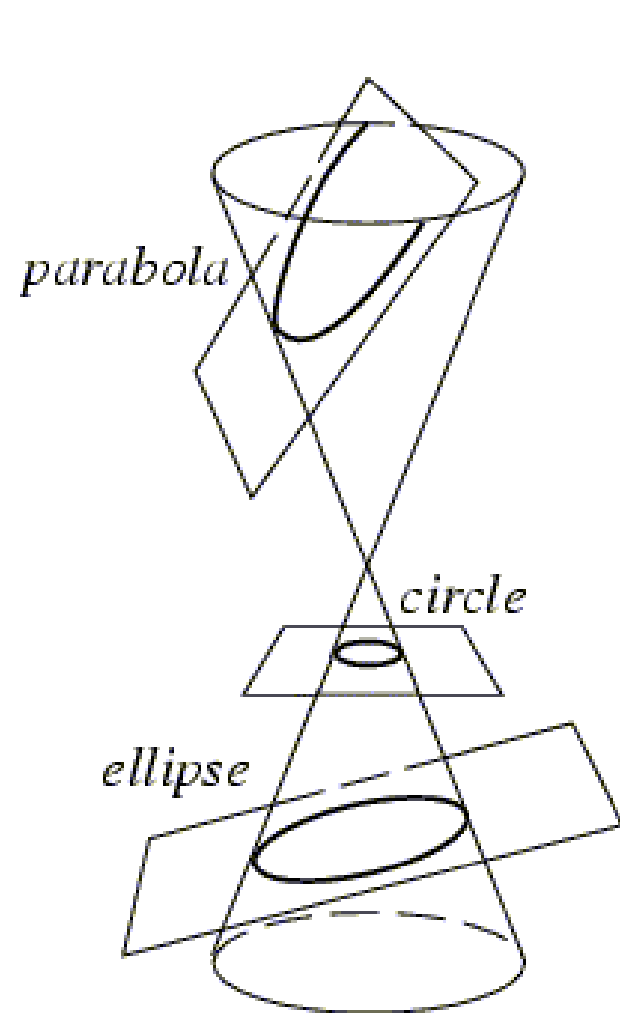
Implicit representations can represent closed or multi value curves

general second degree implicit equation:

$$ax^2 + 2bxy + cy^2 + 2dx + 2ey + f = 0$$

⇒ Conic Sections : parabola  
hyperbola  
ellipse

Explicit and Implicit non parametric Curve representations are axis dependent



## Conic Sections

### *Implicit non-parametric representation of a cone*

Let  $\mathbf{a} = [a_x a_y a_z]$  be the coordinates of the vertex of a cone with its axis in the direction of a unit vector  $\mathbf{n}$ , and with a half angle  $\alpha = \cos^{-1} k, k > 0$ .

The implicit vector equation of the cone, where  $\mathbf{x}$  is any point on the cone is:

$$[(\mathbf{x} - \mathbf{a}) \cdot \mathbf{n}]^2 - k^2(\mathbf{x} - \mathbf{a}) \cdot (\mathbf{x} - \mathbf{a}) = 0$$

### *Parametric equation of a torus*

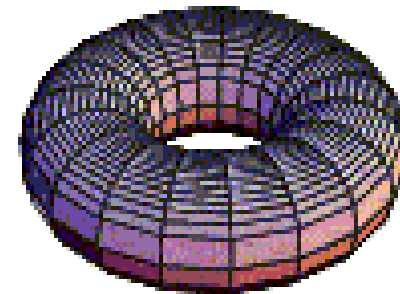
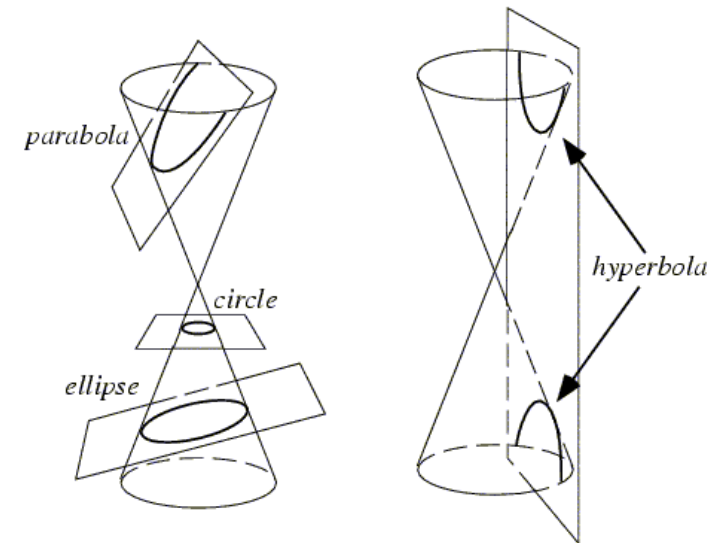
The parametric equation of a torus centered around the point  $x_1, y_1, z_1$  and with its major symmetry axis orientated in the direction  $i, j, k$ , is given by:

$$x(\theta, \varphi) = \frac{V}{L}(R + r \cos \varphi) \cos \theta + \frac{i}{L} r \sin \varphi + x_1$$

$$y(\theta, \varphi) = -\frac{ij}{LV}(R + r \cos \varphi) \cos \theta + \frac{k}{V}(R + r \cos \varphi) \sin \theta + \frac{j}{L} r \sin \varphi + y_1$$

$$z(\theta, \varphi) = -\frac{ij}{LV}(R + r \cos \varphi) \cos \theta - \frac{j}{V}(R + r \cos \varphi) \sin \theta + \frac{k}{L} r \sin \varphi + z_1$$

$$0 \leq \theta < 2\pi \quad -\pi/2 \leq \varphi \leq \pi/2$$





## PARAMETRIC CURVES

- each coordinate of a point on a curve is represented as function of a single parameter
- The position vector of a point on the curve is fixed by the value of the parameter

in Cartesian coordinates :

$$x = x(t)$$

$$y = y(t)$$

Position Vector of point on Curve:

$$P(t) = [x(t) \ y(t)]$$

$$\frac{dy}{dx} = \frac{dy/dt}{dx/dt} = \frac{y'(t)}{x'(t)}$$

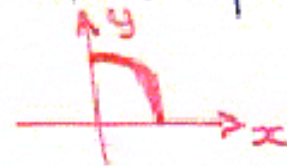
for  $x'(t)=0 \Rightarrow$  Slope infinite

- Since a point on a parametric curve is specified by a single value of the parameter, the parametric form is axis-independent
- The parameter range is often normalized for the curve segment of interest





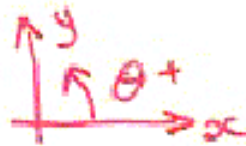
Comparison of non parametric and parametric representations for a circle in the first quadrant



- Non parametric representation for unit circle in first quadrant

$$y = +\sqrt{1-x^2} \quad 0 \leq x \leq 1$$

- Parametric form for unit circle



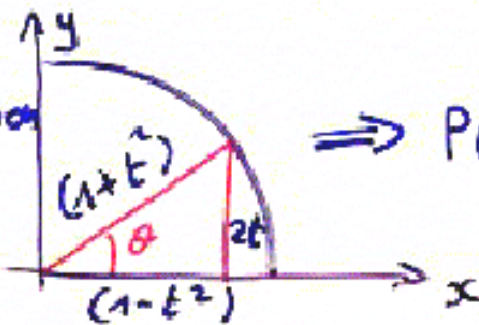
$$\begin{aligned} x &= \cos \theta \\ y &= \sin \theta \quad 0 \leq \theta < 2\pi \end{aligned}$$

$$P(\theta) = [x \ y] = [\cos \theta \ \sin \theta] \quad 0 \leq \theta \leq 2\pi$$

for parametric  $\Rightarrow$  equal parameter increments produce equal arc lengths  
 $\Rightarrow$  computation of trigonometric functions is expensive

THERE IS NO UNIQUE PARAMETRIC REPRESENTATION FOR A CURVE

following representation is less expensive



$$\Rightarrow P(t) = \begin{bmatrix} \overbrace{\frac{(1-t^2)}{(1+t^2)}}^{\cos \theta} & \overbrace{\frac{2t}{1+t^2}}^{\sin \theta} \end{bmatrix}$$

$$0 \leq t \leq 1$$

Determination of a point on a Parametric Curve

Determine the value of  $y$  for a given  $x$

e.g.  $x=0.5$  for unit circle

$\Rightarrow$  explicit representation

$$y = \sqrt{1-x^2} = \sqrt{1-0.5^2} = \sqrt{0.75} = 0.866$$

⇒ parametric representation

I)

$$x = \cos \theta$$

$$y = \sin \theta$$

first solve for the parameter  $t$  in  $x$

$$\theta = \cos^{-1}(x) = \cos^{-1}(0.5) = 60^\circ$$

$$y = \sin(60^\circ) = 0.866$$

II)

$$x = \frac{1-t^2}{1+t^2} \Rightarrow t = \left( \frac{1-x}{1+x} \right)^{1/2} = \frac{1}{\sqrt{3}} = 0.57735$$

$$y = \frac{2t}{1+t^2} \Rightarrow y = \frac{2/\sqrt{3}}{4/3} = \frac{\sqrt{3}}{2} = 0.866$$

⇒ For more complex parametric representations ⇒ iterative techniques

Parametric and Non parametric representations both have advantages and disadvantages !

## Arc Length Parameterization

Because different parametric forms of the same curve may induce different continuity results, one must ask the question: Is there a parameterization that we can trust for continuity discussion? The answer is yes, since mathematicians have solved this problem long time ago. The trick is to use *arc length* as a parameter.

Let a curve segment have length  $s$ . One can parameterize this curve such that  $\mathbf{f}(u)$  is the point having a distance  $u$  from the initial point  $\mathbf{f}(0)$ , where  $u$  is in the range of 0 and  $s$ . With this arc length parameterization, as  $u$  moves from 0 to  $s$ ,  $\mathbf{f}(u)$  moves on the curve from  $\mathbf{f}(0)$  to  $\mathbf{f}(s)$  in the same speed. Therefore, the tangent vector which measures speed is of unit-length. Not only this, many formulas shown on this and previous pages can be simplified.

Why don't we use arc length parameterization to simplify our computation? The answer is quite simple. While arc length parameterization is simple and elegant in theory, it is tedious in computation and impractical. One can easily design a curve in some handy parameterization; but, reparameterizing it with arc length sometime is extremely difficult. That is, finding arc length is not an easy task, since it requires to integrate a function involving the use of square root.

Therefore, even though non-arc length parameterization could cause problems, we will not use the arc length parameterization.

## Geometric Continuity

Many  $C^1$  continuous curves are curvature continuous but not  $C^2$  at the joining point and some of them may not even be twice differentiable. These curves look smooth at the joining point and also look smooth when moving from a segment to the other. Moreover, as mentioned earlier, after a change of variables, some of them could become  $C^2$  at the joining point. But, a reparameterization that can make this to happen could be difficult to find. Therefore, we intend to relax the requirement of  $C^2$  to the following:

**Two curve segments are said  $G^k$  geometric continuous at the joining point if and only if all  $i$ -th derivatives,  $i$  less than or equal to  $k$ , computed with arc length parameters agree at the joining point.**

Oops, we use arc length in the definition! But, don't worry, since arc length parameterization is not required as shown by the following equivalent definition:

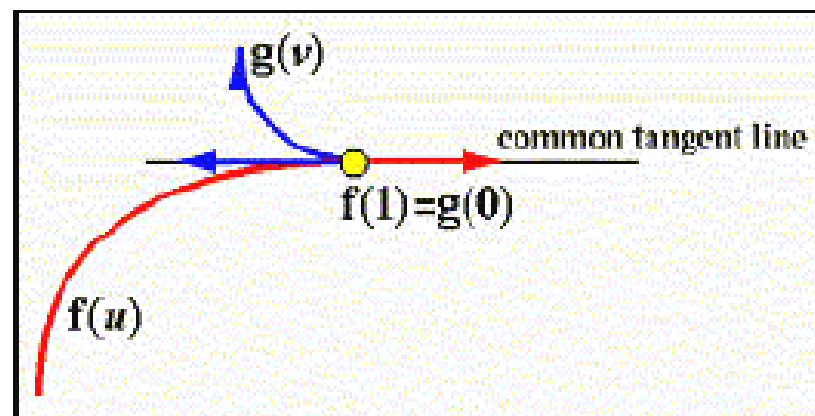
**Two curve segments are said  $G^k$  geometric continuous at the joining point if and only if there exists two parameterizations, one for each curve segment, such that all  $i$ -th derivatives,  $i$  less than or equal to  $k$ , computed with these new parameterizations agree at the joining point.**

This is better but not good enough because we really do not know how to find such parameterizations. Fortunately, the cases of  $k = 1$  and  $k = 2$  are very simple. Let us start with the case of  $k=1$ .

This is better but not good enough because we really do not know how to find such parameterizations. Fortunately, the cases of  $k = 1$  and  $k = 2$  are very simple. Let us start with the case of  $k=1$ .

**Two  $C^0$  curve segments are said  $G^1$  geometric continuous at the joining point if and only if vectors  $f'(u)$  and  $g'(v)$  are in the *same* direction at the joining point. Note that  $f'(u)$  and  $g'(v)$  are evaluated at the joining point.**

Since the tangent vectors at the joining point have the same direction, both curves have the same tangent line at the joining point. **However, the converse does not hold.** More precisely, two curve segments having the same tangent line **does not** imply they are  $G^1$  at the joining point. In the following figure, curves  $f(u)$  and  $g(v)$  have a common point at which the tangent lines are the same. However, the tangent vectors point to the opposite directions, and, as a result, they are not  $G^1$  at the joining point. Based on this definition, all previously discussed example curves are  $G^1$  at the joining points.



## What does a CAD system do with the equations?

### Surface Intersections

<u>Surface Types</u>	<u>Nature of Solution</u>	<u>level of difficulty</u>
1. Two Planes	straight line	trivial
2. Plane section of quadric	planar conic curve	simple
3. two quadric surfaces	quartic Space Curve	semi analytic
4. Plane Section of bicubic Patch	degree 18 plane Curve	numerical
5. two bicubic patches	degree 324 Space Curve	Heuristic



## What does a CAD system do with the equations?

### Other Problems

1. Point on a Curve
2. Point on a Surface
3. Line and a Surface (find all intersections!)
4. Intersection of two Curves (find all intersections!)
5. Intersection of a curve and a surface
6. Interference problem : detect when two geometric objects intersect (output  $\Rightarrow$  YES OR NO)
7. Surface Intersections :
  - hunting phase
  - tracing phase
  - ordering phase
8. Offset curves

## What does a CAD system do with the equations?

### 8. Offset curves

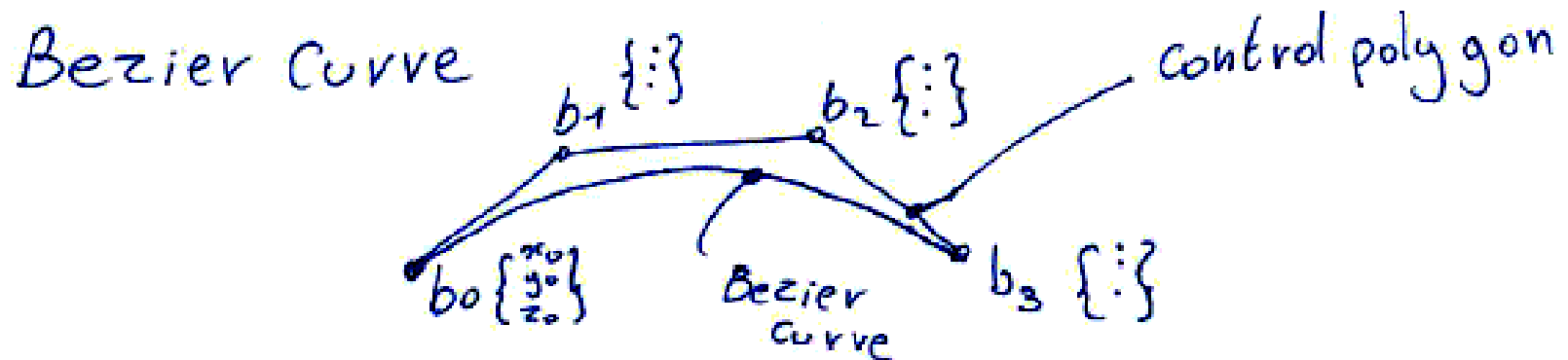
$\vec{r}_0(t) = \vec{r}(t) + d \vec{n}(t)$  in general  $\vec{r}_0(t)$  is not a polynomial (or rational) curve

$\vec{r}_0(t)$  is an algebraic irreducible curve with an implicit equation  $f_0(x,y) = 0$  of degree  $4n-2$

This fact prompted the formulation of several heuristic piece-wise-polynomial approximation schemes for offset curves

9. When a polynomial curve is rendered at a uniform sequence of parameters  $\{t_k\}$  the resulting geometric points  $\{t_k\}$  are not uniformly spaced over the curve. To correct this we need  $s(t)$   
 $s$ : arc length ;  $s(t)$  is a complex integral

## Lesson V: Identifying Unknown Coefficients in Parametric Polynomial Equations of a Curve



$$x(t) = a_n t^n + \dots + a_0$$

$$y(t) = b_n t^n + b_{n-1} t^{n-1} + \dots + b_0$$

$$z(t) = c_n t^n + c_{n-1} t^{n-1} + \dots + c_0$$

Monomial Basis:  $\{t^n, t^{n-1}, \dots, t^0\}$

$(n+1) \times 3$  unknown coefficients

$a_n \dots a_0$	??
$b_n \dots b_0$	
$c_n \dots c_0$	

# Lesson V: Identifying Unknown Coefficients in Parametric Polynomial Equations of a Curve

$$\begin{aligned}
 x(t) &= a_n t^n + \dots + a_0 \\
 y(t) &= b_n t^n + b_{n-1} t^{n-1} + \dots + b_0 \\
 z(t) &= c_n t^n + c_{n-1} t^{n-1} + \dots + c_0
 \end{aligned}$$

$\uparrow \quad \nearrow \quad \rightarrow$   
 Monomial Basis:  $\{t^n, t^{n-1}, \dots, t^0\}$

$(n+1) \times 3$  unknown coefficients

$a_n \dots a_0$   
 $b_n \dots b_0$  ??  
 $c_n \dots c_0$  ??

$$\begin{aligned}
 \Rightarrow x(0) &= x_0 = a_n(0)^n + \dots + a_0 \\
 y(0) &= y_0 = b_n(0)^n + \dots + b_0 \\
 z(0) &= z_0 = c_n(0)^n + \dots + c_0
 \end{aligned}$$

$$\begin{aligned}
 x(t_1) &= x_1 = a_n(t_1)^n + \dots + a_0 \\
 y(t_1) &= y_1 = b_n(t_1)^n + \dots + b_0 \\
 z(t_1) &= z_1 = c_n(t_1)^n + \dots + c_0
 \end{aligned}$$

$\vdots$

$$\begin{aligned}
 x(t_n) &= x_n = a_n(t_n)^n + \dots + a_0 \\
 y(t_n) &= y_n = b_n(t_n)^n + \dots + b_0 \\
 z(t_n) &= z_n = c_n(t_n)^n + \dots + c_0
 \end{aligned}$$

$3 \times (n+1)$  equations

unique solution

$t_n = 1$

## Lesson V: Identifying Unknown Coefficients in Parametric Polynomial Equations of a Curve

a control polygon with  $n+1$  control points:  
 $b_0 \ b_1 \ b_2 \ \dots \ b_n$

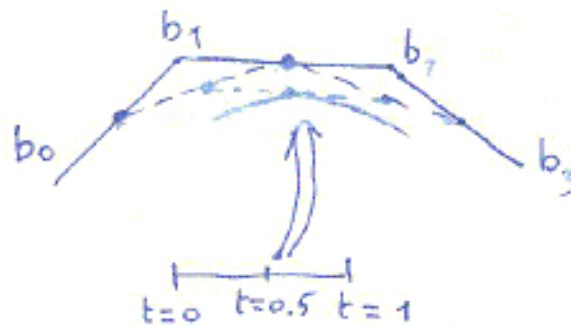
has a unique bezier curve of degree  $n$

$\Rightarrow$  Bernstein basis  $B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$   $B^n(t) = \sum_{j=0}^n b_j B_j^n(t)$

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

control  
point

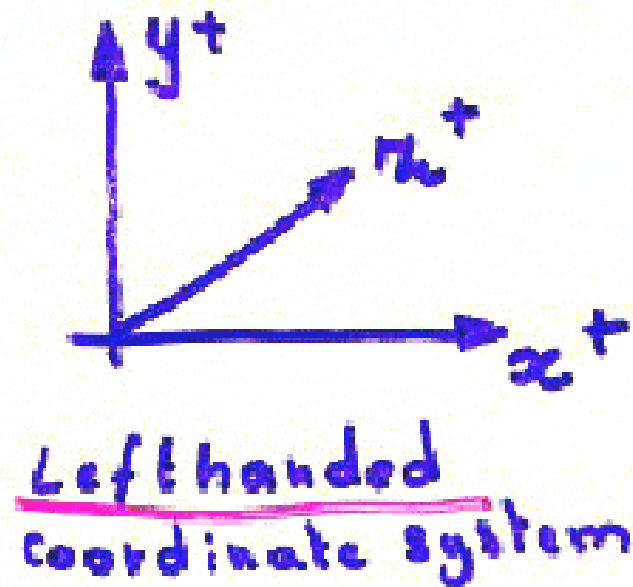
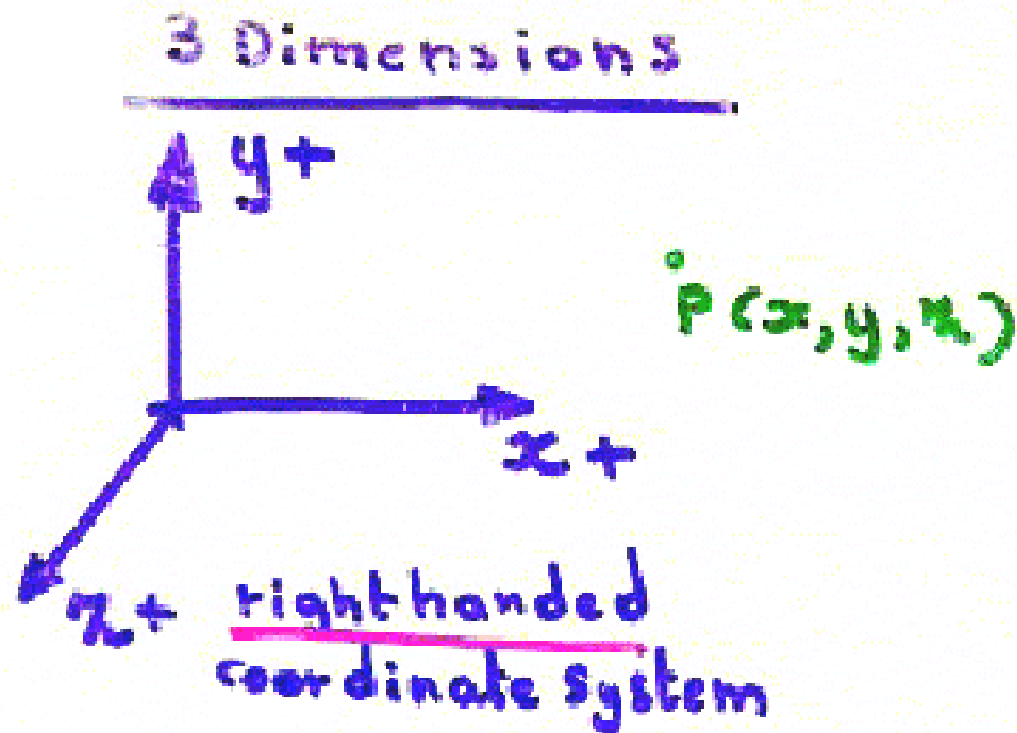
Bernstein  
polynomials



deCasteljau construction

## Lesson VI: Basic 3D Geometry

### Coordinate System



We adopt the right handed system



## Lesson VI: Basic 3D Geometry

### Parametric Line & Plane Equation

$$P_1(x_1, y_1, z_1) \text{ \& \& } P_2(x_2, y_2, z_2)$$

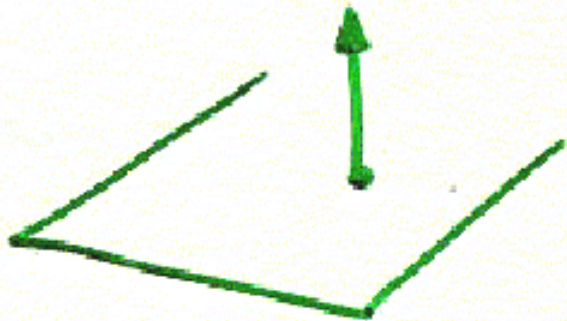
$$\text{LINE} \Rightarrow \left. \begin{aligned} x &= (x_2 - x_1)u + x_1 \\ y &= (y_2 - y_1)u + y_1 \\ z &= (z_2 - z_1)u + z_1 \end{aligned} \right\} \begin{array}{l} \text{Parametric form} \\ \text{of equation of a line} \end{array}$$

$$\text{PLANE} \Rightarrow Ax + By + Cz + D = 0$$
$$ax + by + cz + 1 = 0 \Rightarrow \text{three constants}$$
$$a = \frac{A}{D} ; b = \frac{B}{D} ; c = \frac{C}{D} \text{ are needed to specify a plane}$$

## Lesson VI: Basic 3D Geometry

### Dot and Cross Vector Product

- a plane can be specified by three not colinear point
- a single point in the plane and the perpendicular direction to the plane (normal vector)



Vector DOT-PRODUCT

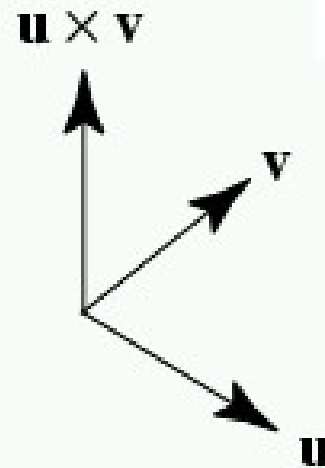
$$A = [A_x, A_y, A_z]$$

$$B = [B_x, B_y, B_z]$$

$$A \cdot B = A_x B_x + A_y B_y + A_z B_z = |A| \cdot |B| \cos \theta$$



# VECTOR CROSS PRODUCT



For **vectors**  $\mathbf{u} = (u_x, u_y, u_z)$  and  $\mathbf{v} = (v_x, v_y, v_z)$  in  $\mathbb{R}^3$ , the cross product is defined by

$$\mathbf{u} \times \mathbf{v} = \hat{\mathbf{x}}(u_y v_z - u_z v_y) - \hat{\mathbf{y}}(u_x v_z - u_z v_x) + \hat{\mathbf{z}}(u_x v_y - u_y v_x) \quad (1)$$

$$= \hat{\mathbf{x}}(u_y v_z - u_z v_y) + \hat{\mathbf{y}}(u_z v_x - u_x v_z) + \hat{\mathbf{z}}(u_x v_y - u_y v_x). \quad (2)$$

This can be written in a shorthand **notation** which takes the form of a **determinant**

$$\mathbf{u} \times \mathbf{v} = \begin{vmatrix} \hat{\mathbf{x}} & \hat{\mathbf{y}} & \hat{\mathbf{z}} \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{vmatrix}. \quad (3)$$

## Lesson VI: Basic 3D Geometry

### 3D Transformations, Scaling, Translation, Rotation

#### 3D transformations

homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ or } [x \ y \ z \ w]$$

↑ we use

⇒ Scaling transformation

$$S = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

⇒ Translation

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

## Lesson VI: Basic 3D Geometry

### 3D Transformations, Scaling, Translation, Rotation

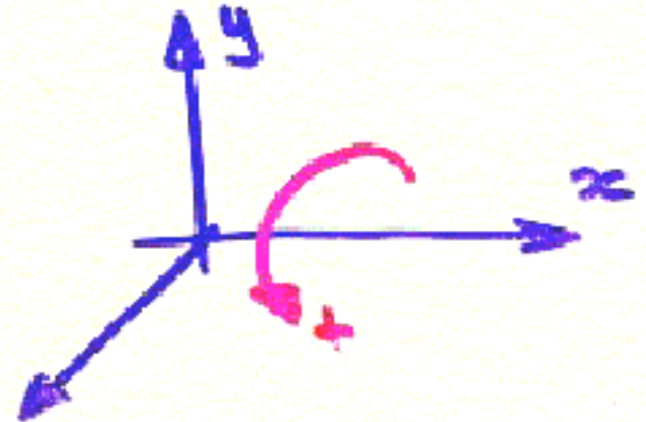
⇒ rotation around z-axis over angle  $\theta$

$$R_z = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



⇒ around x-axis

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



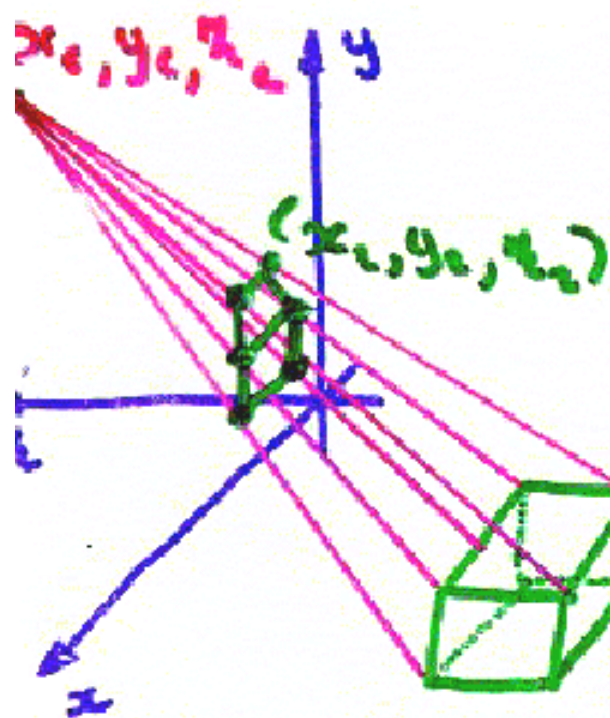
## Lesson VII: Perspective and Parallel Projections & Corresponding Clipping Volumes

### Perspective Projection

the further away an object is from the viewer the smaller it appears (depth cue).

In a perspective projection the lines of projection are not parallel, they converge at a single point called the CENTER OF PROJECTION (e.g. Light ray converging to the viewer's eye).





equation of a projection ray  
through  $x_1, y_1, z_1$

$$\begin{cases} x = x_c + (x_1 - x_c)u \\ y = y_c + (y_1 - y_c)u \\ z = z_c + (z_1 - z_c)u \end{cases}$$

?  $x_2, y_2$ ?  $\Rightarrow z_c = 0$

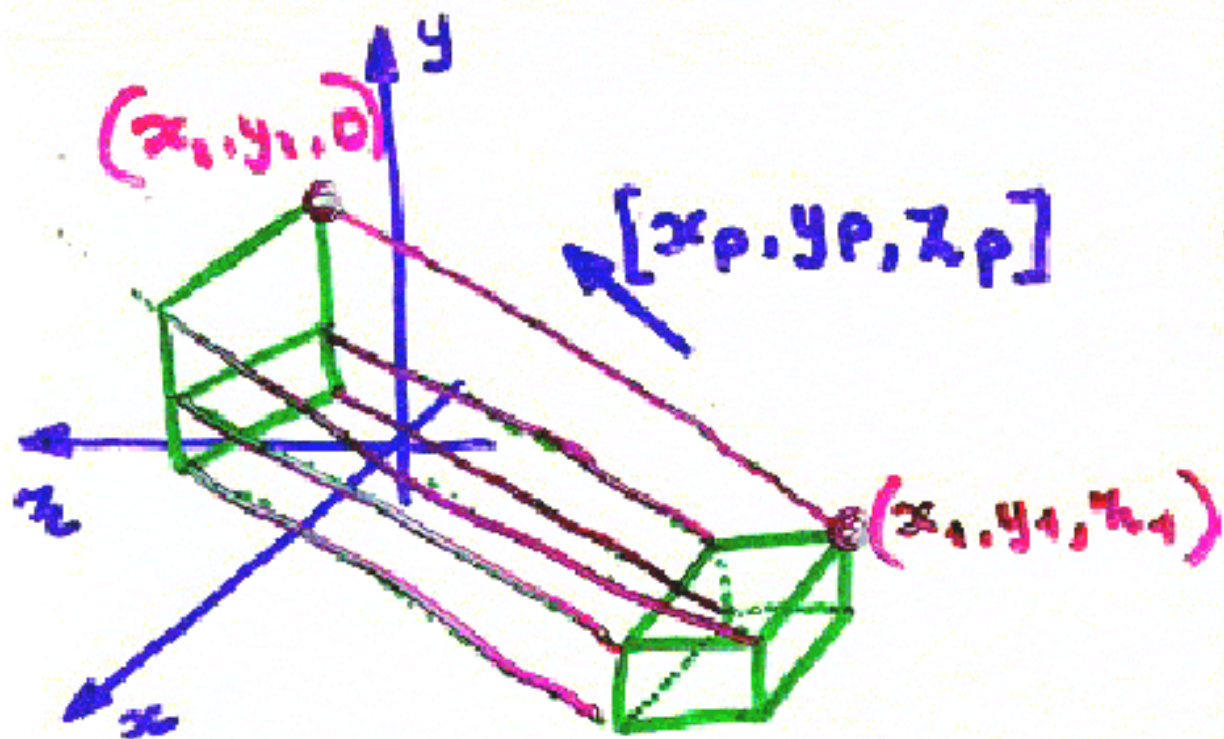
$$\Rightarrow u = -\frac{z_c}{z_1 - z_c}$$

$$\Rightarrow x_2 = x_c - z_c \frac{x_1 - x_c}{z_1 - z_c}$$

$$\Rightarrow y_2 = y_c - z_c \frac{y_1 - y_c}{z_1 - z_c}$$

$$x_2 = \frac{x_c z_1 - x_1 z_c}{z_1 - z_c}$$

$$; y_2 = \frac{y_c z_1 - y_1 z_c}{z_1 - z_c}$$



$$\begin{cases} x = x_1 + x_p u \\ y = y_1 + y_p u \quad (I) \\ z = z_1 + z_p u \end{cases}$$

(I)  $\Rightarrow$  parametric equation of line through  $(x_1, y_1, z_1)$  and in the direction  $[x_p, y_p, z_p]$



To find the parallel projection of  $(x_1, y_1, z_1)$  we must find the transformation matrix which transforms  $(x_1, y_1, z_1)$  into  $(x_2, y_2, 0)$  on the  $x, y$  plane which corresponds to the view plane.

$$(I) \Rightarrow x_2 = 0 = x_1 + x_p u \Rightarrow u = -\frac{x_1}{x_p}$$

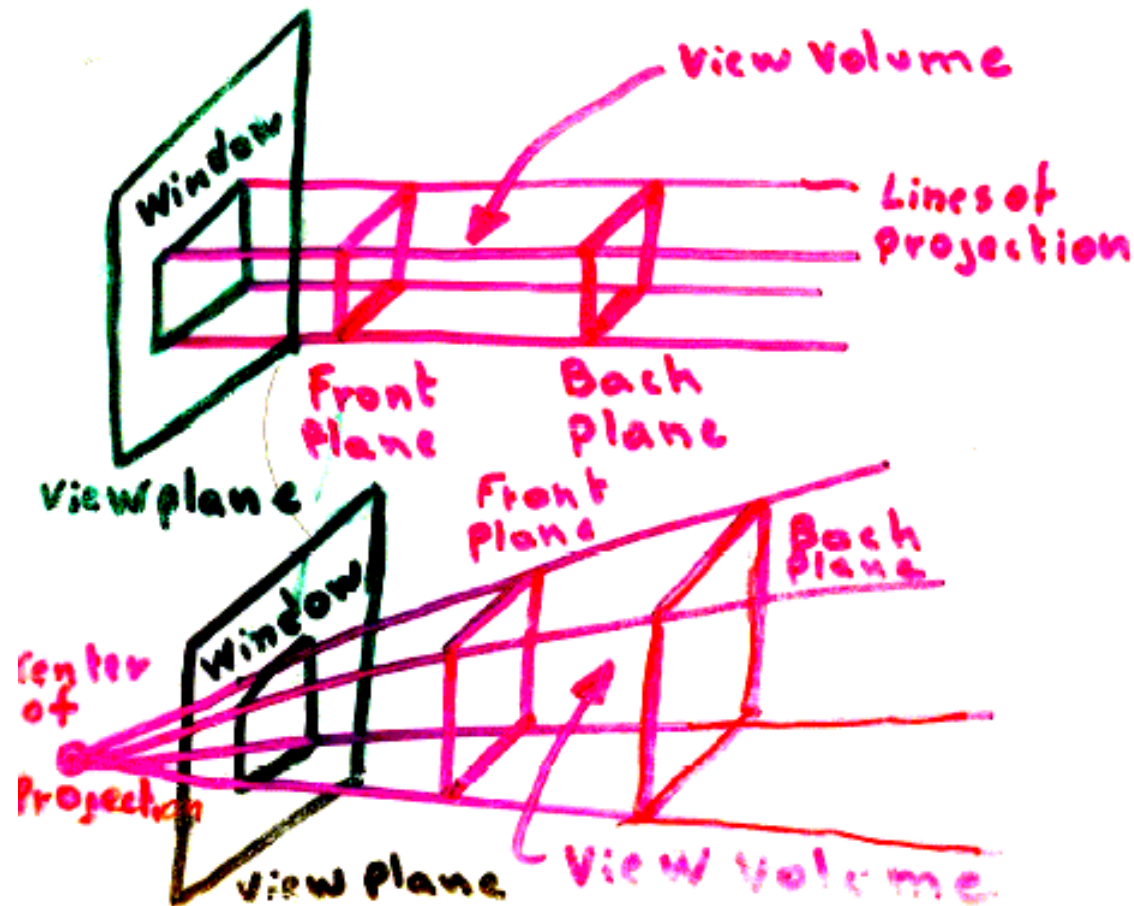
$$x_2 = x_1 - \frac{x_1}{x_p} x_p \quad ; \quad y_2 = y_1 - \frac{x_1}{x_p} y_p$$

Matrix notation

$$[x_2, y_2] = [x_1, y_1, x_1] \begin{vmatrix} 1 & 0 \\ 0 & 1 \\ -\frac{x_p}{x_p} & -\frac{y_p}{x_p} \end{vmatrix}$$



## Clipping Volumes



PARALLEL  
PROJECTION

PERSPECTIVE  
PROJECTION

# Lesson VIII: CONVEX HULL DEFINITION

## Definition of POINTS & VECTORS

### 1. Points & Vectors

mathematical description of an object

⇒ define a coordinate system to describe the object mathematically – analytically

- Space does not possess a preferred coordinate system!
  - We have to define one ourselves!
  - Our interest is in the object and not in its relationship to some arbitrary coordinate system!
  - The methods that we will develop must therefore be independent of the choice of a coordinate system
- ⇒ COORDINATE-FREE METHODS!

**VECTOR = DIFFERENCE OF TWO POINTS**

\* Points are elements of three-dimensional euclidean space  $\mathbb{E}^3$   
or better affine space  
 $a, b \in \mathbb{E}^3$

\* Vectors are elements of the three-dimensional linear space  $\mathbb{R}^3$

\* We adopt the convention here to write them as coordinate columns  $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$  --

\* for any two points  $a$  and  $b$ , there is a unique vector  $v$  that points from  $a$  to  $b$

$$v = b - a ; a, b \in \mathbb{E}^3, v \in \mathbb{R}^3$$

\* given a vector  $v$ , there are infinitely many pairs of points  $a, b$  such that  $v = b - a$

$$\forall w \Rightarrow v = (b+w) - (a+w)$$

\* assigning  $a+w$  to all  $a \in \mathbb{E}^3$  is called a translation

$\Rightarrow$  VECTOR ARE INVARIANT UNDER TRANSLATION; POINTS NOT!

## Lesson VIII: CONVEX HULL DEFINITION

### Definition of BARYCENTRIC COMBINATIONS

- Elements of point space  $\mathbb{E}^3$  can only be subtracted, this operation yields a vector
- Points cannot be added (different coordinate systems will produce different solutions)
- However, addition-like operations are defined for points:  
BARYCENTRIC COMBINATIONS  $\Rightarrow$  weighted sums of points where the weights sum to one:

$$b = \sum_{j=0}^n \alpha_j b_j \quad \begin{array}{l} b_j \in \mathbb{E}^3 \\ \alpha_0 + \dots + \alpha_n = 1 \end{array} \quad (1)$$

(1) can be rewritten as

$$b = b_0 + \sum_{j=1}^n \alpha_j (b_j - b_0)$$

$$\alpha_0 = 1 - (\alpha_1 + \dots + \alpha_n)$$

which is the sum of a point and a vector

example of barycentric combination: centroid  $g$  of a triangle

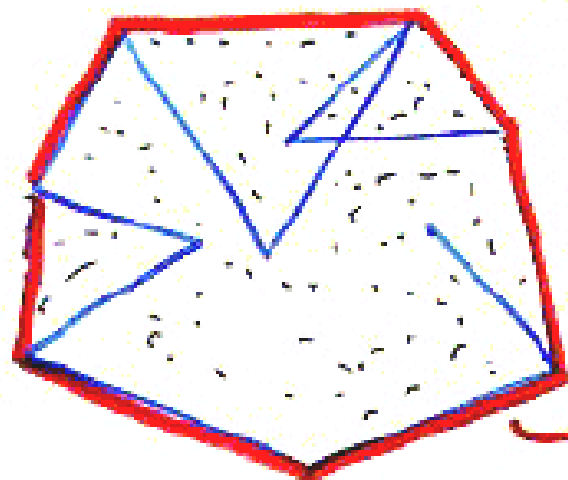
$$a, b, c \in \mathbb{E}^3 \quad g = \frac{1}{3}a + \frac{1}{3}b + \frac{1}{3}c$$

"Bary Center"  $\equiv$  "center of gravity"  $b = \frac{\sum m_j b_j}{\sum m_j}$

SPECIAL CASE OF BARYCENTRIC COMBINATIONS: CONVEX COMBINATIONS

$$\begin{cases} \sum \alpha_j = 1 \\ \alpha_j \geq 0 \quad \forall j \end{cases} \quad \Leftrightarrow \text{always "INSIDE"}$$

$\Rightarrow$  CONVEX HULL OF A POINT SET  
is the set formed by all convex combinations of a point set



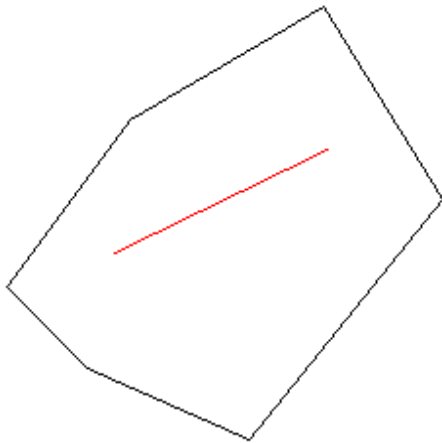
**CONVEX HULL**

- a convex combination



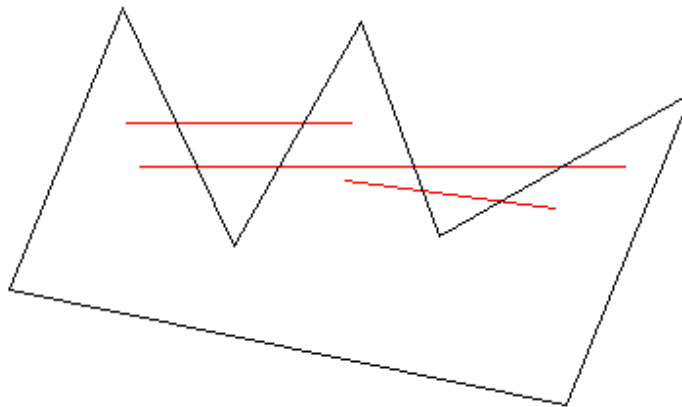
## OTHER DEFINITION OF CONVEX POLYGON:

- For all possible two points inside the polygon
  - take two points inside the polygon
  - connect the two points by a straight line
  - if all points on this line are inside
  - then polygon is convex
  - else it is concave



CONVEX

NON-CONVEX = CONCAVE



## Lesson VIII: CONVEX HULL DEFINITION

### Definition of AFFINE MAPS

#### 2. AFFINE MAPS

A map  $\Phi$  that maps  $\mathbb{E}^3$  into itself is called an affine map if it leaves barycentric combinations invariant

$$x = \sum \alpha_j a_j \quad x, a_j \in \mathbb{E}^3$$

$$\Phi \text{ affine map} \Leftrightarrow \Phi x = \sum \alpha_j \Phi a_j$$

$$\Phi x, \Phi a_j \in \mathbb{E}^3$$

$x = \sum \alpha_j a_j$  specifies how we have to weight the points  $a_j$  so that their weighted average is  $x$ . This relation will still be valid if we apply an affine map to all points  $a_j$  and  $x$ .



In a given coordinate system a point  $x$  is represented by a coordinate triple  $x$

familiar form  $\Phi x = Ax + v$

$A \Rightarrow 3 \times 3$  matrix

$v \Rightarrow$  vector  $v \in \mathbb{R}^3$

Proof

$$\begin{aligned}\Phi(\sum \alpha_j a_j) &= A(\sum \alpha_j a_j) + v \\ &= \sum \alpha_j A a_j + \sum \alpha_j v \\ &= \sum \alpha_j (A a_j + v) \\ &= \sum \alpha_j \Phi a_j\end{aligned}$$

Examples of affine maps

The IDENTITY  $v=0$   $A=I$

A TRANSLATION  $A=I$   $v =$  any vector called Translation vector

A SCALING

A ROTATION

A SHEAR example  $A = \begin{bmatrix} 1 & a & b \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix}$

An important special case of affine maps are the  
EUCLIDEAN MAPS or RIGID BODY MOTIONS which are  
characterized by orthonormal matrices  $A \Leftrightarrow AA^T = I$   
They Leave Lengths and Angles unchanged!  
The rank of  $A$  has an important geometric interpretation  
if  $\text{rank}(A) = 3$  then  $\phi$  maps 3D into 3D objects  
if  $\text{rank}(A) = 2$  then  $\phi$  is a parallel projection onto  
a plane (or line if  $\text{rank}(A) = 1$ )

## Lesson VIII: CONVEX HULL DEFINITION

### Definition of LINEAR INTERPOLATION

#### 3. LINEAR INTERPOLATION

Let  $a, b$  be two distinct points in  $\mathbb{E}^3$

The set of all points  $x \in \mathbb{E}^3$  of the form

$$x = x(t) = (1-t)a + tb; \quad t \in \mathbb{R}$$

is called the STRAIGHT LINE through  $a$  and  $b$

any three points on a straight line are said to be COLINEAR

$t=0$      $x(0)=a$  passes through  $a$

$t=1$      $x(1)=b$  passes through  $b$

for  $0 \leq t \leq 1$  the point  $x$  is between  $a$  and  $b$ , for all other values of  $t$  it is outside

$x(t) = (1-t)a + tb$  is a barycentric combination of two points in  $\mathbb{E}^3$

The same holds for the three points  $0, t, 1$  in  $\mathbb{E}^1$ :

$$t = (1-t) \cdot 0 + t \cdot 1$$

So  $t$  is related to  $0$  and  $1$  by the same barycentric combination that relates  $x$  to  $a$  and  $b$ . But then by the definition of affine maps, the three points  $a, b, c$  in three space are an affine map of the three points  $0, t, 1$  in one space! **THUS LINEAR INTERPOLATION IS AN AFFINE MAP OF THE REAL LINE ONTO A STRAIGHT LINE IN  $\mathbb{E}^3$**   
**AFFINE INTERPOLATION  $\equiv$  LINEAR INTERPOLATION**

Linear interpolation is affinely Invariant

affine map  $\bar{\phi}: \mathbb{E}^3 \rightarrow \mathbb{E}^3$

$$x = x(t) = (1-t)a + tb; \quad t \in \mathbb{R}$$

$$\bar{\phi} x = (1-t)\bar{\phi} a + t\bar{\phi} b$$



Closely Related to Linear Interpolation is the Concept of  
BARYCENTRIC COORDINATES (due to Moebius)

Let  $a, x, b$  be three collinear point  $a, x, b \in \mathbb{E}^3$

$$x = \alpha a + \beta b ; \alpha + \beta = 1$$

$\alpha$  and  $\beta$  are called Barycentric coordinates  
with respect to  $a$  and  $b$

— if  $\alpha = (1-t)$   
 $\beta = t$  we obtain the linear interpolation formula

— barycentric Coordinates do not always have to be positive :  
for  $t \notin [0,1]$  either  $\alpha$  or  $\beta$  is negative

- for three colinear points  $a, b, c$  the barycentric coordinates of  $b$  are given

$$\alpha = \frac{\text{Vol}_1(b, c)}{\text{Vol}_1(a, c)} \quad ; \quad \beta = \frac{\text{Vol}_1(a, b)}{\text{Vol}_1(a, c)}$$

$\text{Vol}_1$  is the one dimensional volume between two points (here the signed distance)

- Barycentric coordinates can also be defined on the plane

- Concept of ratio : the ratio of three colinear points  $a, b, c$  is defined by

$$\text{ratio}(a, b, c) = \frac{\text{Vol}_1(a, b)}{\text{Vol}_1(b, c)}$$

## Lesson IX: BEZIER CURVES

### CASTELJAU ALGORITHM

#### The de Casteljau Algorithm

##### Parabolas

- Simple construction for the generation of a parabola  
generalization will lead to Bézier curves

$$b_0, b_1, b_2 \in \mathbb{E}^3 \quad t \in \mathbb{R}$$

$$\begin{aligned} b_0^1(t) &= (1-t)b_0 + t b_1 \\ b_1^1(t) &= (1-t)b_1 + t b_2 \end{aligned} \quad (1)$$

$$b_0^2(t) = (1-t)b_0^1(t) + t b_1^1(t) \quad (2)$$

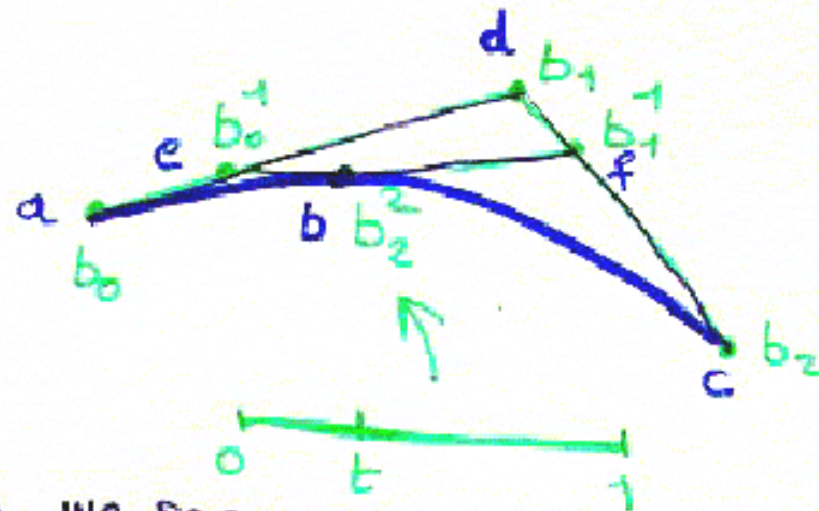
$$(1) \rightarrow \text{in } (2) \quad b_0^2(t) = (1-t)^2 b_0 + 2t(1-t)b_1 + t^2 b_2$$



## Lesson IX: BEZIER CURVES

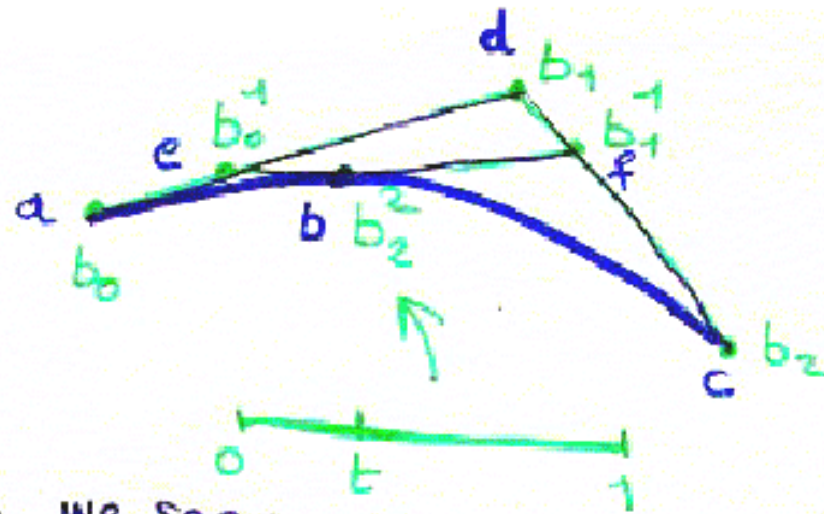
### CASTELJAU ALGORITHM

$b_0^2(t)$  traces out a parabola as  $t$  varies from  $-\infty$  to  $+\infty$   
 We denote this parabola by  $b^2$



Parabola construction by repeated Linear interpolation

from fig we see :



from fig we see:

$$\frac{t}{1-t} = \text{ratio}(b_0, b_0^1, b_1) = \text{ratio}(b_1, b_1^1, b_2) = \text{ratio}(b_0^1, b_2^2, b_1^1)$$

- our construction of a parabola is affinely invariant, because piecewise linear interpolation is affinely invariant
- We obtain a plane curve, due to the fact that  $b^2(t)$  is always a barycentric combination of three points
- Let  $a, b, c$  be three distinct points on a parabola. Let the tangent at  $b$  intersect the tangents at  $a$  and  $c$  in  $e$  and  $f$ , respectively. Let the tangents at  $a$  and  $c$  intersect in  $d$ . Then  $\text{ratio}(a, e, d) = \text{ratio}(d, f, c) = \text{ratio}(e, b, f)$  **THREE TANGENT THEOREM**

## de CASTELJAU ALGORITHM

generalisation of parabola construction to generate a polynomial space curve of arbitrary degree  $n$

Given:  $b_0, b_1, \dots, b_n \in \mathbb{E}^3$  and  $t \in \mathbb{R}$

set

$$b_i^r(t) = (1-t)b_i^{r-1}(t) + t b_{i+1}^{r-1}(t) \quad \begin{cases} r=1, \dots, n \\ i=0, \dots, n-r \end{cases}$$

$b_i^0(t) = b_i$  then  $b_0^n(t)$  is the point with parameter value  $t$  on the BEZIER curve  $b^n$

- The polygon  $P$  formed by  $b_0, \dots, b_n$  is called the Bezier polygon of the curve  $b^n$
- The polygon vertices  $b_i$  are called Control points - Bezier Points
- Notation  $b^n(t) = \mathcal{B}[b_0, \dots, b_n; t] = \mathcal{B}[P; t]$   
shorter  $b^n = \mathcal{B}[b_0, \dots, b_n] = \mathcal{B}P$



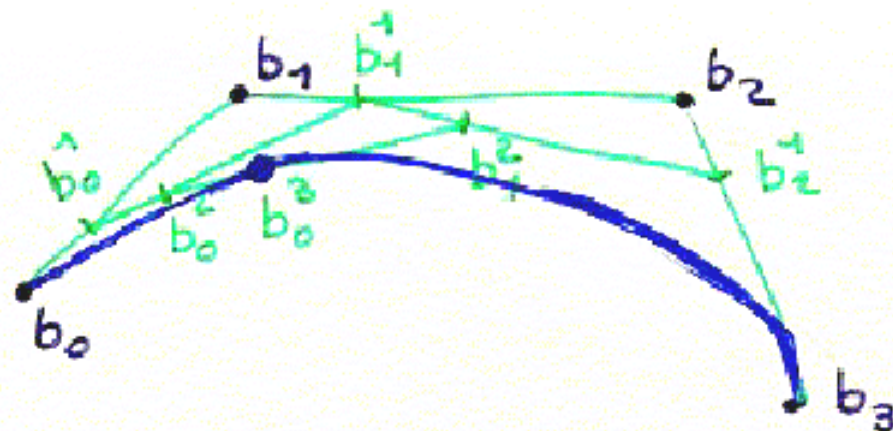
$B$ : the (Linear) Operator that associates the Bézier curve with its control polygon

$B[b_0, \dots, b_n]$  is the Bernstein-Bézier approximation to the control polygon

-  $b_i^k(t)$  intermediate coefficients  $\Rightarrow$  triangular array  $\Rightarrow$  Casteljau scheme

$b_0$   
 $b_1 \quad b_0^1$   
 $b_2 \quad b_1^1 \quad b_0^2$   
 $b_2 \quad b_1^1 \quad b_0^2 \quad b_1^3$

$\Rightarrow$  this also suggest the use of a 2D array for code writing space efficiency??



cubic Case

$n=3$

$t=1/4$



# **Lesson IX: BEZIER CURVES BERNSTEIN POLYNOMIALS**

## The BERNSTEIN Form of a Bézier Curve

recursive algorithm  $\rightarrow$  de Casteljau

Non-recursive Formula  $\rightarrow$  Bernstein Polynomials

### Bernstein Polynomials

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

recursion  $B_i^n(t) = (1-t) B_i^{n-1}(t) + t B_{i-1}^{n-1}(t)$

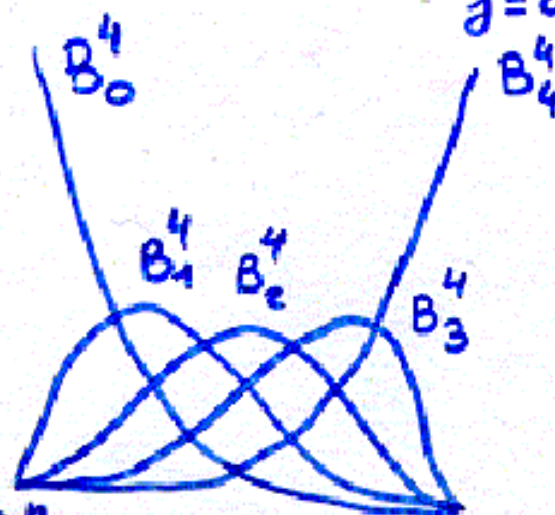
$$B_0^0(t) \equiv 1 \quad ; \quad B_j^n(t) \equiv 0 \text{ for } j \notin \{0, \dots, n\}$$

Proof  $\left[ \begin{array}{l} B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i} \quad \text{with } \binom{n}{i} = \binom{n-1}{i} + \binom{n-1}{i-1} \\ \quad = \binom{n-1}{i} t^i (1-t)^{n-i} + \binom{n-1}{i-1} t^i (1-t)^{n-i} \\ \quad = (1-t) B_i^{n-1}(t) + t B_{i-1}^{n-1}(t) \end{array} \right.$

Bernstein Polynomials form a partition of Unity

$$\sum_{j=0}^n B_j^n(t) \equiv 1 \quad \text{with} \quad (x+a)^n = \sum_{j=0}^n \binom{n}{j} a^j x^{n-j}$$

Proof  $\left[ 1 = (t + (1-t))^n = \sum_{j=0}^n \binom{n}{j} t^j (1-t)^{n-j} = \sum_{j=0}^n B_j^n(t) \right]$



eg Bernstein Polynomials  
Quartic Case

the  $B_i^n$  are nonnegative over the interval  $[0, 1]$



The intermediate de Casteljau points  $b_i^r$  can be expressed in terms of Bernstein polynomials of degree  $r$ :

$$b_i^r(t) = \sum_{j=0}^r b_{i+j} B_j^r(t) \quad \begin{array}{l} r \in \{0, n\} \\ i \in \{0, n-r\} \end{array}$$

This equation shows exactly how the intermediate points  $b_i^r$  depends on the given Bézier points  $b_i$

$r=n \Rightarrow$  The corresponding deCasteljau point is the point on the curve and is given by

$$b^n(t) = \sum_{j=0}^n b_j B_j^n(t)$$

The recursive definition of the intermediate de Casteljau points is

$$\begin{aligned} b_i^n(t) &= (1-t) b_i^{n-1}(t) + t b_{i+1}^{n-1}(t) \quad \text{see before} \\ &= (1-t) \sum_{j=i}^{i+n-1} b_j B_{j-i}^{n-1}(t) \\ &\quad + t \sum_{j=i+1}^{i+n} b_j B_{j-i-1}^{n-1}(t) \end{aligned}$$

$$B_j^n(t) \equiv 0 \text{ for } j \notin \{0, 1, \dots, n\}$$

$$\begin{aligned} b_i^n(t) &= (1-t) \sum_{j=i}^{i+n} b_j B_{j-i}^{n-1}(t) + t \sum_{j=i}^{i+n} b_j B_{j-i-1}^{n-1}(t) \\ &= \sum_{j=i}^{i+n} b_j [(1-t) B_{j-i}^{n-1}(t) + t B_{j-i-1}^{n-1}(t)] \end{aligned}$$

## Some Properties of Bezier Curves

the Costello algorithm allows us to infer several important Properties of Bezier Curves through the geometry underlying the algorithm (they can also be derived analytically).

- **AFFINE INVARIANCE** - linear Interpolations are affine maps!

- Barycentric combinations

$$b^n(t) = \sum_{j=0}^n b_j B_j^n(t) \quad \text{with} \quad \sum_{j=0}^n B_j^n(t) \equiv 1 \quad (*)$$

- Bezier Curves are not projectively invariant!  
e.g. perspective projection

- Invariance under affine parameter transformations

$$t \in [0, 1] \quad \text{or} \quad u \quad a \leq u \leq b \quad u \in [a, b]$$



$$t = \frac{u-a}{b-a}$$

$$\sum_{i=0}^n b_i B_i^n(t) = \sum_{i=0}^n b_i B_i^n\left(\frac{u-a}{b-a}\right)$$

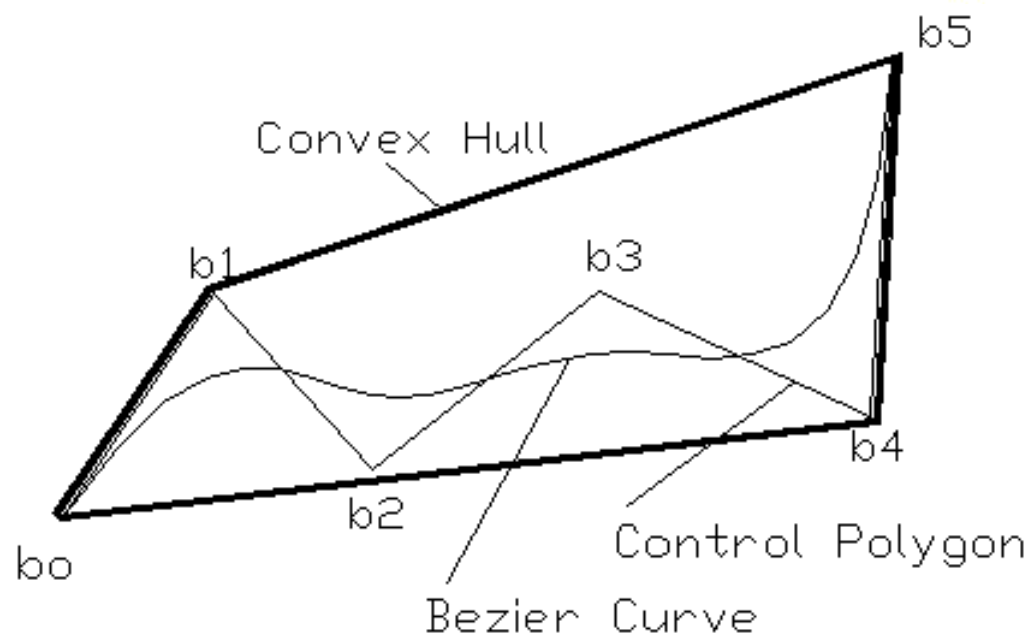


- CONVEX HULL property

for  $t \in [0,1]$ ,  $b^n(t)$  lies in the convex hull of the control Polygon

- \* every intermediate point  $b_i^n$  is obtained as a convex barycentric combination of the previous  $b_j^{n-1}$
- \* another simple consequence is that a planar control Polygon produces a planar curve

since for  $t \in [0,1]$  all  $B_i^n(t) \geq 0$  see definition of convex combination



- Endpoint Interpolation : the Bézier curve passes through  $b_0, b_n$   
 $b^n(0) = b_0, b^n(1) = b_n$

$$\sum_{j=0}^n B_j^n(t) \equiv 1 \quad B_i^n(0) = 1 \text{ if } i=0; B_i^n(1) = 1 \text{ if } i=n$$

- **Symmetry** Labeling the Bézier points  $b_0, b_1, \dots, b_n$  or  $b_n, b_{n-1}, \dots, b_0$   

$$\sum_{j=0}^n b_j B_j^n(t) = \sum_{j=0}^n b_{n-j} B_j^n(1-t) \quad \text{because } B_j^n(t) = B_{n-j}^n(1-t)$$

we say that Bernstein polynomials are symmetric with respect to  $t$  and  $1-t$

- **Invariance under barycentric combinations**

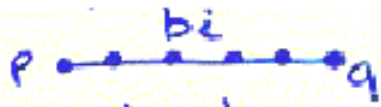
$B$  : operator that produces a Bézier curve from its Bézier polygon - It leaves Barycentric combinations invariant :  
 for  $\alpha + \beta = 1$  we get

$$\sum_{j=0}^n (\alpha b_j + \beta c_j) B_j^n(t) = \alpha \sum_{j=0}^n b_j B_j^n(t) + \beta \sum_{j=0}^n c_j B_j^n(t)$$

Important for the definition of Tensor product surfaces



## - Linear Precision

a useful identity is  $\sum_{j=0}^n \frac{1}{n} B_j^n(t) = t$    $\Rightarrow$  application  
uniform points  $b_i$  on  
straight line

$b_j = (1 - \frac{j}{n})p + \frac{j}{n}q \quad j=0, \dots, n$

$\Rightarrow$  the Bézier curve reproduces the initial straight line

## - Pseudo-Local Control

$B_i^n(t)$  has only one maximum  
at  $t = i/n$  if we move only  $b_i$  curve will change most around  
 $b_i$

## - CAD with Bézier curves

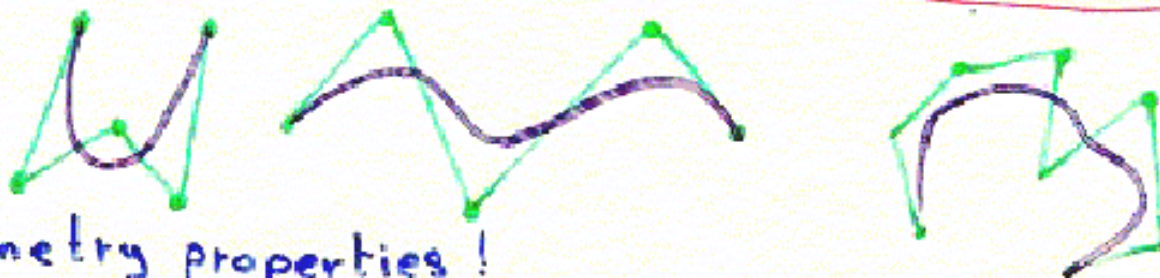
- invariance for affine maps  $\left\{ \begin{array}{l} \text{translation} \\ \text{rotation} \\ \text{scaling} \\ \text{shear} \end{array} \right.$
- endpoint control
- curve "mimics" control polygon

- Not projectively invariant

- degree: (number of control point - 1)

Not  
So good!!

$\Rightarrow$  Rational / cur.  
NURBS



- Symmetry properties!



## **Lesson X: B-SPLINE CURVES**

## Lesson X: B-SPLINE CURVES

### UNIFORM NONRATIONAL B-SPLINES

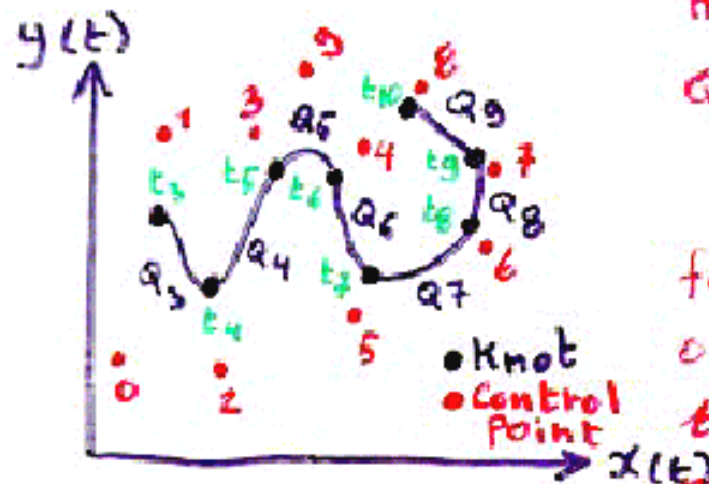
Spline : flexible strip of metal used by draft persons to lay out the surfaces of airplanes, cars & ships  
"Ducks" weights were used to pull the spline in various directions

Metal splines had second-order continuity.

The mathematical equivalent of these strips, the NATURAL CUBIC spline is a  $C^0$ ,  $C^1$ , and  $C^2$  continuous cubic polynomial that interpolates (passes through) the control points. This is 1 more degree of continuity than is inherent in the Hermite & Bezier forms.  $\Rightarrow$  splines are smoother

Polynomial coefficients for natural splines depend on all control points, and their calculation involves inverting an  $n+1$  by  $n+1$  matrix.  $\Rightarrow$  No Local Control  
Computational Complex

## Cubic B-splines



$m+1$  points  $P_0, P_1, \dots, P_m$   $m \geq 3$

$m-2$  cubic curves  $Q_3, Q_4, \dots, Q_m$

$Q_i$  is defined in  $t_i \leq t < t_{i+1}$

$$3 \leq i \leq m$$

for each  $i \geq 4$  there is a join point or KNOT between  $Q_{i-1}$  and  $Q_i$  at  $t = t_i$ , the KNOT value, there are  $m-1$  knots

UNIFORM means that the knots are spaced at equal intervals, we can assume without loss of generality that  $t_3 = 0$  &  $t_{i+1} - t_i = 1$

Each of the  $m-2$  curve segments is defined by four of the  $m+1$  control points

$$Q_i \text{ is defined by } G_{B_{S_i}} = \begin{bmatrix} P_{i-3} \\ P_{i-2} \\ P_{i-1} \\ P_i \end{bmatrix}, \quad 3 \leq i \leq m$$

## Recursive definition of B-splines

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,h}(t) \quad t_{\min} \leq t < t_{\max}$$
$$2 \leq h \leq n+1$$

$B_i$  :  $n+1$  defining polygon vertices

$N_{i,h}(t)$  : normalized B-spline basis functions  
defined by the Cox-de Boor recursion formula

$$N_{i,h}(t) = \frac{(t - x_i) N_{i,h-1}(t)}{x_{i+h-1} - x_i} + \frac{(x_{i+h} - t) N_{i+1,h-1}(t)}{x_{i+h} - x_{i+1}}$$

$\{x_i\}$  knots such  $x_i \leq x_{i+1}$

$0/0 = 0$  is adopted

$P(t)$  is a polynomial spline of order  $k$  (degree  $k-1$ )  
 $P(t)$  and its derivatives of order  $1, 2, \dots, k-2$  are all continuous over the entire curve

$$N_{i,k} \geq 0 ; \sum_{i=1}^{n+1} N_{i,k}(t) \equiv 1$$

$\Rightarrow$  convex hull property

Local control

move a control point  
 it will only influence  
 $k/2$  spans on both sides



# Lesson X: B-SPLINE CURVES

## B-SPLINE PROPERTIES

### SUMMARY B-Spline properties

B-spline : a B-spline curve is defined as a polynomial function of order  $k$  (degree  $k-1$ ), it satisfies the following conditions :

- The function is a polynomial of degree  $k-1$  on each interval  $x_i \leq t < x_{i+1}$
- Its derivatives of order  $1, 2, \dots, k-2$  are all continuous over the entire curve

Important properties of B-splines :

- The maximum order of the curve is equal to the number of defining polygon vertices
- The curve exhibits the variation diminishing property. Thus the curve does not oscillate about any straight line more often than its defining polygon
- The curve generally follows the shape of the defining polygon



- Any affine transformation can be applied to the curve by applying it to the defining polygon vertices; i.e. the curve is transformed by transforming the defining polygon vertices.

The curve lies within the convex hull of its defining polygon. The convex hull property of B-spline curves is stronger than that for Bézier curves. For a B-spline curve of order  $k$  (degree  $k-1$ ) a point on the curve lies within the convex hull of  $k$  neighboring points. Thus all points on a B-spline curve must lie within the union of all such convex hulls formed by taking  $k$  successive defining polygon vertices

example:  $k=2$  convex hull is control polygon  
 $\Rightarrow$  so B-spline is piecewise linear, it is the control polygon

example: •  $k=2$  convex hull is control polygon  
 $\Rightarrow$  So B-spline is piecewise linear, it is the control polygon

- if  $L$  colinear polygon vertices occur in a non colinear defining polygon, then straight portions of the defining curve start and end at least  $k-2$  spans from the beginning and end of the series of colinear polygon vertices

If at least  $k-1$  coincident defining polygon vertices occur i.e.  $d_i = d_{i+1} = d_{i+k-2}$ , then the convex hull of  $d_i$  to  $d_{i+k-2}$  is the vertex itself. Hence the resulting B-spline must pass through vertex  $d_i$ . Further, since a B-spline curve is everywhere  $C^{k-2}$  continuous, it is  $C^{k-2}$  continuous at  $d_i$ .

The choice of knot vector has a significant influence on the B-spline. The only requirement for a knot vector is that it satisfy the relation  $x_i \leq x_{i+1}$ ; i.e., it is a monotonically increasing series of real numbers. Fundamentally three types of knot vector are used: UNIFORM, OPEN UNIFORM and NONUNIFORM.

UNIFORM: knot values are evenly spaced

[0 1 2 3 4]

PEN UNIFORM: has multiplicity of knot values at the ends equal to the order  $k$  of the B-spline function. Internal knot values are evenly spaced

( $n=1$ )  $k=2$  [0 0 1 2 3 4 4]

( $n=2$ )  $k=3$  [0 0 0 1 2 3 4 4 4]

( $n=3$ )  $k=4$  [0 0 0 0 1 2 3 4 4 4 4]

special case single Bézier curve:

[0 0 0 0 1 1 1 1]

NONUNIFORM: may be either unequally spaced and/or multiple internal knot values

[0 0 0 1 1 2 2 2]

[0 1 2 2 3 4]

For multiple knot values within the knot vector a cusp occurs

## Lesson X: B-SPLINE CURVES

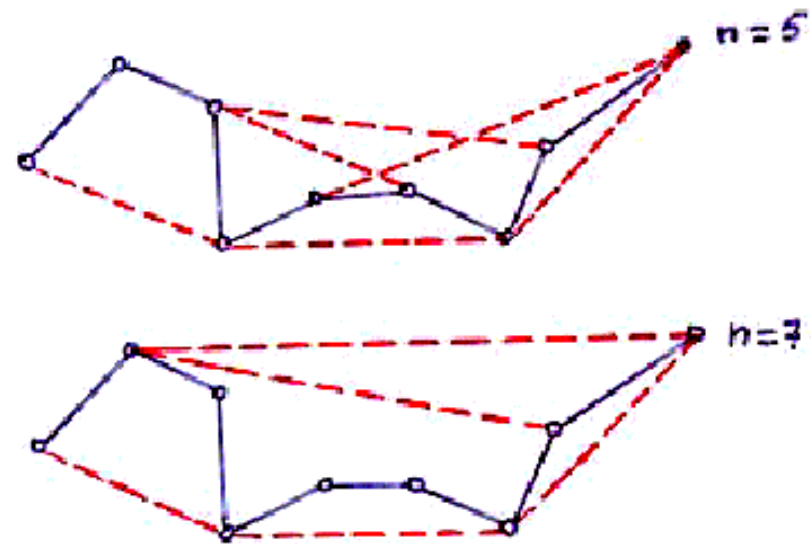
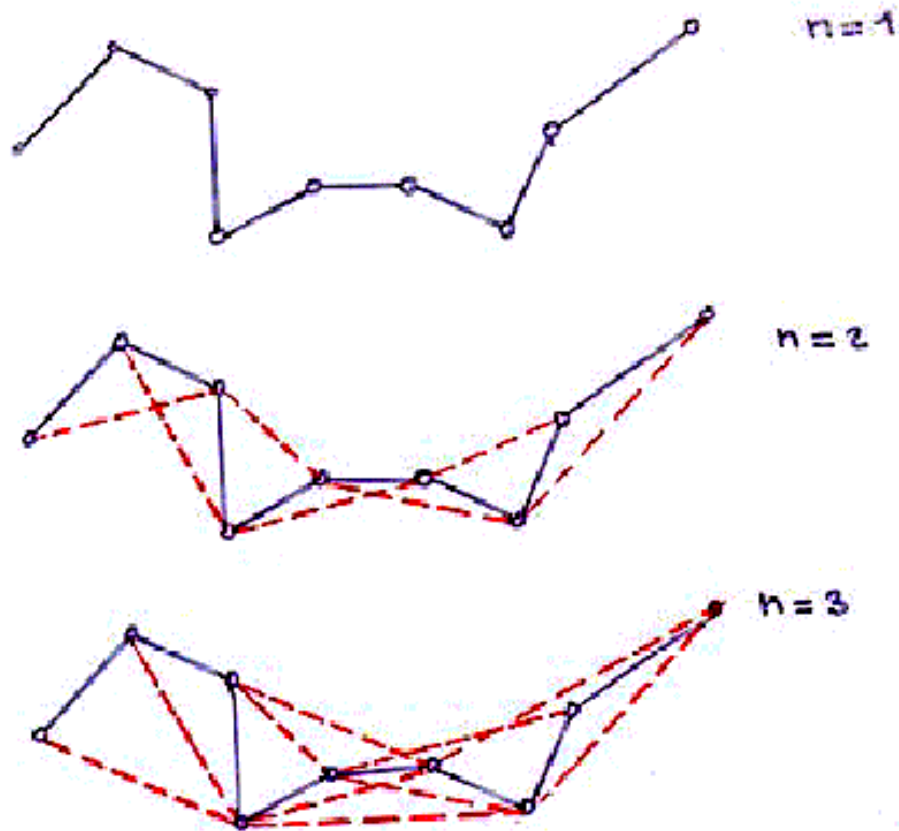
### B-SPLINE CONTROL HANDLES

Different types of control 'handles' are used to influence the shape of the curve:

- changing the type of knot vector
  - changing the order
  - changing the number and position of the defining polygon vertices
  - using multiple polygon vertices
  - using multiple knot values in the knot vector
- multiple knot value introduces a span of zero length. Further multiple interior knot values, in contrast to multiple polygon vertices, reduce the differentiability at  $x_i$  to  $C^{k-m-1}$  where  $m \leq k-1$  is the multiplicity of the interior knot value

# Lesson X: B-SPLINE CURVES

## B-SPLINE CONVEX HULL

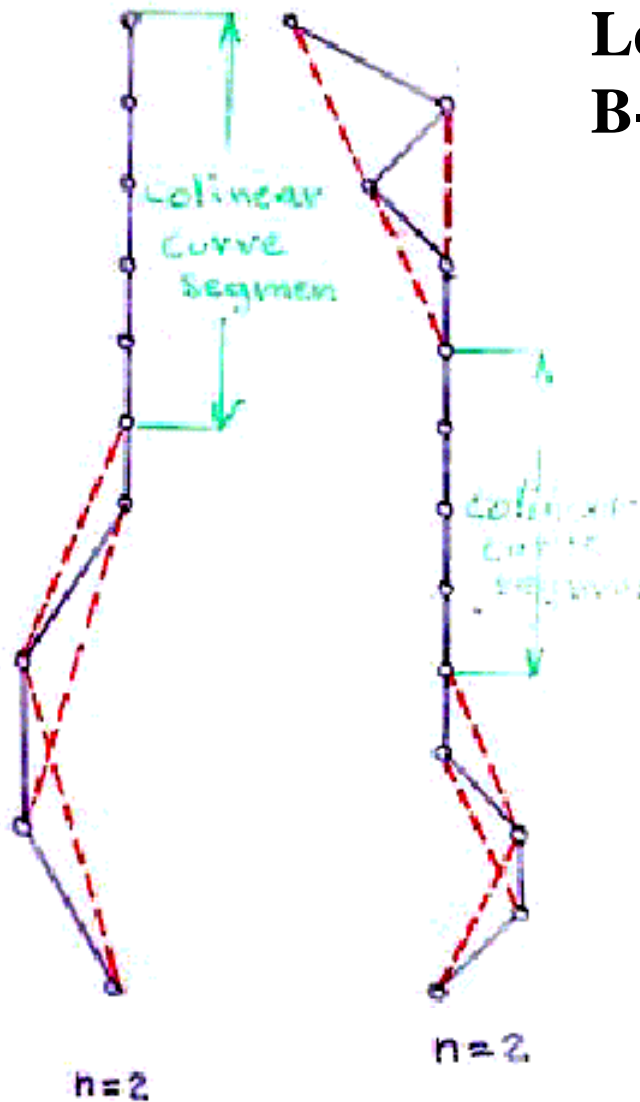


Convex hull properties of B-splines



# Lesson X: B-SPLINE CURVES

## B-SPLINE EMBEDDED LINE

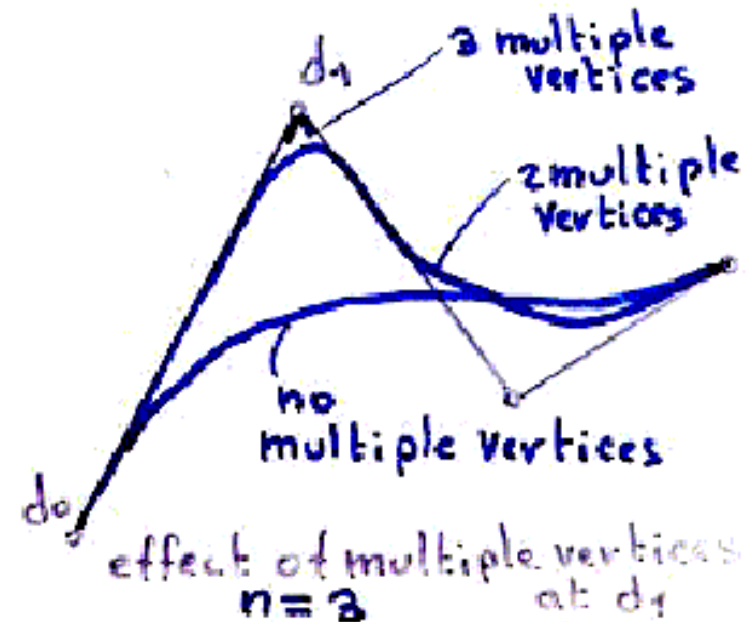
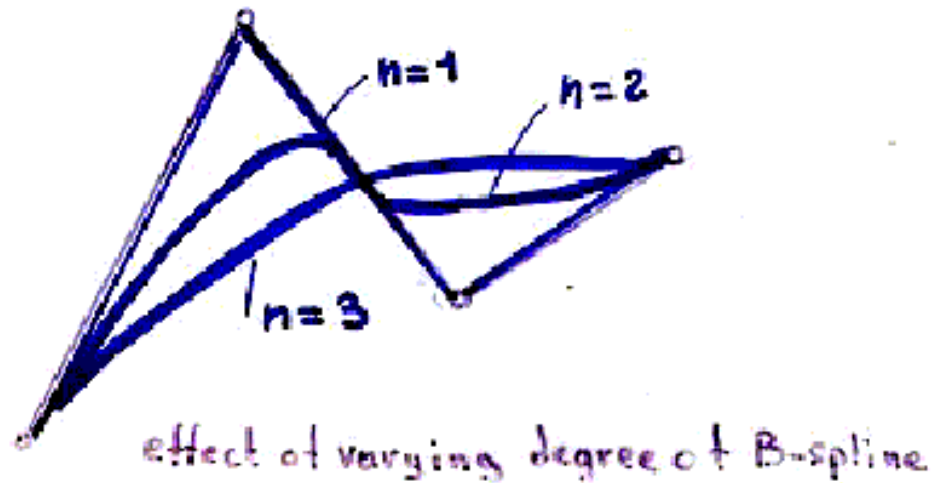


B-spline convex hull  
properties for colinear  
curve segments

# Lesson X: B-SPLINE CURVES

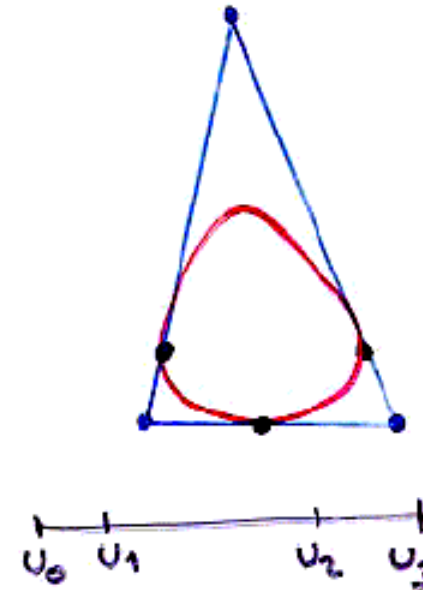
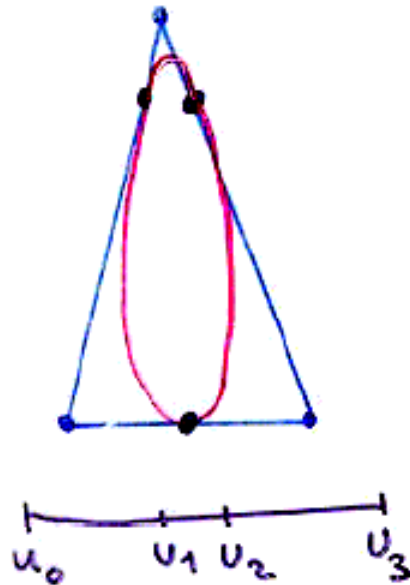
## RELATION DEGREE/SHAPE

### MULTIPLE KNOTS



# Lesson X: B-SPLINE CURVES

## KNOT INTERVAL INFLUENCE ON SHAPE

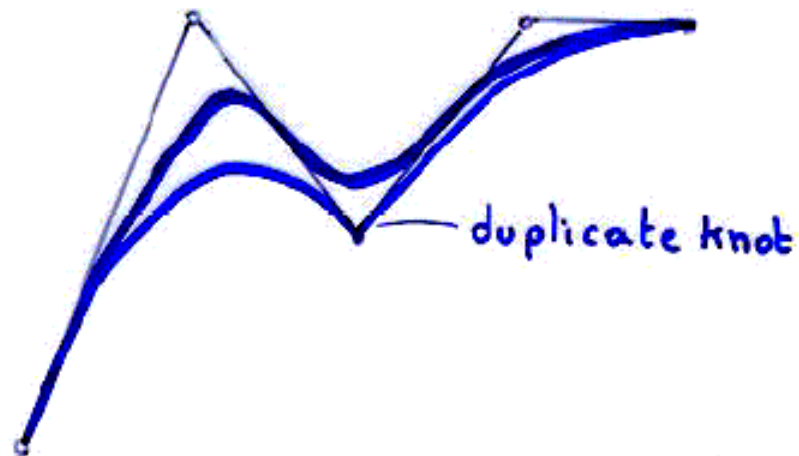


Two quadratic closed  
B-spline curves with  
the same control polygon  
but different knot sequences

# Lesson X: B-SPLINE CURVES

## DUPLICATE KNOTS

B-splines - Duplicate knots - Discontinuity



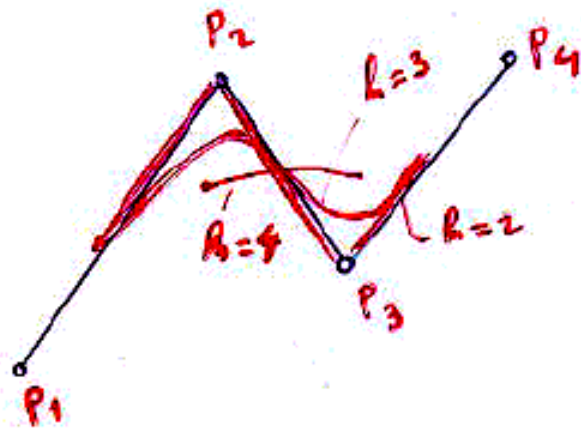
Non Uniform B-spline curves  $n=2$ .

a) knot vector  $[0\ 0\ 0\ 1\ 2\ 3\ 3\ 3]$

b) knot vector  $[0\ 0\ 0\ 1\ 2\ 2\ 3\ 3\ 3]$

## Lesson X: B-SPLINE CURVES

### UNIFORM B-SPLINES – NOT USED IN CAD – BECAUSE NO ENDPOINT CONTROL

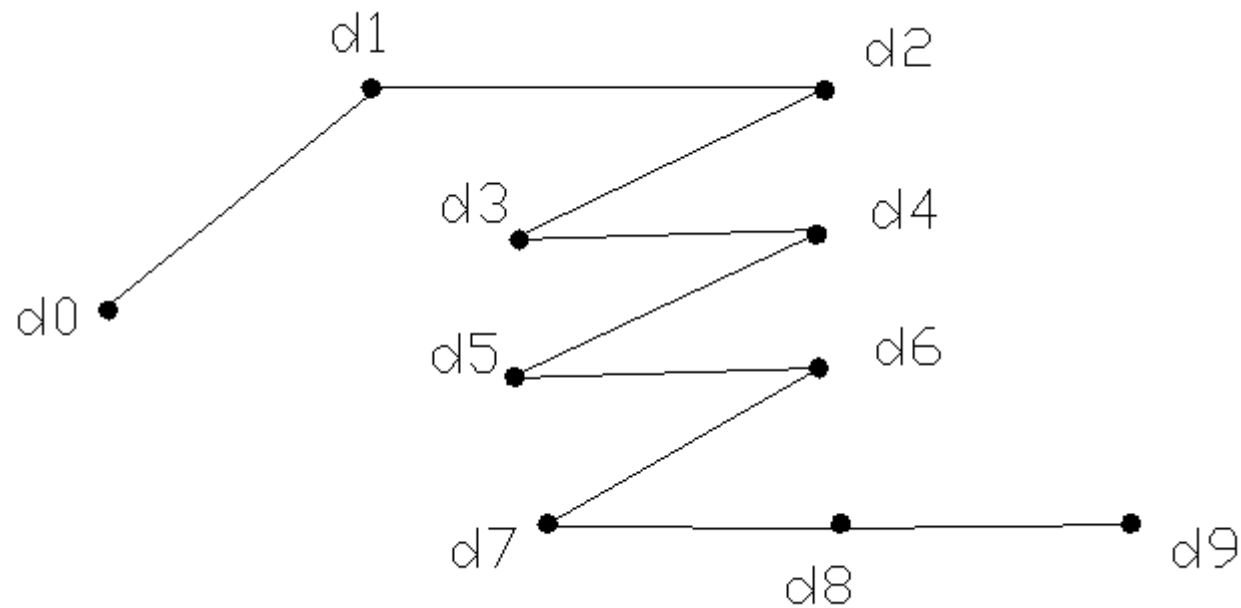


Uniform B-spline  
knotvector [      ]

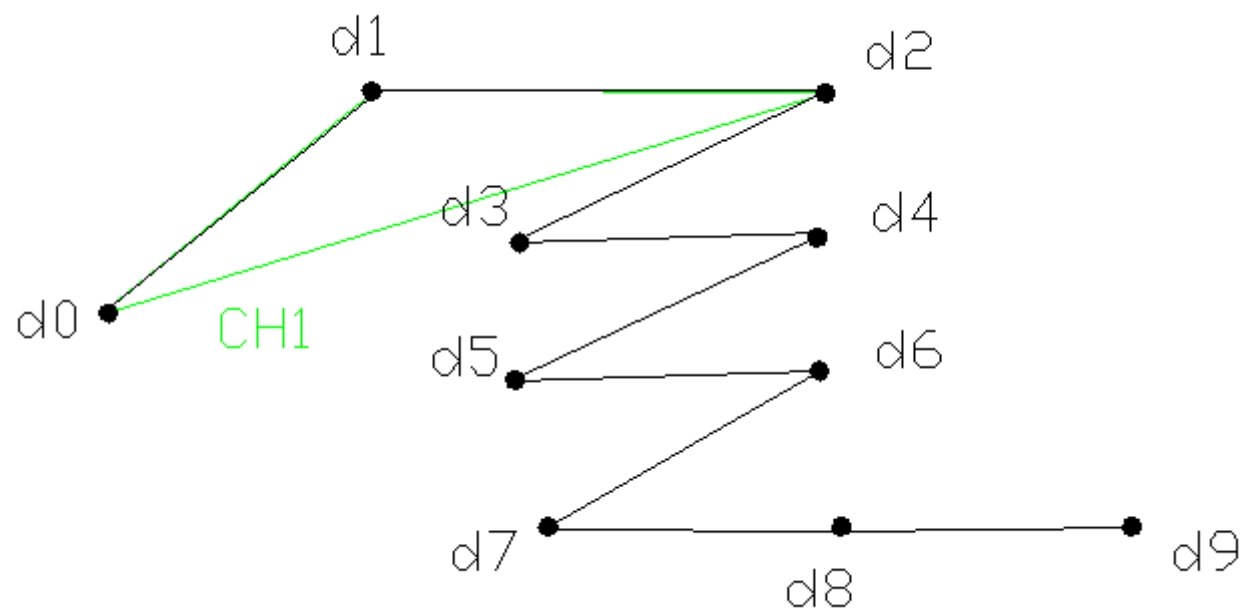


# **B-Spline Properties:**

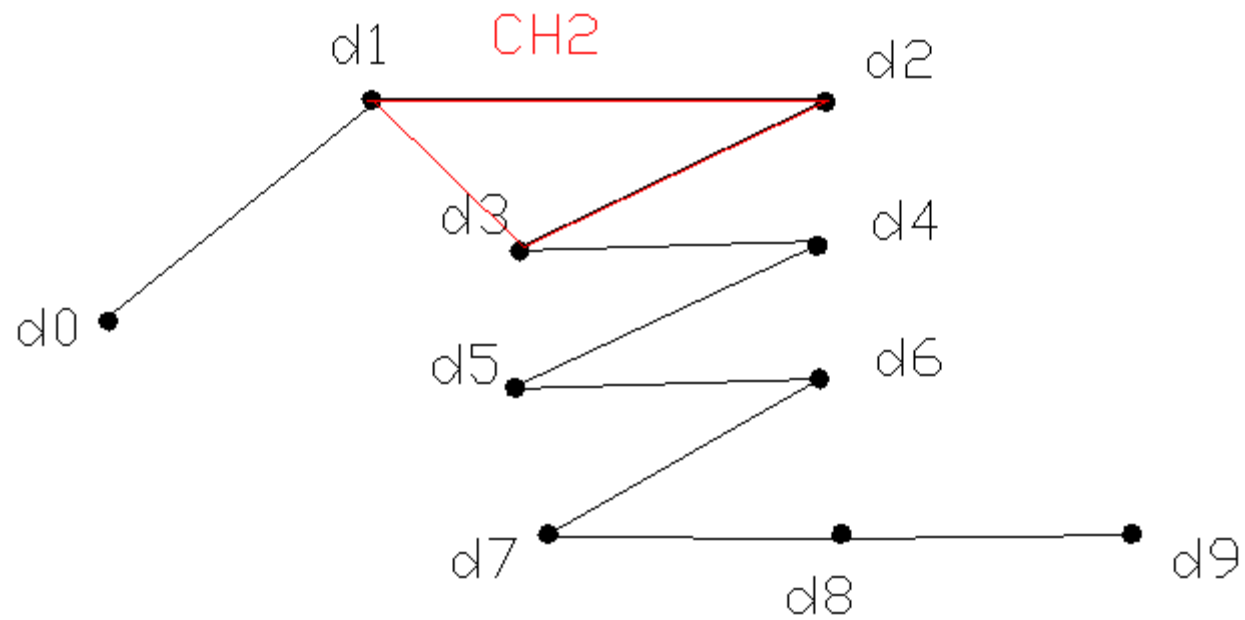
- Convex Hull**
- Local Control**
- Embedded Straight Line**
- Conic Section**
- Degree**



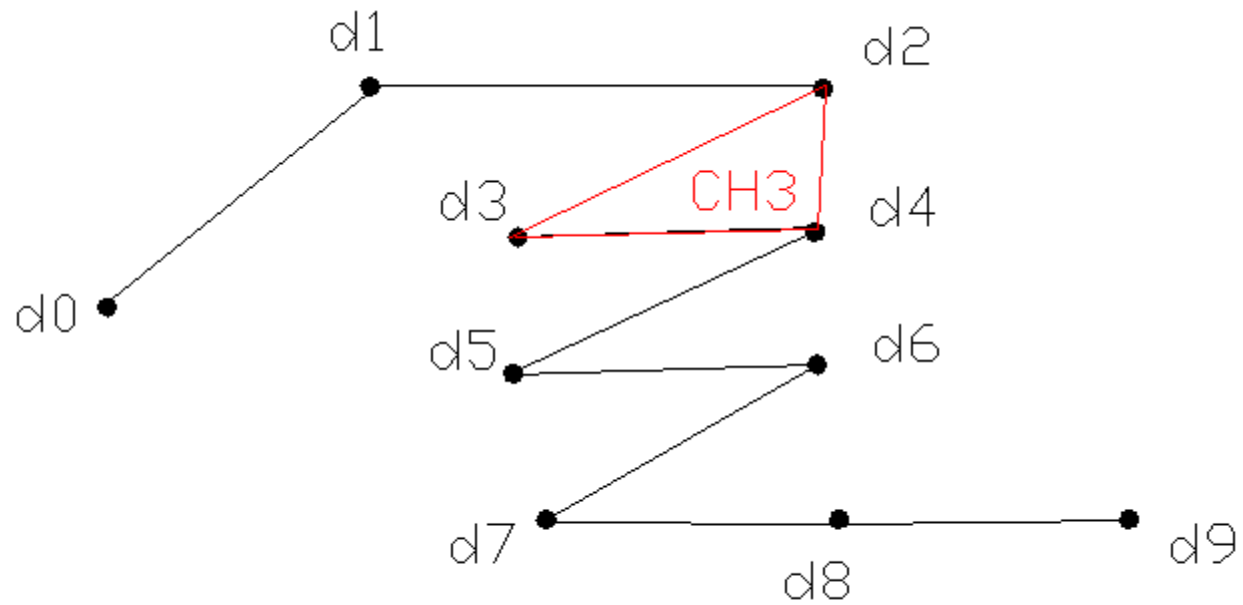
## B-spline Control Polygon



**Convex Hull of B-spline : Partial Convex Hull 1**

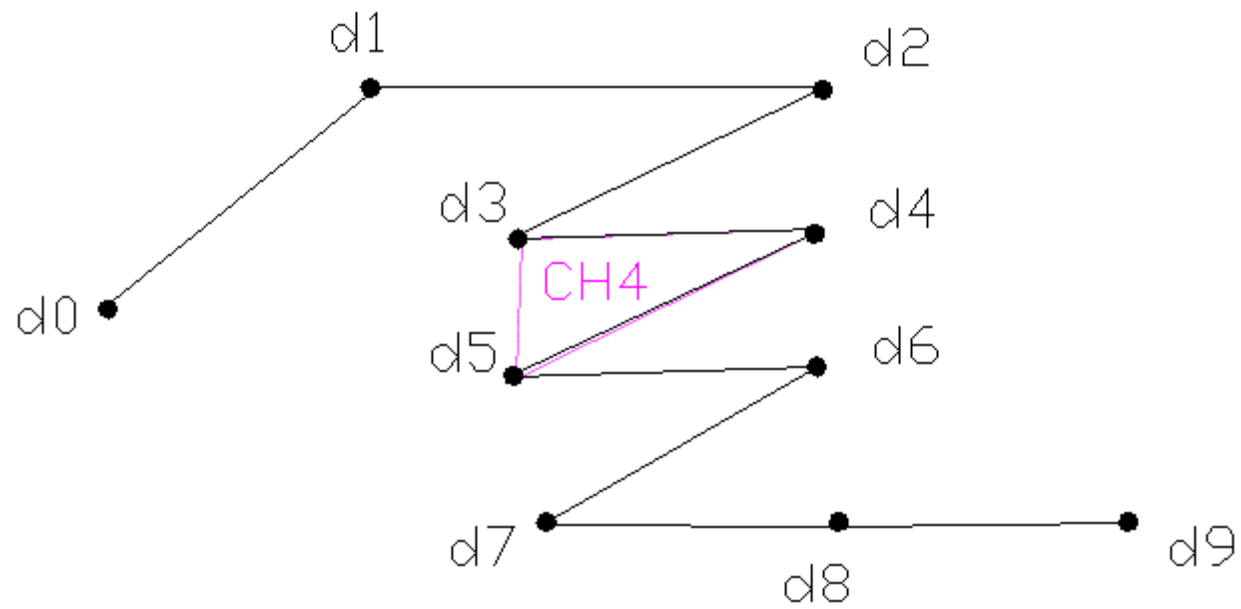


**Convex Hull of B-spline : Partial Convex Hull 2**

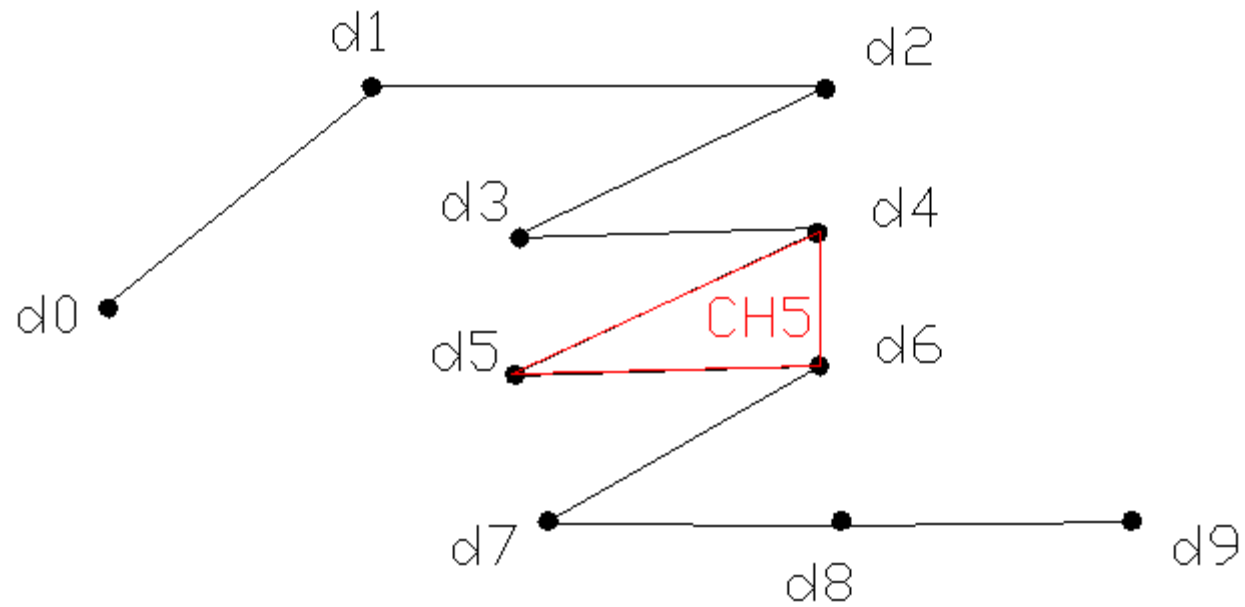


## Convex Hull of B-spline : Partial Convex Hull 3

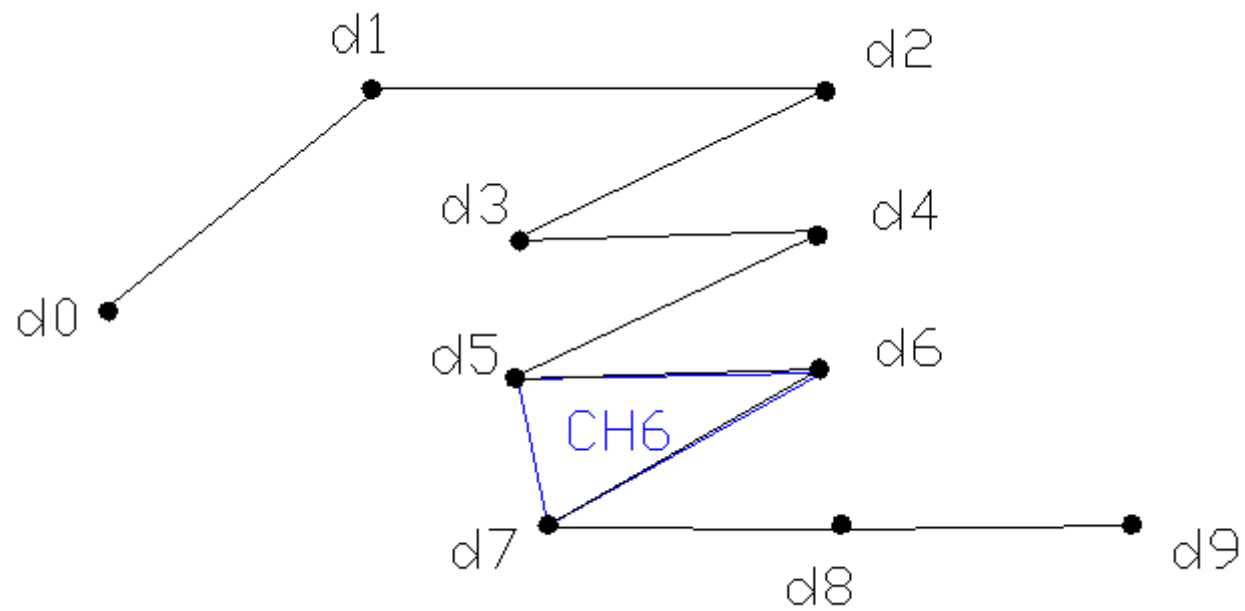




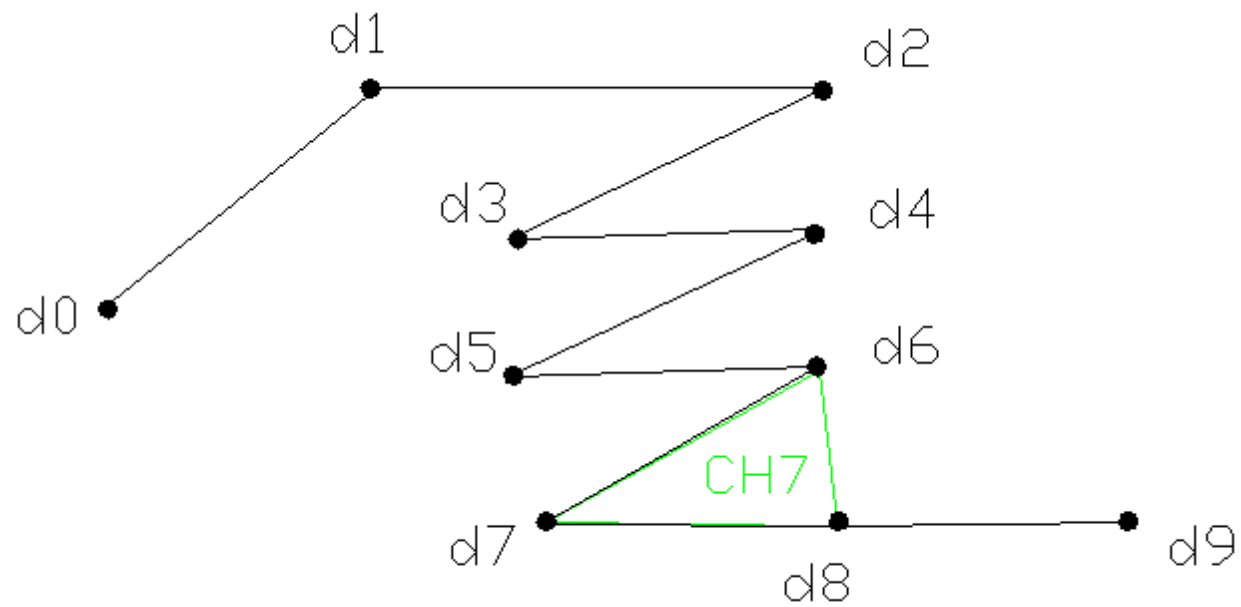
**Convex Hull of B-spline : Partial Convex Hull 4**



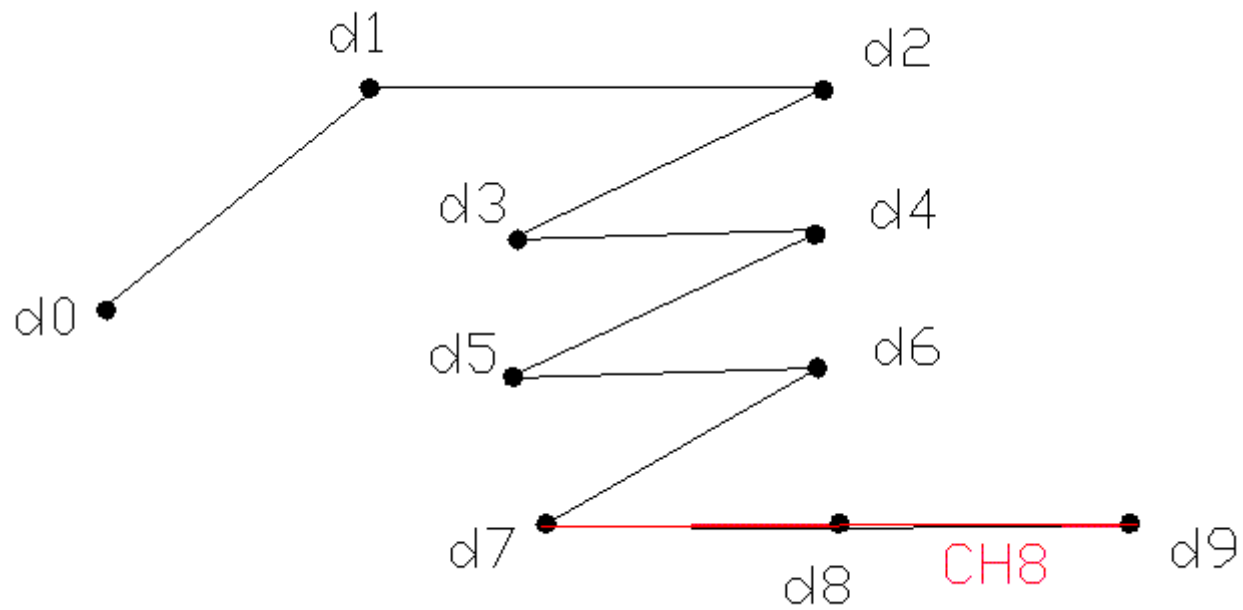
**Convex Hull of B-spline : Partial Convex Hull 5**



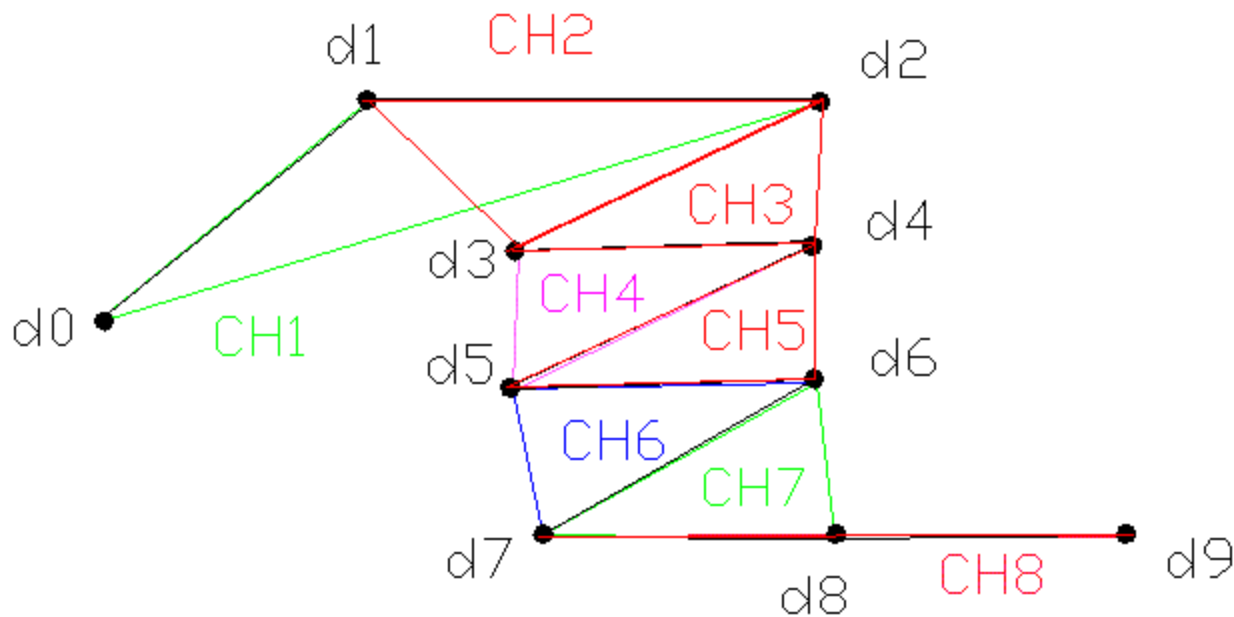
**Convex Hull of B-spline : Partial Convex Hull 6**



**Convex Hull of B-spline : Partial Convex Hull 7**



**Convex Hull of B-spline : Partial Convex Hull 8**



**Convex Hull of B-spline = Union of Partial Convex Hulls**

**KNOT VECTOR: (8 Pieces of Curve because we have 8 Convex Hulls)**

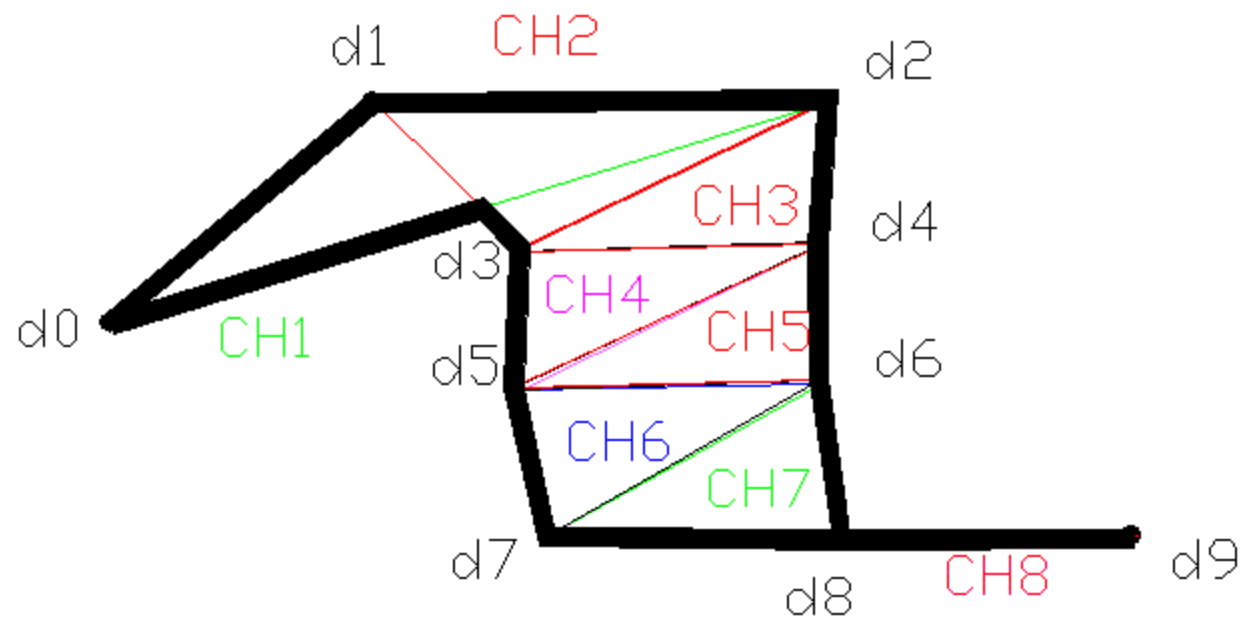
**$\{U_0 U_0 U_0 U_1 U_2 U_3 U_4 U_5 U_6 U_7 U_8 U_8 U_8\}$**

**Length of intervals for knotvector:**

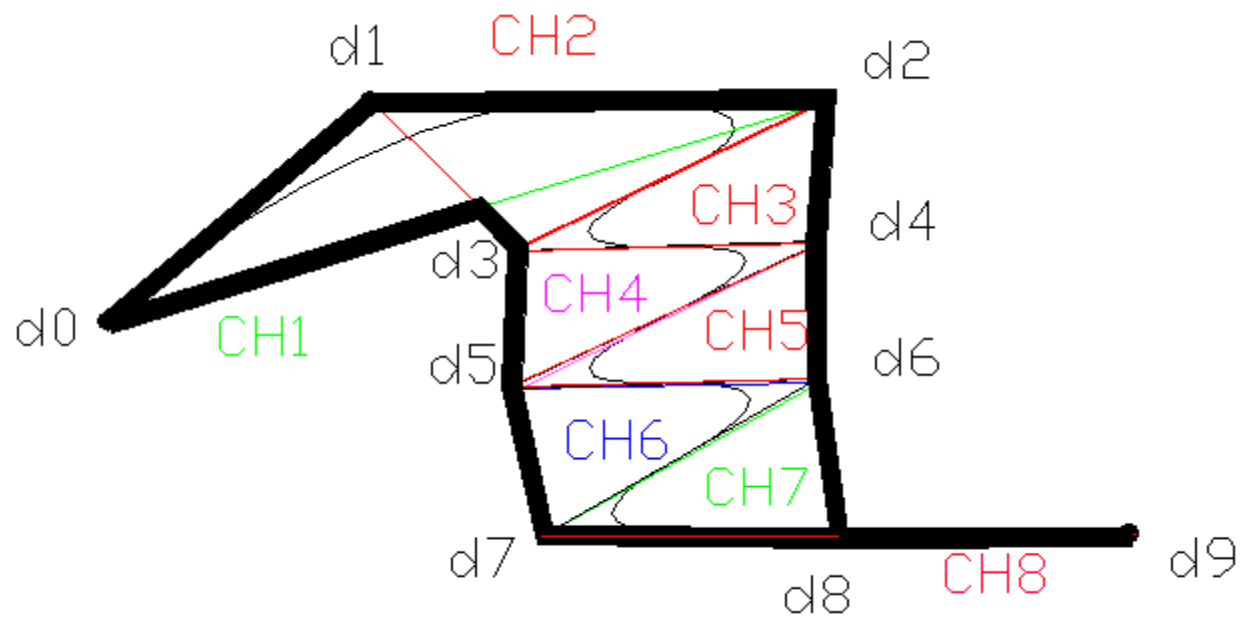
**$U_0U_1 = d_0d_1d_2; U_1U_2 = d_1d_2d_3; U_2U_3 = d_2d_3d_4; U_3U_4 = d_3d_4d_5;$**

**$U_5U_6 = d_5d_6d_7; U_7U_8 = d_7d_8d_9$**

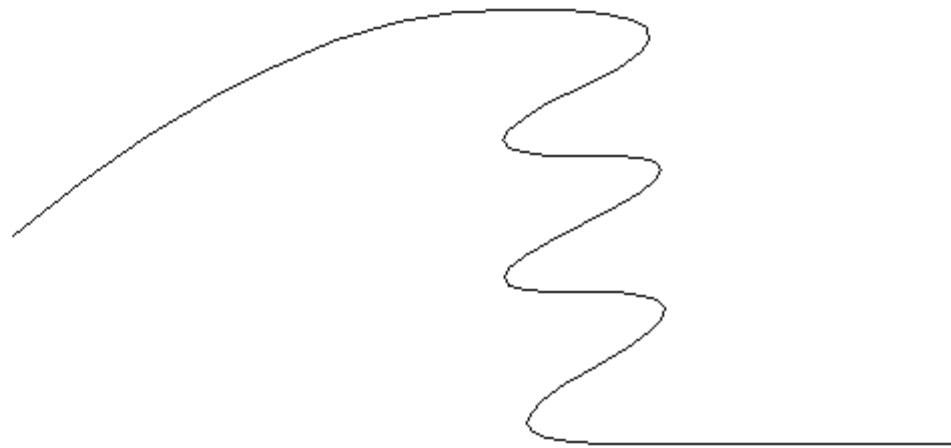




**Convex Hull of B-spline = Union of Partial Convex Hulls**



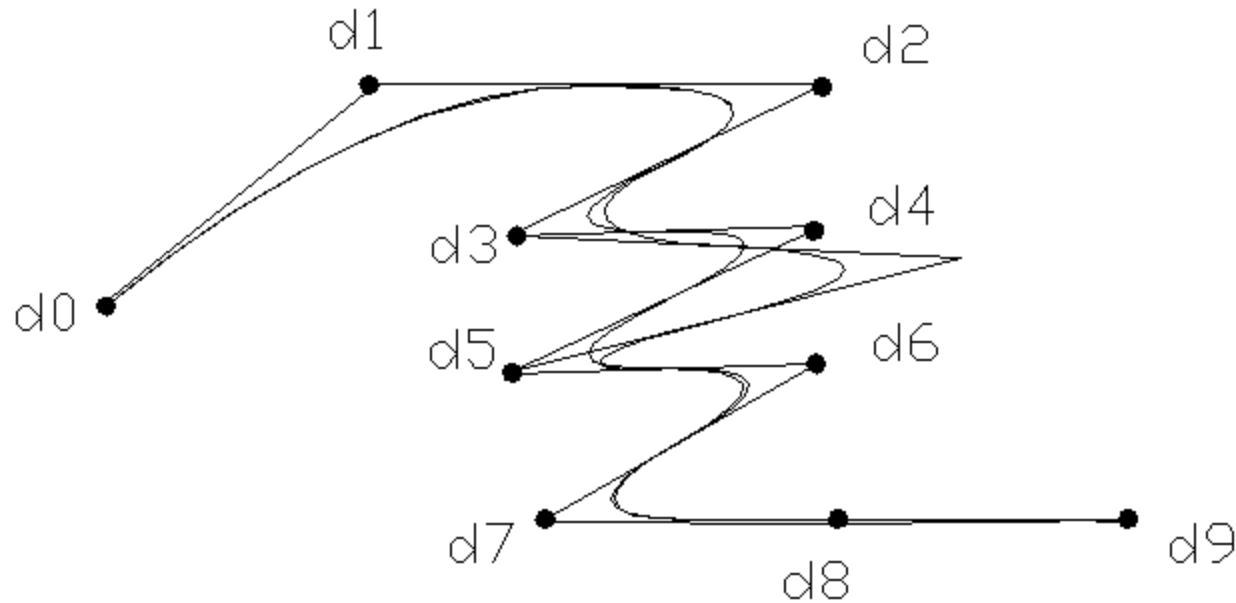
**B-Splines is Inside Convex Hull**



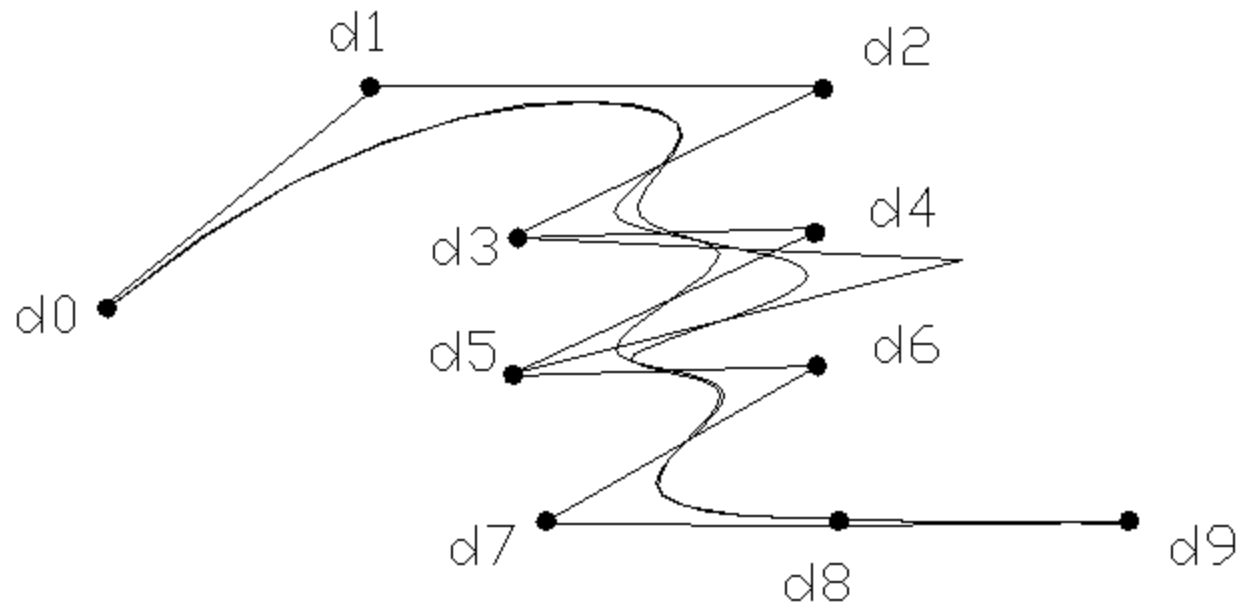
**B-Spline Smoothly Embeds Straight Line Segment**

# Local Control of B-Spline:

- better the lower the degree




**B-Spline of Degree  $n = 2$**




## **B-Spline of Degree $n = 3$**

**-Less local control**

**-To embed straight line will require more control points in line**



**$n = 3$**



**$n = 2$**



B-spline curve

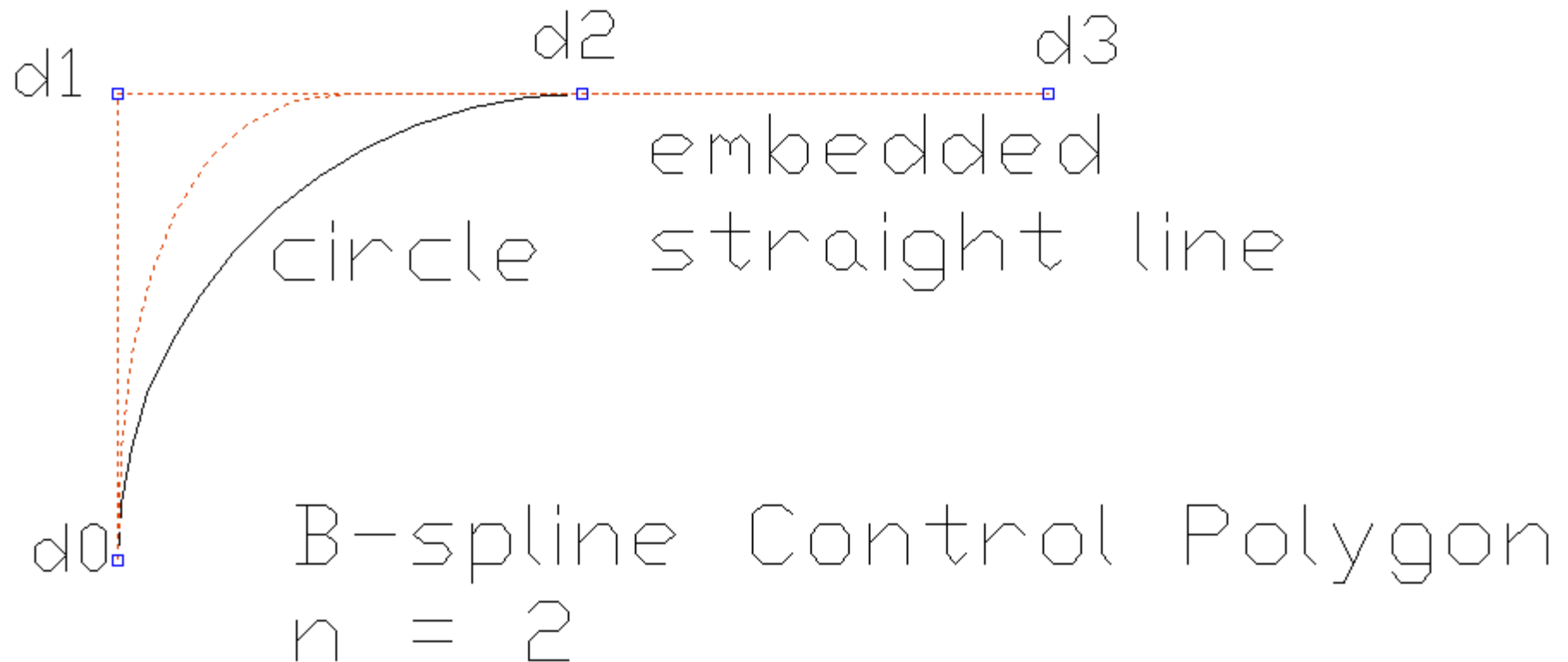
- Embedded Straight Line

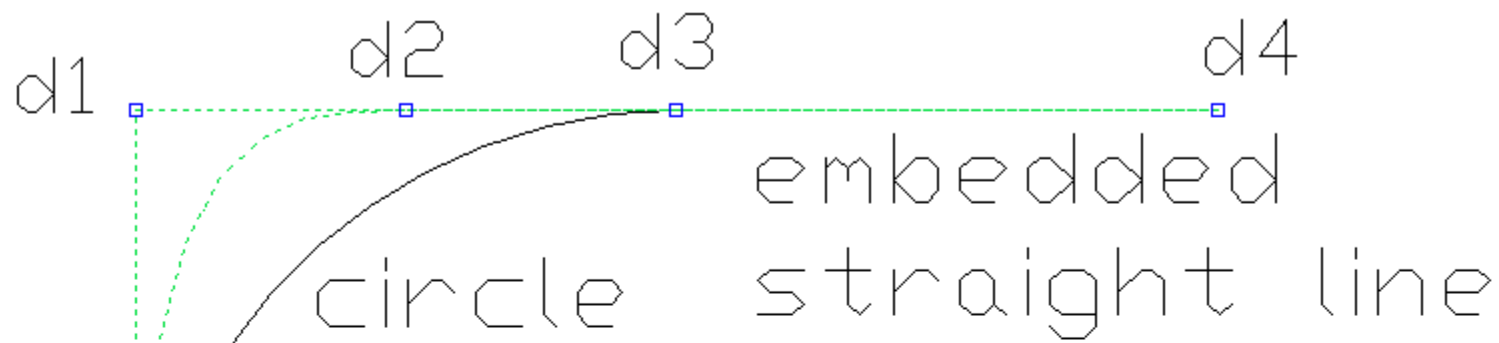
- Circle Approximation

NURBS

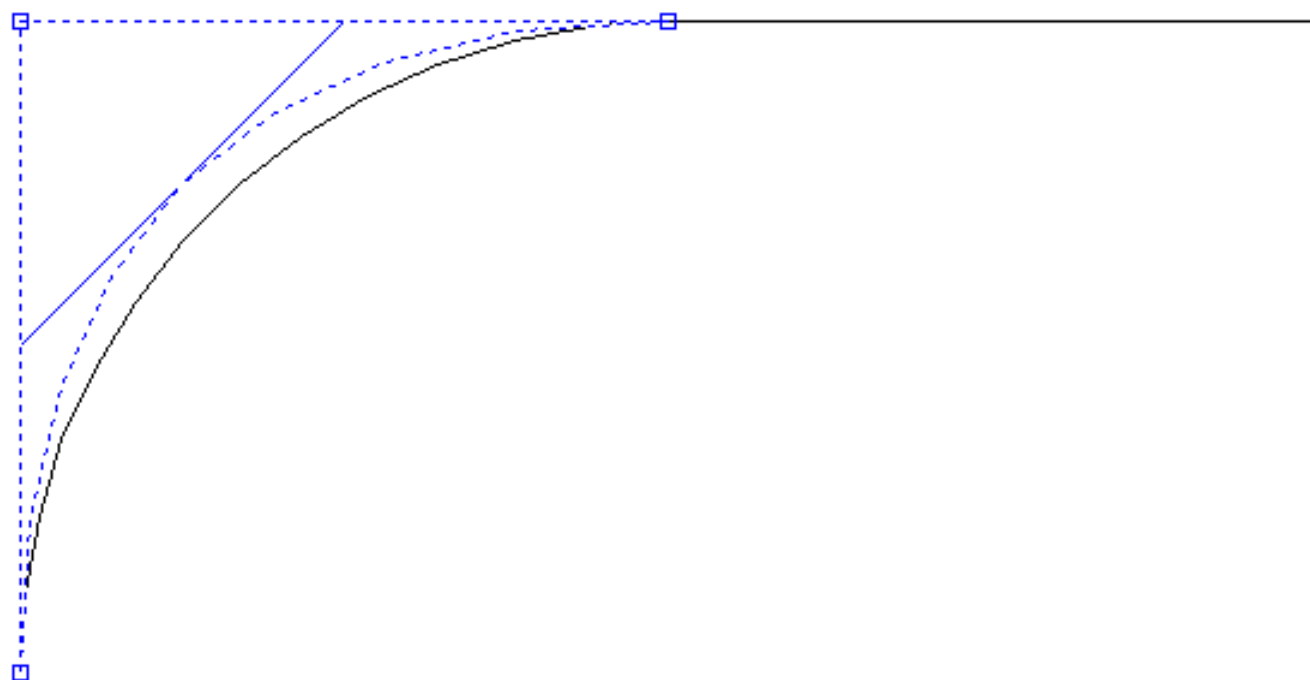
## Lesson X: B-SPLINE CURVES

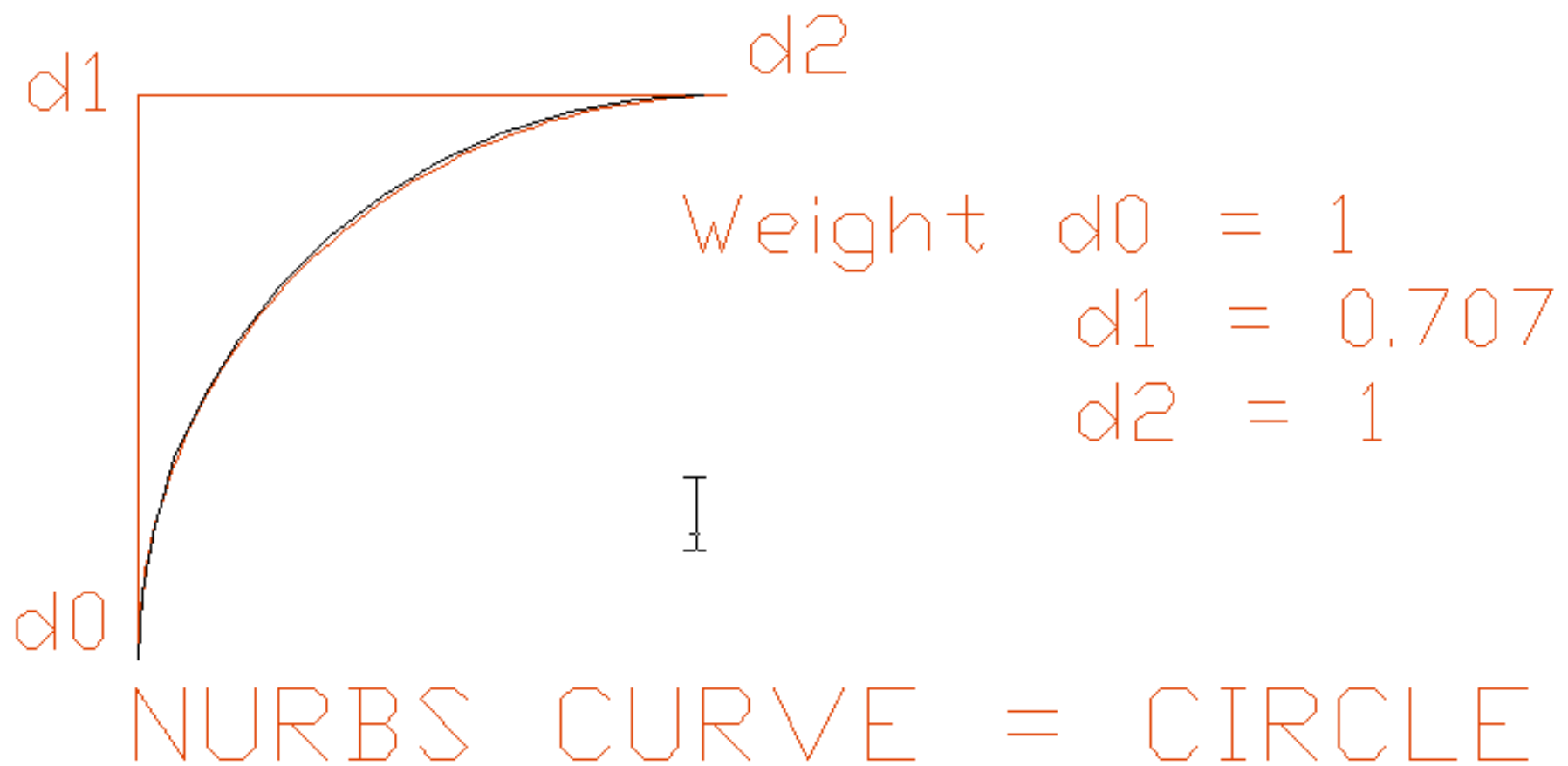
### B-SPLINE PROPERTIES – B-SPLINE CANNOT MODEL CONIC SECTIONS





B-spline Control Polygon  
 $n = 3$





## Lesson XI: NURBS

### Non Uniform Rational B-spline

#### Rational Bézier and B-spline Curves

Rational B-spline curves (NURBS) are becoming the standard surface description in the field of CAD and graphics. (Non Uniform Rational B-splines)

#### Rational Bézier curves

A rational Bézier curve of degree  $n$  in  $\mathbb{E}^3$  is the projection of an  $n^{\text{th}}$  degree Bézier curve in  $\mathbb{E}^4$  into the hyperplane  $w=1$ . We may view this 4D hyperplane as a copy of  $\mathbb{E}^3$ . A point in  $\mathbb{E}^4$  is given by its coordinates  $[x \ y \ z \ w]^T$ .



It can be shown that an  $n$ th degree rational Bézier curve is given by:

$$x(t) = \frac{w_0 b_0 B_0^n(t) + \dots + w_n b_n B_n^n(t)}{w_0 B_0^n(t) + \dots + w_n B_n^n(t)} ; x(t), b_i \in E^3$$

$w_i$  are called weights, the  $b_i$  form the control polygon.

It is the projection of the 4D control polygon  $[w_i b_i w_i]$  of the non rational 4D pre-image of  $x(t)$ .

If all the weights are equal one (or all equal), we obtain the standard non rational Bézier curve; in that case, the denominator is identically equal to one. If some  $w_i$  are negative, singularities may occur; we will therefore only deal with nonnegative  $w_i$ . Rational Bézier curves enjoy all the properties that their non rational counterparts possess; for example, they are affinely invariant.

Rational Bézier curves enjoy all the properties that their nonrational counterparts possess; for example, they are affinely invariant. We can see this from:

$$\alpha(t) = \sum_{i=0}^n b_i \frac{w_i B_i^n(t)}{\sum_{j=0}^n w_j B_j^n(t)}$$

The basis functions are now however:

$$\frac{w_i B_i^n(t)}{\sum_{j=0}^n w_j B_j^n(t)}$$

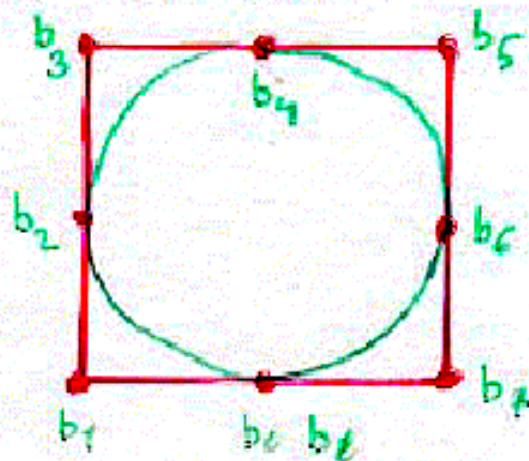
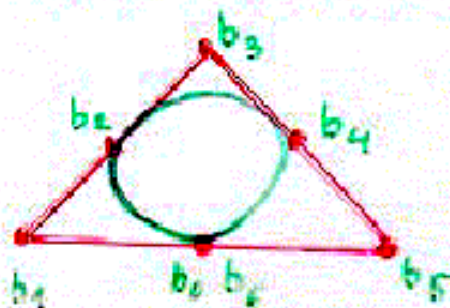
They sum up to one identically, thus asserting affine invariance. If all  $w_i$  are non-negative we have the convex hull property.

We also have symmetry

- invariance under affine parameter transformations
- endpoint interpolation
- variation diminishing property

The  $w_i$  are typically used as shape parameters. If we increase one  $w_i$ , the curve is pulled toward the corresponding  $b_i$

- additional properties
- perspective transform invariant
- conic sections



e.g.  
Circle

## Lesson XII: SURFACES

### Tensor Product Bézier Surfaces

The first person to consider this class of surfaces was de Casteljau (1959-1963). Initially, Bézier patches were only used to approximate a given surface. It took some time until people realized that any B-spline surface can also be written in piecewise Bézier form.

## Bilinear Interpolation

Tensor product Bézier surfaces are based on the concept of BILINEAR INTERPOLATION. While linear interpolation fits the "simplest" curve between two points, bilinear interpolation fits the "simplest" surface between four points.

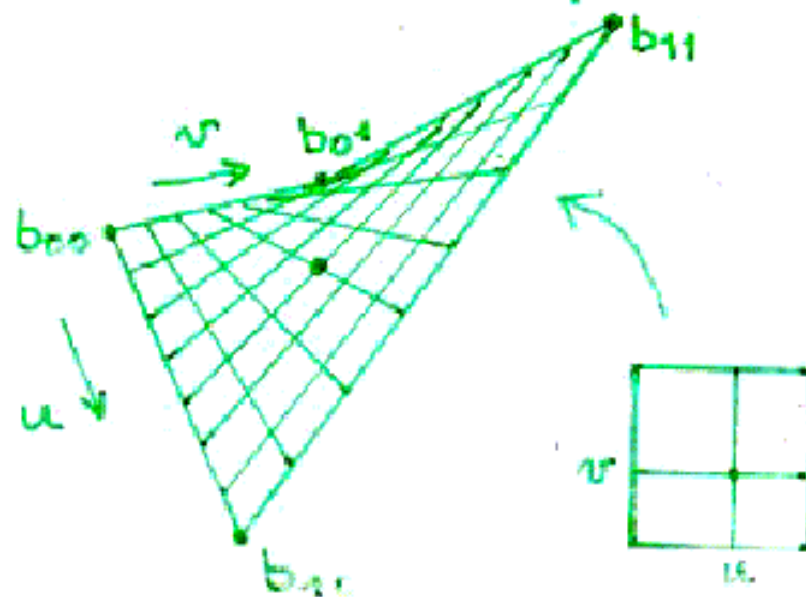
Let  $b_{0,0}, b_{0,1}, b_{1,0}, b_{1,1}$  be four distinct points in  $\mathbb{E}^3$ . The set of all points  $x \in \mathbb{E}^3$  to form:

$$x(u,v) = \sum_{i=0}^1 \sum_{j=0}^1 b_{i,j} B_i^1(u) B_j^1(v)$$

is called a hyperbolic paraboloid through the four  $b_{i,j}$ .

$$x(u,v) = \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} b_{00} & b_{10} \\ b_{10} & b_{11} \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

$x$  is called the bilinear interpolant





The bilinear interpolant can be viewed as a map of the unit square  $0 \leq u, v \leq 1$  in the  $uv$  plane. We say that the unit square is the domain of the interpolant, while the surface  $x$  is its range. A line parallel to one of the axes in the domain corresponds to a curve in the range; it is called an isoparametric curve. Every isoparametric curve of the hyperbolic paraboloid is a straight line, thus hyperbolic paraboloids are ruled surfaces. In particular, the isoparametric line  $u=0$  is mapped onto the straight line through  $b_{0,0}$  and  $b_{0,1}$ .

Instead of evaluating the bilinear interpolant directly we can apply a two stage process:

$$b_{0,0}^{0,1} = (1-v)b_{0,0} + vb_{0,1}$$

$$b_{1,0}^{0,1} = (1-v)b_{1,0} + vb_{1,1}$$

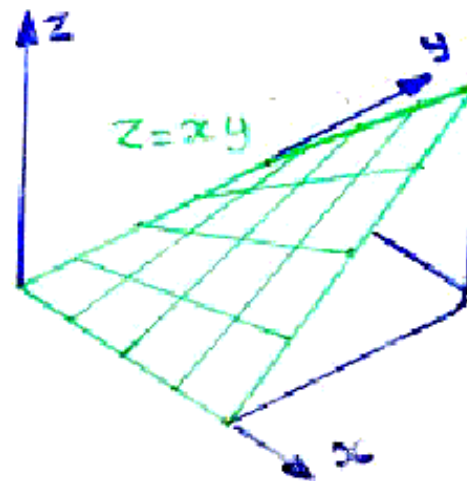
$$b_{0,0}^{0,1} = (1-v)b_{0,0} + vb_{0,1}$$

$$b_{1,0}^{0,1} = (1-v)b_{1,0} + vb_{1,1}$$

$$x(u,v) = b_{0,0}^{1,1}(u,v) = (1-u)b_{0,0}^{0,1} + ub_{1,0}^{0,1}$$

This amounts to computing the coefficients of the isoparametric line  $v = \underline{cte}$  first and then evaluating this isoparametric line at  $u = \underline{cte}$ .  
The term hyperbolic paraboloid comes from analytic geometry.

- consider surface  $z = xy$  this can be considered as the bilinear interpolant to the four points:  $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$
- a plane // to  $x, y$  plane gives as intersection a hyperbola
  - a plane containing the  $z$  axis e.g.  $y = x$  gives a parabola



## The Tensor Product Approach

Intuitive definition of a surface : A surface is the locus of a curve that is moving through space and thereby changing its shape.

Assume that the moving curve is a Bézier curve of constant degree  $m$ . At any time, the moving curve is then determined by a set of control points. Each original control point moves through space on a curve. We assume that this curve is also a Bézier curve, and that the curves on which the control points move are all of the same degree.

$$b^m(u) = \sum_{i=0}^m b_i B_i^m(u)$$

Let each  $b_i$  traverse a Bézier curve of degree  $n$

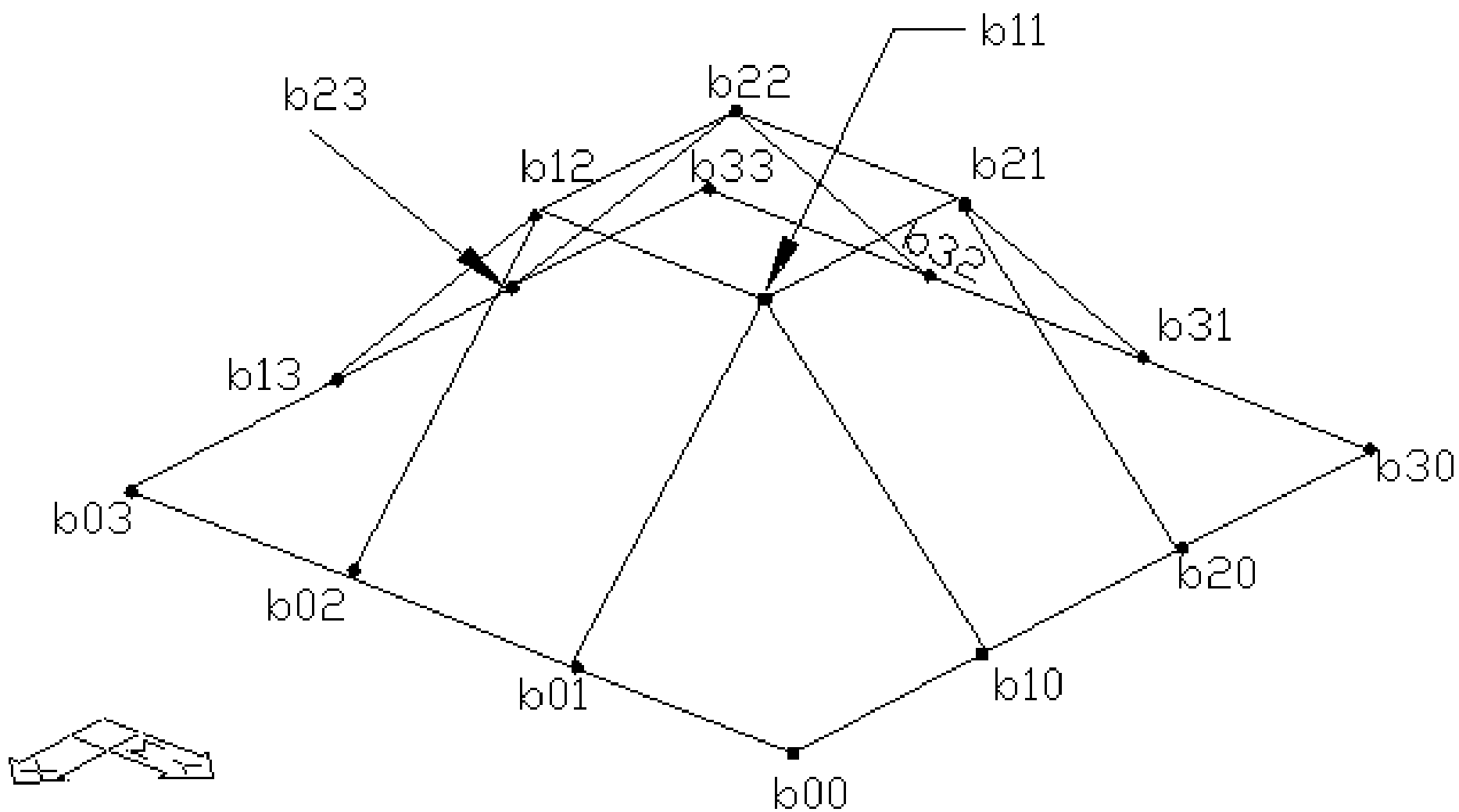
$$b_i = b_i(v) = \sum_{j=0}^n b_{i,j} B_j^n(v)$$

$$b^{m,n}(u,v) = \sum_{i=0}^m \sum_{j=0}^n b_{i,j} B_i^m(u) B_j^n(v)$$

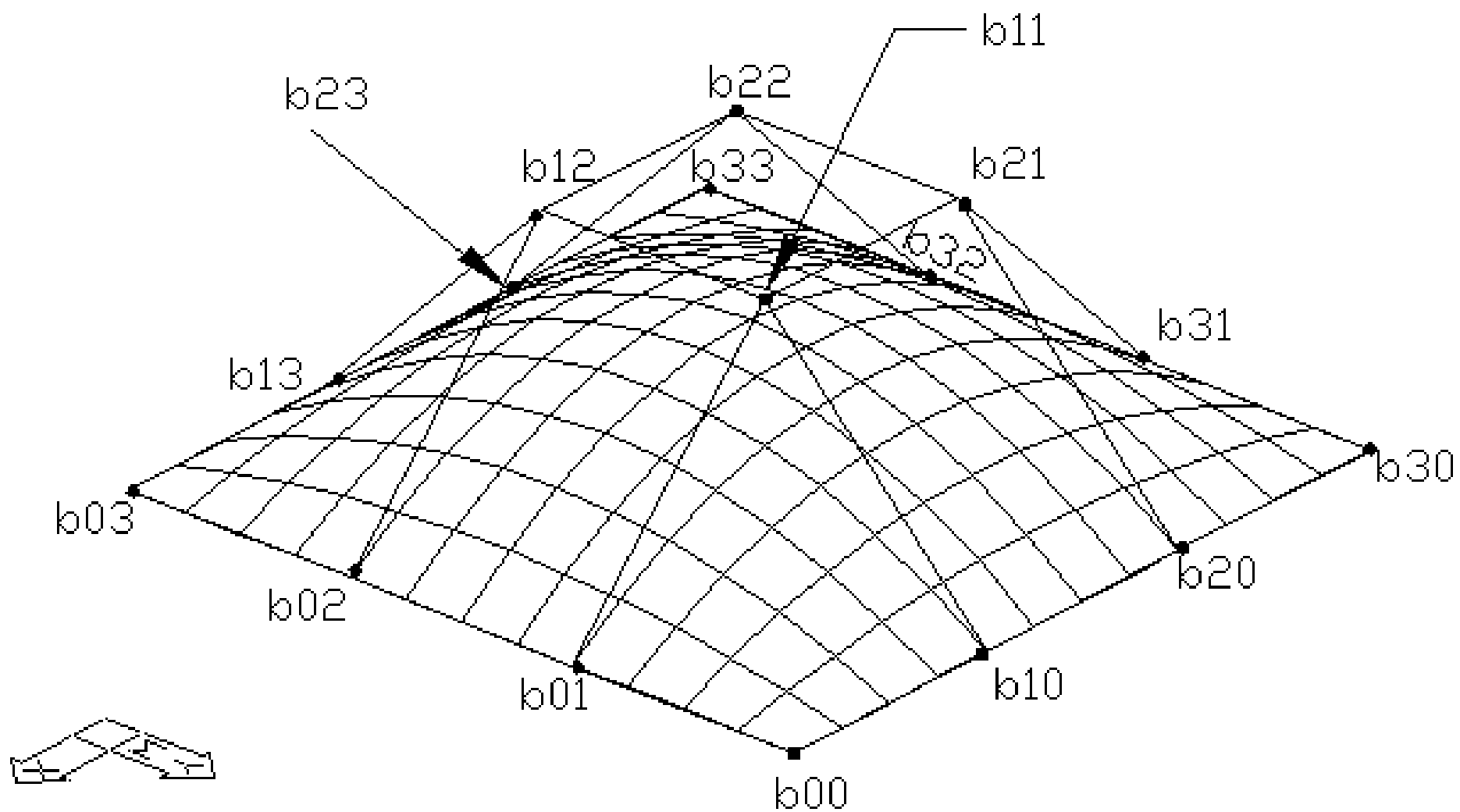
The original curve  $b^m(u)$  now has Bézier points  $b_{i,0}, i=0, \dots, m$

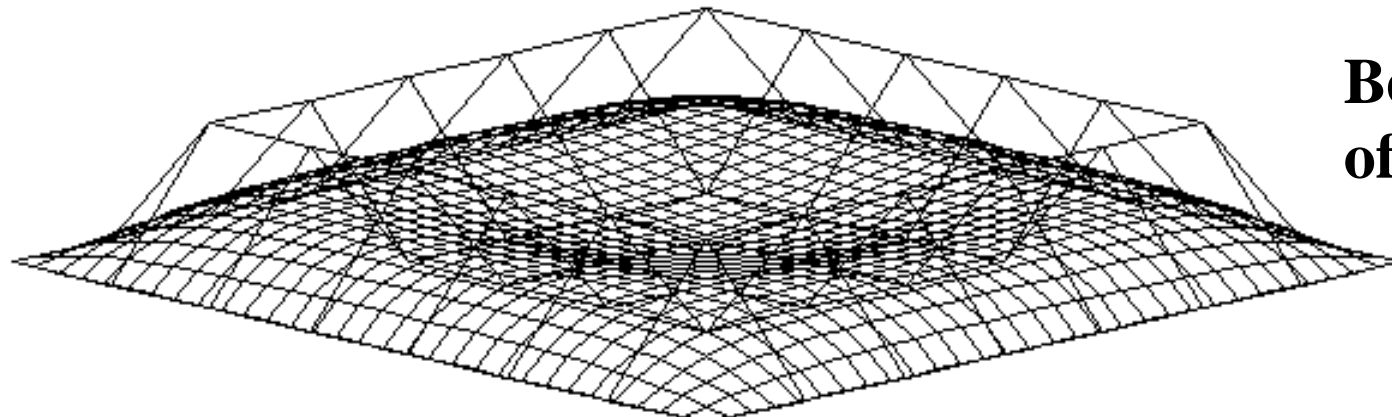
### Properties

- Affine invariance, barycentric combinations:  $\sum_{i=0}^m \sum_{j=0}^n B_i^m(u) B_j^n(v) \equiv 1$
  - Convex hull property
  - Boundary curves: The boundary curves of the patch  $b^{m,n}$  are binomial curves. Their Bézier polygons are given by the boundary polygons of the control net. The four corners of the control net lie on the patch.
  - Variation diminishing property: Definition not clear for the bivariate case
- It is easy to visualize a patch that is intersected by a straight line while its control net is not

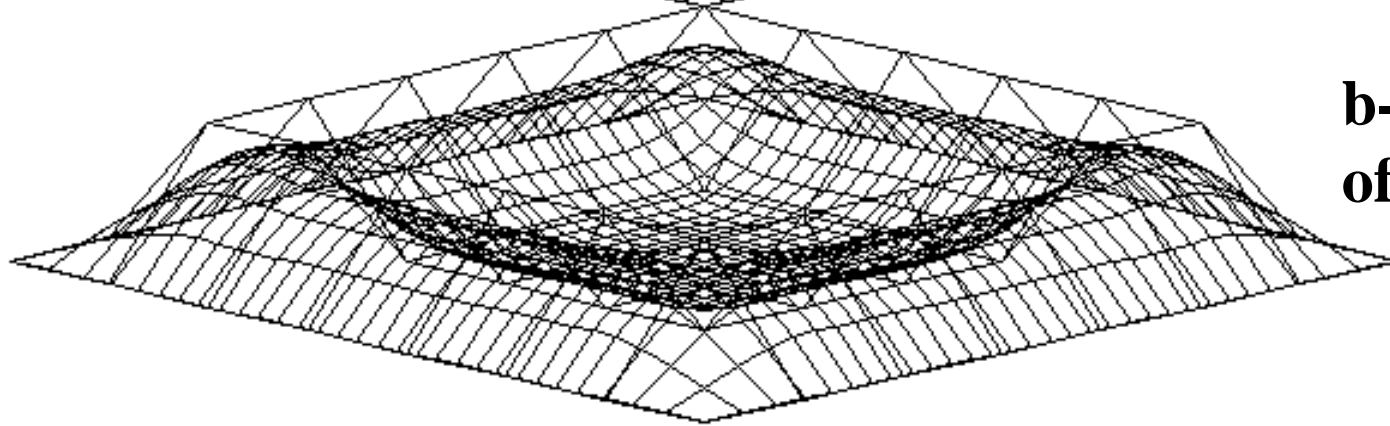




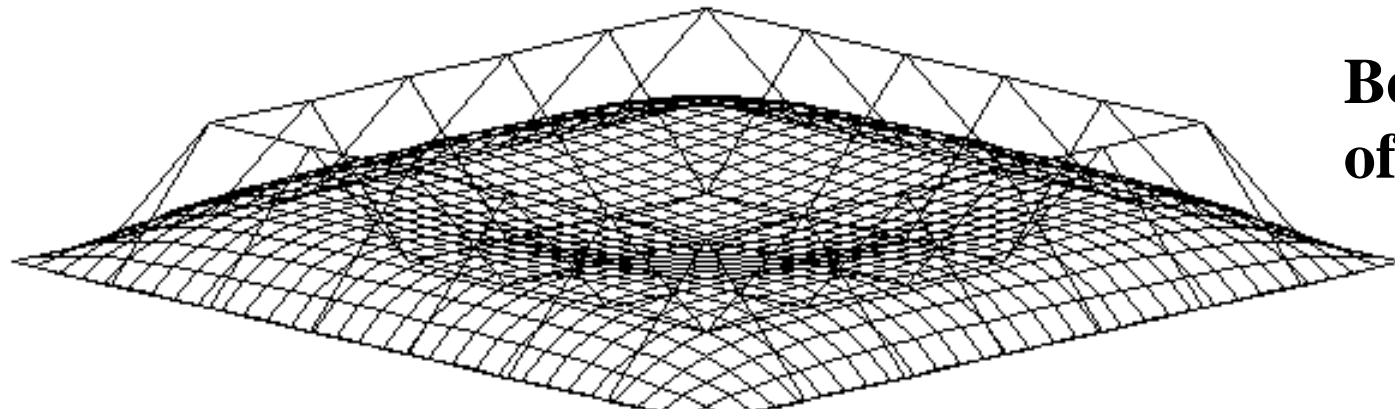




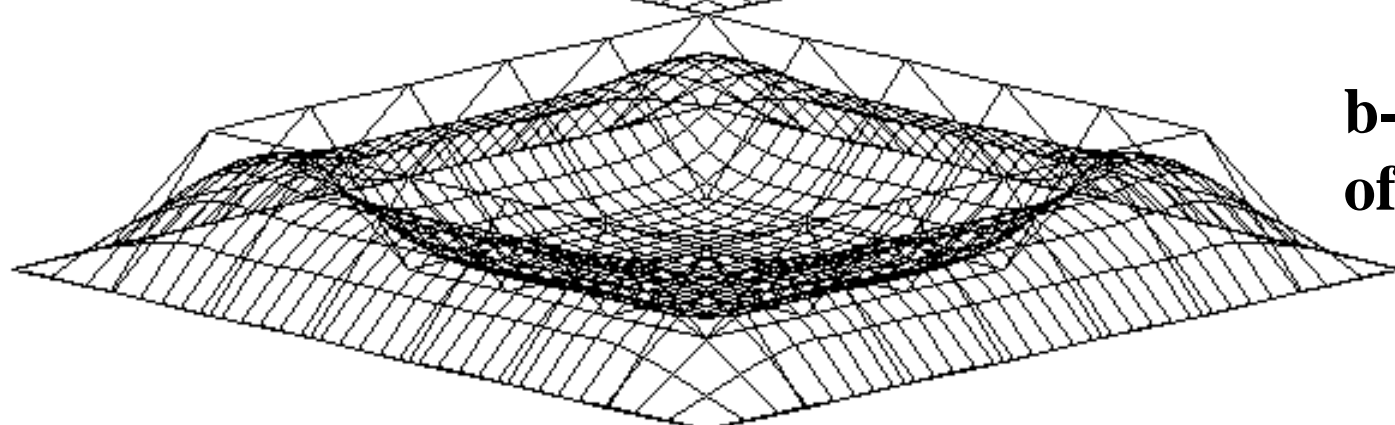
**Bezier Surface  
of degree 7**



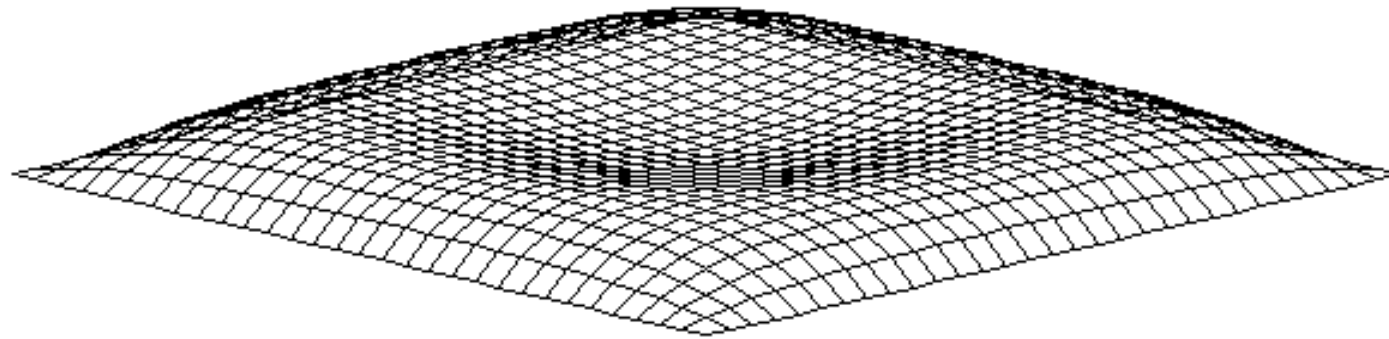
**b-spline Surface  
of degree 3**



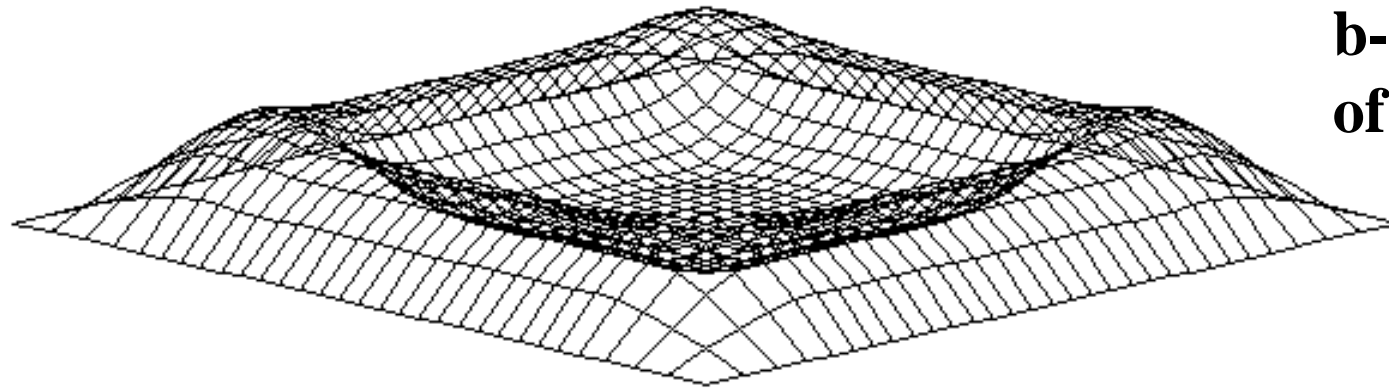
**Bezier Surface  
of degree 7**



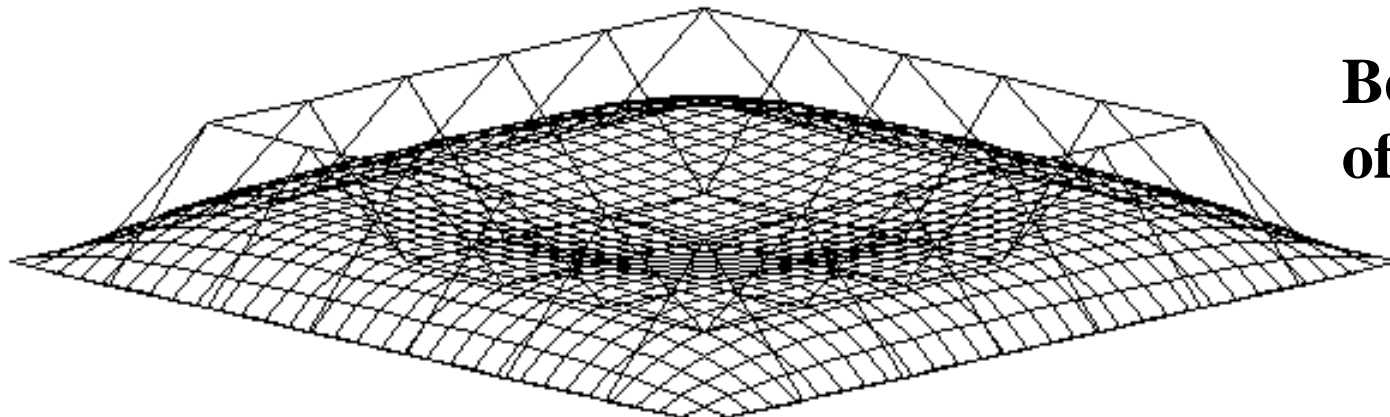
**b-spline Surface  
of degree 3**



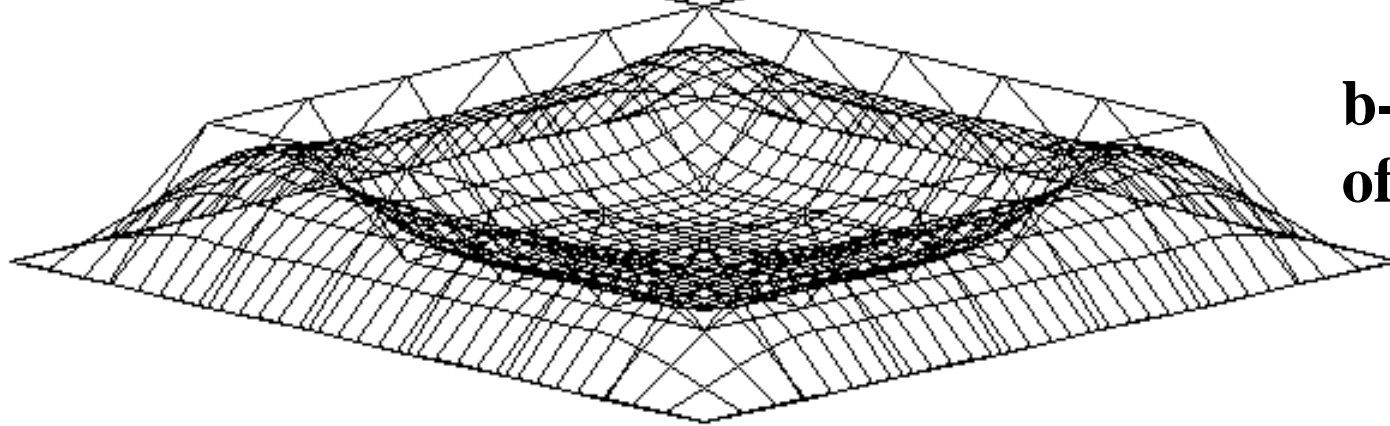
**Bezier Surface  
of degree 7**



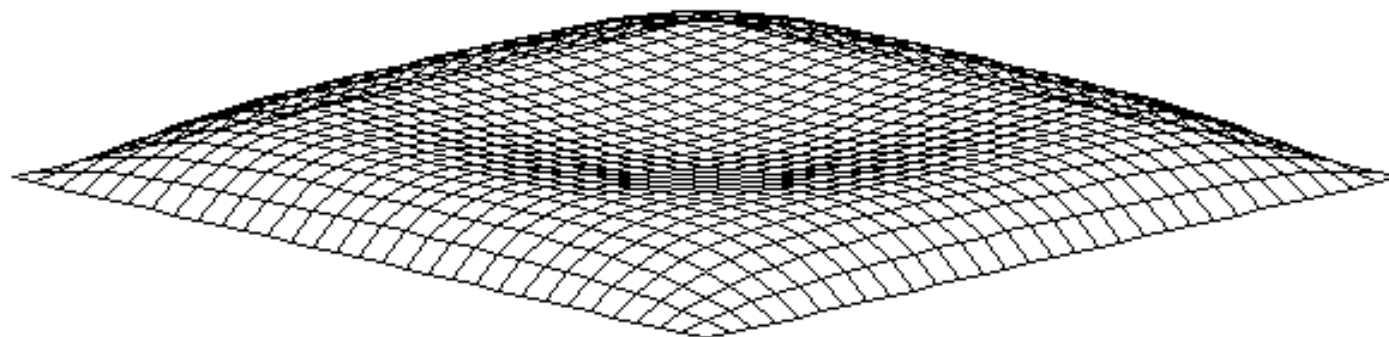
**b-spline Surface  
of degree 3**



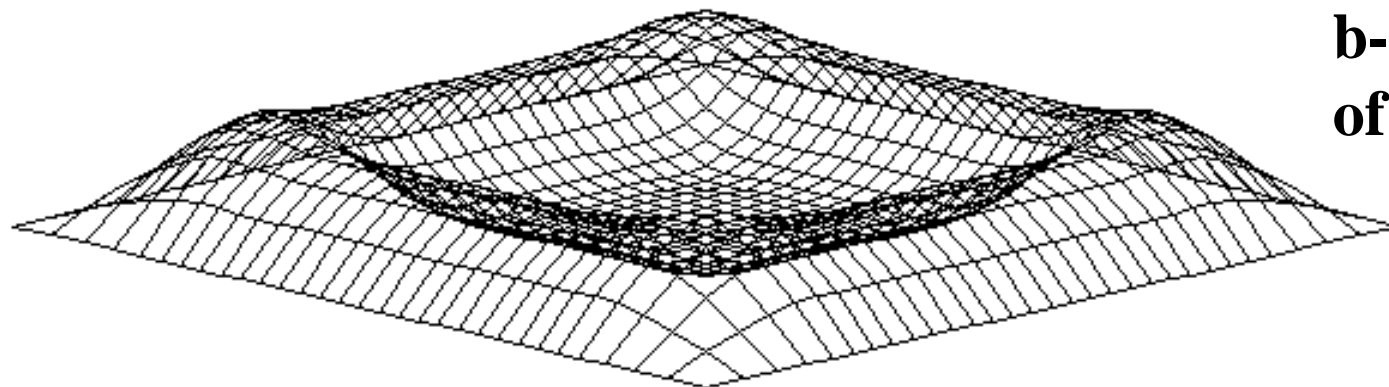
**Bezier Surface  
of degree 7**



**b-spline Surface  
of degree 3**

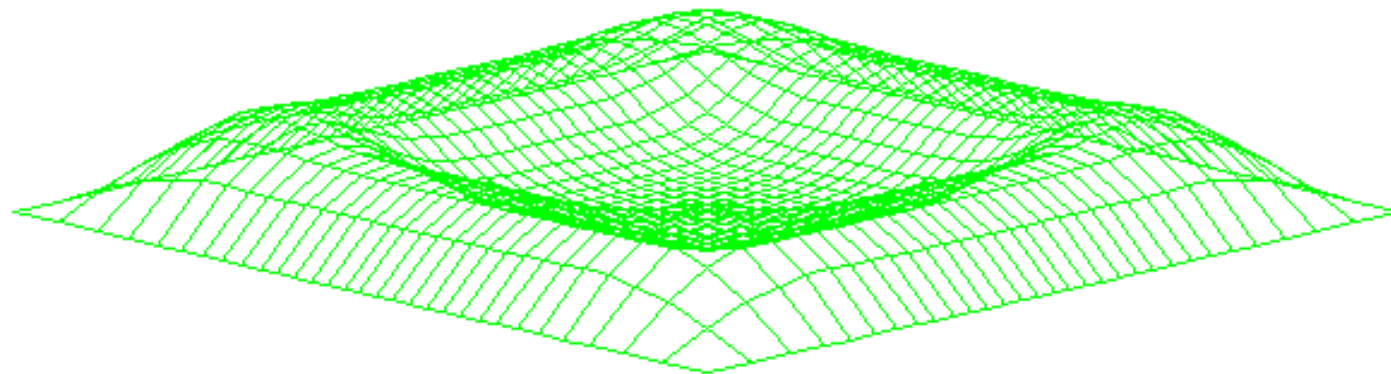


**Bezier Surface  
of degree 7**

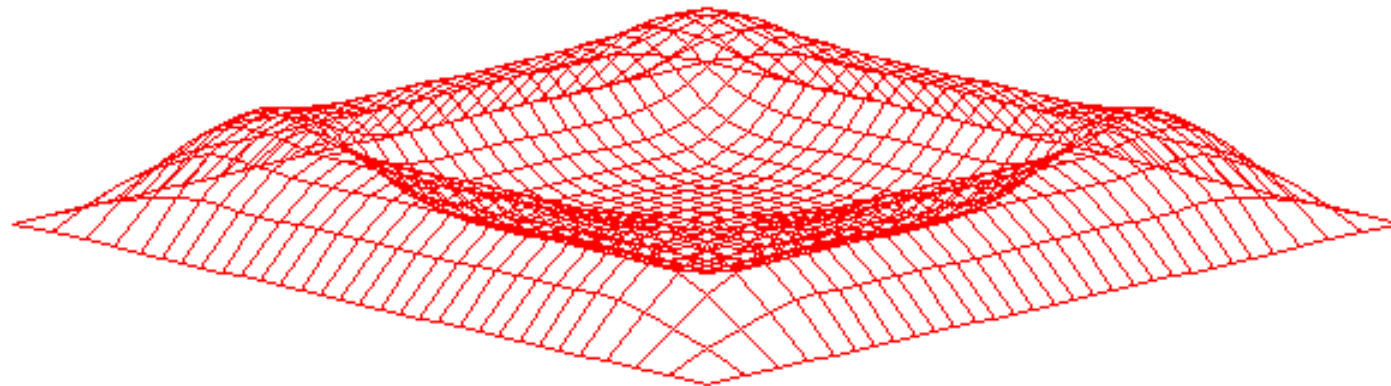


**b-spline Surface  
of degree 3**



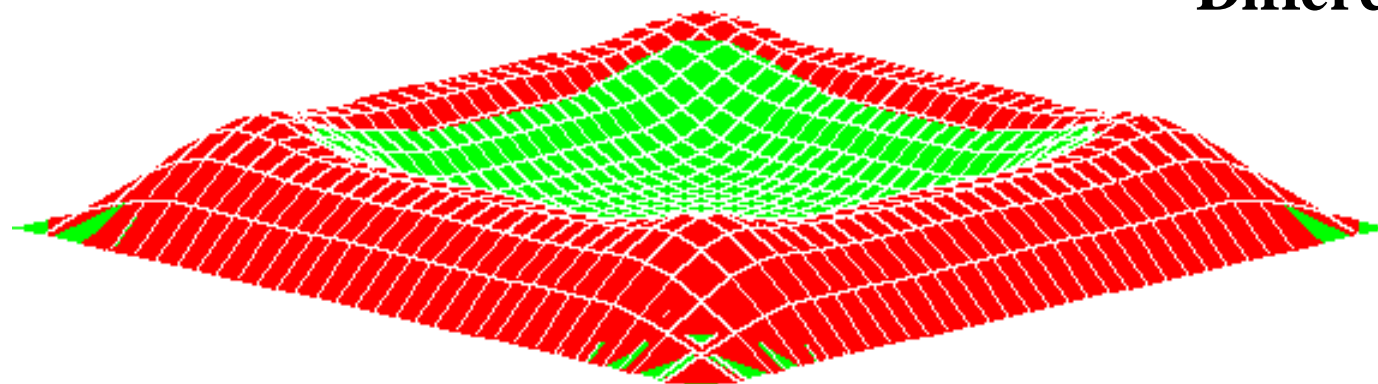


**b-spline surface  
degree 3**



**b-spline surface  
degree 2**

**Difference**



## Normal Vectors - Tensor Product Bézier Surfaces

The normal vector  $n$  of a surface is a normalized vector that is normal to the surface at a given point.

Cross product of any two vectors that are tangent to that surface at that point.  $\partial/\partial u$  &  $\partial/\partial v$

$\times$  : cross product

$\| \cdot \|$  : Modulus

$$n(u,v) = \frac{\frac{\partial}{\partial u} b^{m,n}(u,v) \times \frac{\partial}{\partial v} b^{m,n}(u,v)}{\| \frac{\partial}{\partial u} b^{m,n}(u,v) \times \frac{\partial}{\partial v} b^{m,n}(u,v) \|}$$

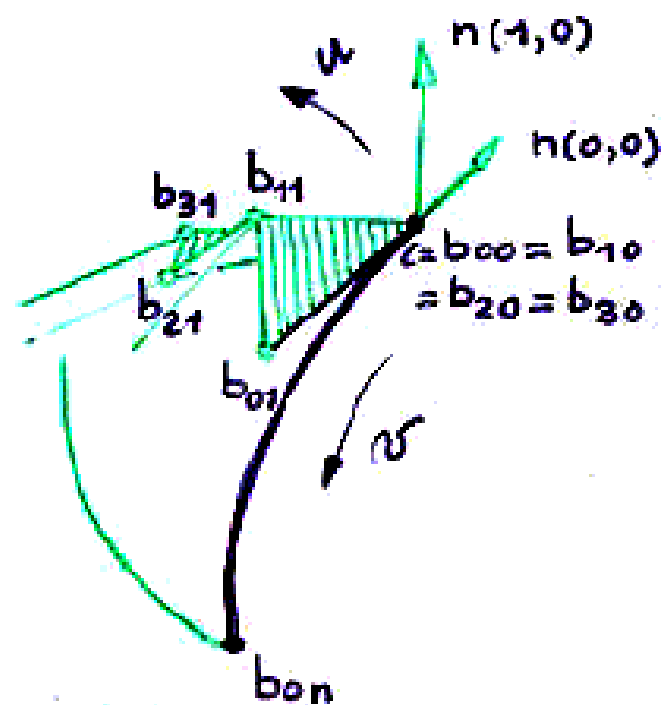
!! the four corners of the patch we obtain simply differences of boundary points:

$$n(0,0) = \frac{\Delta^{1,0} b_{0,0} \times \Delta^{0,1} b_{0,0}}{\| \Delta^{1,0} b_{0,0} \times \Delta^{0,1} b_{0,0} \|}$$

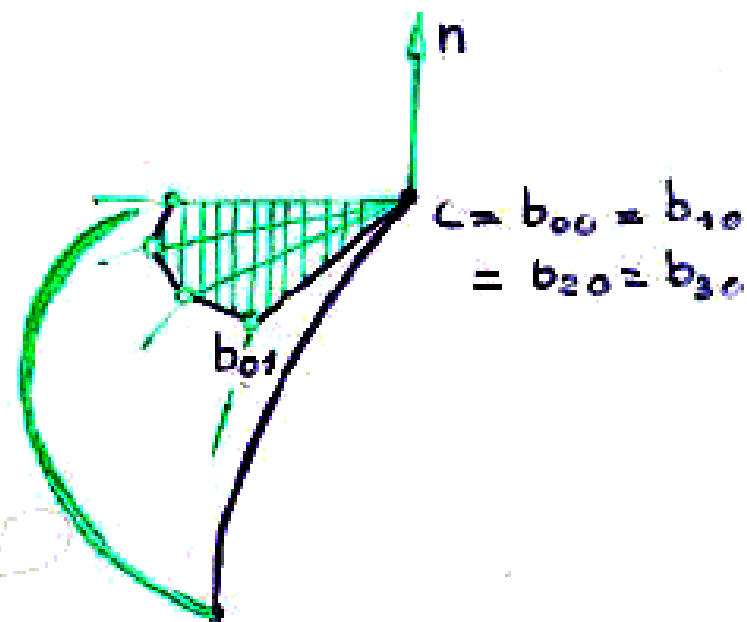
idem!

$b_{1,0}; b_{1,1}; b_{0,1}$

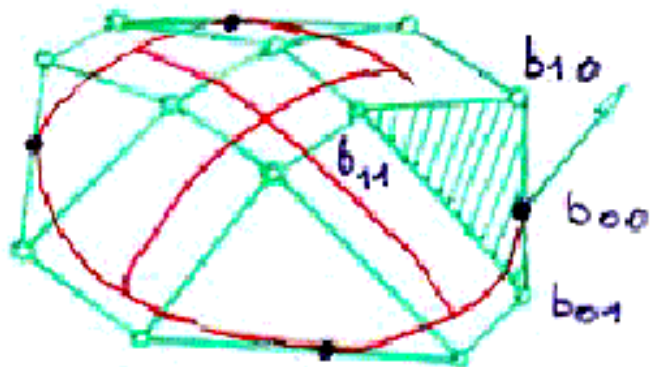
The normal at one of the corners is undefined if  $\Delta^{1,0} b_{0,0}$  and  $\Delta^{0,1} b_{0,0}$  are linearly dependent : we find a degenerate expression  $\frac{0}{0}$ . The corresponding patch is then called degenerate



a whole boundary curve collapsed into a point



if all  $b_{i,j}$  and  $c$  are collinear then the normal vector  $c$  is perpendicular to that plane.

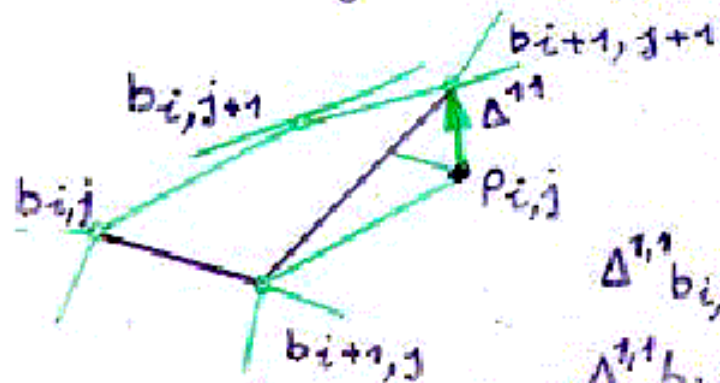


Normals at all four corners are determined by triangles

Whole boundary curve collapsed into a single point e.g.  
 $b_{00} = b_{10} = \dots = b_{m0} = c$  then  $b(u, v)$  would degenerate into a single point. In such cases, the normal vector at  $v=0$  may or may not be defined. Consider the tangents of the isoparametric lines  $u = \hat{u}$ , evaluated at  $v=0$ . These tangents must be perpendicular to the normal vector, if it exists. So a condition for the existence of the normal vector at  $c$  is that all  $v$ -partials, evaluated at  $v=0$ , are coplanar. But that is equivalent to  $b_{01}, b_{11}, \dots, b_{m1}$  and  $c$  being coplanar.

## TWISTS

The twist of a polynomial surface is its mixed partial  $\partial^2 / \partial u \partial v$ .  
 The twist surface of  $b^{m,n}$  is a Bézier surface of degree  $(m-1, n-1)$ , and its vector coefficients have the form  $m n \Delta^{1,1} b_{i,j}$ . These coefficients have a nice geometric interpretation.



The point  $P_{i,j}$  is the fourth point on the parallelogram  $b_{i,j}; b_{i+1,j}; b_{i,j+1}$ .

$$P_{i,j} - b_{i+1,j} = b_{i,j+1} - b_{i,j}$$

$$\Delta^{1,1} b_{i,j} = (b_{i+1,j+1} - b_{i+1,j}) - (b_{i,j+1} - b_{i,j})$$

$$\Delta^{1,1} b_{i,j} = b_{i+1,j+1} - P_{i,j}$$

Thus  $\Delta^{1,1} b_{i,j}$  measure the deviation of each subquadrilateral from a parallelogram.

## Twists

The twists at the four corners determine the deviation of the respective corner subquadrilaterals of the control net from parallelograms for example

$$\frac{\partial^2}{\partial u \partial v} b^{m,n}(0,0) = mn \Delta^{1,1} b_{00}$$

This twist vector is a measure for the deviation of  $b_{11}$  from the tangent plane at  $b_{00}$

An interesting class of surfaces is obtained if all subquadrilaterals  $b_{i,j}; b_{i+1,j}; b_{i,j+1}; b_{i+1,j+1}$  are parallelograms; in that case the twist vanishes everywhere. Such surfaces are called translational surfaces



## Coons Patches

physical model  $\rightarrow$  digitized  $\rightarrow$  feature lines (fitted curves through digitized points)  $\rightarrow$  The resulting network of curves now must be completed in order to generate a full surface description of the model  $\rightarrow$  This can be solved using Coons and Gordon surfaces

### Ruled Surfaces (lofted surfaces)

Problem: given two space curves  $C_1$  and  $C_2$ , both defined over the same parameter interval  $u \in [0,1]$ , find a surface  $\alpha$  that contains both curves as "opposite boundary curves":  
find  $\alpha$  such that

$$\alpha(u,0) = C_1(u), \quad \alpha(u,1) = C_2(u)$$

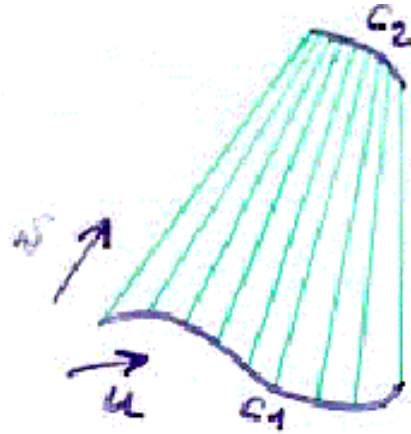


The stated problem has infinitely many solutions, so we pick the "Simplest" one :

$$x(u, v) = (1-v)C_1(u) + vC_2(u)$$

$$x(u, v) = (1-v)x(u, 0) + vx(u, 1)$$

We see that ruled surfaces have the familiar flavor of linear interpolation: every isoparametric line  $u = \text{const}$  is a straight line segment. Now we interpolate to whole curves not just points!



One important aspect of ruled surfaces is the generality that is allowed for the input curves  $\alpha(u, 0)$  and  $\alpha(u, 1)$ ; there is no restriction on them other than having to be defined over the same parameter interval.

cubic polynomial curve - the other one a spline or even a polygon

### Coons patches : Bilinearly Blended

A ruled surface interpolates to two boundary curves - a rectangular surface, however, has four boundary curves, and that is precisely to what a Coons patch interpolates.



	Degree n	Local Control	Embedded Line Segment	Model Conic Sections	Invariant Under Affine Transform	Invariant Under Parallel Projection	Invariant Under Perspective Projection	Convex Hull
<b>Bezier</b>	Fixed: Number of Control Points - 1	NO	NO	NO	YES	YES	NO	Conex Hull of all Control Points
<b>B-spline</b>	Variable: From 1 to Number of Control Points -1	YES +/- (n+1)/2 Spans	YES	NO	YES	YES	NO	Union of Convex Hulls of n+1 Control Points
<b>NURBS</b>	Variable: From 1 to Number of Control Points - 1	YES +/- (n+1)/2 Spans	YES	YES	YES	YES	YES	Union of Convex Hulls of n+1 Control Points

## **Lesson XIII: SOLID MODELING**

- ✓ **Construction Solid Geometry (CSG) Boolean Operations**
- ✓ **Boundary Representation**
- ✓ **Volume Elements (VOXELS) Octree/Quatree**
- ✓ **Hyperpatch Parametric Polynomial  $X(u,v,w)$**

# Boolean Operations

Boolean operations can be used to construct the geometric Model

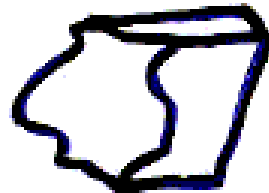
## Primitives



cube

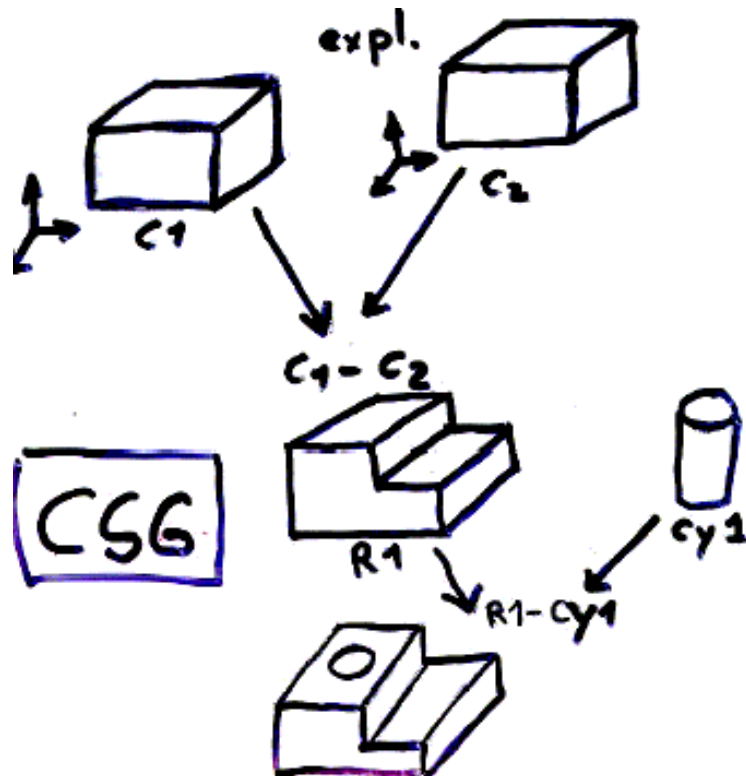


cyl



SURF SOLID

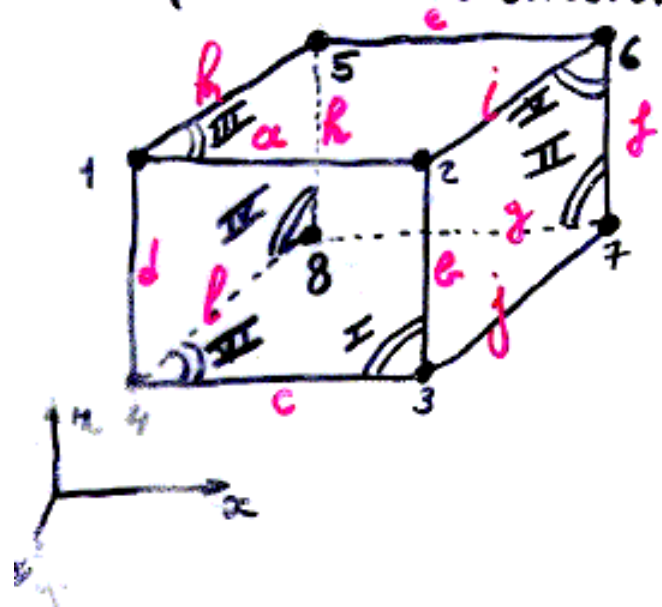
...



$$R_1 = (C_2 - C_1)$$

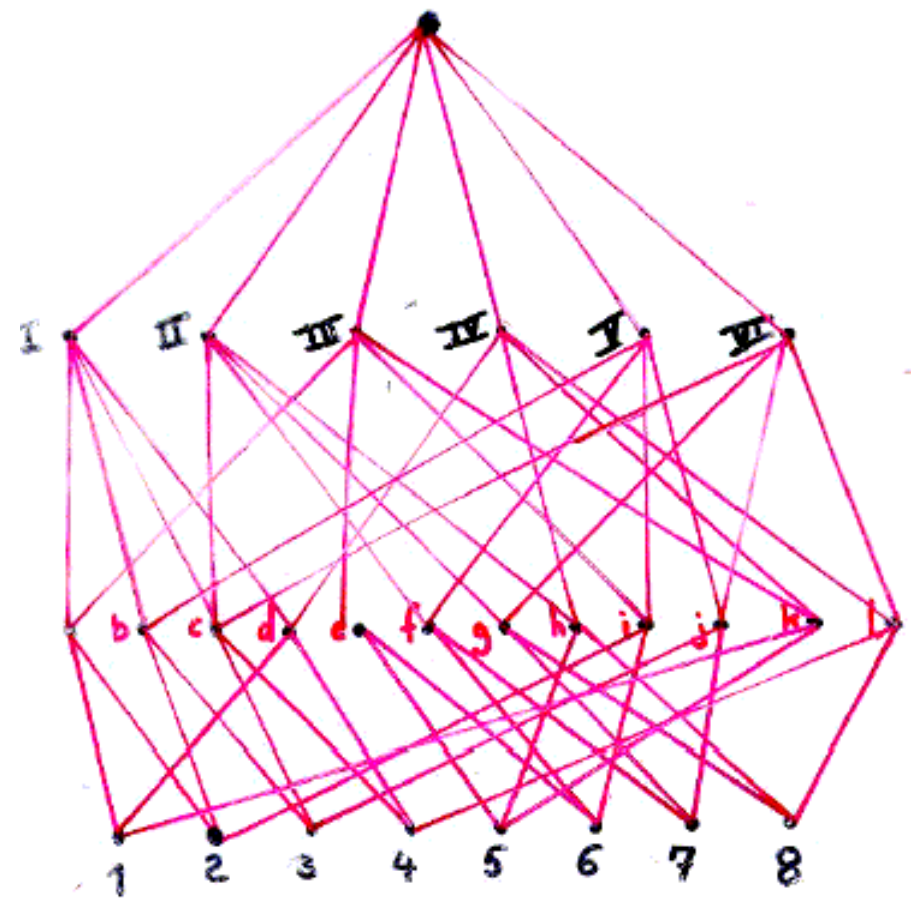
$$R = (C_2 - C_1) - CY_1$$

example of a data structure



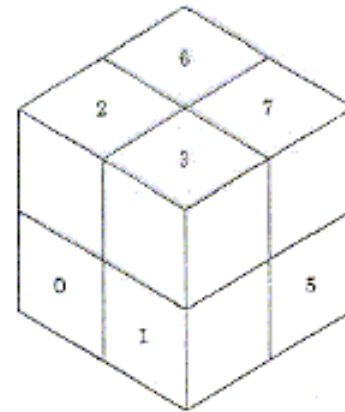
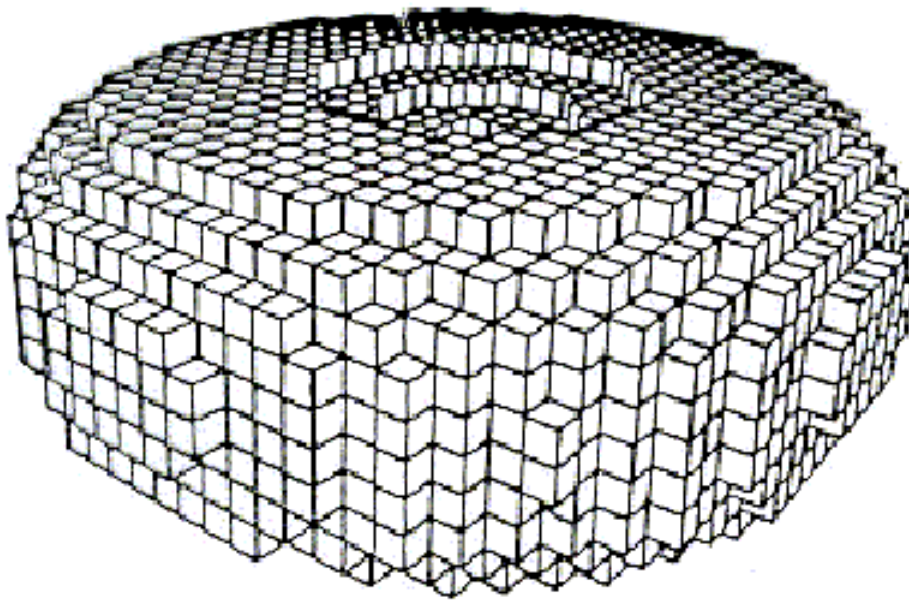
POINTS	EDGES	FACES	OBJECT
1 $x_1, y_1, z_1$	a 1→2	I a b c d	I
2 $x_2, y_2, z_2$	b 2→3	II c h g f	II
3 $x_3, y_3, z_3$	c 3→4	III a i e h	III
4 $x_4, y_4, z_4$	d 4→1	IV d k l e	IV
5 $x_5, y_5, z_5$	e 5→6	V b j f i	V
6 $x_6, y_6, z_6$	f 6→7	VI c l g j	VI
7 $x_7, y_7, z_7$	g 7→8		
8 $x_8, y_8, z_8$	h 8→5		
	i 6→2		
	j 3→7		
	k 1→5		
	l 4→8		

OBJECT/SOLID

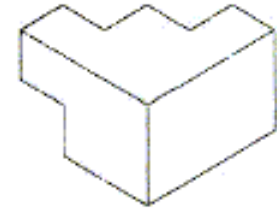


B-rep  
Boundary Representation

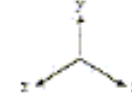




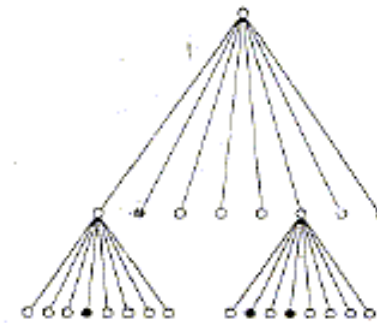
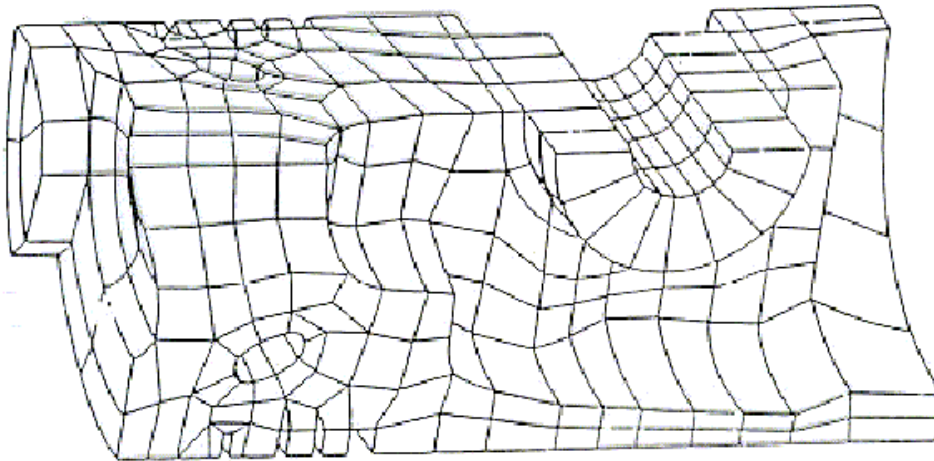
(a)



(b)

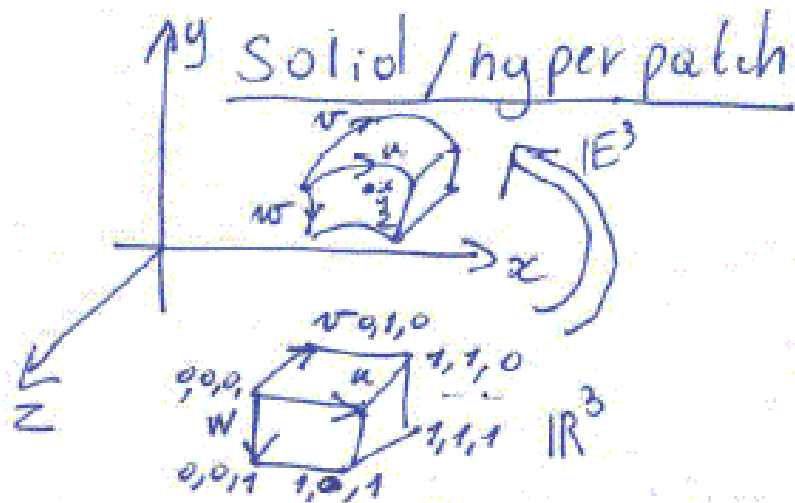


## Cell Decomposition Voxels



**OCTREE – 3D**  
**QUATREE – 2D**



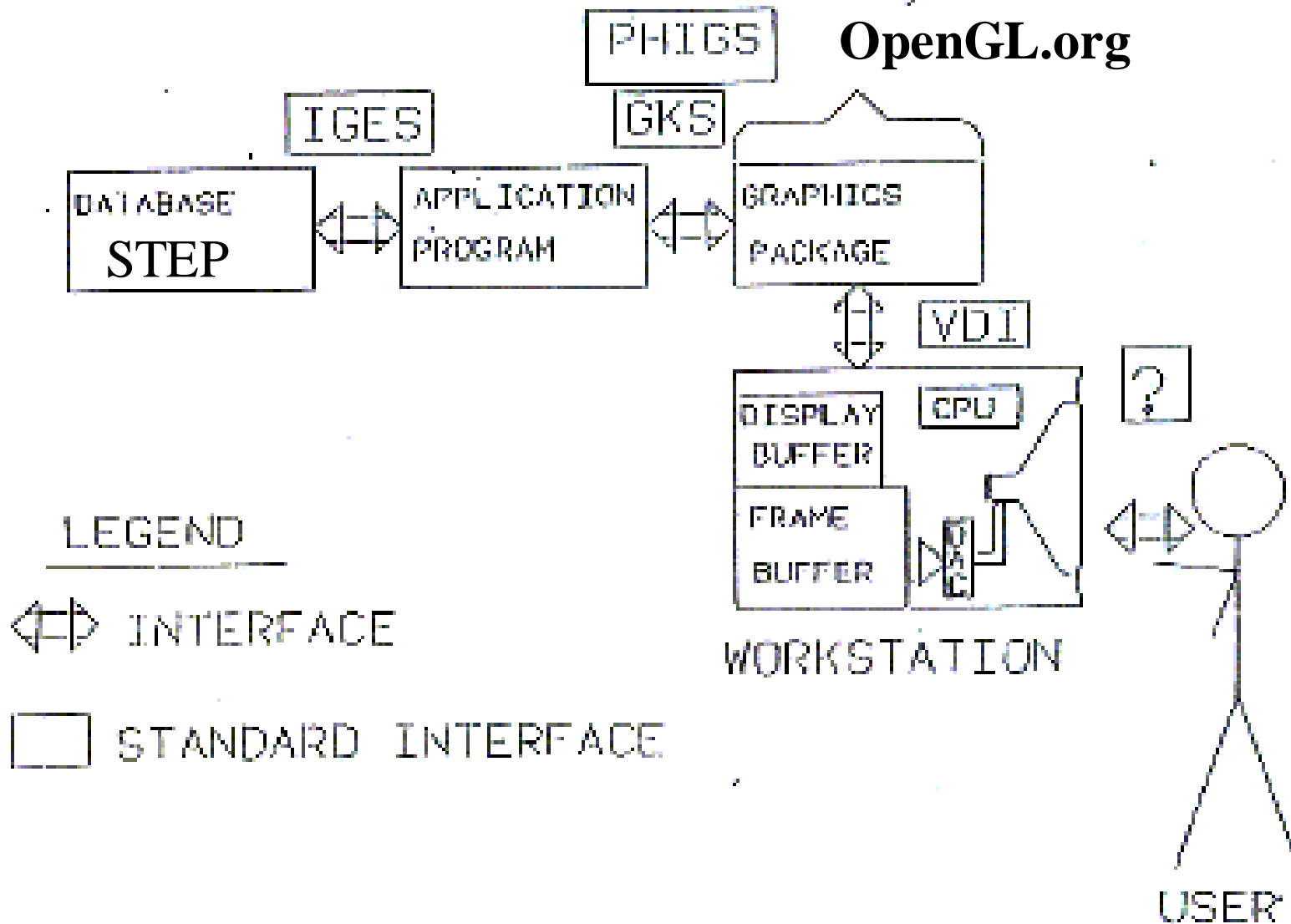


$$\begin{aligned} x(u, v, w) &= f_x(u, v, w) \\ y(u, v, w) &= f_y(u, v, w) \\ z(u, v, w) &= f_z(u, v, w) \end{aligned}$$

functions of  
3 parameters

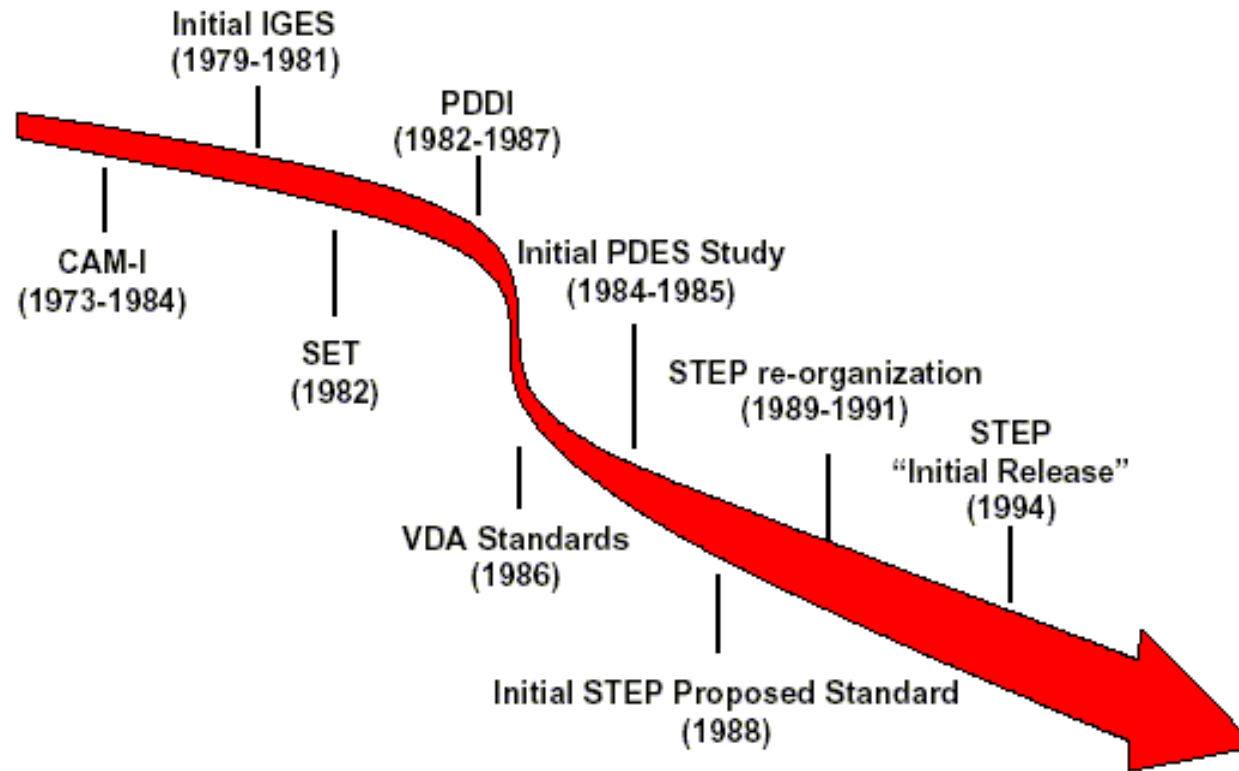
**Most used Solid Modeling systems are based on CSG & B-Rep  
Translators are needed to Translate from one representation  
to another**

## Lesson XIV: CAD SYSTEM ARCHITECTURE

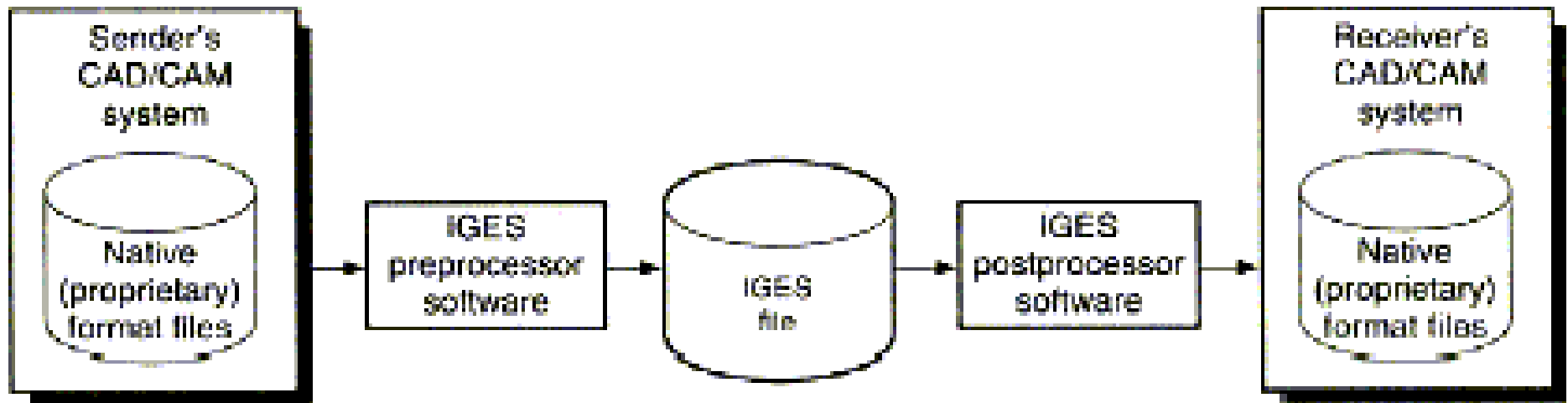


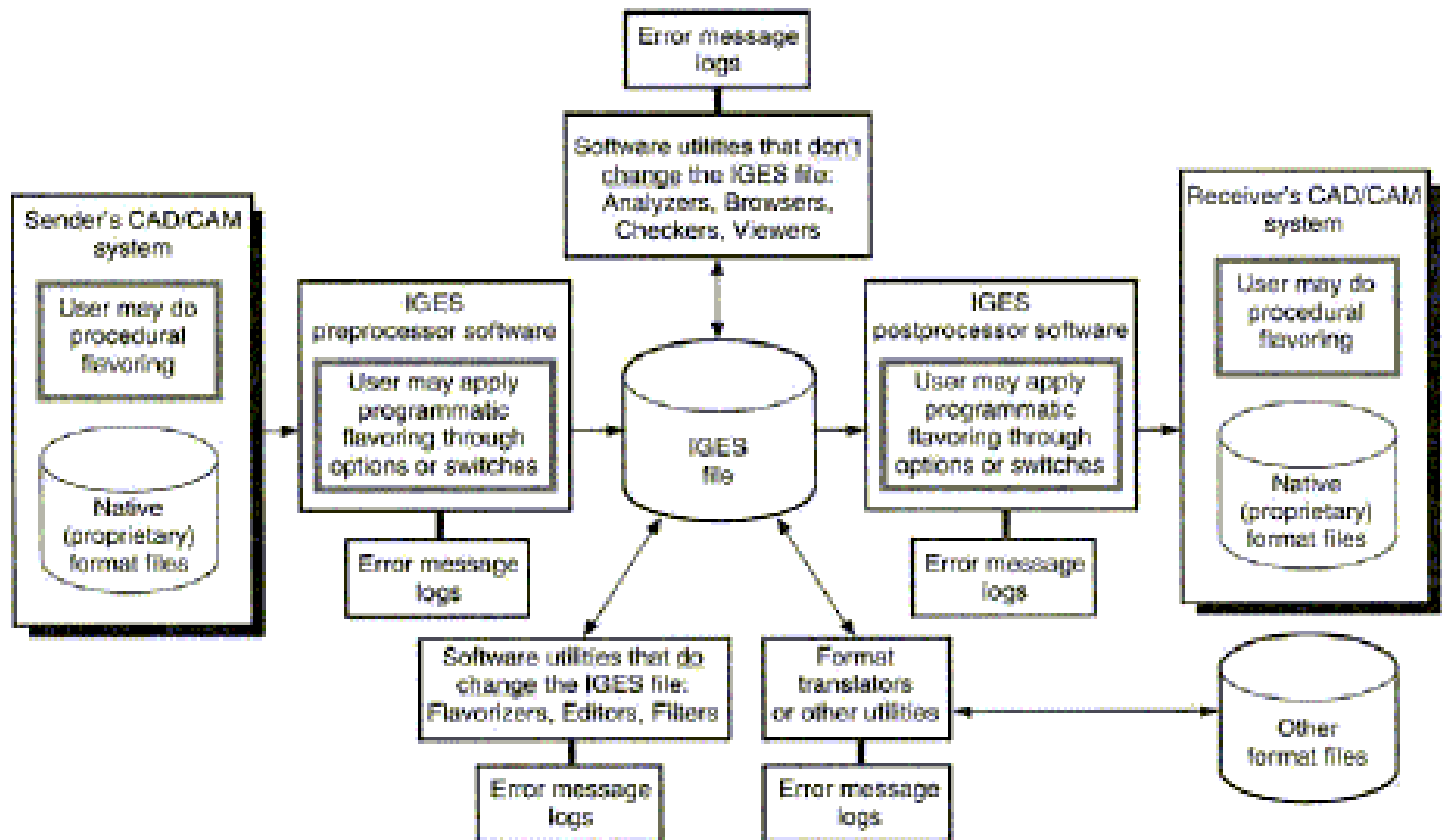
**IGES: Initial Graphics Exchange Specification ISO**

# Evolution of Standards in CAD/CAM



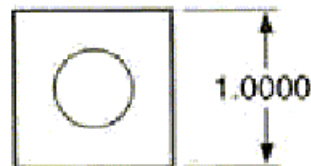
## Lesson XV: IGES – STEP



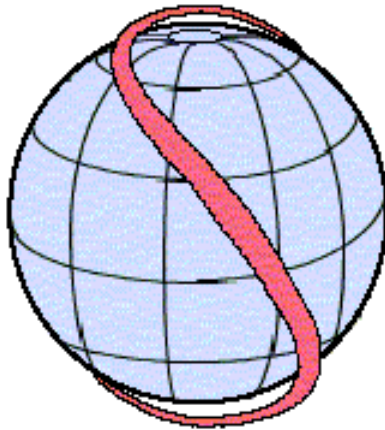


Start	IGES file generated from an AutoCAD drawing by the IGES	800000001
	translator from Autodesk, Inc., translator version IGESOUT-3.04.	800000002
Global	,,3HTMP,24HG:\COMMON\LESSON.IGS.IGS,10HAutoCAD-11,12HIGESOUT-3.04,32,38,600000001	600000001
	6,99,15,3HTMP,1.0,1,4HINCH,32767,3.2767D1.13H910029.065145,	600000002
	1.9044561766567D-9,1.9044561766567,15H T.C. IGES UC	600000003
	..6,0;	600000004
	110 1 1 1	000000000D00000001
	110 1 1	D00000002
	110 2 1 1	000000000D00000003
	110 3 1 1	D00000004
	110 3 1 1	000000000D00000005
	110 4 1 1	D00000006
	110 4 1 1	000000000D00000007
	110 5 1 1	D00000008
	100 5 1 1	000000000D00000009
	100 6 1 1	D00000010
Directory entry	212 6 1 1	00010100D00000011
	212 7 1 2	D00000012
	106 8 1 1	00010100D00000013
	106 9 1 1 40	D00000014
	106 9 1 1 40	00010100D00000015
	106 10 1 1 40	D00000016
	214 10 1 1 3	00010100D00000017
	214 11 1 1 3	D00000018
	214 12 1 1 3	00010100D00000019
	214 13 1 1 3	D00000020
	216 14 1 1 0	00000101D00000021
	216 15 1 1	D00000022
	110,0.0,0.0,0.0,0.0,1.0,0.0,0.0;	1P00000001
	110,1.0,0.0,0.0,0.0,1.0,1.0,0.0;	3P00000002
	110,1.0,1.0,0.0,0.0,1.0,0.0,0.0;	5P00000003
	110,0.0,1.0,0.0,0.0,0.0,0.0,0.0;	7P00000004
	100,0.0,0.5,0.5,0.75,0.5,0.75,0.5;	9P00000005
Parameter data	212,1.6,0.72,0.18,1.0,0.0,0.0,1.18444561766567D0,0.41,0.0,6H1.0000	11P00000006
	;	11P00000007
	106,1.3,0.0,1.0,1.0,1.0625,1.0,1.7244561766567D0,1.0;	13P00000008
	106,1.3,0.0,1.0,0.0,1.0625,0.0,1.7244561766567D0,0.0;	15P00000009
	214,1.0.18,6.0D-2,0.0,1.5444561766567D0,1.0,1.5444561766567D0,	17P00000010
	0.68;	17P00000011
	214,1.0.18,6.0D-2,0.0,1.5444561766567D0,0.0,1.5444561766567D0,	19P00000012
	0.32;	19P00000013
Terminate	216,11,17,19,13,15;	21P00000014
	800000002G00000004D00000022P00000014	T09000001

Equivalent graphics:



# STEP



- **The Standard for the Exchange of Product Model Data**
- **An International Standard supporting:**
  - ◆ exchange of information between engineering applications
  - ◆ long-term archiving of product information
  - ◆ implementation of shared product databases
- **ISO 10303 “Product data representation and exchange”**



## ■ STEP

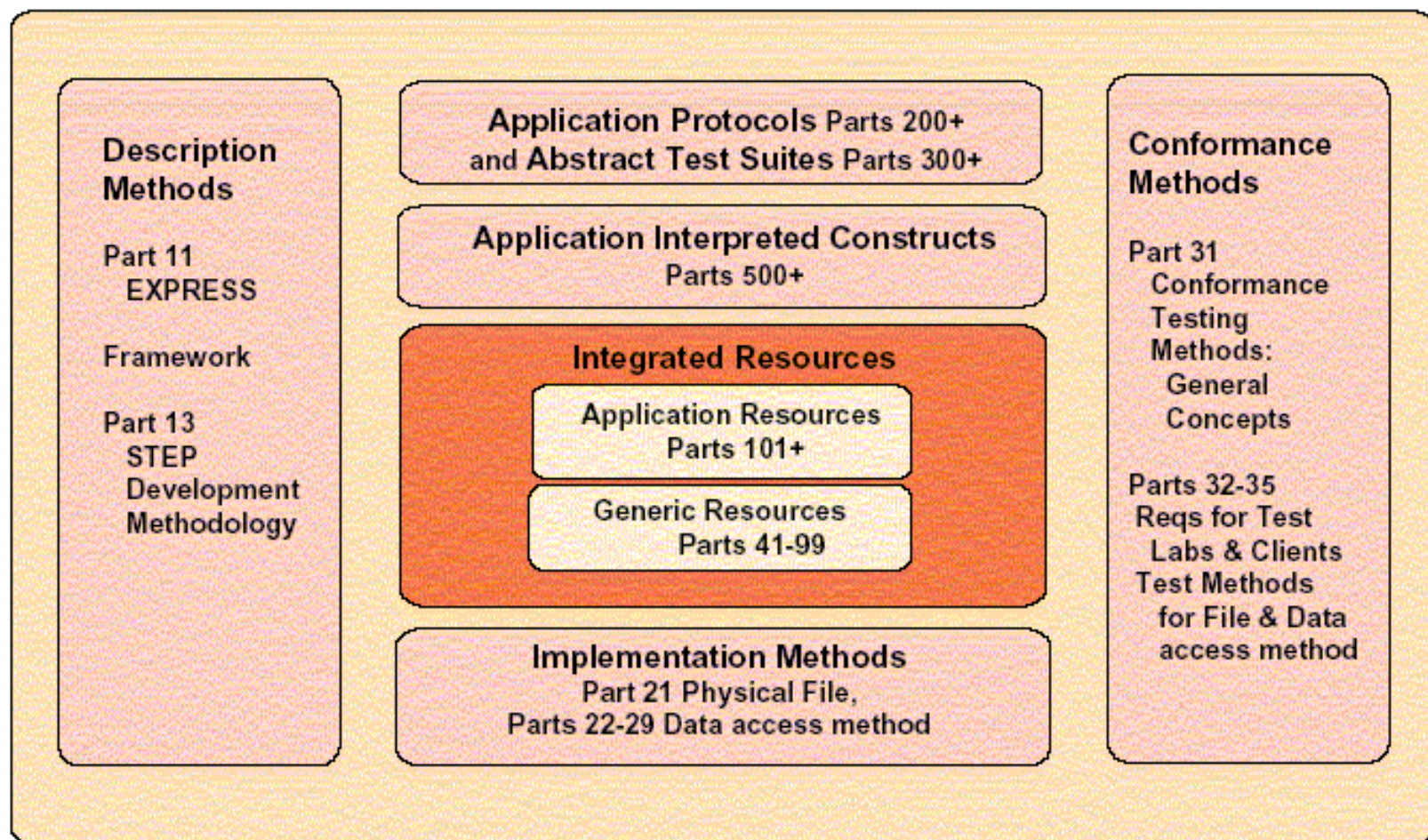
- ◆ is an ISO standard
- ◆ defines a reference model for describing products
- ◆ supports the full life cycle of the product in multiple application domains
- ◆ more and more systems are supporting STEP - STEP is in use
- ◆ STEP development is ongoing

## ■ STEP and Documentation

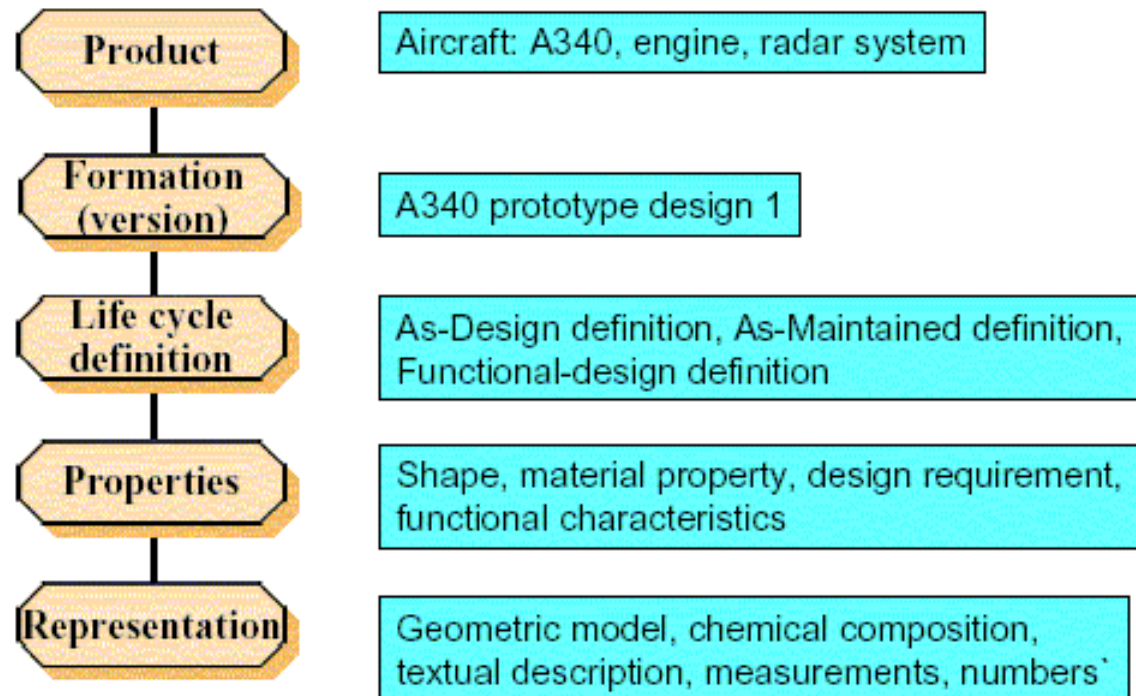
- ◆ product models are a beneficial resource for documentation applications
- ◆ documentation data is one component of product model data

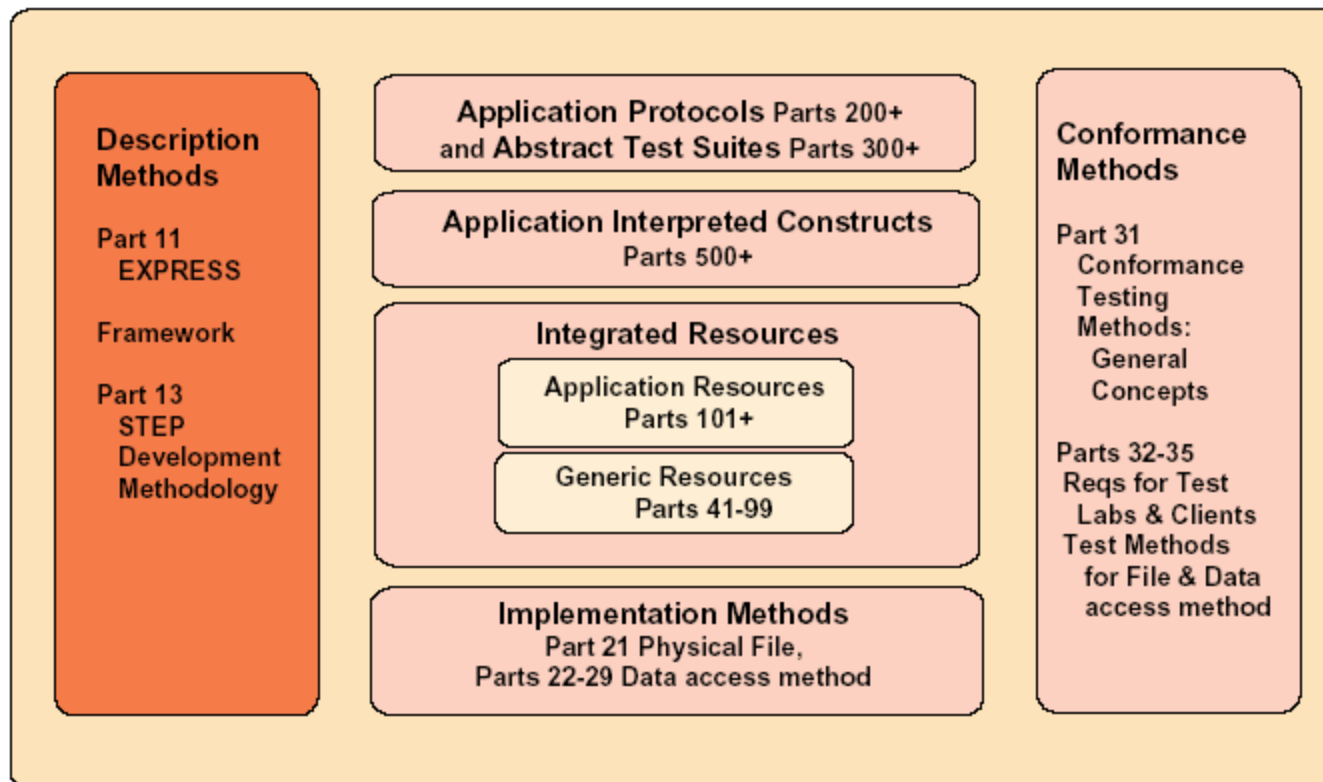
## ■ Information on STEP in the WWW

- ◆ NIST Server : <http://www.nist.gov/sc4/www/stepdocs.htm>
- ◆ STEP on a Page : <http://www.mel.nist.gov/sc5/soap/>
- ◆ STEP Entity Graph Viewer :  
[http://www.vx.com/cgi-bin/STEP/ent\\_grf\\_util.cgi](http://www.vx.com/cgi-bin/STEP/ent_grf_util.cgi)



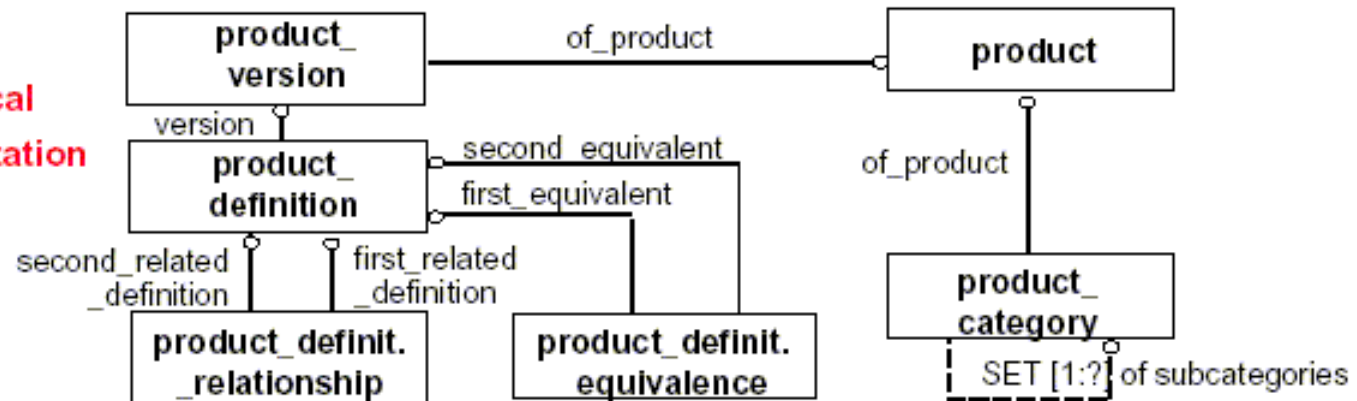
# INTEGRATED RESOURCES





# EXPRESS Language – Description Methods

Graphical  
Representation



Textual  
Representation

```

SCHEMA
  product_definition;
  
```

```

TYPE identifier = STRING;
END_TYPE;
  
```

```

TYPE label = STRING;
END_TYPE;
  
```

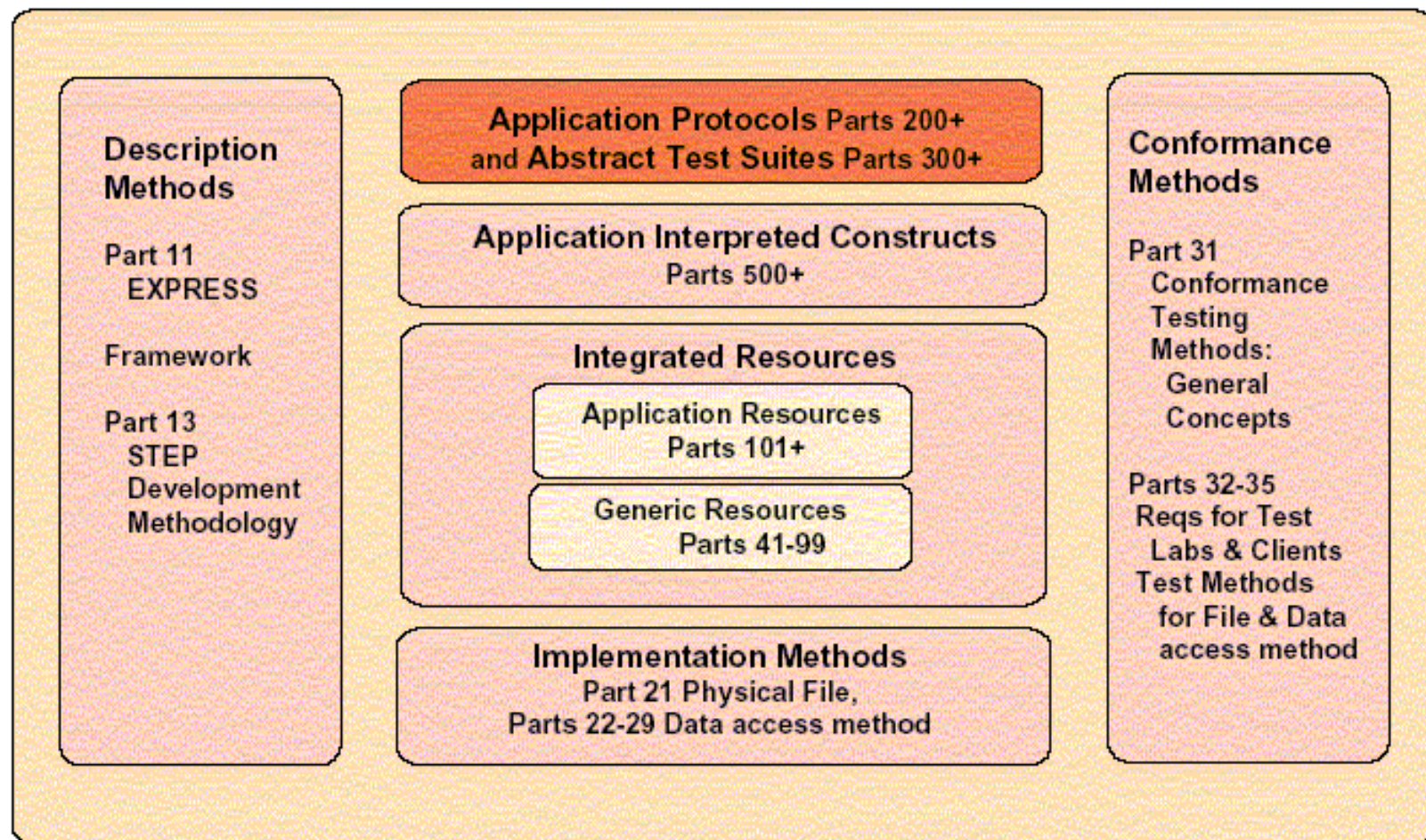
```

ENTITY product_category;
  name    : label;
  subcategories : OPTIONAL SET [1:?]
                OF product_category;
  of_product : product;
END_ENTITY;
  
```

```

ENTITY product_version;
  ident    : identifier;
  name     : label;
  description : OPTIONAL text;
  of_product : product;
END_ENTITY;
  
```

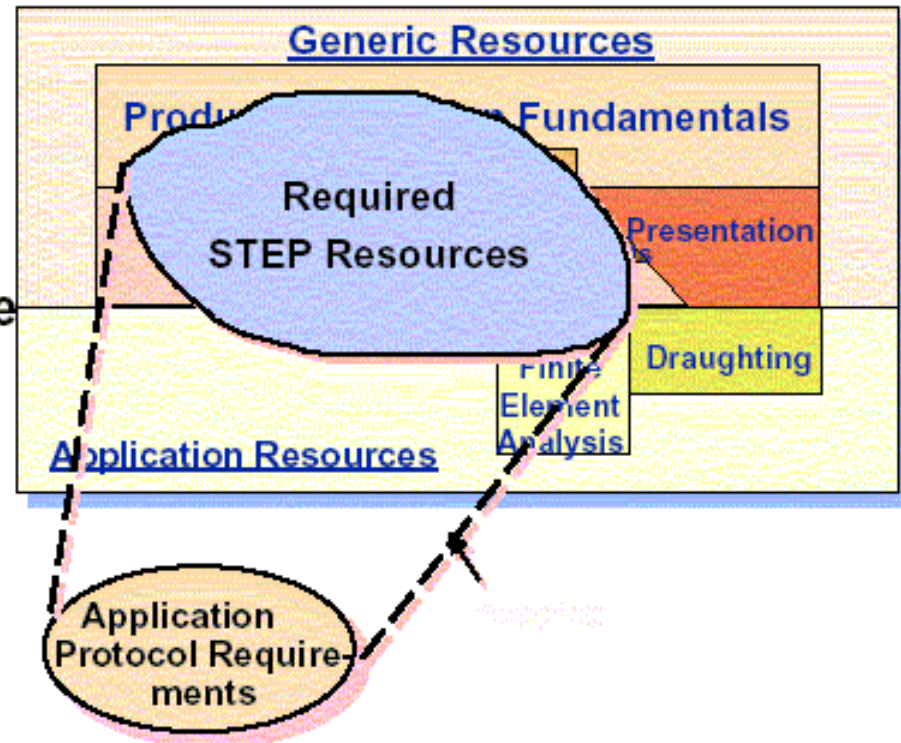






# APPLICATION PROTOCOLS

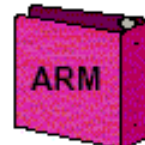
- An AP defines the usage of STEP Product Data for a given application context.
- An AP represents a measurable and shareable subset of STEP capability that is expressed in an industry's or discipline's terminology.





### ■ Application Activity Model

A function model that describes the activities and processes of defined application context domain. This is a requirement document.



### ■ Application Reference Model

An information model that describes the information requirements and constraints for an application context area. The model uses application-specific terminology and rules that are familiar to experts in the application area.



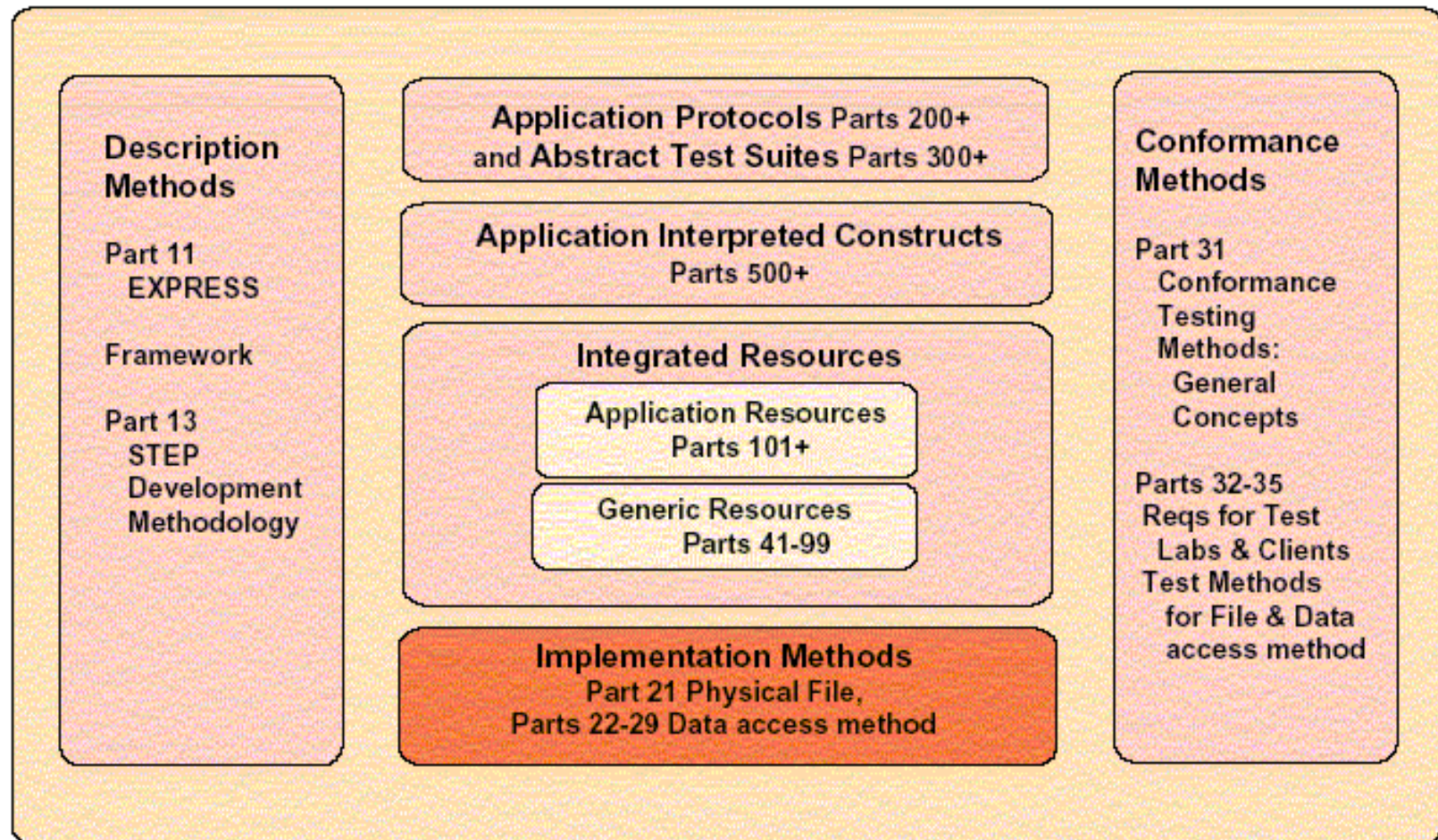
### ■ Application Interpreted Model

An information model that describes the STEP data structures required for functional equivalence with the application contexts' AAM's and ARM's.



### ■ Conformance Classes

Descriptions of the valid populations of the file, which serve to define conformant uses of the AP.





# IMPLEMENTATION METHODS

## ■ Part 21: Clear text encoding of the exchange structure

- ◆ Is the format used in the exchange of data via a physical file format.
- ◆ Result is a sequential file of ASCII characters described in a computer-interpretable form.
- ◆ Defines the mapping from EXPRESS into the exchange file:

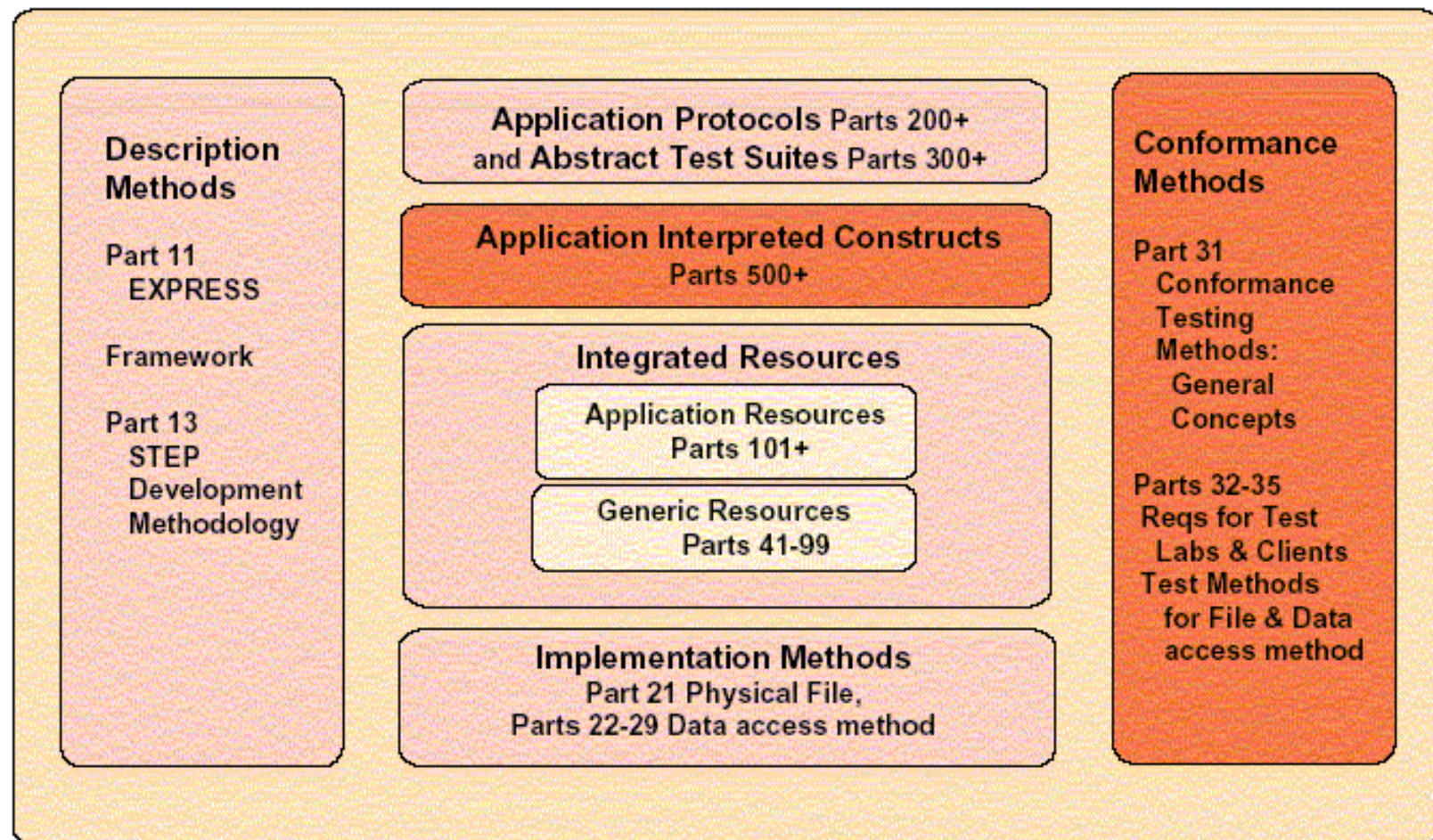
```
ENTITY product_category;  
  name      : label;  
  subcategories : OPTIONAL SET [1:?]  
                OF product_category;  
  of_product : OPTIONAL product;  
END_ENTITY;
```

```
ENTITY product_version;  
  ident      : identifier;  
  name       : label;  
  description : OPTIONAL text;  
  of_product : product;  
END_ENTITY;
```



```
#1=product_category("process vessel",(#2),$)  
#2=product_category("tank",$, #3)
```

```
#4=product_version("ver-001","Original Design",$, #3)
```



- **Basic STEP capability has been available since December 1994.**
  - ◆ 12 Parts registered as ISO 10303 international standards in December 1994. This is the “Initial Release”.
  - ◆ This includes 2 AP’s (Part 201) address the exchange of 2D drawing data and (Part 203) 3D design data under configuration control and included in the 12 parts.
- **Additional work on STEP is under development**
  - ◆ 29 AP’s are in different stages of ballot cycle to be standardized
  - ◆ Parametric capabilities
  - ◆ Interoperability of APs
- **CAX-Systems support STEP**
  - ◆ AutoCAD, CADDs, Catia, Euclid, IDEAS Master Series, Pro/Engineer, ROBCAD, Solid Designer, Syrko, Unigraphics, etc.

# **SUPPORTED INDUSTRY SECTORS**

## **■ Discrete Manufacturing**

- ◆ **Product Design Configuration Management**
- ◆ **Core Data for Automobile Industry**
- ◆ **CAD System Shape Geometry for Mechanical Product Design**
- ◆ **Engineering Drawing Information**
- ◆ **Manufacturing Technologies: Composites, NC Process Planning, Casting, Forging**

## **■ Electrical and Electronics**

- ◆ **PCA, PCB and Component Design**
- ◆ **Electronic Test Diagnostic and Remanufacture**
- ◆ **Electrotechnical Plant Design and Installation**



# SUPPORTED INDUSTRY SECTORS

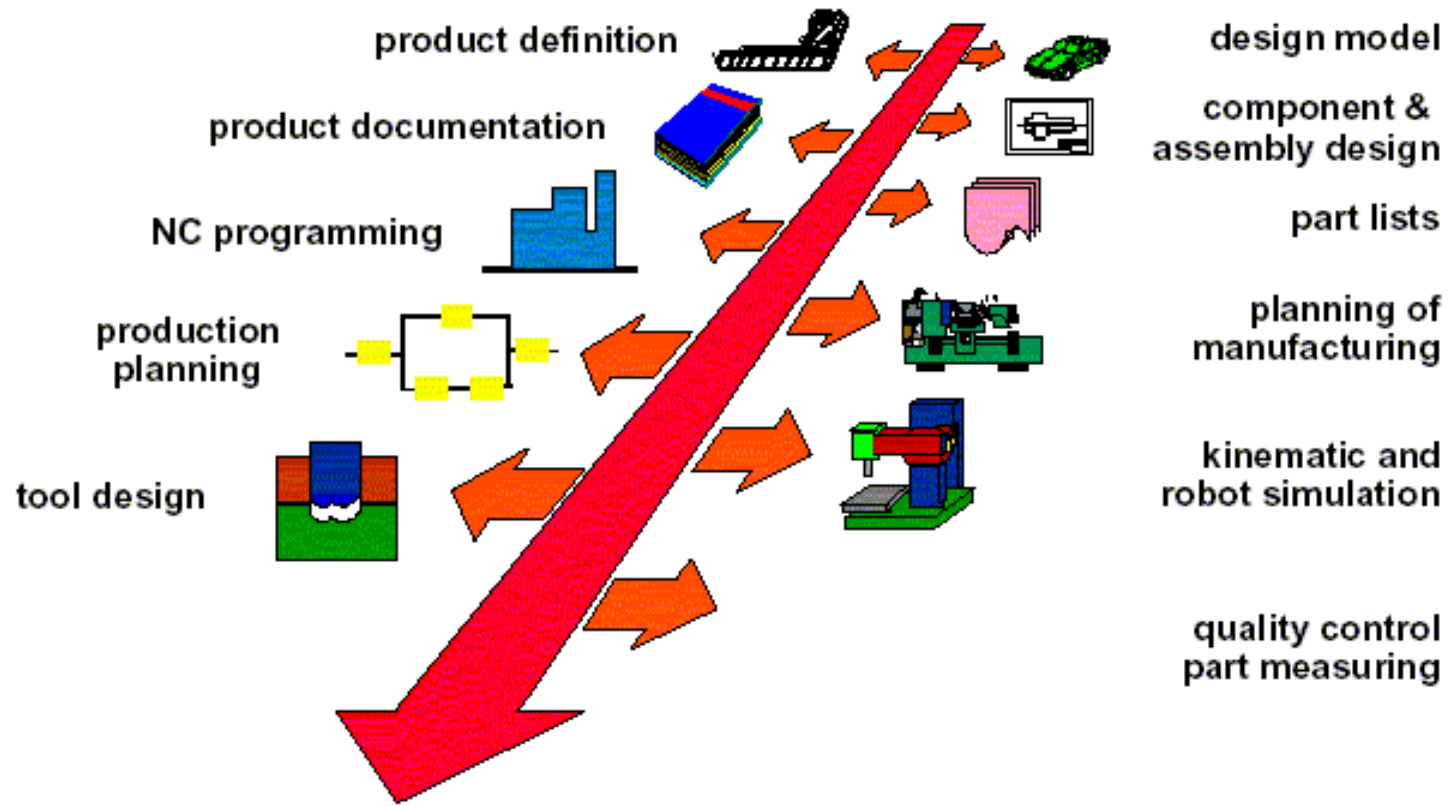
## ■ Architecture and Construction

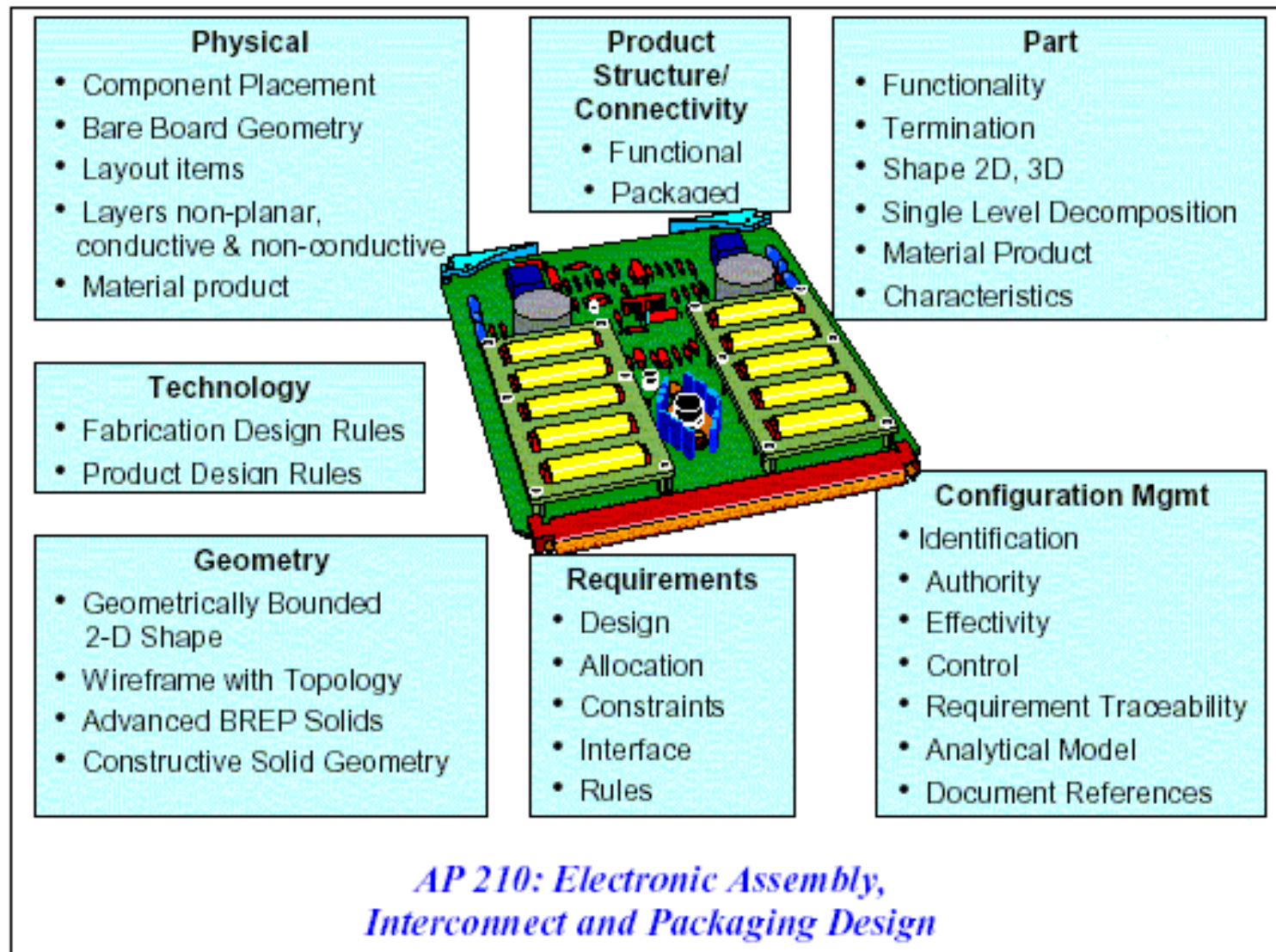
- ◆ Building Element Shape Design
- ◆ Ship Building
- ◆ Building Construction Core Model
- ◆ Building Services: Heating, Ventilation and AC
- ◆ Building Structural Frame: Steelwork

## ■ Process Plants

- ◆ Plant Spatial Configuration
- ◆ Plant Functional Design
- ◆ Process Engineering

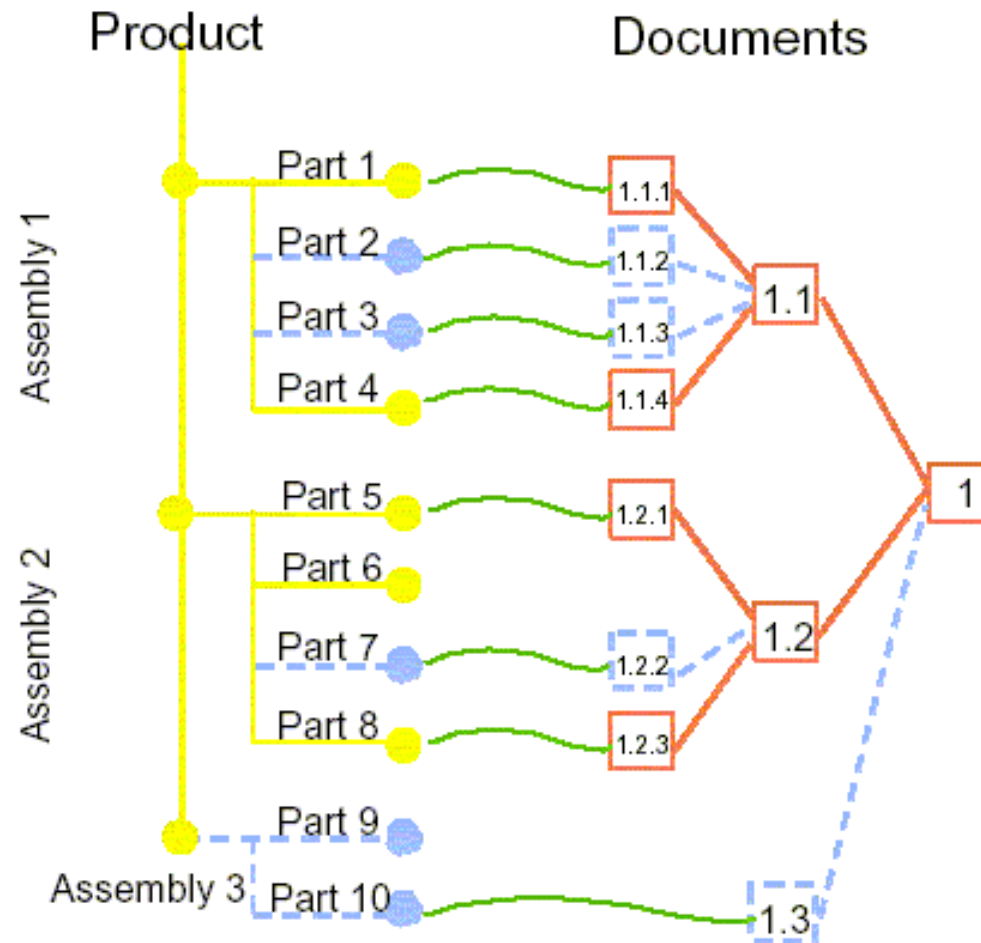
## AP214 - Core data for automotive mechanical design processes





# PRODUCT BOM

# DOCUMENT BOM

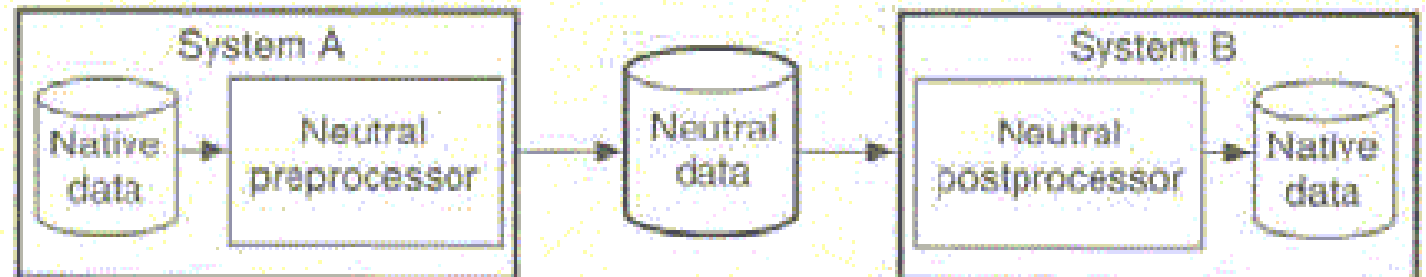


**WRONG**



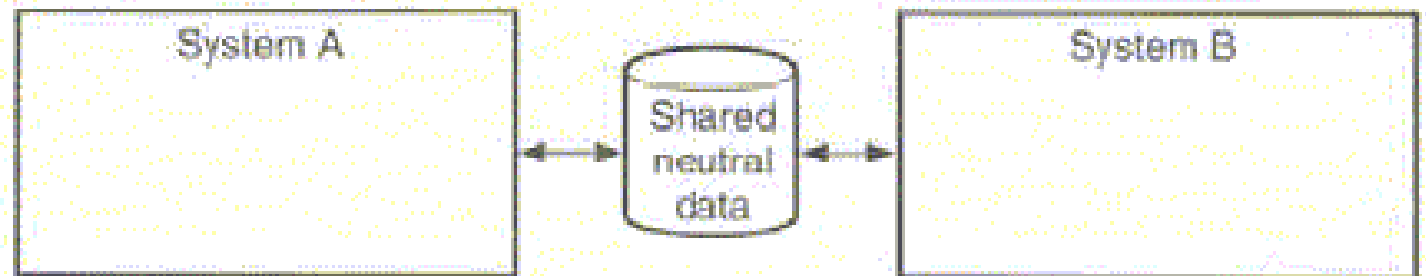
(a)

**TODAY**

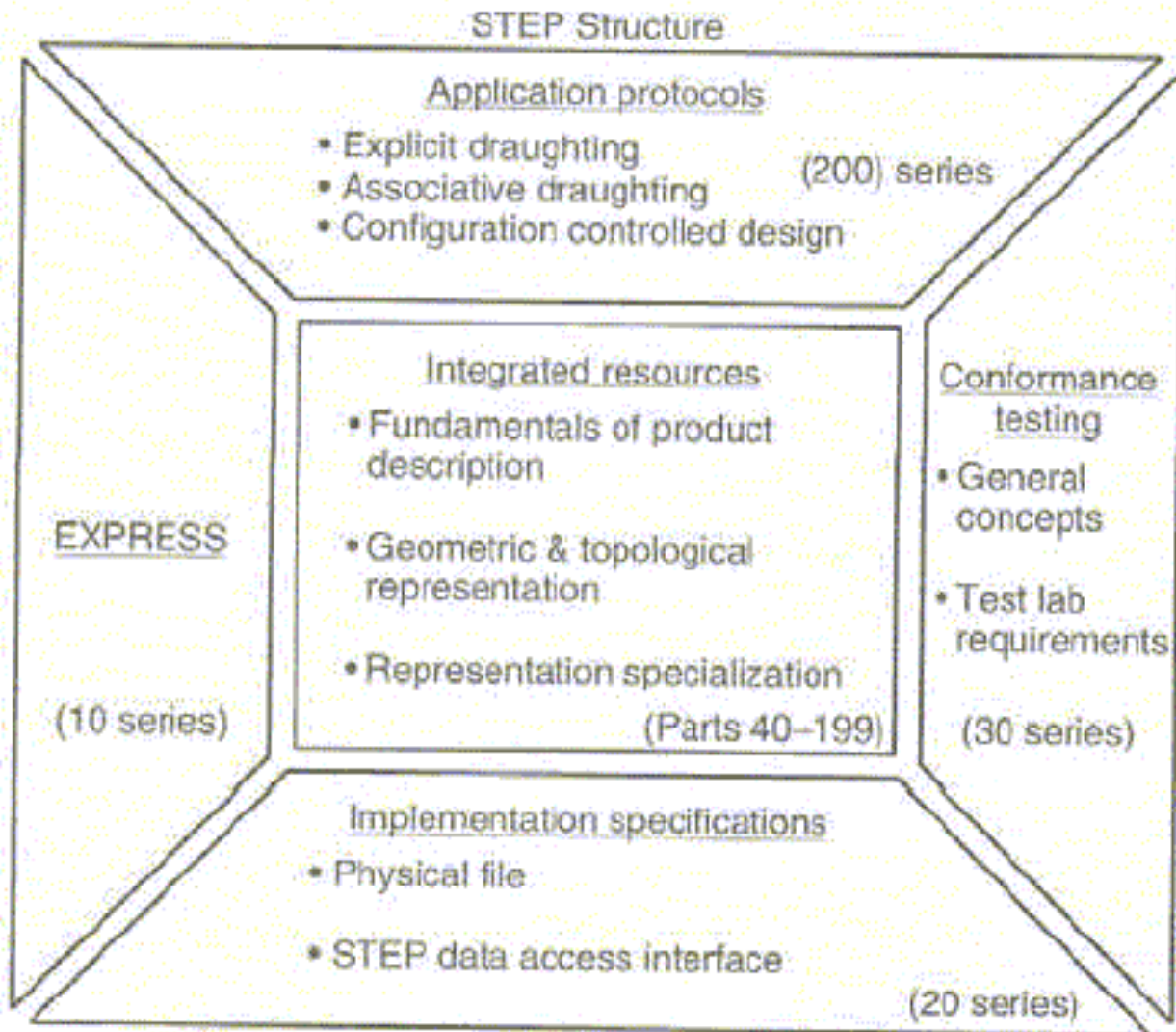


(b)

**FUTURE**



(c)





## STEP Series

**The 10-Series.** The 10-series parts comprise the computer-interpretable area of STEP. This area allows all users to operate by the same guidelines and rules necessary to maintain consistent, accurate data exchange. EXPRESS is the data modeling language used to make STEP computer-interpretable. This language can be compiled to produce "C" structures, SQL statements, or other similar types of information. This language is an important advantage of STEP over IGES, which offers nothing comparable.

**The 20-Series.** The 20-series parts define the physical file and database-sharing exchange area and are the enabling tools for STEP data transfer.

**The 30-Series.** The 30-series parts define conformance testing requirements and are used for data and application verification.

# STEP Series

## Solid Modeling and Product Data Exchange Using STEP (PDES/STEP)

---

**The 40-Series.** The 40-series parts are considered to be the bread and butter of STEP. These parts contain such generic resource information as raw geometry and display attributes, among other things. These and the 100-series parts are the tools used to create application protocols (APs).

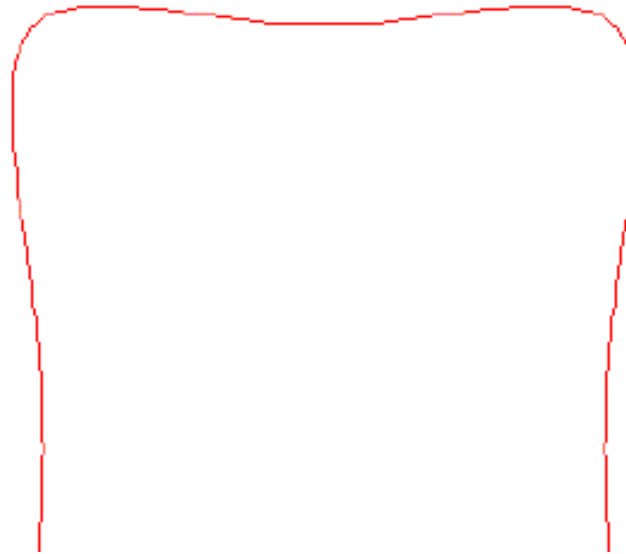
**The 100-Series.** The 100-series parts are similar in concept to the 40-series parts in their use to create application protocols. The difference between the two is that the 100-series is specific to an application area.

For example, multiple electronic APs may use the concept of a drilled hole in a circuit board. The 40-series parts do not contain anything as specific as this, but a 100-series part could be created that defines a drilled hole. Then, the multiple electronic APs would reference this 100-series part. This avoids multiple definitions of the same physical component in similar APs.

**The 200-Series.** The APs in the 200-series parts are where STEP meets the real world. They are specific applications of this technology in various industries. Today, the automotive, sheet-metal, and electronic industries are just a few examples of companies defining how STEP will be used in their specific applications. As this technology is more widely implemented, this area of STEP will be in a constant state of change and growth as more and different applications are developed.

# IGES Entity Type 126:0

## Rational B-Spline Curve - Form 0



### IGES File:

```

S 1
1H,,1H;;,7H126-000,11H126-000.IGS,9H{unknown},9H{unknown},32,38,15,308, G 1
15,7H126-000,1.,1,4HINCH,8,0.016,15H19970830.164828,0.0001,0., G 2
21Hdennette@wiz-worx.com,23HLegacy PDD AP Committee,11,3, G 3
13H920717.080000,23HMIL-PRF-28000B0,CLASS 1; G 4
126 1 0 1 0 0 0 000000001D 1
126 0 2 5 0 D 2
126,8,3,1,0,1,0,0.,0.,0.,0.,1.,2.,3.,4.,5.,6.,6.,6.,6.,1.,1.,1., 1P 1
1.,1.,1.,1.,1.,1.,7.,7.,0.,7.01111,7.15385,0.,7.03333,7.46154, 1P 2
0.,6.86667,8.15385,0.,7.5,7.92308,0.,8.133330000000001,8.15385, 1P 3
0.,7.96667,7.46154,0.,7.98889,7.15385,0.,8.,7.,0.,0.,6.,0.,0., 1P 4
1.; 1P 5
S 1G 4D 2P 5 T 1
```

The last DE number in '...\igs\126-000.IGS' is 1

\*\*\*\*\* Global Section \*\*\*\*\*

Parameter Delimiter Character = ","  
Record Delimiter Character = ";"  
Product ID from Sender = "126-000"  
File Name = "126-000.IGS"  
System ID = "{unspecified}"  
Pre-processor Version = "IGESXTRACT(tm) Version 5.3 (Jul 29 1997)"  
Number of Bits for Integers = 32  
Single Precision Magnitude = 38  
Single Precision Significance = 15  
Double Precision Magnitude = 308  
Double Precision Significance = 15  
Product ID for Receiver = "126-000"  
Model Space Scale = 1.00000  
Unit Flag = 1 - Inches  
Units = "INCH"  
Maximum Number of Line Weights = 8  
Size of Maximum Line Width = 0.0160000  
Date & Time Stamp = "970731.034906"  
Minimum User-intended Resolution = 0.000100000  
Approximate Maximum Coordinate = 0.000000  
Name of Author = "dennette@wiz-worx.com"  
Author's Organization = "Legacy PDD AP Committee"  
IGES Version Number = 11 - USPRO/IPO-100-1996 (IGES 5.3)  
Drafting Standard Code = 0 - None Specified  
Model Creation/Change Date = "920717.080000"  
Application Protocol/Subset ID = <default>

\*\*\*\*\* Terminate Section \*\*\*\*\*

1 records in Start Section  
4 records in Global Section  
2 records in Directory Entry Section (1 entities)  
5 records in Parameter Data Section

\*\*\*\*\* DE = 1 \*\*\*\*\*

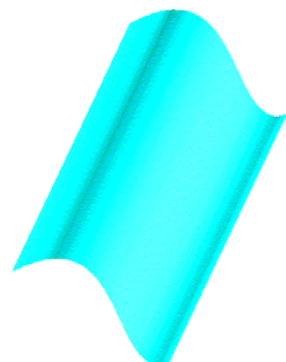
Entity Type Number = 126 - Rational B-Spline Curve  
Parameter Data (Count) = 1 (5)  
Structure = 0  
Line Font Pattern = 1 - Solid  
Level = 0  
View = 0  
Transformation Matrix = 0  
Label Display = 0  
Blank Status = 0 - Visible  
Subord. Entity Switch = 0 - Independent  
Entity Use Flag = 0 - Geometry  
Hierarchy = 1 - Global defer  
Line Weight Number = 0  
Color Number = 2 - Red  
Form Number = 0 - <computationally derived>  
Entity Label (Subscript) = <default> (<default>)

\*\*\*\*\* Parameter Data Record \*\*\*\*\*

K	=	8
M	=	3
PROP1	=	1 - Planar
PROP2	=	0 - Open Curve
PROP3	=	1 - Polynomial
PROP4	=	0 - Non-periodic
T(-3)	=	0.000000
T(-2)	=	0.000000
T(-1)	=	0.000000
T(0)	=	0.000000
T(1)	=	1.00000
T(2)	=	2.00000
T(3)	=	3.00000
T(4)	=	4.00000
T(5)	=	5.00000
T(6)	=	6.00000
T(7)	=	6.00000
T(8)	=	6.00000
T(9)	=	6.00000
W(0)	=	1.00000
W(1)	=	1.00000
W(2)	=	1.00000
W(3)	=	1.00000
W(4)	=	1.00000
W(5)	=	1.00000
W(6)	=	1.00000
W(7)	=	1.00000
W(8)	=	1.00000



XYZ(0)	=	7.00000,	7.00000,	0.000000
XYZ(1)	=	7.01111,	7.15385,	0.000000
XYZ(2)	=	7.03333,	7.46154,	0.000000
XYZ(3)	=	6.86667,	8.15385,	0.000000
XYZ(4)	=	7.50000,	7.92308,	0.000000
XYZ(5)	=	8.13333,	8.15385,	0.000000
XYZ(6)	=	7.96667,	7.46154,	0.000000
XYZ(7)	=	7.98889,	7.15385,	0.000000
XYZ(8)	=	8.00000,	7.00000,	0.000000
V(0)	=	0.000000		
V(1)	=	6.00000		
NORM	=	0.000000,	0.000000,	1.00000
***** Associativities & Properties *****				
NA	=	<default>		
NP	=	<default>		



# IGES Entity Type 128:0

## Rational B-Spline Surface - Form 0

```

S 1
1H,,1H;;7H128-000,11H128-000.IGS,9H{unknown},9H{unknown},16,6,15,13,15, G 1
7H128-000,1.,1,4HINCH,8,0.016,15H19970830.165254,0.0001,0., G 2
21Hdennette@wiz-worx.com,23HLegacy PDD AP Committee,11,3, G 3
13H920717.080000,23HMIL-PRF-28000B0,CLASS 1; G 4
128 1 0 1 0 0 0 000000001D 1
128 0 2 17 0 D 2
128,3,7,3,5,0,0,1,0,0,0.,0.,0.,0.,1.,1.,1.,1.,0.,0.,0.,0.,0.,0., 1P 1
1.,2.,3.,3.,3.,3.,3.,3.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1., 1P 2
1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,8.5, 1P 3
9.5,1.,8.5,9.25,0.666667,8.5,9.,0.333333,8.5,8.75,0.,8.49394, 1P 4
9.4465900000000001,1.,8.49394,9.1965900000000001,0.666667,8.49394, 1P 5
8.9465900000000001,0.333333,8.49394,8.6965900000000001,0., 1P 6
8.4363600000000001,9.39545,1.,8.4363600000000001,9.14546,0.666667, 1P 7
8.4363600000000001,8.89545,0.333333,8.4363600000000001,8.64545,0., 1P 8
8.21364,9.485799999999999,1.,8.21364,9.235799999999999,0.666667, 1P 9
8.21364,8.985799999999999,0.333333,8.21364,8.735799999999999,0., 1P 10
7.78636,9.82671,1.,7.78637,9.57671,0.666667,7.78637,9.32671, 1P 11
0.333333,7.78636,9.07671,0.,7.56363,9.804539999999999,1., 1P 12
7.56363,9.554539999999999,0.666667,7.56363,9.304539999999999, 1P 13
0.333333,7.56363,9.054539999999999,0.,7.50606,9.628410000000001, 1P 14
1.,7.50606,9.378410000000001,0.666667,7.50606,9.128410000000001, 1P 15
0.333333,7.50606,8.878410000000001,0.,7.5,9.5,1.,7.5,9.25, 1P 16
0.666667,7.5,9.,0.333333,7.5,8.75,0.,0.,1.,0.,3.; 1P 17
S 1G 4D 2P 17 T 1

```

The last DE number in '.\igs\128-000.IGS' is 1

\*\*\*\*\* Global Section \*\*\*\*\*

Parameter Delimiter Character = ","  
Record Delimiter Character = ";"  
Product ID from Sender = "NENTITY"  
File Name = "128-000.igs"  
System ID = "{unknown}"  
Pre-processor Version = "{unknown}"  
Number of Bits for Integers = 16  
Single Precision Magnitude = 6  
Single Precision Significance = 15  
Double Precision Magnitude = 13  
Double Precision Significance = 15  
Product ID for Receiver = "NENTITY"  
Model Space Scale = 1.00000  
Unit Flag = 1 - Inches  
Units = "INCH"  
Maximum Number of Line Weights = 8  
Size of Maximum Line Width = 0.0160000  
Date & Time Stamp = "19970830.165254"  
Minimum User-intended Resolution = 0.000100000  
Approximate Maximum Coordinate = 0.000000  
Name of Author = "dennette@wiz-worx.com"  
Author's Organization = "Legacy PDD AP Committee"  
IGES Version Number = 10 - USPRO/IPO-100 (IGES 5.2) [USPRO93]  
Drafting Standard Code = 3 - ANSI  
Model Creation/Change Date = "920717.080000"

\*\*\*\*\* Terminate Section \*\*\*\*\*

1 records in Start Section  
4 records in Global Section  
2 records in Directory Entry Section (1 entities)  
17 records in Parameter Data Section

\*\*\*\*\* DE = 1 \*\*\*\*\*

Entity Type Number = 128 - Rational B-Spline Surface  
Parameter Data (Count) = 1 (17)  
Structure = 0  
Line Font Pattern = 1 - Solid  
Level = 0  
View = 0  
Transformation Matrix = 0  
Label Display = 0  
Blank Status = 0 - Visible  
Subord. Entity Switch = 0 - Independent  
Entity Use Flag = 0 - Geometry  
Hierarchy = 1 - Global defer  
Line Weight Number = 0  
Color Number = 2 - Red  
Form Number = 0 - <computationally derived>  
Entity Label (Subscript) = <default> (<default>)

\*\*\*\*\* Parameter Data Record \*\*\*\*\*

K1	=	3	W(0,0)	=	1.00000
K2	=	7	W(1,0)	=	1.00000
M1	=	3	W(2,0)	=	1.00000
M2	=	5	W(3,0)	=	1.00000
PROP1	=	0 - Not Closed	W(0,1)	=	1.00000
PROP2	=	0 - Not Closed	W(1,1)	=	1.00000
PROP3	=	1 - Polynomial	W(2,1)	=	1.00000
PROP4	=	0 - Non-periodic in 1st direction	W(3,1)	=	1.00000
PROP5	=	0 - Non-periodic in 2nd direction	W(0,2)	=	1.00000
S(-3)	=	0.000000	W(1,2)	=	1.00000
S(-2)	=	0.000000	W(2,2)	=	1.00000
S(-1)	=	0.000000	W(3,2)	=	1.00000
S(0)	=	0.000000	W(0,3)	=	1.00000
S(1)	=	1.00000	W(1,3)	=	1.00000
S(2)	=	1.00000	W(2,3)	=	1.00000
S(3)	=	1.00000	W(3,3)	=	1.00000
S(4)	=	1.00000	W(0,4)	=	1.00000
T(-5)	=	0.000000	W(1,4)	=	1.00000
T(-4)	=	0.000000	W(2,4)	=	1.00000
T(-3)	=	0.000000	W(3,4)	=	1.00000
T(-2)	=	0.000000	W(0,5)	=	1.00000
T(-1)	=	0.000000	W(1,5)	=	1.00000
T(0)	=	0.000000	W(2,5)	=	1.00000
T(1)	=	1.00000	W(3,5)	=	1.00000
T(2)	=	2.00000	W(0,6)	=	1.00000
T(3)	=	3.00000	W(1,6)	=	1.00000
T(4)	=	3.00000	W(2,6)	=	1.00000
T(5)	=	3.00000	W(3,6)	=	1.00000
T(6)	=	3.00000	W(0,7)	=	1.00000
T(7)	=	3.00000	W(1,7)	=	1.00000
T(8)	=	3.00000	W(2,7)	=	1.00000
			W(3,7)	=	1.00000

XYZ(0,0)	=	8.50000,	9.50000,	1.00000
XYZ(1,0)	=	8.50000,	9.25000,	0.666667
XYZ(2,0)	=	8.50000,	9.00000,	0.333333
XYZ(3,0)	=	8.50000,	8.75000,	0.000000
XYZ(0,1)	=	8.49394,	9.44659,	1.00000
XYZ(1,1)	=	8.49394,	9.19659,	0.666667
XYZ(2,1)	=	8.49394,	8.94659,	0.333333
XYZ(3,1)	=	8.49394,	8.69659,	0.000000
XYZ(0,2)	=	8.43636,	9.39545,	1.00000
XYZ(1,2)	=	8.43636,	9.14546,	0.666667
XYZ(2,2)	=	8.43636,	8.89545,	0.333333
XYZ(3,2)	=	8.43636,	8.64545,	0.000000
XYZ(0,3)	=	8.21364,	9.48580,	1.00000
XYZ(1,3)	=	8.21364,	9.23580,	0.666667
XYZ(2,3)	=	8.21364,	8.98580,	0.333333
XYZ(3,3)	=	8.21364,	8.73580,	0.000000
XYZ(0,4)	=	7.78636,	9.82671,	1.00000
XYZ(1,4)	=	7.78637,	9.57671,	0.666667
XYZ(2,4)	=	7.78637,	9.32671,	0.333333
XYZ(3,4)	=	7.78636,	9.07671,	0.000000
XYZ(0,5)	=	7.56363,	9.80454,	1.00000
XYZ(1,5)	=	7.56363,	9.55454,	0.666667
XYZ(2,5)	=	7.56363,	9.30454,	0.333333
XYZ(3,5)	=	7.56363,	9.05454,	0.000000
XYZ(0,6)	=	7.50606,	9.62841,	1.00000
XYZ(1,6)	=	7.50606,	9.37841,	0.666667
XYZ(2,6)	=	7.50606,	9.12841,	0.333333
XYZ(3,6)	=	7.50606,	8.87841,	0.000000
XYZ(0,7)	=	7.50000,	9.50000,	1.00000
XYZ(1,7)	=	7.50000,	9.25000,	0.666667
XYZ(2,7)	=	7.50000,	9.00000,	0.333333
XYZ(3,7)	=	7.50000,	8.75000,	0.000000

U(0)	=	0.000000
U(1)	=	1.00000
V(0)	=	0.000000
V(1)	=	3.00000

\*\*\*\*\* Associativities & Properties \*\*\*\*\*

NA	=	<default>
NP	=	<default>

# **CAD HARDWARE – COMPUTER GRAPHICS**

## **•WORKSTATIONS**

- Vector Display**
- Raster Scan Display**
- Flat Panel Display**
- Mouse Locator Selector**
- Light pen**

## **•PLOTTERS**

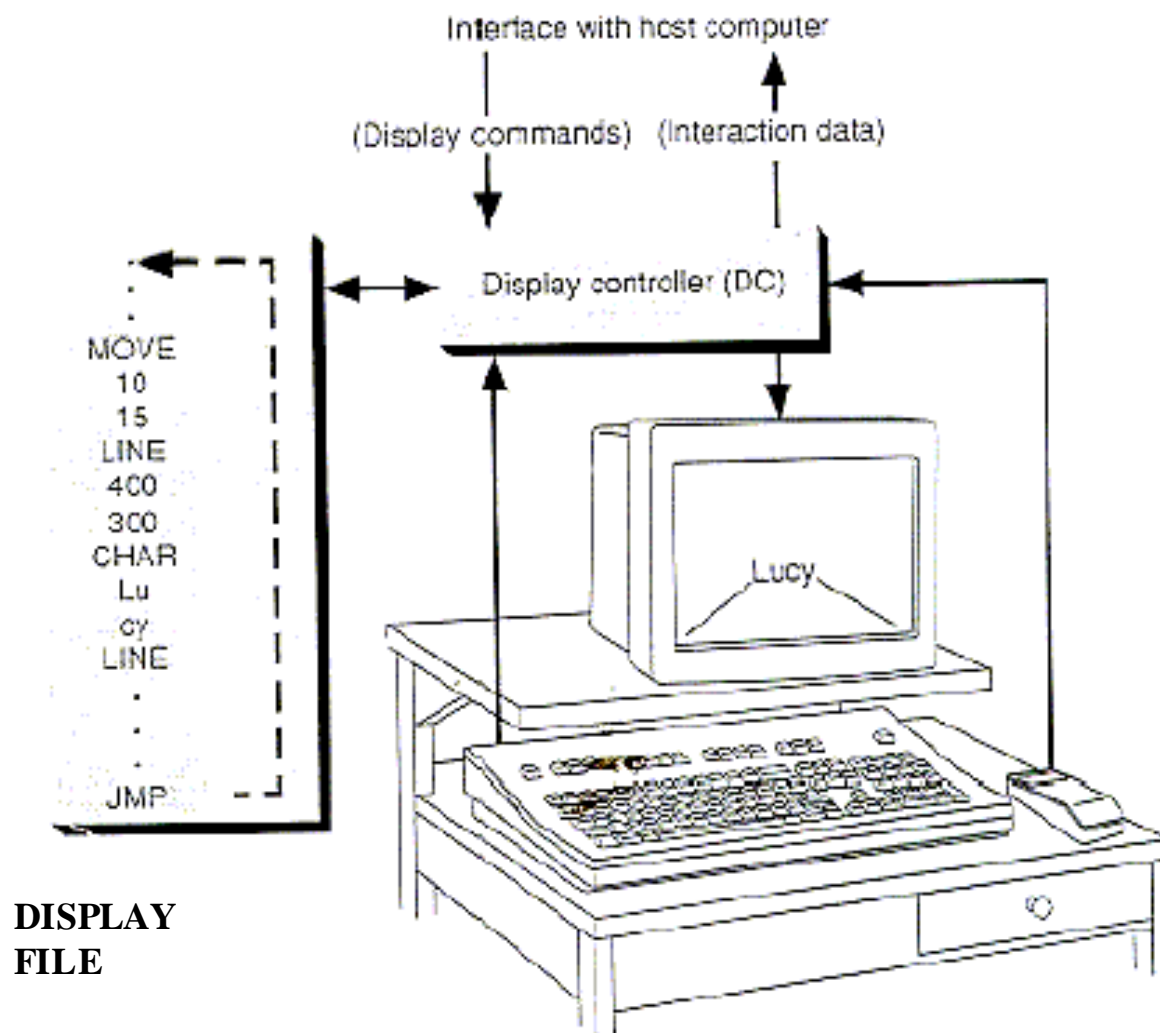
- Pen Plotter – Vector Plotter**
- LASER Printer – Raster printer/Plotter**
- Inkjet Printer – Raster printer/Plotter**
- Electro-Static Printer – Raster printer/Plotter**

## **•SCANNERS**

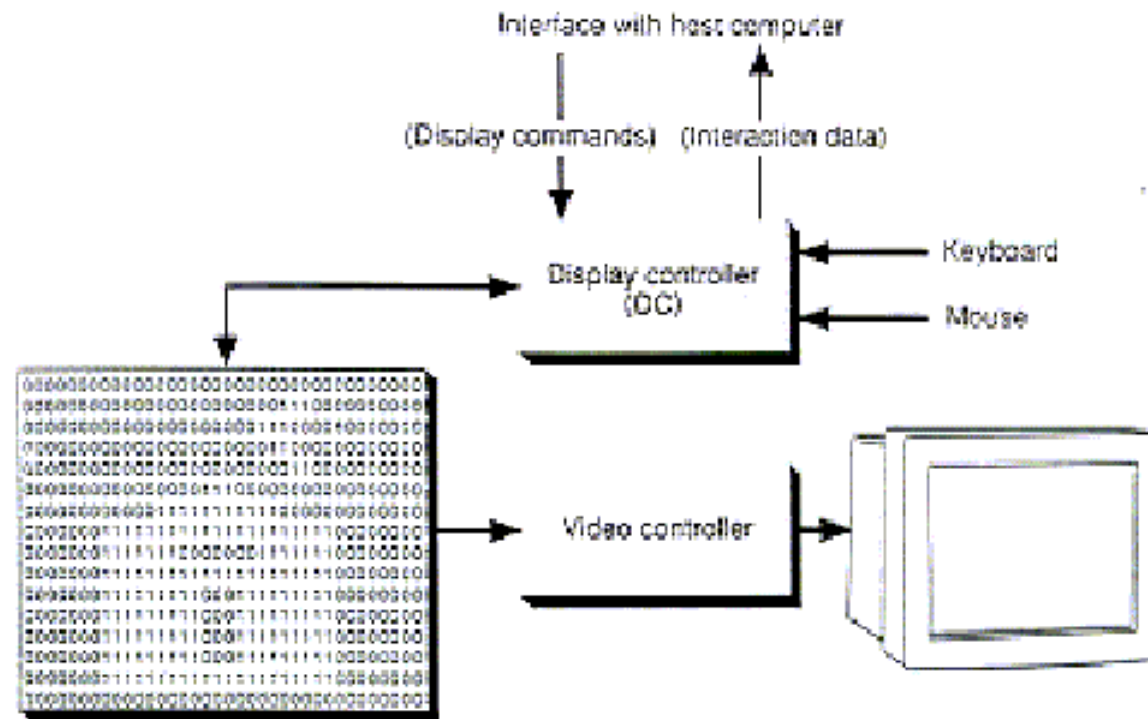
## **•DIGITIZERS - 2D/3D**



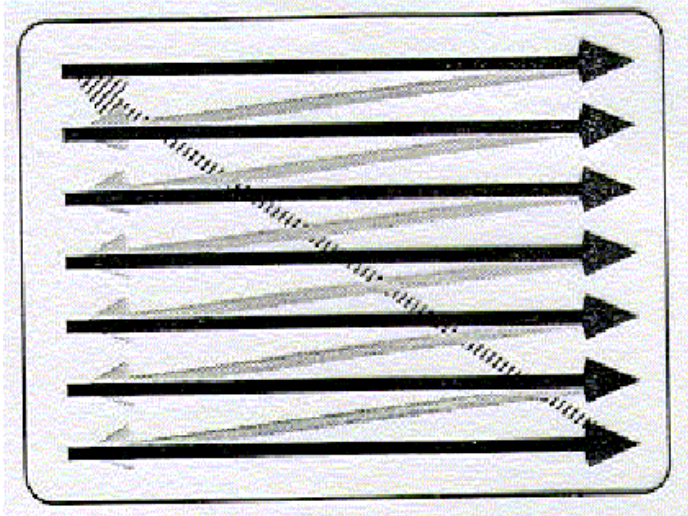
# VECTOR DISPLAY



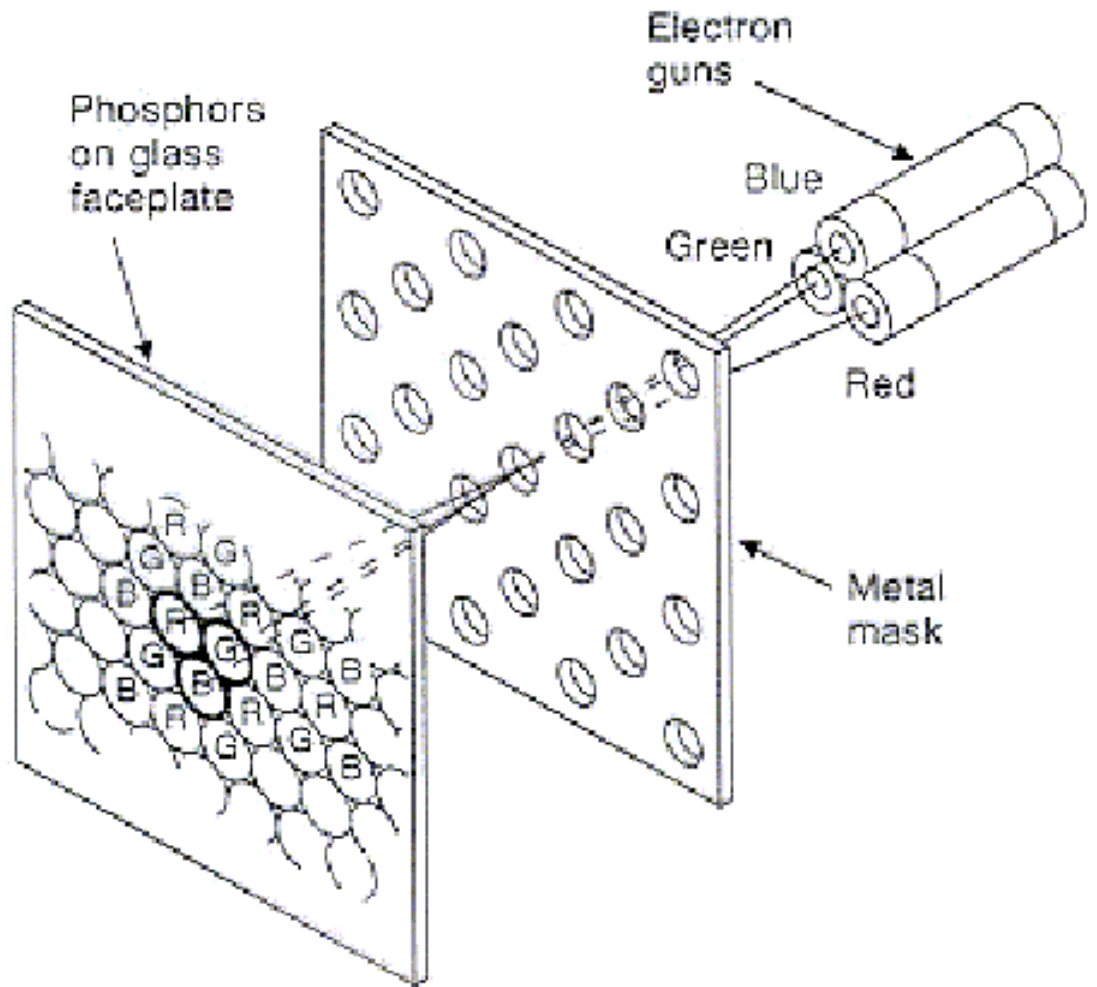
# RASTER DISPLAY



## FRAME BUFFER



**SCAN LINES**



**COLOR RASTER  
DISPLAY**



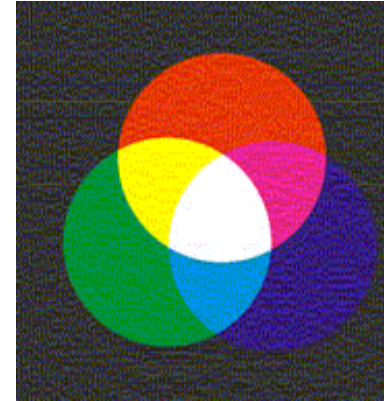
Subtractive Colors

Cyan

Magenta

Yellow

Used For Printers  
(White Background)



Additive Colors

Red

Green

Blue

Used for displays  
Black Background

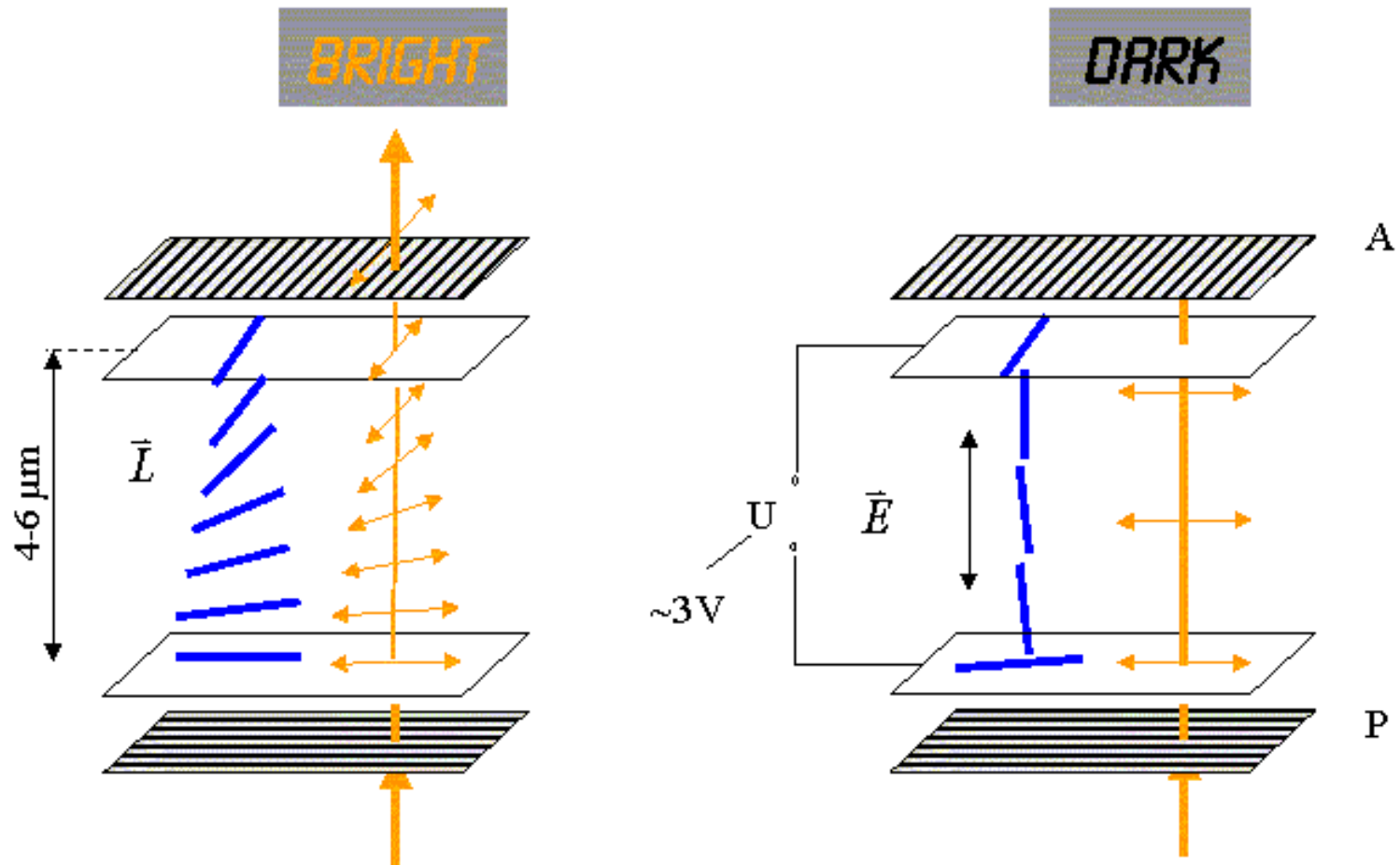
## Colour perception

Visible light falls between 380nm (violet) and 780nm (red) on the electromagnetic spectrum, sandwiched between ultraviolet and infrared. White light comprises approximately equal proportions of all the visible wavelengths, and when this shine on or through an object, some wavelengths are absorbed and others are reflected or transmitted. It's the reflected or transmitted light that gives the object its perceived colour. Leaves, for example, are their familiar colour because chlorophyll absorbs light at the blue and red ends of the spectrum and reflects the green part in the middle.

The "temperature" of the light source, measured in Kelvin (K), affects an object's perceived colour. White light, as emitted by the fluorescent lamps in a viewing box or by a photographer's flashlight, has an even distribution of wavelengths, corresponding to a temperature of around 6,000K, and doesn't distort colours. Standard light bulbs, however, emit less light from the blue end of the spectrum, corresponding to a temperature of around 3,000K, and cause objects to appear more yellow.

Humans perceive colour via a layer of light-sensitive cells on the back of the eye called the retina. The key retinal cells are the cones that contain photo-pigments that render them sensitive to red, green or blue light (the other light-sensitive cells, the rods, are only activated in dim light). Light passing through the eye is regulated by the iris and focused by the lens onto the retina, where cones are stimulated by the relevant wavelengths. Signals from the millions of cones are passed via the optic nerve to the brain, which assembles them into a colour image.

# LIQUID CRYSTAL DISPLAY

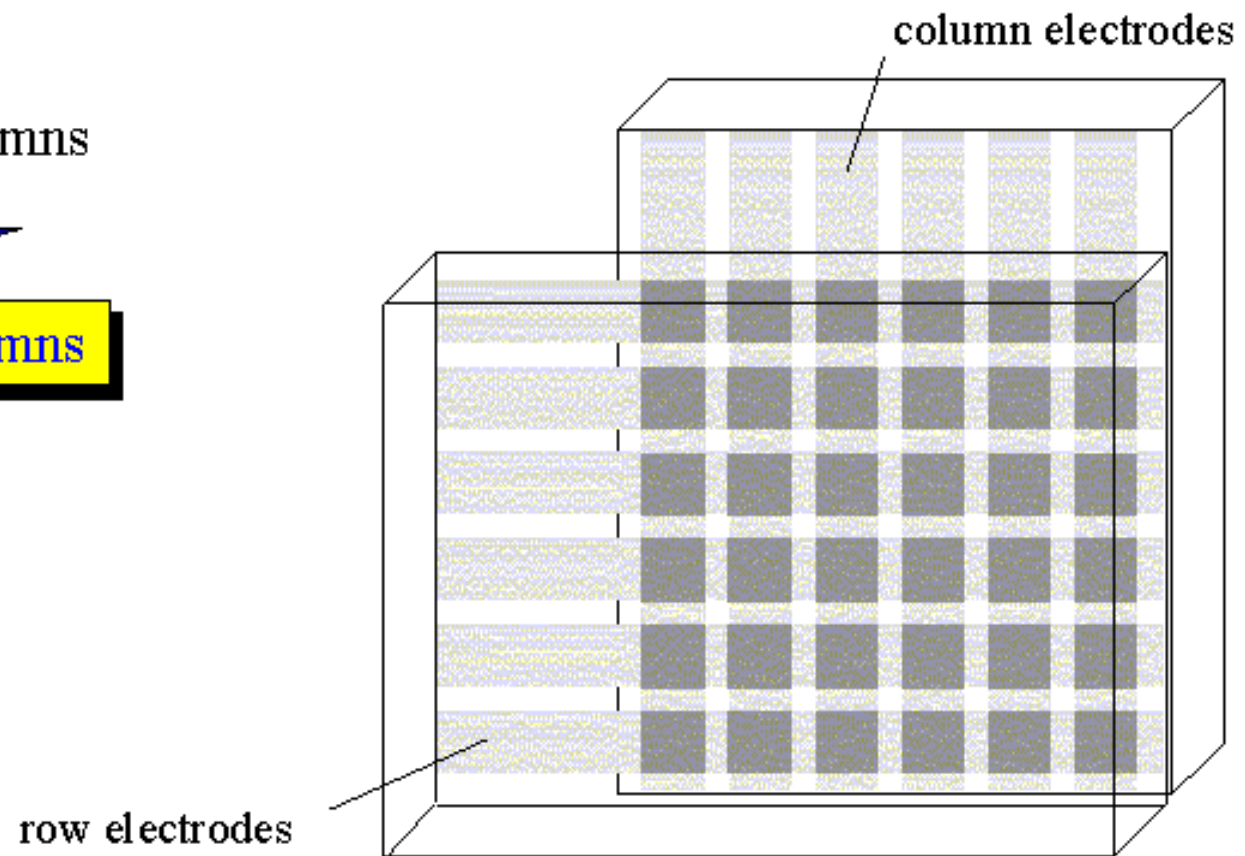


**Twisted Nematic Effect**

rows x columns



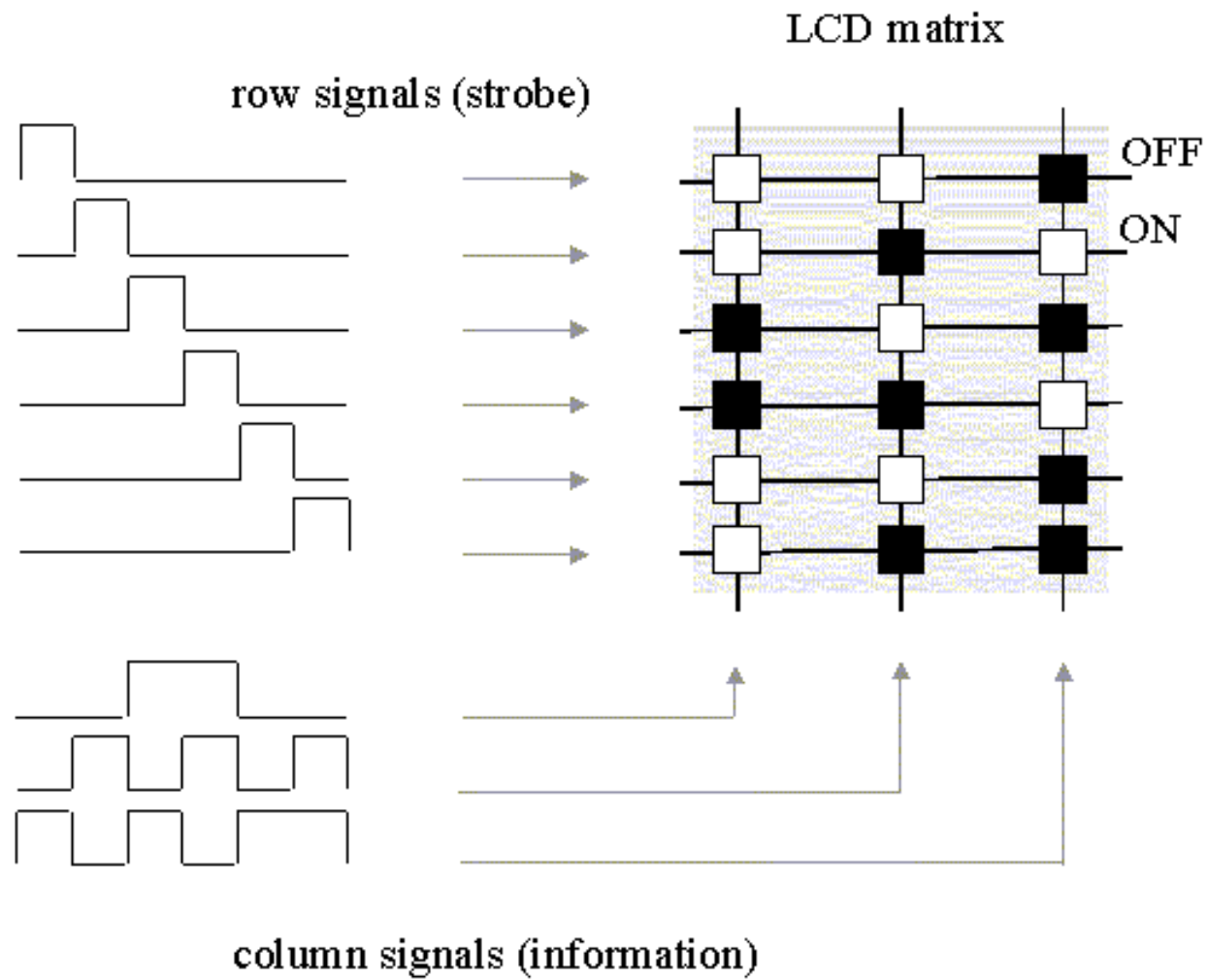
rows + columns



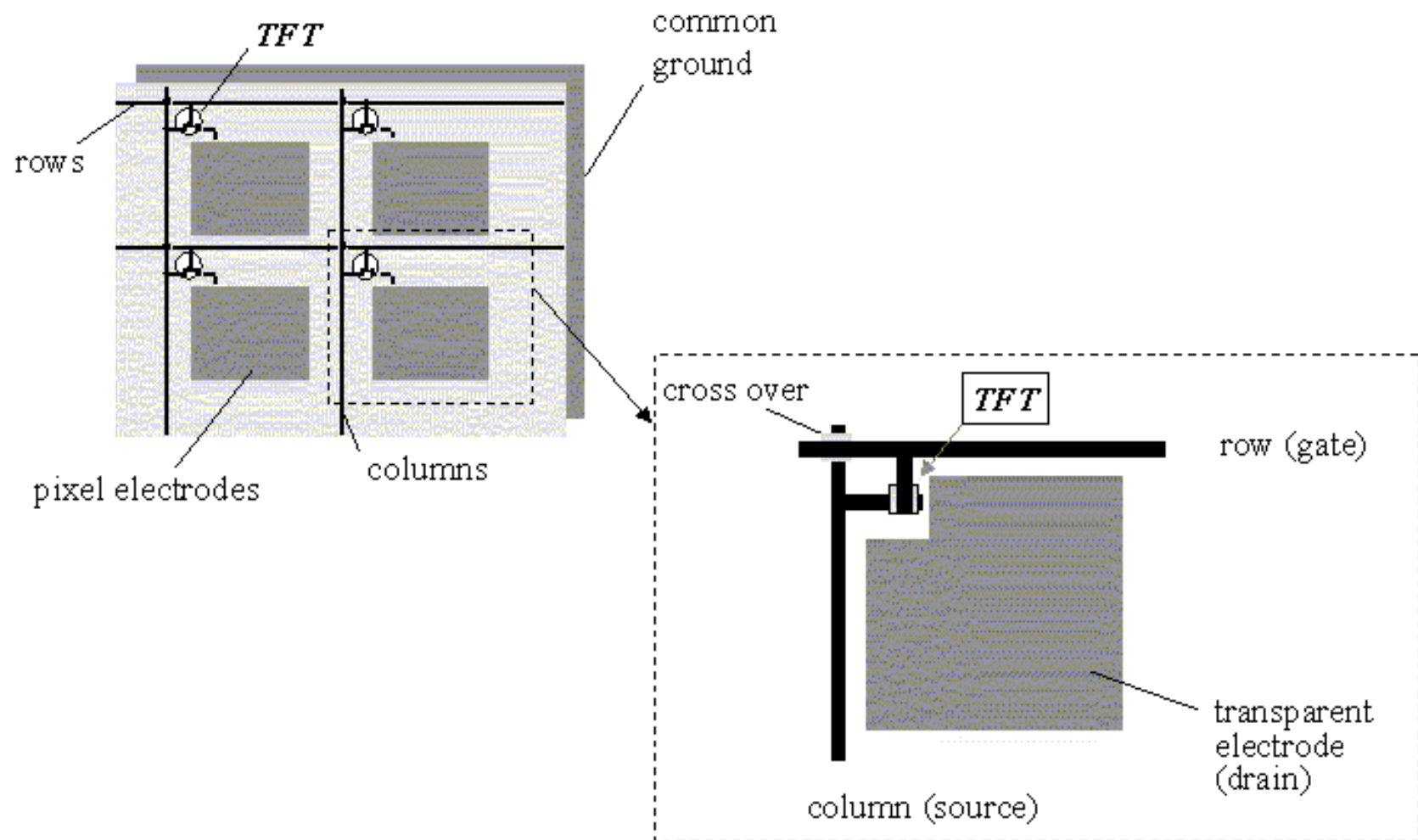
Passive matrix displays



# Time Multiplexing in LCD



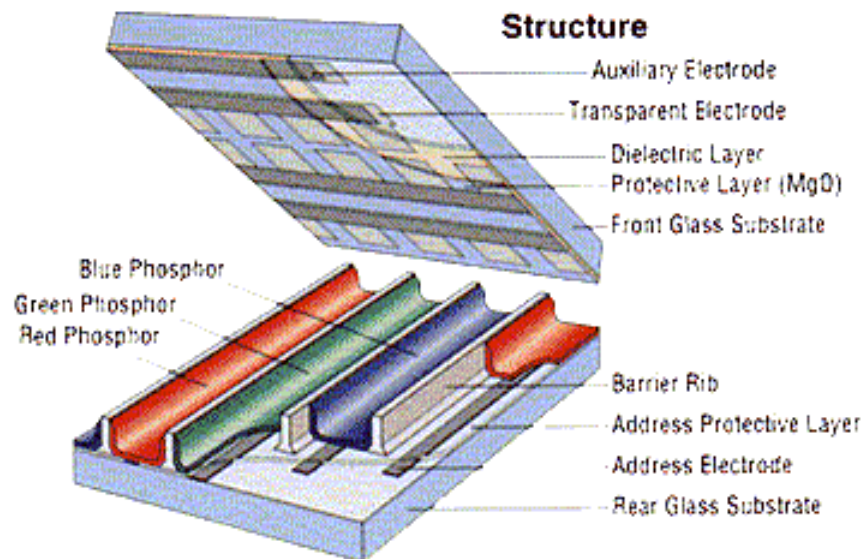
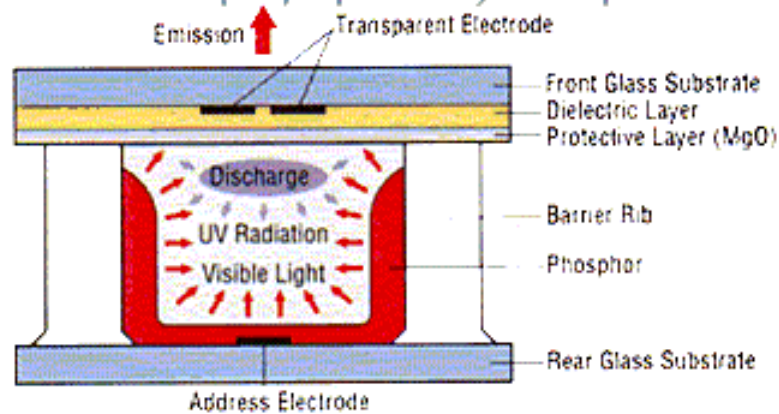
## *Thin-Film-Transistor or active matrix displays*

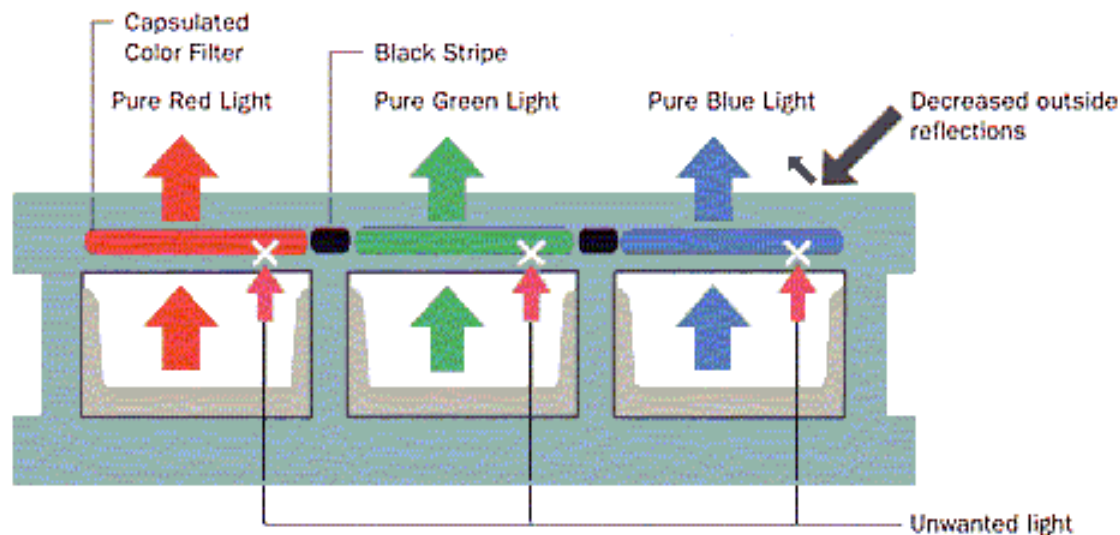


Thin-Film-Transistor or active matrix displays

# Illustration of Plasma New Technologies

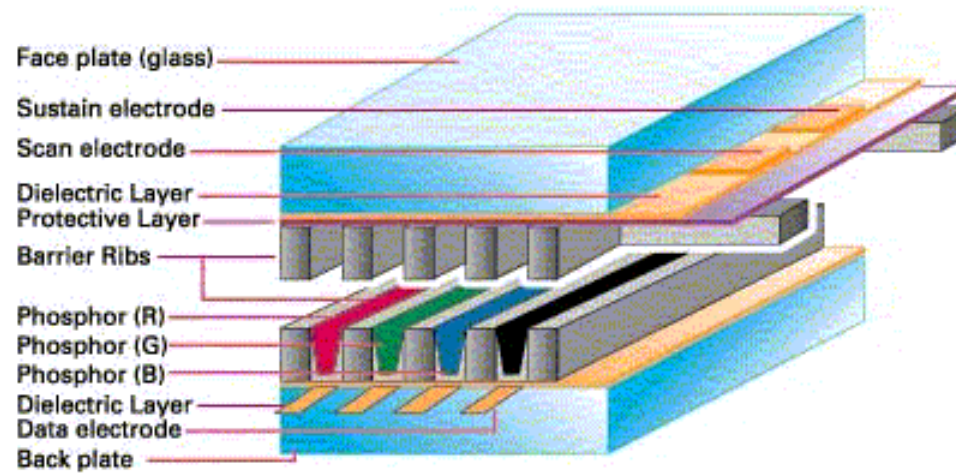
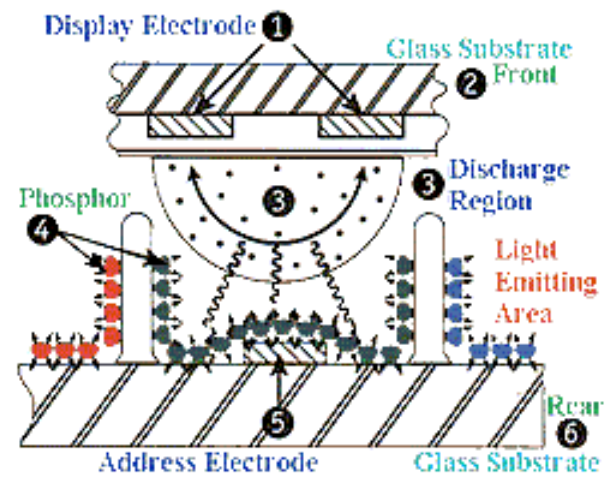
## Plasma Display Operating Principle





**Exclusive Capsulated Color Filter gas plasma technology  
utilizes embedded filters and a black matrix to improve color purity and contrast.**

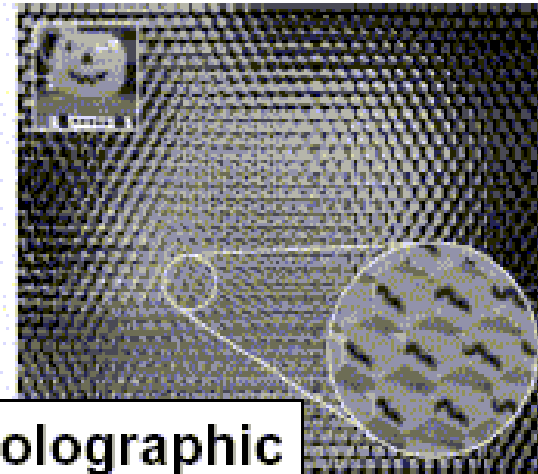
Plasma vision is the latest display technology and the best way to achieve flat panel displays with excellent image quality and large screen sizes that are easily viewable in any environment. Plasma vision is an array or cells, known as pixels, corresponding to the colors red, green, and blue. Gas in a plasma state is used to react with phosphors in each sub pixel to produce colored light (red, green, or blue). Three phosphors are the same types used in Cathode Ray Tube (CRT) devices such as televisions and standard computer monitors. You get the rich, dynamic colors that you expect. Each sub pixel is individually controlled by advanced electronics to produce over 16 million different colors. All of this means that are easily viewable in a display that is less than 6 inches thick.



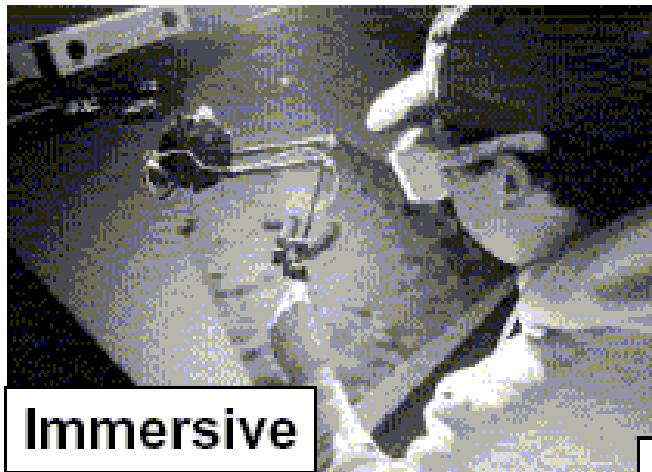
# Display Devices



Plasma



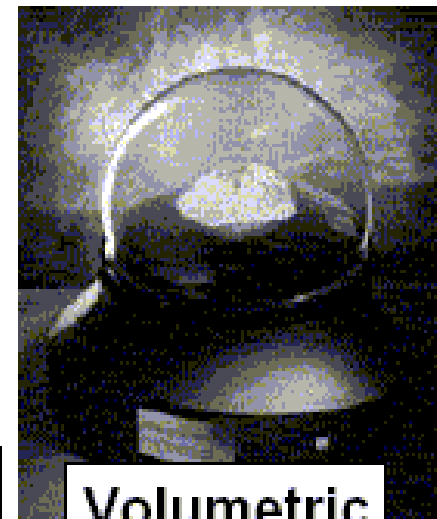
Holographic



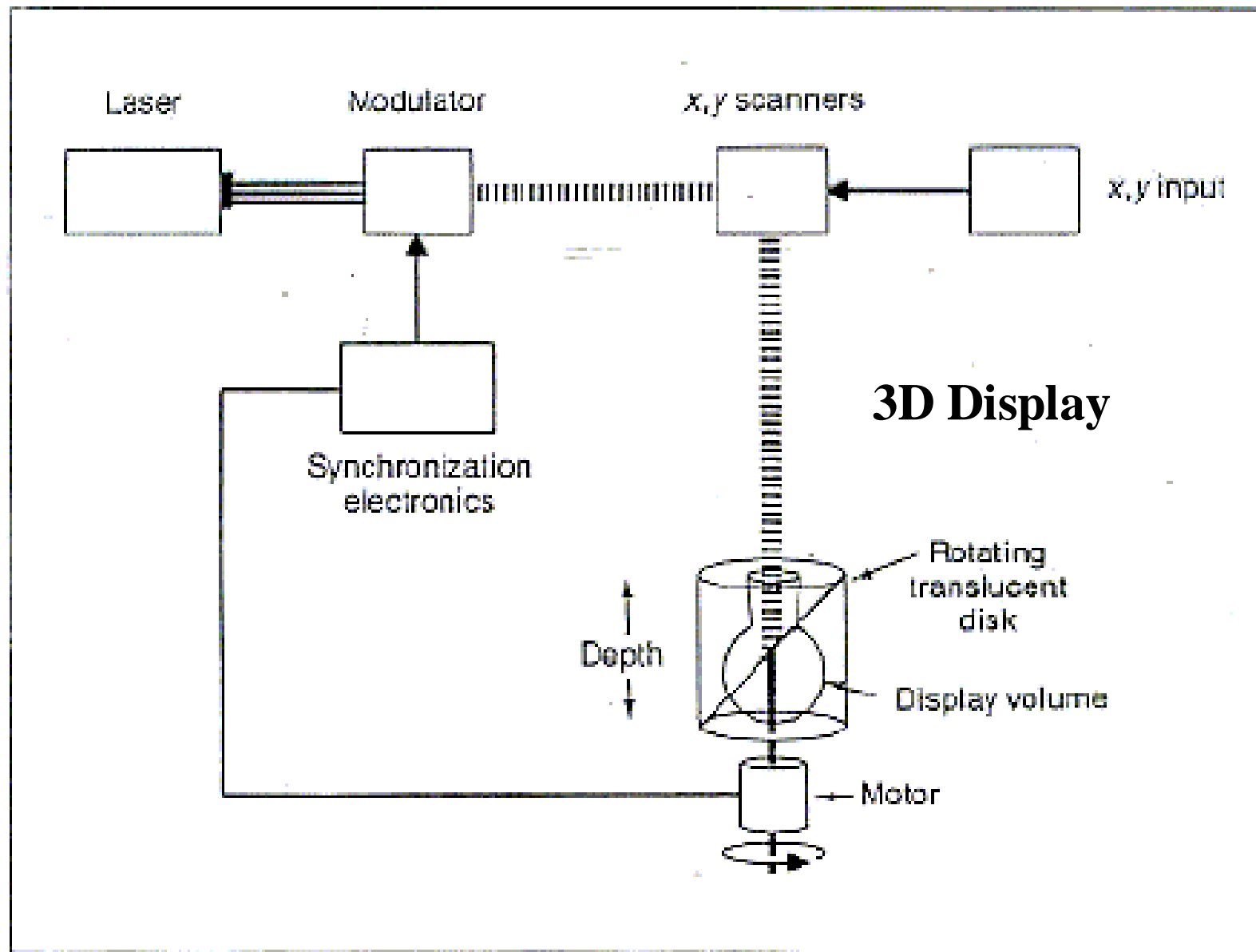
Immersive



Head-mounted



Volumetric



*Will this be the wave of the future? A prototype display system uses a laser and rotating translucent disk to produce the elusive third dimension. (Diagram modified. Original courtesy of Texas Instruments, Inc.)*





# **INTERACTION DEVICES**

**-KEY BOARD**

**-MOUSE = PURE LOCATOR DEVICE**

**-LIGHT PEN = PURE SELECTOR DEVICE**

**KEY = EVENT DRIVEN DEVICE**

**MOUSE = SAMPLED DEVICE**

**LIGHT PEN = EVENT DRIVEN DEVICE**

**TOUCH SCREEN**

# INTERACTION

## USER response to images

- select a particular object on the display
- Specify a position on the display
- Alter an objects orientation, size or location
- Alter new object on the display

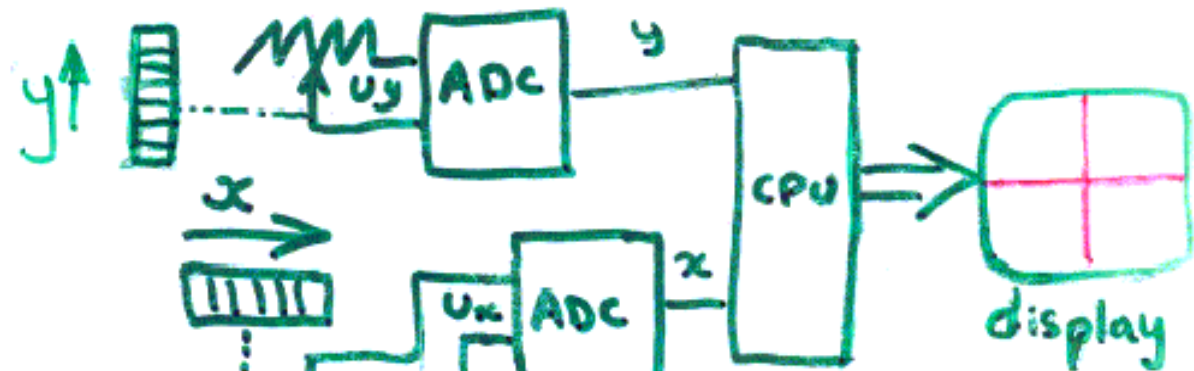
## HARDWARE

2 classes : LOCATORS & SELECTORS

Locators  $\Rightarrow$  devices which give position information  
ex. computer receives  $x, y$  data from  
locator

Selectors  $\Rightarrow$  devices used to select a particular object  
on the display

## Locator devices thumbwheels



feedback  
 $\Rightarrow$  screen cursor

Joystick  
mouse  
Tablet + cursor

enable & disable button

Selector device Light pen  $\Rightarrow$  EVENT DRIVEN

photo cell in pen

allow selection in a refresh display

When the light pen senses the phosphor beneath it illuminated, it can interrupt the display processor's interpreting of the display file  $\Rightarrow$  instruction register tell which display file instruction was interrupted.

once information extracted processor continue

locator can be sampled at any time

The processor must wait on the light pen until it encounters an illuminated pixel (phosphor) on the screen

KEYBOARD  $\Rightarrow$  EVENT DRIVEN : computer must wait for a key to be pressed before it can determine which char.

Two different ways to handle event driven devices

1. Enter a **POLLING LOOP** : a loop which checks the status of the device until the event occurs (e.g. terminal input in high level languages)  
disadvantage during polling processor idle
2. **Enable an interrupt.**  
Interrupt processor from normal execution and transfers control to special interrupt handling routine.

With interrupts events may occur a times other than the processor expects.

Also user may anticipate input. With polling this could result in losing the information. With interrupts the processor could stop and store the information for further use.

### EVENT PROCESSING :

1. When event occurs information is stored in  
EVENT QUEUE

2. When information from event driven device is needed, the event queue is searched to see what events have transpired. if no event has occurred or if event queue is empty a polling loop can still be used to wait for right event

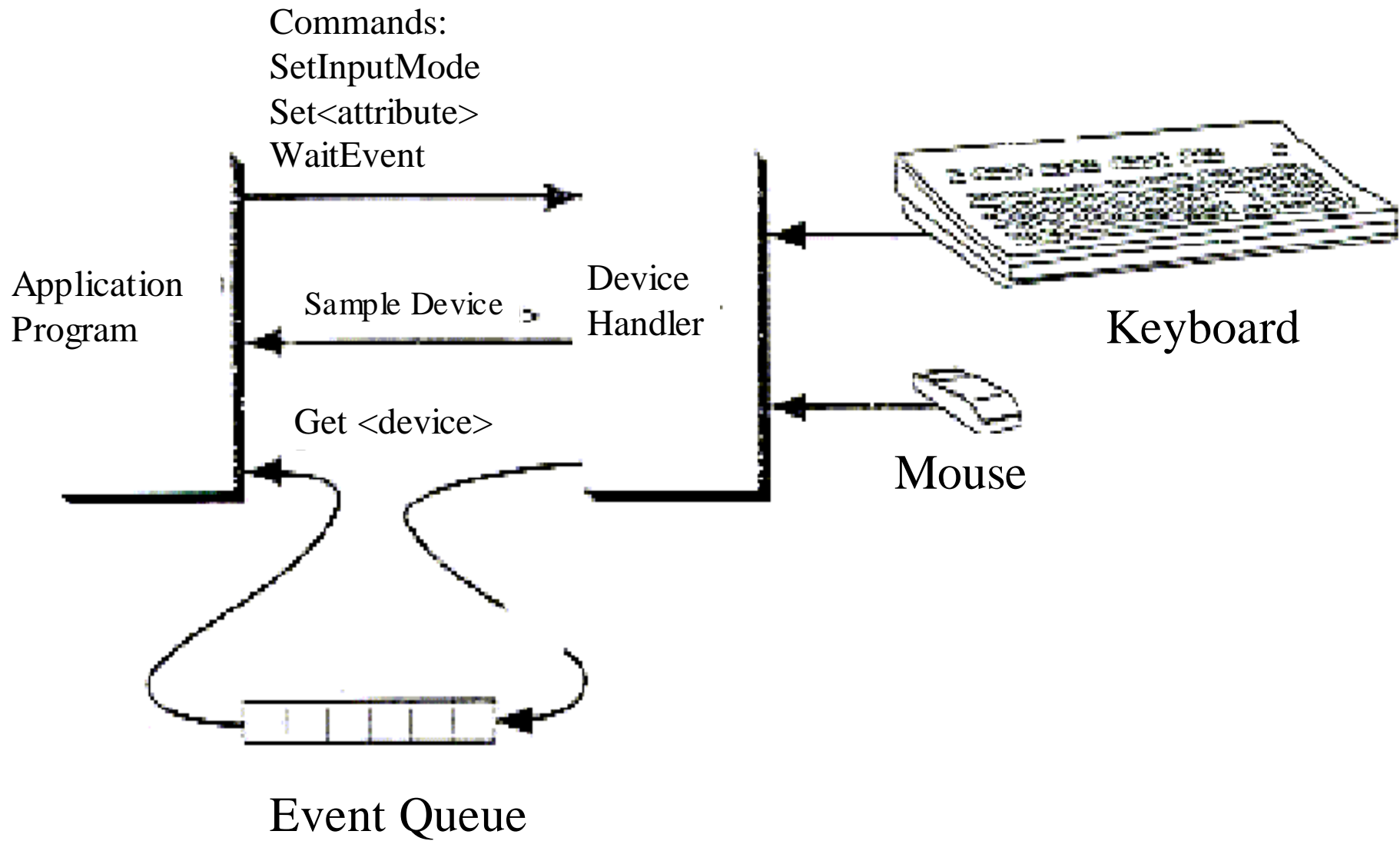


Several event driven devices may be available giving different input data formats, so queue entry may be used to save a POINTER to the right data.  
Or specialised queue for each class of device.

## EVENT-HANDLING

- interrupt processor
- disable interrupt
- save processor status
- branch to interrupt handling routine
- restore processor status
- enable interrupts.

Strings will be handled differently because they need more storage.



## SIMULATING A LOCATOR WITH A PICK

Light pen can be used to select object on screen, but it gives no data about the position. If we want to use the light pen for positioning then a tracking cross is employed.



## SIMULATING A PICK WITH A LOCATOR

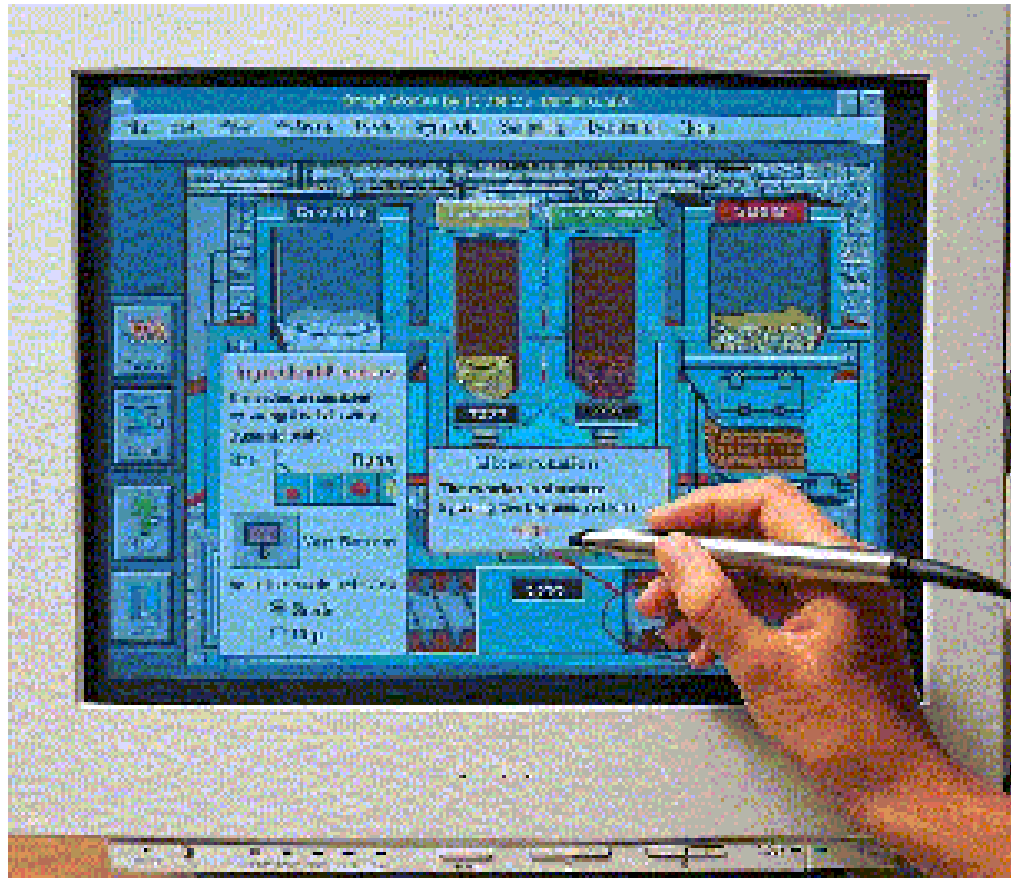
STEP THROUGH display file instructions and check how close the lines and characters are to the locator position

$$D = \sqrt{(x - x_c)^2 + (y - y_c)^2} \quad D < \text{Aperture}$$

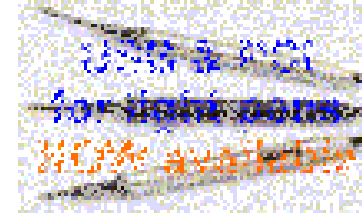


$$D^2 = (x - x_c)^2 + (y - y_c)^2$$

Algorithm need Multiplication



## LIGHT PEN RANGE



## INTERFACES

### APPLICATIONS:

What is a light pen?

Where can I use a light pen?

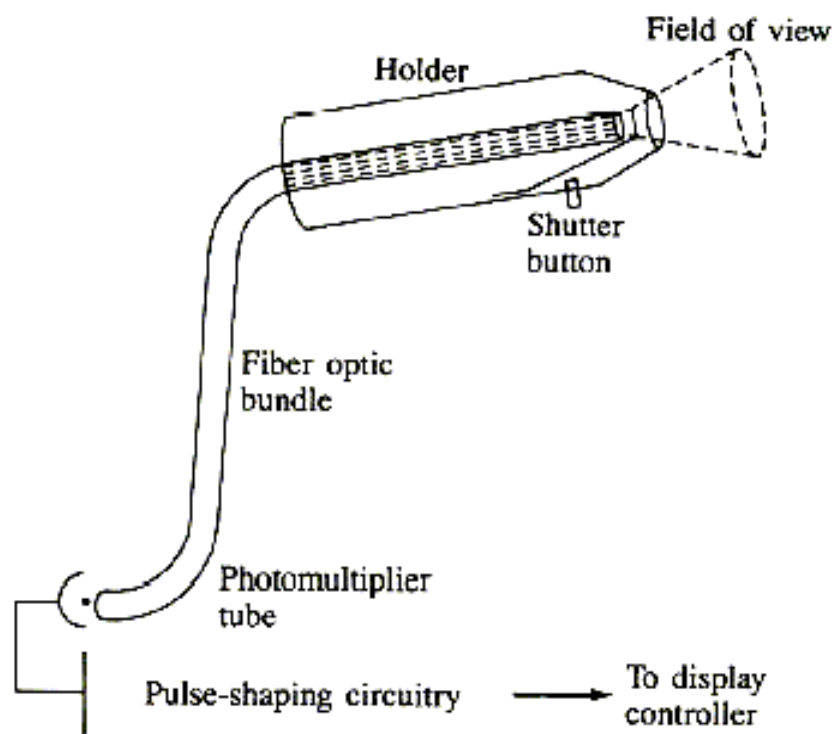
Will it work on my application?

Where light pens are used now

## LIGHT PENS V TOUCH SCREENS

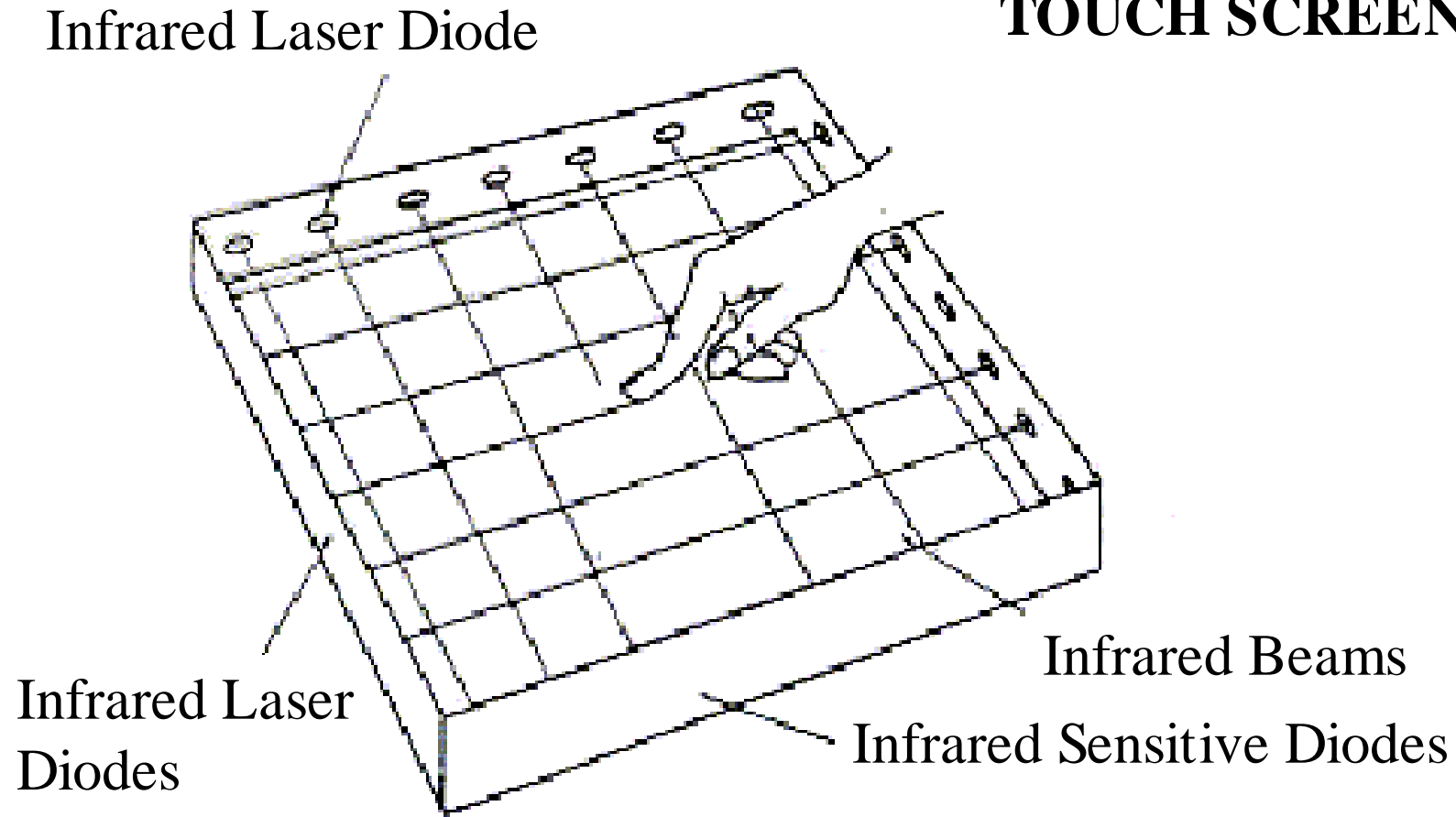
## LIGHT PEN PRICES

**LIGHT PEN = PURE SELECTOR DEVICE**



Schematic of a light pen.

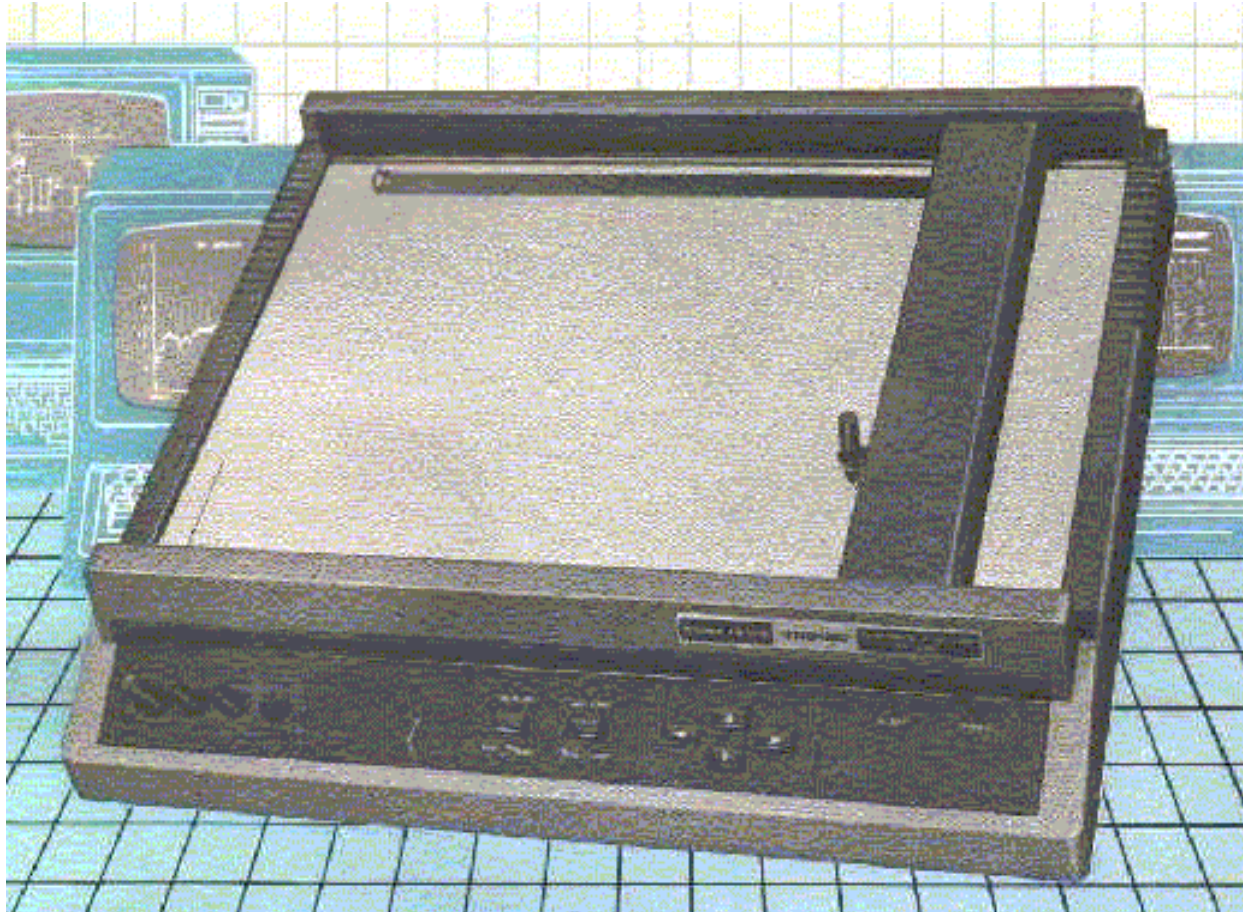
## TOUCH SCREEN







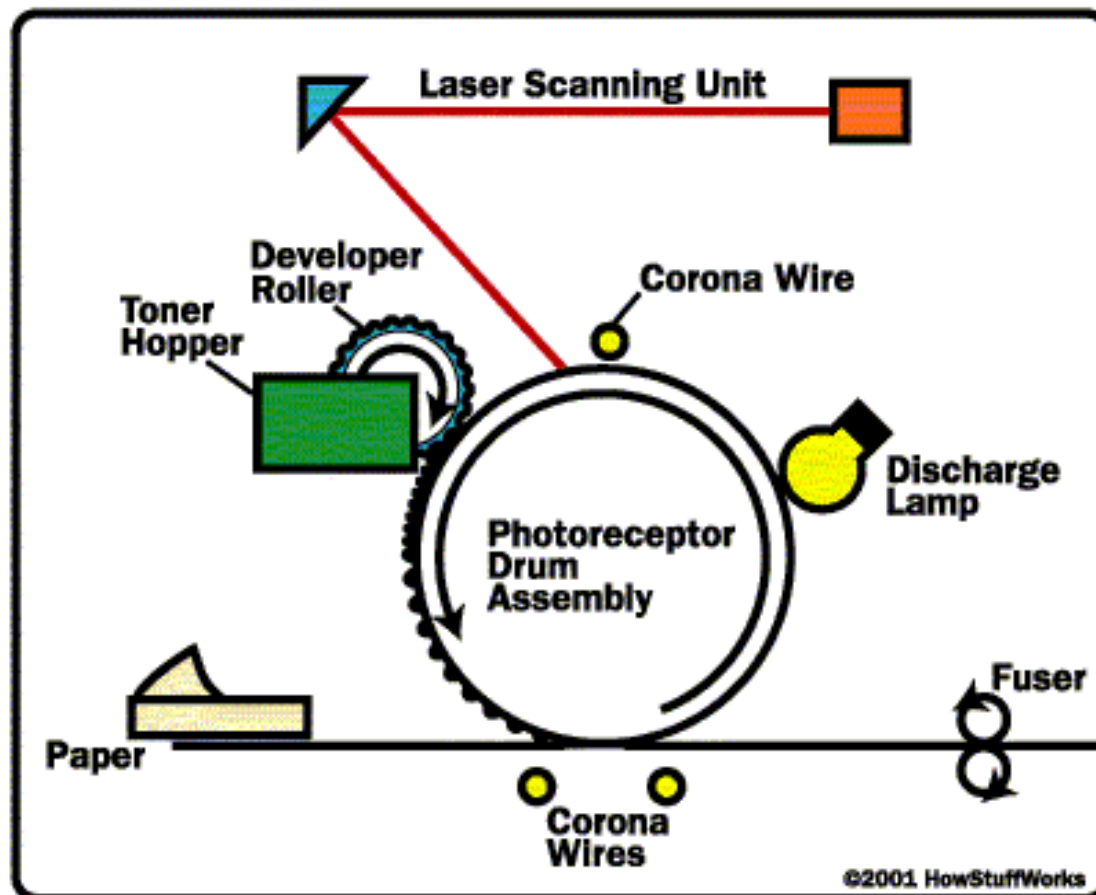
## PEN PLOTTER – VECTOR PLOTTER



# LASER PRINTER – RASTER PRINTER

## The Basic Process

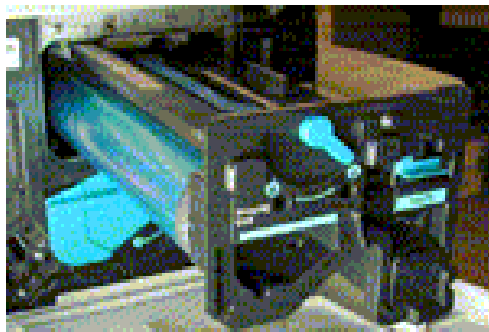
The primary principle at work in a laser printer is static electricity, the same energy that makes clothes in the dryer stick together or a lightning bolt travel from a thundercloud to the ground. Static electricity is simply an electrical charge built up on an insulated object, such as a balloon or your body. Since oppositely charged atoms are attracted to each other, objects with opposite static electricity fields cling together.



The basic components of a laser printer

A laser printer uses this phenomenon as a sort of "temporary glue." The core component of this system is the **photoreceptor**, typically a revolving drum or cylinder. This **drum assembly** is made out of highly photoconductive material that is discharged by light photons.

Initially, the drum is given a total **positive charge** by the **charge corona wire**, a wire with an electrical current running through it. (Some printers use a **charged roller** instead of a corona wire, but the principle is the same.) As the drum revolves, the printer shines a tiny laser beam across the surface to discharge certain points. In this way, the laser "draws" the letters and images to be printed as a pattern of electrical charges -- an electrostatic image. The system can also work with the charges reversed -- that is, a positive **electrostatic image** on a negative background.



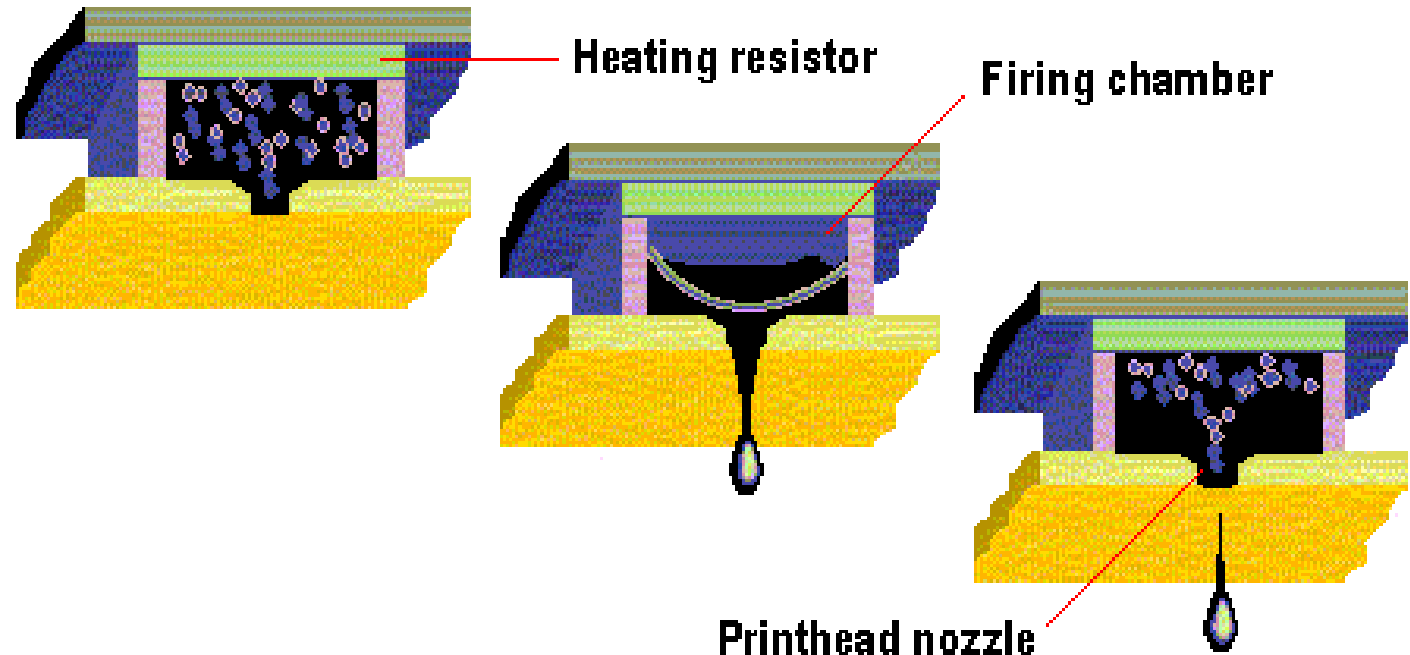
**The laser "writes" on a photoconductive revolving drum.**

After the pattern is set, the printer coats the drum with positively charged toner -- a fine, black powder. Since it has a positive charge, the toner clings to the negative discharged areas of the drum, but not to the positively charged "background." This is something like writing on a soda can with glue and then rolling it over some flour: The flour only sticks to the glue-coated part of the can, so you end up with a message written in powder.

# INKJET PRINTER – RASTER PRINTER

## Thermal technology

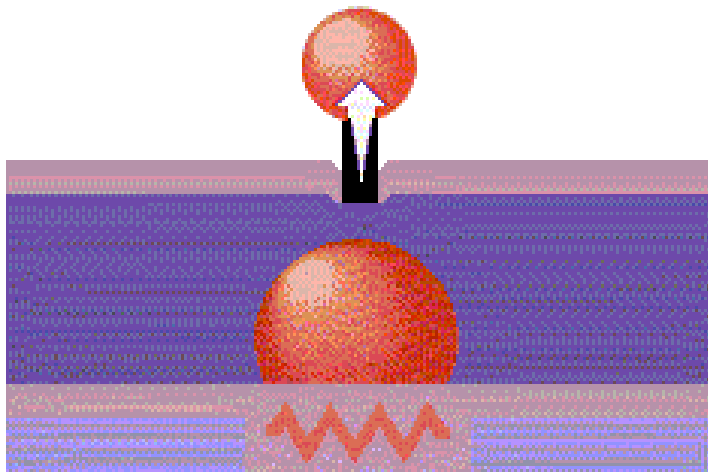
Most inkjets use thermal technology, whereby heat is used to fire ink onto the paper. There are three main stages with this method. The squirt is initiated by heating the ink to create a bubble until the pressure forces it to burst and hit the paper. The bubble then collapses as the element cools, and the resulting vacuum draws ink from the reservoir to replace the ink that was ejected. This is the method favoured by Canon and Hewlett-Packard.



Thermal technology imposes certain limitations on the printing process in that whatever type of ink is used, it *must* be resistant to heat because the firing process is heat-based. The use of heat in thermal printers creates a need for a cooling process as well, which levies a small time overhead on the printing process.

# INKJET PRINTER – RASTER PRINTER

Thermal technology imposes certain limitations on the printing process in that whatever type of ink is used, it *must* be resistant to heat because the firing process is heat-based. The use of heat in thermal printers creates a need for a cooling process as well, which levies a small time overhead on the printing process.



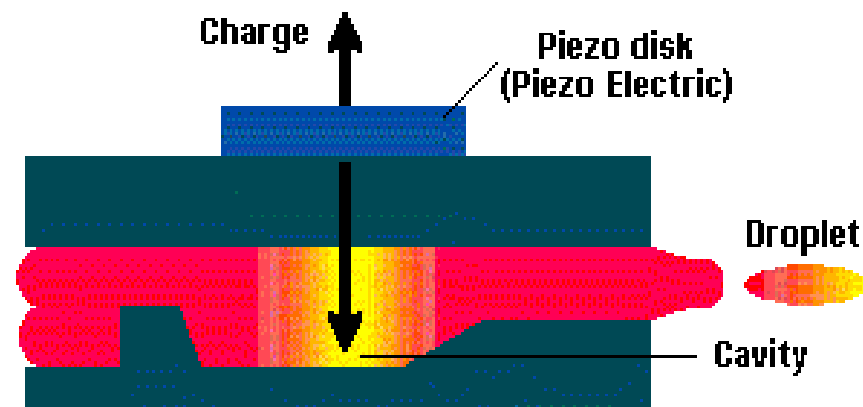
Tiny heating elements are used to eject ink droplets from the print-head's nozzles. Today's thermal inkjets have print heads containing between 300 and 600 nozzles in total, each about the diameter of a human hair (approx. 70 microns). These deliver drop volumes of around 8 - 10 picolitres (a picolitre is a million millionth of a litre), and dot sizes of between 50 and 60 microns in diameter. By comparison, the smallest dot size visible to the naked eye is around 30 microns. Dye-based cyan, magenta and yellow inks are normally delivered via a combined **CMY** print-head. Several small colour ink drops -

typically between four and eight - can be combined to deliver a variable dot size, a bigger palette of non-halftoned colours and smoother **halftones**. Black ink, which is generally based on bigger pigment molecules, is delivered from a separate print-head in larger drop volumes of around 35pl.

Nozzle density, corresponding to the printer's native **resolution**, varies between 300 and 600**dpi**, with enhanced resolutions of 1200dpi increasingly available. Print speed is chiefly a function of the frequency with which the nozzles can be made to fire ink drops and the width of the swath printed by the print-head. Typically this is around 12MHz and half an inch respectively, giving print speeds of between 4 to 8ppm (pages per minute) for monochrome text and 2 to 4ppm for colour text and graphics.

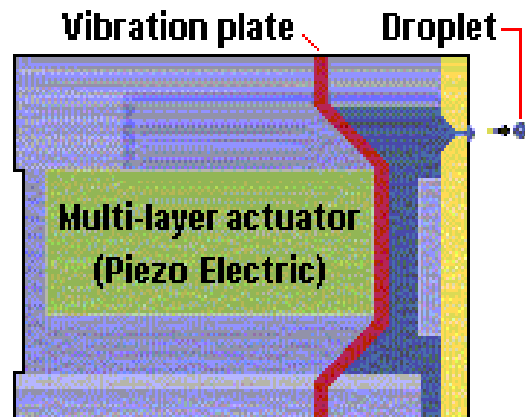
## Piezo-electric technology

Epson's proprietary inkjet technology uses a **piezo** crystal at the back of the ink reservoir. This is rather like a loudspeaker cone - it flexes when an electric current flows through it. So, whenever a dot is required, a current is applied to the piezo element, the element flexes and in so doing forces a drop of ink out of the nozzle.



There are several advantages to the piezo method.

The process allows more control over the shape and size of ink droplet release. The tiny fluctuations in the crystal allow for smaller droplet sizes and hence higher nozzle density. Also, unlike with thermal technology, the ink does not have to be heated and cooled between each cycle. This saves time, and the ink itself is tailored more for its absorption properties than its ability to withstand high temperatures. This allows more freedom for developing new chemical properties in inks.

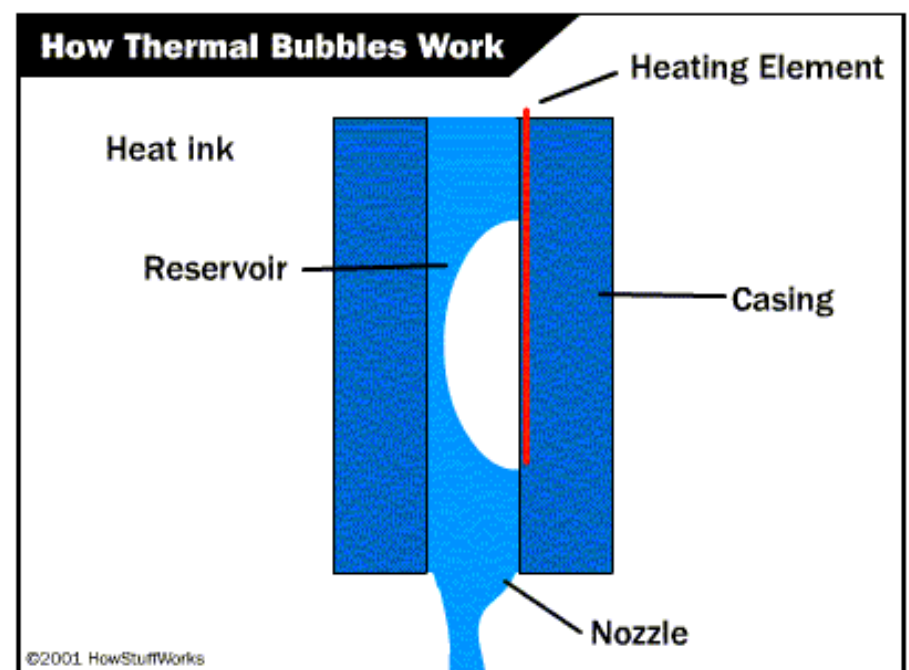
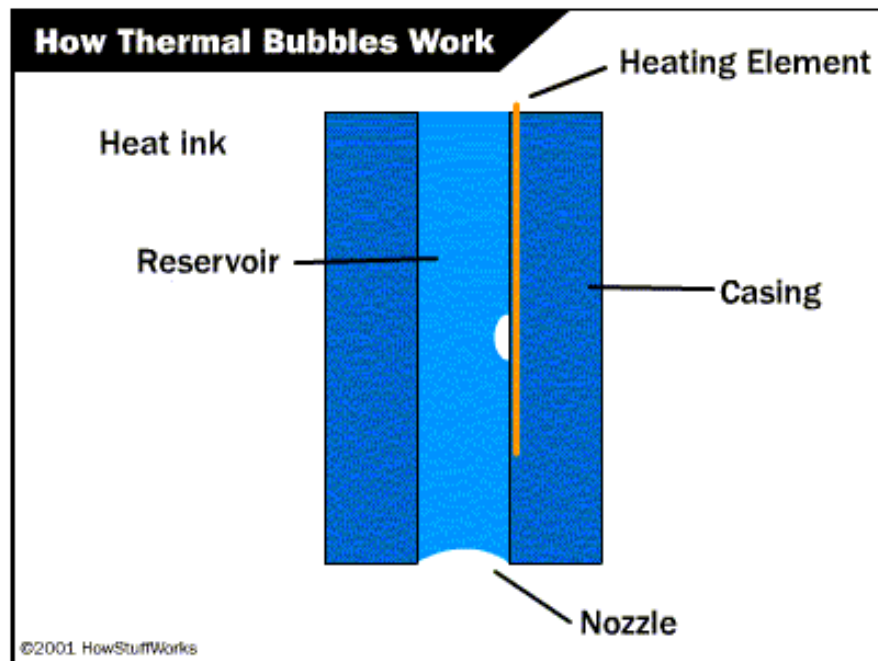


Epson's latest mainstream inkjets have black print-heads with 128 nozzles and colour (CMY) print-heads with 192 nozzles (64 for each colour), addressing a native **resolution** of 720 by 720dpi. Because the piezo process can deliver small and perfectly formed dots with high accuracy, Epson is able to offer an enhanced resolution of 1440 by 720dpi - although this is achieved by the print-head making two passes, with a consequent reduction in print speed. The tailored inks Epson has developed for use with its piezo technology are solvent-based and extremely quick-drying. They penetrate the paper and maintain their

shape rather than spreading out on the surface and causing dots to interact with one another. The result is extremely good print quality, especially on coated or glossy paper.

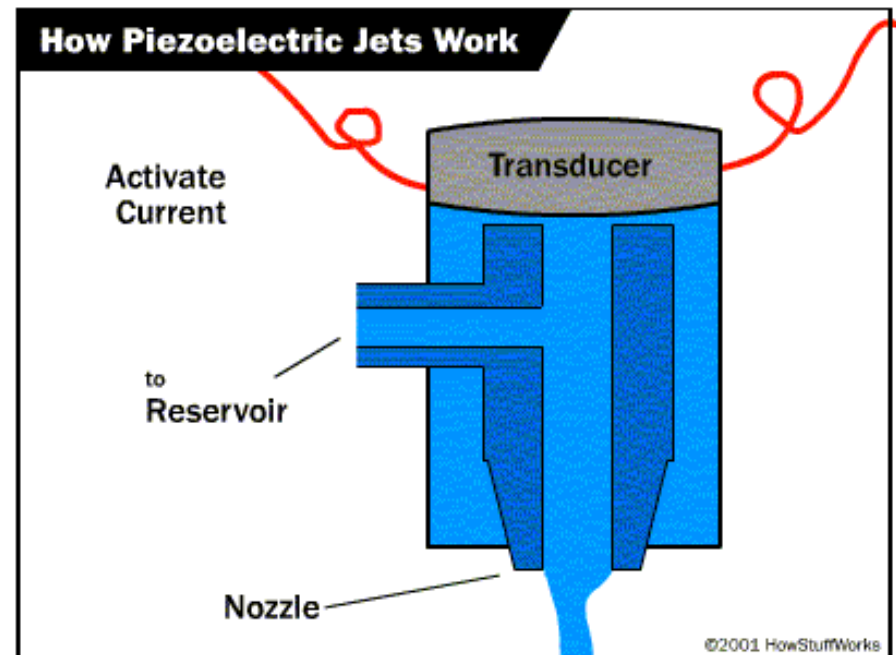
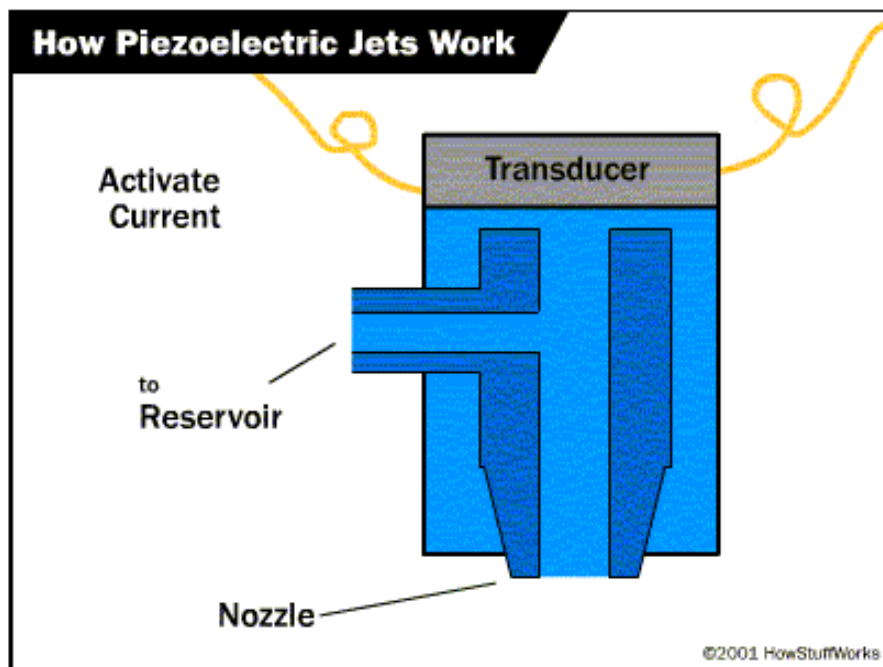
## INKJET PRINTER – RASTER PRINTER

- **Thermal bubble** - Used by manufacturers such as [Canon](#) and [Hewlett Packard](#), this method is commonly referred to as **bubble jet**. In a thermal inkjet printer, tiny resistors create heat, and this heat vaporizes ink to create a bubble. As the bubble expands, some of the ink is pushed out of a nozzle onto the paper. When the bubble "pops" (collapses), a vacuum is created. This pulls more ink into the print head from the cartridge. A typical bubble jet print head has 300 or 600 tiny nozzles, and all of them can fire a droplet simultaneously.





- **Piezoelectric** - [Patented by Epson](#), this technology uses **piezo crystals**. A crystal is located at the back of the ink reservoir of each nozzle. The crystal receives a tiny electric charge that causes it to vibrate. When the crystal vibrates inward, it forces a tiny amount of ink out of the nozzle. When it vibrates out, it pulls some more ink into the reservoir to replace the ink sprayed out.



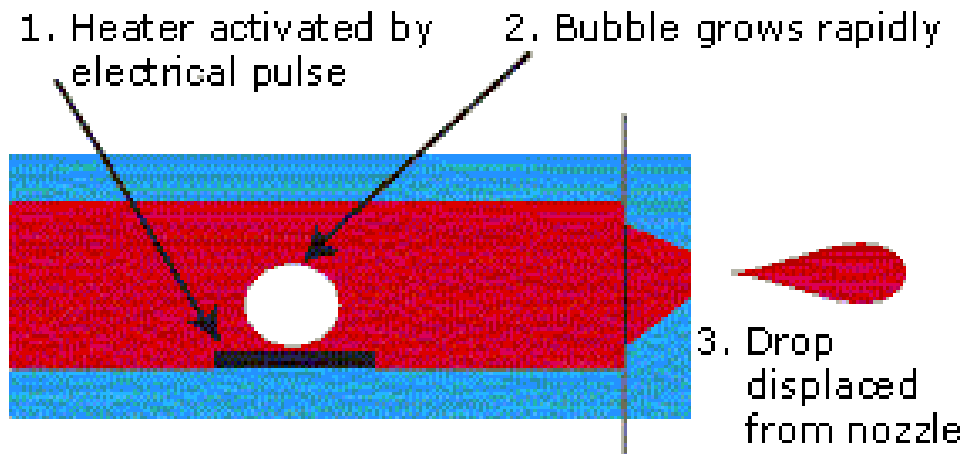
# Technology Tutorial

Liquid inkjet printers generally fall into one of two classes - continuous and drop-on-demand. In a continuous inkjet printer, a continuous spray of ink droplets is produced, the unneeded droplets are deflected before they reach the paper.

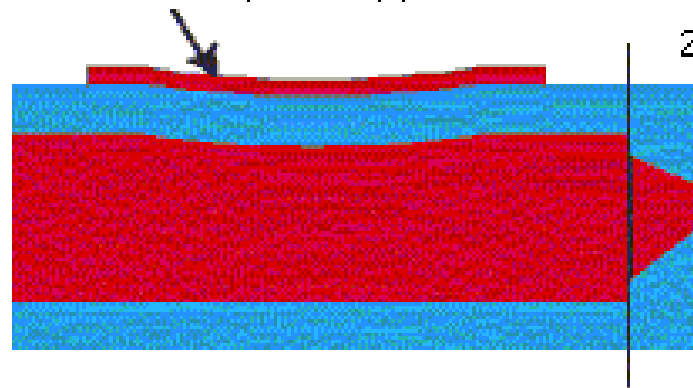
**Continuous inkjet** technology permits very high-speed drop generation, one million drops per second or faster, but is expensive to manufacture and, because of the wasted ink, expensive to operate. Two classes of continuous inkjet products are available today. High-speed industrial printers are used for applications such as carton and product marking and addressing and personalizing direct mail. The other is the proofing printer which offers the best print quality of any non-photographic device, but they are much slower - less than an inch per second. Although the resolutions are not that high (e.g. 300 dpi), the variable-sized dots make photographic quality possible.

**Drop-on-demand** inkjet printers produce ink droplets only when needed. The two most common technologies to drive the droplets out of the printhead are thermal (used by Hewlett Packard, Lexmark, Canon, Olivetti, Océ and others) and piezo-electric (used by Epson). Thermal have been by far the most successful because they can be produced inexpensively.

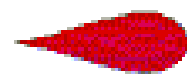
Thermal inkjet printers have a small resistor built into each nozzle. These resistors heat up when an electrical current is applied to them. The bubbles formed by vaporizing provides the force needed for ejecting the ink from the nozzle onto the paper.



1. Piezo crystal deforms when electrical pulse applied

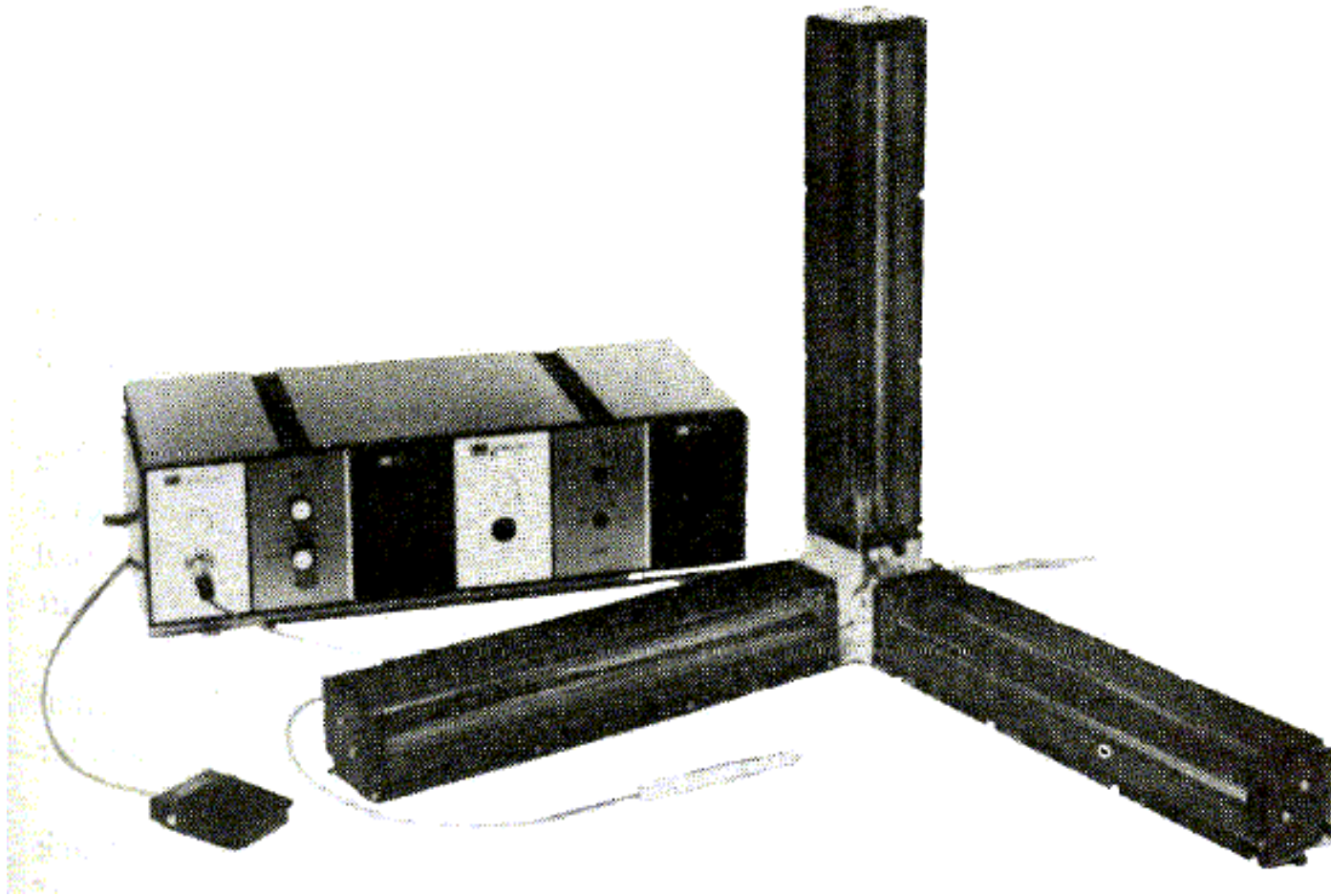


2. Drop displaced from nozzle

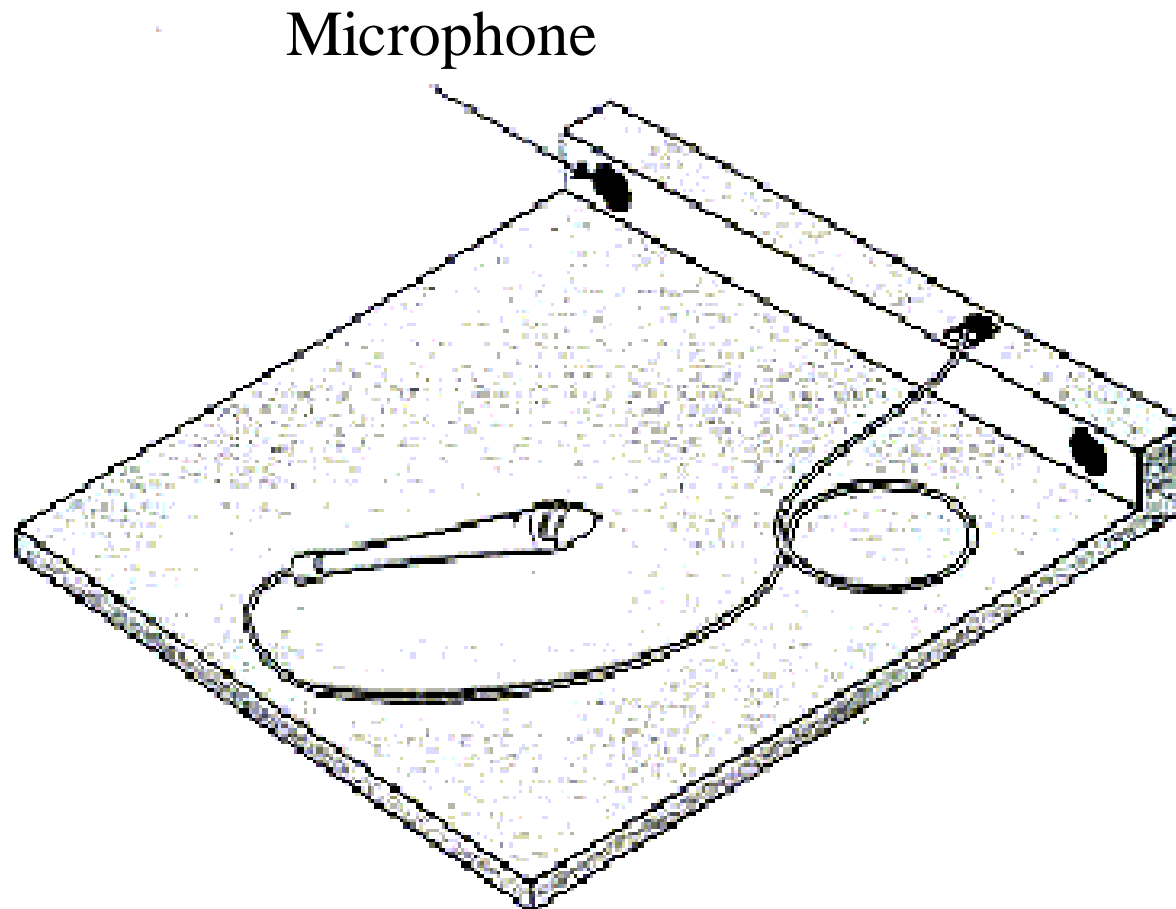


Piezo-electric inkjet printers harness the piezo-electric technology, which causes certain crystalline materials to change shape when a voltage is applied across them. In these printers, each nozzle of the printhead is housed inside an integrated piezo crystal. A small electrical current makes the crystal contract slightly, squeezing ink out of the nozzle onto the paper

The biggest challenge for piezo-electric inkjet technology is the cost and difficulty of producing print heads. Today Epson has a very successful line of piezo-electric colour printers, namely the Stylus color and Stylus photo family of printers offering photographic quality.



**3D ULTRASONIC DIGITIZER**



## **2D ULTRASONIC DIGITIZER**

