

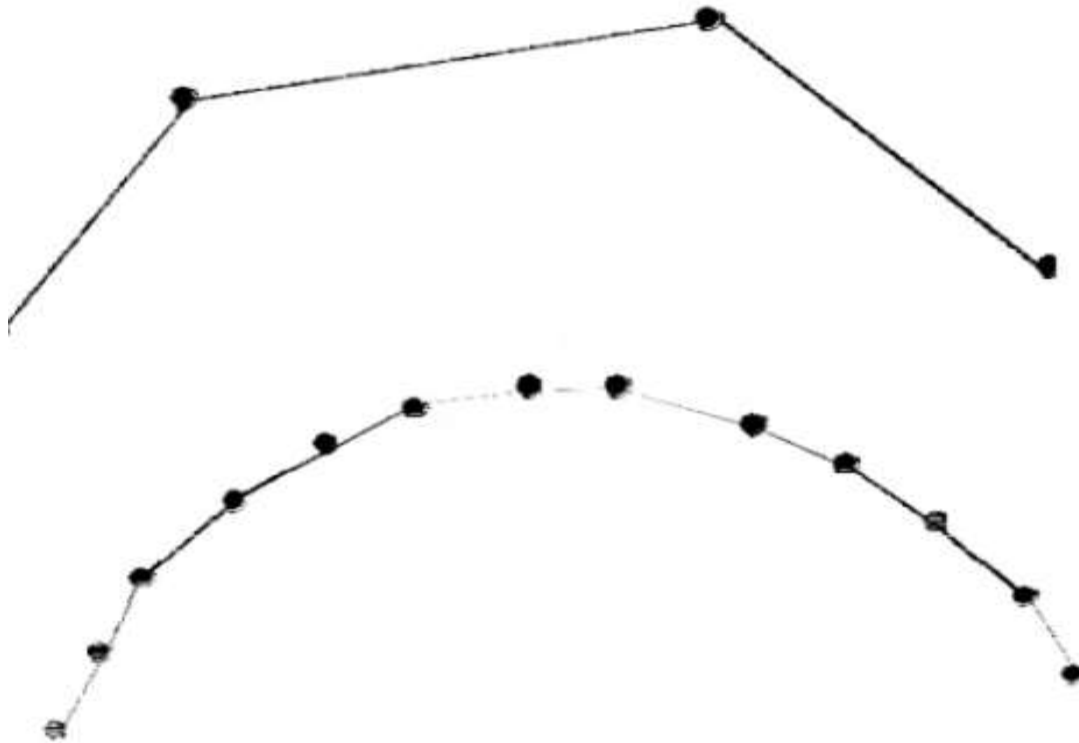
# Curves

- Curves are used to create high resolution graphics. They can be stored easily and scaled to any resolution without losing smoothness.
- In computer graphics, all curves are specified by parametric equations.
- Using parametric equations allows the curves that can double back and cross themselves.
- Simply, by changing the parameter,  $u$ , results in moving a fixed distance along the curve.

# Curves

- Curves are used to create high resolution graphics. They can be stored easily and scaled to any resolution without losing smoothness.
  - In computer graphics, all curves are specified by parametric equations.
  - Using parametric equations allows the curves that can double back and cross themselves.
  - Simply, by changing the parameter,  $u$ , results in moving a fixed distance along the curve.
-

“Computers can't draw  
curves.”



The more points/line segments that are used, the smoother the curve.

# Why have curves ?

- Representation of “irregular surfaces”
  - Example: Auto industry (car body design)
    - Artist’s representation
    - Clay / wood models
    - Digitizing
    - Surface modeling (“body in white”)
    - Scaling and smoothening
    - Artificial objects
-

# Curve representation

- In computer graphics we often need to draw different types of objects onto the screen.
- Objects are not flat all the time and we need to draw curves many times to draw an object.
- Types of curves:-
  - Curve is infinitely large set of points.
  - Each point has two neighbors except end point.

# Types of curves

1. Implicit curve
2. Explicit curve
3. Parametric curve

## 1. Implicit curve:-

I

Implicit curve representations define the set of points on a curve by employing a procedure that can test to see if a point is on the curve. Usually, an implicit curve is defined by an implicit function of the form  $f(x,y) = 0$ .

---

2. Explicit curve:- A mathematical function  $y = f(x)$  can be used to plot a curve. Such a function is an explicit representation of the curve. The explicit representation is not general, since it cannot represent vertical lines and is also single valued.

For each value of  $x$ , only a single value of  $y$  is normally computed by the function.



3. Parametric curve:- Curves having parametric form are called parametric curves. The explicit and implicit curve representations can be used only when the function is known. In practice the parametric curves are used. A two-dimensional parametric curve has the following form –

$$P(t) = f(t), g(t)$$

OR

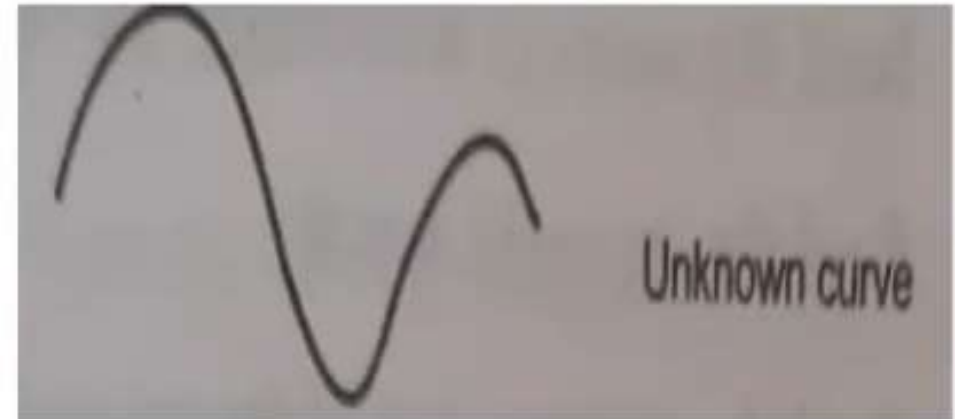
$$P(t) = x(t), y(t)$$

The functions  $f$  and  $g$  become the  $x$ ,  $y$  coordinates of any point on the curve, and the points are obtained when the parameter  $t$  is varied over a certain interval  $[a, b]$ , normally  $[0, 1]$ .

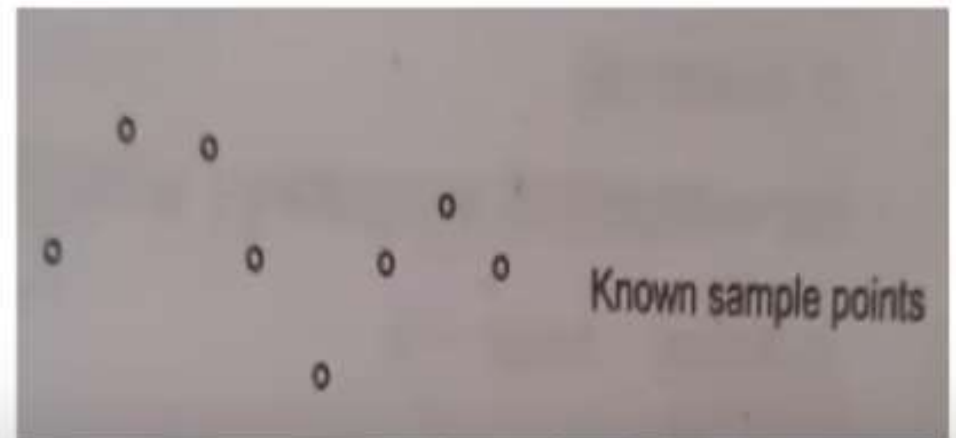


# Interpolation, Approximation and Blending function

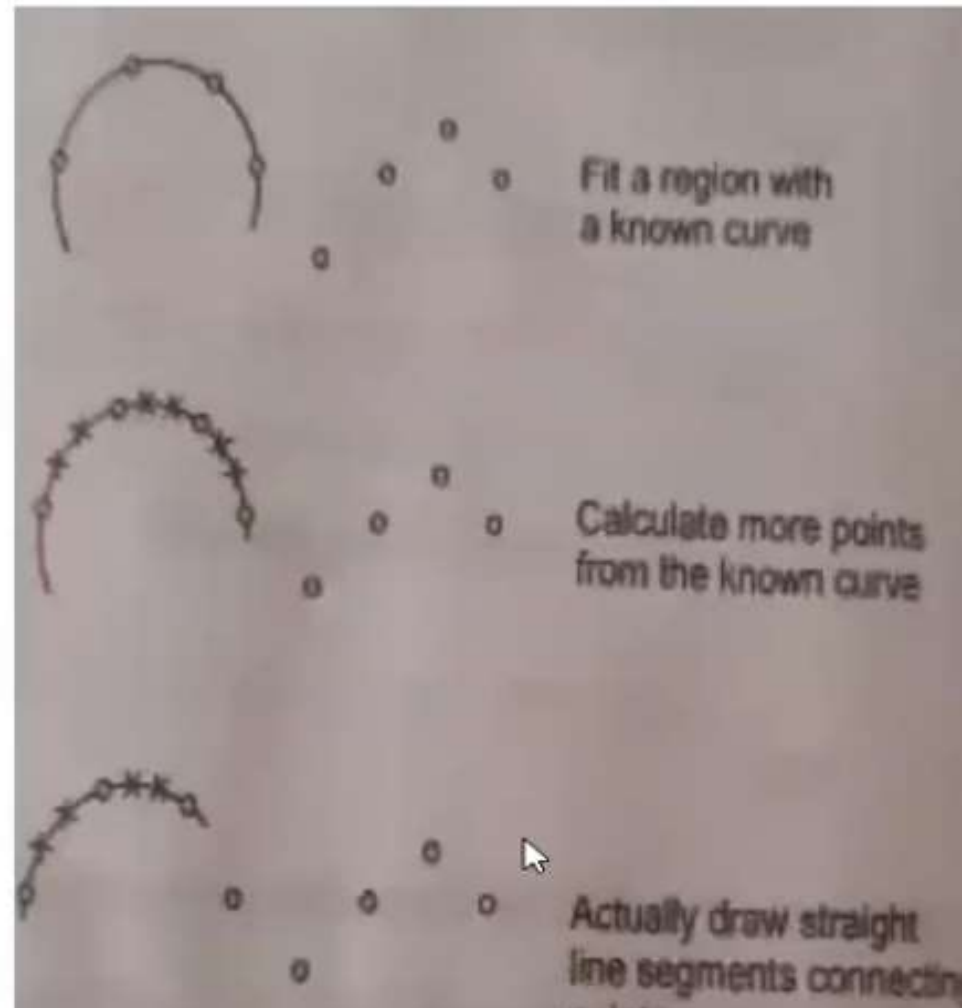
There are some complex curves for which no direct mathematical function is available. Such curves can be drawn using approximation methods.



If we have set of sample points which lie on the required curve, then we can draw the required curve by filling portions of the curve with the pieces of known curves which pass through nearby sample points.



- The gap between the sample points can be filled by finding the co-ordinates of the points along the known approximating curve and connecting these points with line segments as shown in the figure.



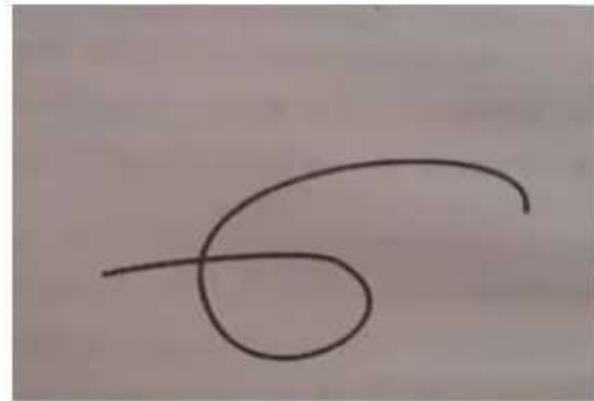
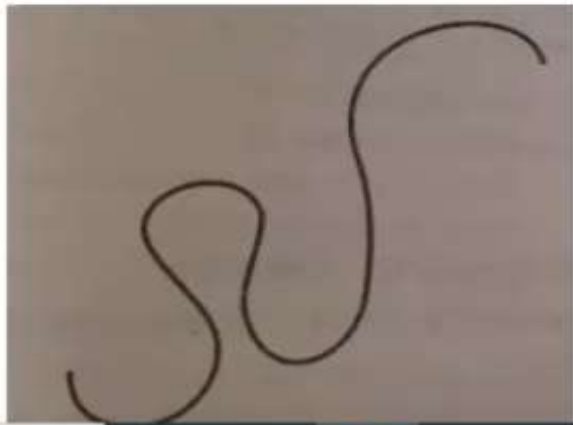
- 
- The main task in this process is to find the suitable mathematical expression for the known curve.
  - There are polynomial, trigonometric, exponential and other classes of functions that can be used to approximate the curve.
  - Usually polynomial functions in the parametric form are preferred. The polynomial functions in the parametric form can be given as,

$$x = f_x(u)$$

$$y = f_y(u)$$

$$z = f_z(u)$$

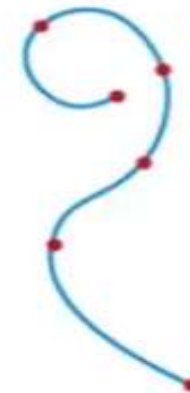
- We can realize from above equation that the difference between 2 and 3 dimensions is just the addition of the third equation for  $z$ .
- Further the parametric form treats all the three dimensions equally and allows multiple values (several values of  $y$  or  $z$  for a given  $x$  value).
- Due to these multiple values curves can double back or even cross themselves, as shown in the figure below.



- We have seen that, we have to draw the curve by determining the intermediate points between the known sample points. This can be achieved using interpolation techniques.
  - Interpolation is a method of constructing new data points within range of discrete set of known data points. The number of data points obtained by sampling or experimentation represents values of function for limited number of values of independent variable.
  - The main task of Interpolation is to find suitable mathematical expression for known curve. This technique is used when we have to draw curve by determining intermediate points between known sample points.
-

In computer graphics, interpolation is the creation of new values that lie between known values. For example, when objects are rasterized into two-dimensional images from their corner points (vertices), all the pixels between those points are filled in by an interpolation algorithm, which determines their color and other attributes

– **Interpolation** - the curve passes through all of the control points



Suppose we want a polynomial curve that will pass through  $n$  sample points.  $(x_1, y_1, z_1), (x_2, y_2, z_2), \dots, (x_n, y_n, z_n)$ . We will construct the function as the sum of terms, one term for each sample point. These functions can be given as

$$f_x(\mathbf{u}) = \sum_{i=1}^n x_i B_i(\mathbf{u})$$

$$f_y(\mathbf{u}) = \sum_{i=1}^n y_i B_i(\mathbf{u})$$

$$f_z(\mathbf{u}) = \sum_{i=1}^n z_i B_i(\mathbf{u})$$



- 
- The function  $B_i(u)$  is called 'blending function'. For each value of parameter  $u$ , the blending function determines how much the  $i^{\text{th}}$  sample point affects the position of the curve.
  - In other words we can say that each sample point tries to pull the curve in its direction and the function  $B_i(u)$  gives the **strength** of the pull.
  - If for some value of  $u$ ,  $B_i(u) = 1$  for unique value of  $i$  (i.e.  $B_i(u) = 0$  for other values of  $i$ ) then  $i^{\text{th}}$  sample point has complete control of the curve and the curve will pass through  $i^{\text{th}}$  sample point.
  - for different value of  $u$ , some other sample point may have complete control of the curve. In such case the curve will pass through that point as well.
  - In general, the blending functions give control of the curve to each of the sample points in-turn for different values of  $u$ .
  - Let us assume that the first sample point  $(x_1, y_1, z_1)$  has complete control when  $u = -1$ , the second when  $u = 0$ , the third when  $u = 1$ , and so on i.e
-



- When  $u = -1 \Rightarrow B_1(u) = 1$  and 0 for  $u = 0, 1, 2, \dots, n-2$
- When  $u = 0 \Rightarrow B_2(u) = 1$  and 0 for  $u = -1, 1, \dots, n-2$
- :        :        :
- :        :        :
- When  $u = (n-2) \Rightarrow B_n(u) = 1$  and 0 for  $u = -1, 0, \dots, (n-1)$

- To get  $B_1(u) = 1$  at  $u = -1$  and 0 for  $u = 0, 1, 2, \dots, n-2$ , the expression for  $B_1(u)$  can be given as

$$B_1(u) = \frac{u(u-1)(u-2)\dots[u-(n-2)]}{(-1)(-2)\dots(1-n)}$$

when denominator term is used constantly. In general form  $i^{\text{th}}$  blending function which is 1 at  $u = i-2$  and 0 for other integers can be given as:

$$B_i(u) = \frac{(u+1)(u)(u-1)\dots[u-(i-3)][u-(i-1)]\dots[u-(i-2)]}{(i-1)(i-2)(i-3)\dots(1)(-1)\dots(i-n)}$$

$\vdots$   
 $\vdots$   
 $\vdots$   
 ■ When  $u=(n - 2) \Rightarrow B_n(u) = 1$  and 0 for  $u = - 1, 0, \dots , (n - 1)$

- To get  $B_1(u)= 1$  at  $u= -1$  and 0 for  $u= 0,1,2,\dots,n-2$ , the expression for  $B_1(u)$  can be given as

$$B_1(u) = \frac{u (u - 1) (u - 2) \dots [u - (n - 2) ]}{(-1) (-2) \dots (1 - n)}$$

when denominator term is used constantly. In general form  $i^{\text{th}}$  blending function which is 1 at  $u= i-2$  and 0 for other integers can be given as:

$$B_i(u) = \frac{(u + 1)(u) (u - 1) \dots [u - (i - 3) ] [u - (i - 1) ] \dots [u - (i - 2) ]}{(i - 1) (i - 2) (i - 3) \dots (1) (-1) \dots (i - n)}$$

- The approximation of the curve using above expression is called Lagrange interpolation. From the above expression blending functions for four sample points can be given as:

$$B_1(u) = \frac{u(u-1)(u-2)}{(-1)(-2)(-3)}$$

$$B_2(u) = \frac{(u+1)(u-1)(u-2)}{1(-1)(-2)}$$

$$B_3(u) = \frac{(u+1)u(u-2)}{(2)(1)(-1)}$$

$$B_1(u) = \frac{u(u-1)(u-2)}{(-1)(-2)(-3)}$$

$$B_2(u) = \frac{(u+1)(u-1)(u-2)}{1(-1)(-2)}$$

$$B_3(u) = \frac{(u+1)u(u-2)}{(2)(1)(-1)}$$

$$B_4(u) = \frac{(u+1)u(u-1)}{(3)(2)(1)}$$

Using above blending functions, the expressions for the curve passing through sampling points can be realized as follows:-

$$x = x_1 B_1(u) + x_2 B_2(u) + x_3 B_3(u) + x_4 B_4(u)$$

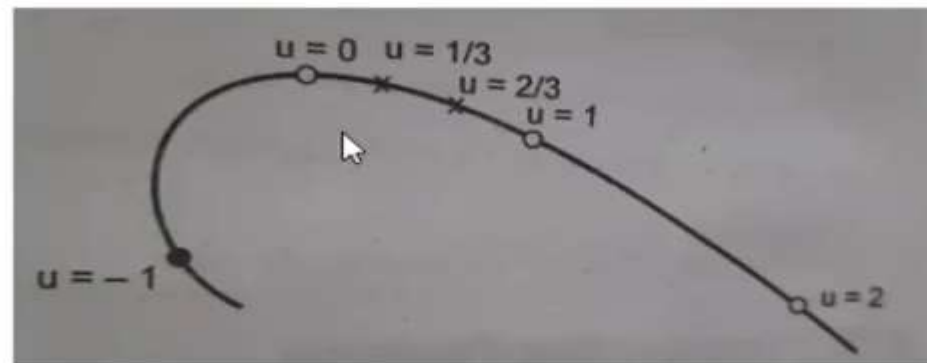
$$y = y_1 B_1(u) + y_2 B_2(u) + y_3 B_3(u) + y_4 B_4(u)$$

$$z = z_1 B_1(u) + z_2 B_2(u) + z_3 B_3(u) + z_4 B_4(u)$$

It is possible to get intermediate points between two sampling points by taking values of  $u$  between the values of  $u$  related to the two sample points under consideration.

- It is possible to get intermediate points between two sampling points by taking values of  $u$  between the values of  $u$  related to the two sample points under consideration.

- For example, we can find the intermediate points between second and third sample points for which values of  $u$  are 0 and 1, respectively; by taking values of  $u$  between 0 and 1. This is shown in figure



### Determining intermediate points for approximation of curve

- The subsequent intermediate points can be obtained by repeating the same procedure. Finally the points obtained by this procedure are joined by small straight line segments to get the approximated curve.

- Initially, sample points(1,2,3,4) are considered and intermediate points between (2,3) are obtained. Then sample point at one end is discarded and sample point at the other end is added to get new sample points(2,3,4,5).
- Now the curve between the sample points(3,4) is approximated. The subsequent intermediate points can be obtained by repeating the same procedure.
- The initial and final portions of the curve require special treatment. For the first four points (1,2,3,4) we have to draw region between points (1,2) with  $u$  values between -1 and 0.
- Similarly the blending function for very last step of the curve should be evaluated with  $u$  values between 1 and 2.

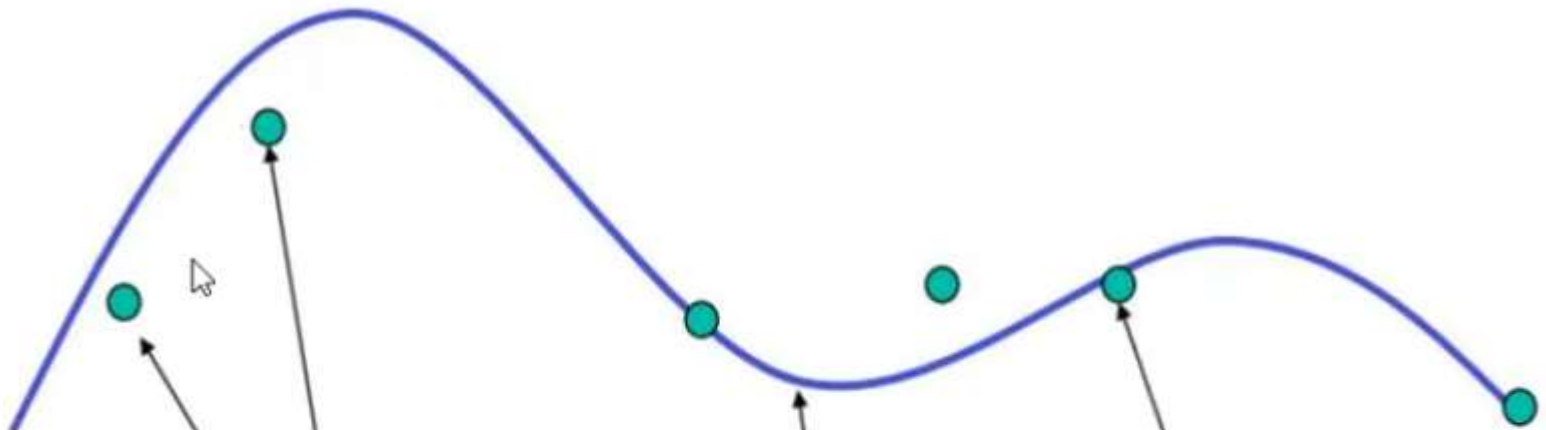


- Instead of using readymade algorithm, we are approximating the curve by small straight lines. For this we have to use interpolation technique. We can draw an approximation to a curve if we have an array of sample points.

– **Approximation** - the curve does not pass through all of the control points

I



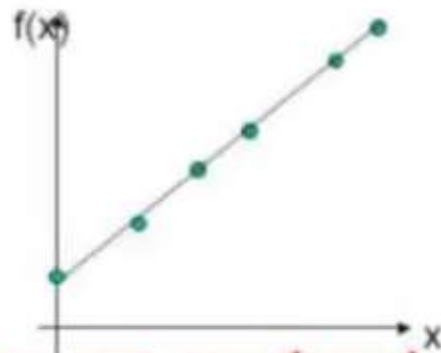


data points

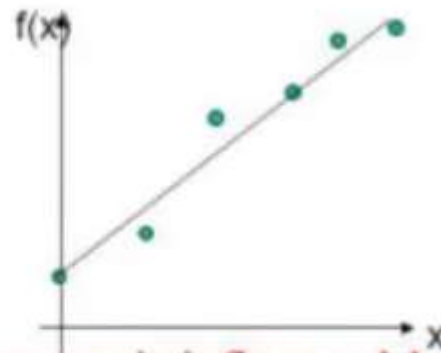
approximating curve

interpolating data point

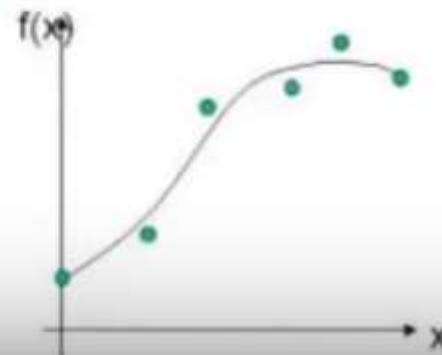
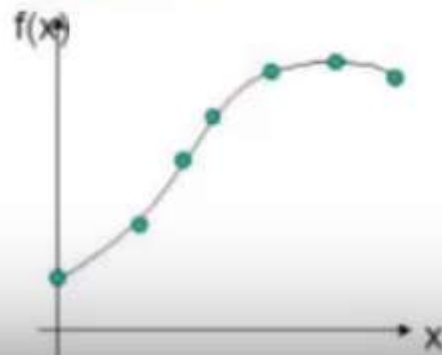
# Interpolation vs approximation



curve must pass through control points



curve is influenced by control points



# B-Spline Curve

⇒ Bezier Curve (global control) ⇒ Control points

⇒ B-spline curve (local control)

⇓

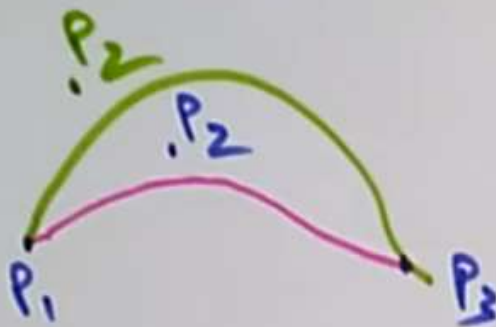
degree of poly Does not depends upon ~~the~~ no: of control points

⇒ It depends upon the order of polynomial.

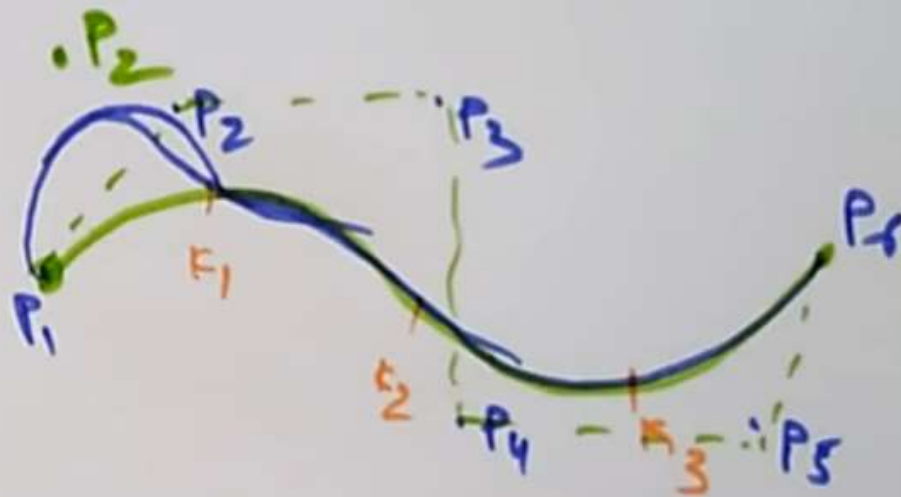
ie based on the no: of control points  
if order varies then blending function  
degree will be diff.

Bezier (global control)

Eg 1



B-spline (local control)



## Properties of B-spline Curve

→ The sum of B-spline basis function at any parameter 'u' equal to 1 i.e.

$$\sum_{i=1}^{n+1} N_{i,k}(u) = 1$$

$n+1$  = no. of control point

$k$  = order of B-spline curve

→ The basis function is +ve or zero for all parameter values i.e.  $N_{i,k}(u) \geq 0$ . Except for  $k=1$  each basis function has one maximum value

→ The maximum order of the curve is equal to the number of vertices of defining polygon.

→ The degree of B-spline polynomial is independent on the no: of ~~vert~~ vertices of defining polygon.

→ B-spline allow the control (ie local control) over the curve surface.

→ The curve lies within the convex hull of its defining polygon.

→ The curve generally follow the spo shape of defining polygon.

→ The B-spline curve generally represented as,

$$P(u) = \sum_{i=1}^{n+1} P_i N_{i,k}(u).$$

{ Blending function  
for B-spline curve }

where  $u_{\min} \leq u \leq u_{\max}$

$$2 \leq k \leq n+1$$

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

$$N_{i,k}(u) = \begin{cases} 1 & \alpha_i \leq u \leq \alpha_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \alpha_i &= 0 & \text{if } i < k \\ \alpha_i &= i - k + 1 & \text{if } k \leq i \leq n \\ \alpha_i &= n - k + 2 & \text{if } i > n \end{aligned}$$



When we designing the B-spline curve we have to evaluate knot, vector based on the no: of control points & order of curve.

⇒ knot vector can be evaluated by using

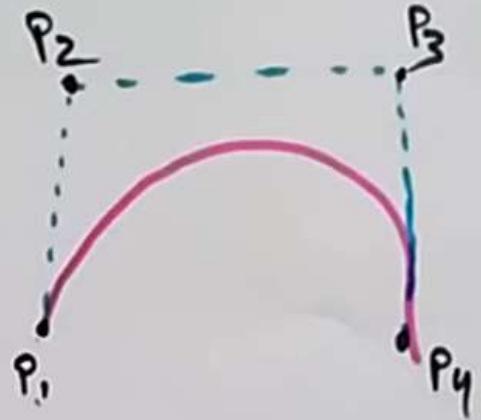
- i) Uniform
- ii) Non-uniform
- iii) open-uniform

# Bezier Curve #

defined by { control points }

$P_1, P_2, P_3, P_4$

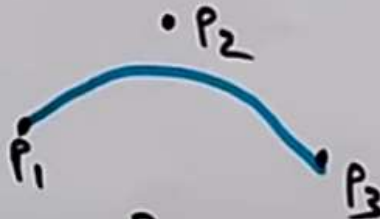
Control points



⇒ 2 pt Curve ⇒



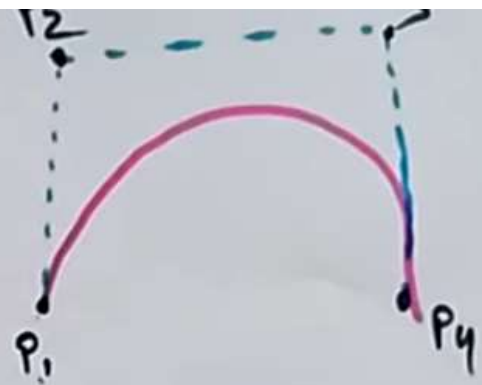
⇒ 3 pt Curve ⇒



⇒ 4 pt Curve ⇒



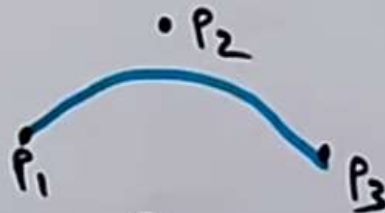
defined by  $\{ \text{control points} \}$   
 $P_1, P_2, P_3, P_4$   
 $\Downarrow$   
 Control points



$\Rightarrow$  2 pt Curve  $\Rightarrow$



$\Rightarrow$  3 pt Curve  $\Rightarrow$



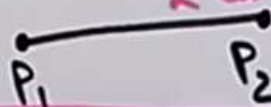
$\Rightarrow$  4 pt Curve  $\Rightarrow$



defined by  $\{ \text{control points} \}$   
 $P_1, P_2, P_3, P_4$   
 Control points



$\Rightarrow$  2 pt Curve  $\Rightarrow$



2 control points  
linear curve

$\Rightarrow$  3 pt Curve  $\Rightarrow$

3  $\Rightarrow$  C.P  
 $n-1$   
 degree = 2



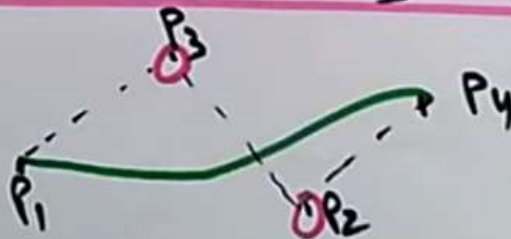
3 control pts

quadratic  
parabolic

① pts are not always on curve

$\Rightarrow$  4 pt Curve  $\Rightarrow$

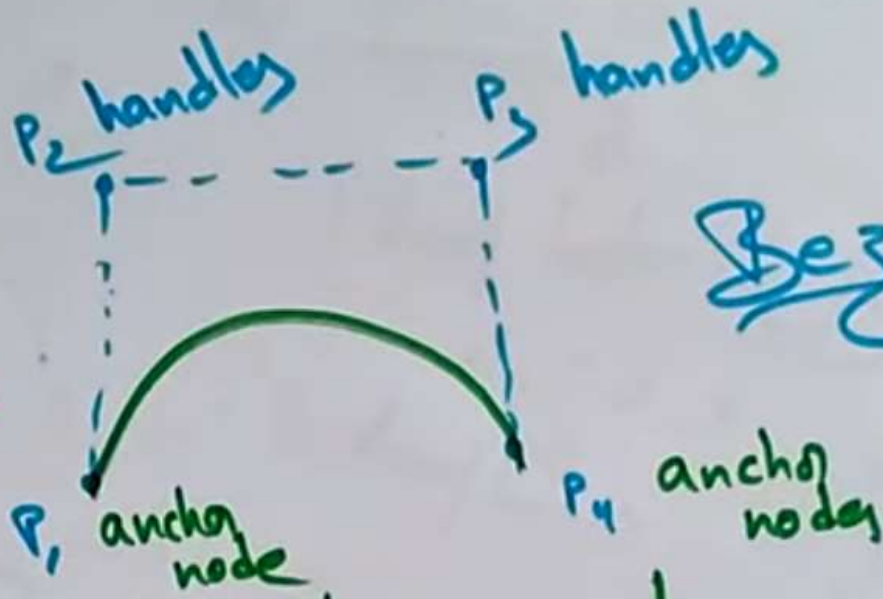
Cubic  $n-1 = 4-1$   
 degree = 3



Cubic Curve.

Eg 1-

degree = 3  
4



Bezier Cubic

Curve is defined as 4 pts

## Properties of Bezier Curve

They always pass through the first & last control points



They are contained in the Convex hull of their defining control points.



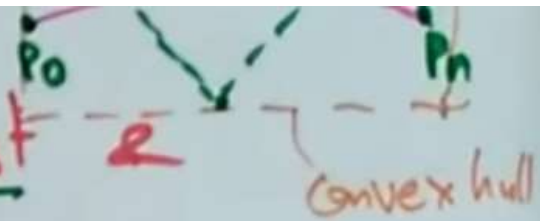
The degree of polynomial defining the curve segment is one less than the no: of defining, polygon pt.

→ They always pass through the first & last control points

→ They are contained in the Convex hull of their defining control points.

→ The degree of polynomial defining the curve segment is one less than the no. of defining, polygon pt.

4  $\Rightarrow$  control points  
 $\rightarrow$  degree =  $4 - 1 = 3 \Rightarrow$  Cubic poly

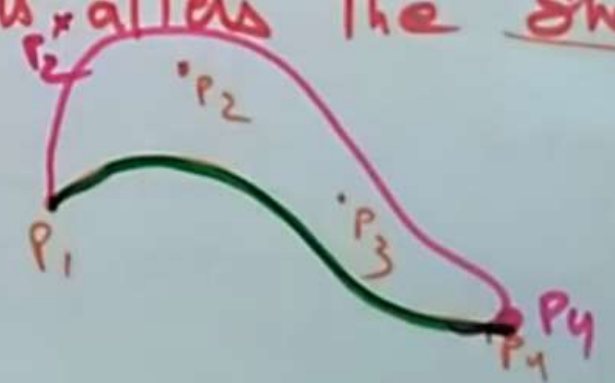
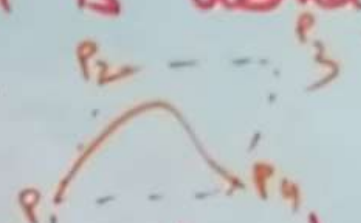


degree
1 - linear poly
2 - quadratic poly
3 - cubic poly

→ The shape of defining polygon is usually followed by Bezier Curve

→ Bezier Curves are tangent to their first & last edges of control polyline.

→ Bezier Curves exhibit global control point ~~alter~~ means moving a control point ~~alter~~ the shape of the whole curve.





# Derivation of Bezier Curve

⇒ Blending / basis function

$$P(u) = \sum_{i=0}^n P_i \mathcal{B}_{i,n}(u)$$

$0 \leq u \leq 1$   
 $n+1 \Rightarrow$  control p

where

$$\mathcal{B}_{i,n}(u) = c(n, i) u^i (1-u)^{n-i}$$

∥,

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

$P(u) = P_0 \underline{B_{0,3}}(u) + P_1 \underline{B_{1,3}}(u) + P_2 \underline{B_{2,3}}(u) + P_3 \underline{B_{3,3}}(u)$

$\Rightarrow B_{0,3}(u) = C(3,0) \cdot u^0 \cdot (1-u)^{3-0}$   
 $= \frac{3!}{0! 3!} (1-u)^3 \Rightarrow (1-u)^3 \text{ ——— ①}$

$\Rightarrow B_{1,3}(u) = C(3,1) \cdot u^1 \cdot (1-u)^{3-1}$   
 $= 3u \cdot (1-u)^2 \text{ ——— ②}$

$\Rightarrow B_{2,3}(u) = C(3,2) \cdot u^2 \cdot (1-u)^{3-2}$   
 $=$

$$+ P_3 \underline{B_{3,3}(u)}$$

$$\Rightarrow B_{0,3}(u) = C(3,0) \cdot u^0 \cdot (1-u)^{3-0}$$
$$= \frac{3!}{0! 3!} (1-u)^3 \Rightarrow (1-u)^3 \text{ ——— } \textcircled{1}$$

$$\Rightarrow B_{1,3}(u) = C(3,1) \cdot u^1 \cdot (1-u)^{3-1}$$
$$= 3u \cdot (1-u)^2 \text{ ————— } \textcircled{2}$$

$$\Rightarrow B_{2,3}(u) = C(3,2) \cdot u^2 \cdot (1-u)^{3-2}$$
$$= 3u^2 \cdot (1-u) \text{ ————— } \textcircled{3}$$

$$\Rightarrow \cancel{B} B_{3,3}(u) = C \cdot (3,3) u^3 \cdot (1-u)^{3-3}$$
$$= u^3 \quad \text{---} \quad \textcircled{4}$$

So, blending function,

$$P(u) = P_0(1-u)^3 + P_1 \cdot 3u \cdot (1-u)^2 + P_2 \cdot 3u^2 \cdot (1-u) + P_3 \cdot u^3$$

$$P(u_x) = P_{x_0}(1-u)^3 + P_{x_1} \cdot 3u \cdot (1-u)^2 + P_{x_2} \cdot 3u^2 \cdot (1-u) + P_{x_3} \cdot u^3$$

$$P(u_y) = P_{y_0}(1-u)^3 + P_{y_1} \cdot 3u \cdot (1-u)^2 + P_{y_2} \cdot 3u^2 \cdot (1-u) + P_{y_3} \cdot u^3$$

$$P(u_z) = P_{z_0}(1-u)^3 + P_{z_1} \cdot 3u \cdot (1-u)^2 + P_{z_2} \cdot 3u^2 \cdot (1-u) + P_{z_3} \cdot u^3$$

Example: Design a bezier curve with  
points  $A(1,1)$   $B(2,3)$   $C(4,3)$   $D(6,4)$

Solution:-

Control points = 4

$$n = 4 - 1 = 3$$

$$A(1,1) = P_0$$

$$C(4,3) = P_2$$

$$B(2,3) = P_1$$

$$D(6,4) = P_3$$

Eq<sup>n</sup> of bezier curve,

$$P(u) = \sum_{i=0}^n P_i B_{i,n}(u)$$

points  $A(1,1)$  ...

Solution:-

Control points = 4

$$n = 4 - 1 = 3$$

$$C(4,3) = P_2$$

$$D(6,4) = P_3$$

$$A(1,1) = P_0$$

$$B(2,3) = P_1$$

Eq<sup>n</sup> of bezier curve,

$$P(u) = \sum_{i=0}^n P_i B_{i,n}(u)$$

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} \cdot u^i \cdot (1-u)^{n-i}$$

$$\begin{aligned} P(u) &= P_0 B_{0,3}(u) + P_1 B_{1,3}(u) + P_2 B_{2,3}(u) + P_3 B_{3,3}(u) \\ &= P_0 (1-u)^3 + P_1 \cdot 3u (1-u)^2 + P_2 \cdot 3u^2 (1-u) + P_3 u^3 \end{aligned}$$



$$P(u) = P_0 B_{0,3}(u) + P_1 B_{1,3}(u) + P_2 B_{2,3}(u) + P_3 B_{3,3}(u)$$

$$= P_0(1-u)^3 + P_1 \cdot 3u(1-u)^2 + P_2 \cdot 3u^2(1-u) + P_3 u^3$$

find  $x(u)$  &  $y(u)$

$$P_0 = \binom{3}{0} \begin{matrix} 1 \\ 0 \\ 0 \\ 0 \end{matrix}$$

$$P_1 = \binom{3}{1} \begin{matrix} 2 \\ 1 \\ 0 \\ 0 \end{matrix}$$

$$P_2 = \binom{3}{2} \begin{matrix} 1 \\ 2 \\ 1 \\ 0 \end{matrix}$$

$$P_3 = \binom{3}{3} \begin{matrix} 0 \\ 0 \\ 0 \\ 3 \end{matrix}$$

$$x(u) = 2_0(1-u)^3 + 6u(1-u)^2 + 12u^2(1-u) + 6u^3$$

$$y(u) = (1-u)^3 + 9u(1-u)^2 + 9u^2(1-u) + 4u^3$$

u

0.2

0.2

0.2

0.2

0.2

$u$	$x(u)$	$y(y)$
0	1	1
0.2	1.712	0.984
0.4	2.616	2.632
0.6	3.664	3.088
0.8	4.808	3.496
1	6	4

h = 0.2

u	$x(u)$	$y(u)$
0	1	1
0.2	1.712	0.984
0.4	2.616	2.632
0.6	3.664	3.088
0.8	4.808	3.496
1	6	4

$P_0(1,1)$   
 $P_1(2,3)$   
 $P_2(2,3)$   
 $P_3(6,4)$



# What do you mean by fractal

- Fractals are very complex pictures generated by a computer from a single formula. They are created using iterations. This means one formula is repeated with slightly different values over and over again, taking into account the results from the previous iteration.

# Where are Fractals are used ?

- **Astronomy** – For analyzing galaxies, rings of Saturn, etc.
- **Biology/Chemistry** – For depicting bacteria cultures, Chemical reactions, human anatomy, molecules, plants,
- **Others** – For depicting clouds, coastline and borderlines, data compression, diffusion, economy, fractal art, fractal music, landscapes, special effect, etc.

# Generation of Fractals

- Fractals can be generated by repeating the same shape over and over again as shown in the following figure. In figure a, shows an equilateral triangle. In figure b, we can see that the triangle is repeated to create a star-like shape. In figure c, we can see that the star shape in figure b is repeated again and again to create a new shape.
- We can do unlimited number of iteration to create a desired shape. In programming terms, recursion is used to create such shapes.



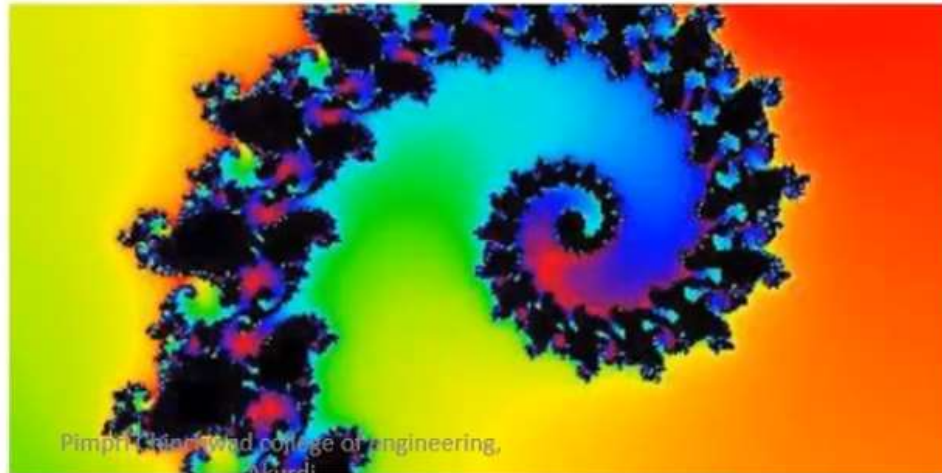
(a) Zeroth Generation



(b) First Generation



(c) Second Generation



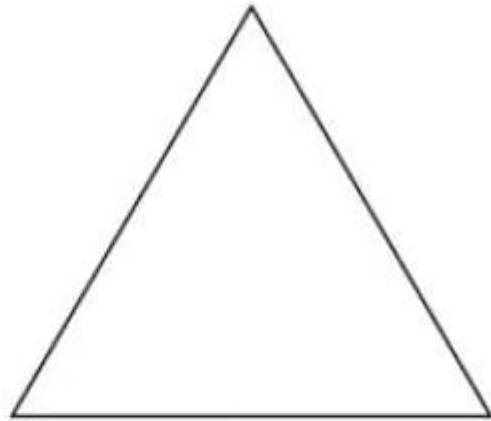
Pimpri Chinchwad college of engineering,  
Pune

# Geometric Fractals

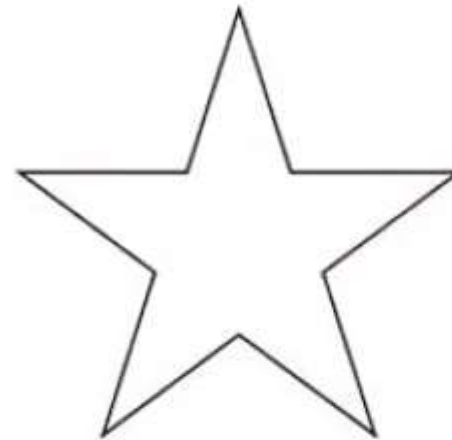
- Geometric fractals deal with shapes found in nature that have non-integer or fractal dimensions. To geometrically construct a deterministic nonrandom self-similar fractal, we start with a given geometric shape, called the initiator. Subparts of the initiator are then replaced with a pattern, called the generator.








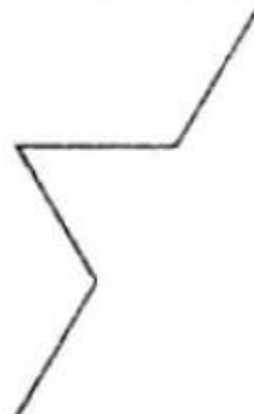

**Initiator**



**Generator**

- As an example, if we use the initiator and generator shown in the above figure, we can construct good pattern by repeating it. Each straight-line segment in the initiator is replaced with four equal-length line segments at each step. The scaling factor is  $1/3$ , so the fractal dimension is  $D = \ln 4 / \ln 3 \approx 1.2619$ .

Also, the length of each line segment in the initiator increases by a factor of  $4/3$  at each step, so that the length of the fractal curve tends to infinity as more detail is added to the curve as shown in the following figure


Segment Length = 1	Segment Length = $1/3$	Segment Length = $1/9$
		
Length = 1	Length = $4/3$	Length = $16/9$

# Classification of fractals


- The fractals can be classified as
  1. Self similar
  2. Self affine
  3. Invariant



# Self affine

- Self-affinity is distorted self-similarity. In case of self-similarity, the objects is scaled by the same amount in all directions, but in self-affinity scaling is not necessary identical in all directions. 
- An affine transformation is imposed on self-affine structure to witness its self-similarity.
- These fractals are commonly used to model water, clouds and terrain.

# Invariant fractals

- In these fractals, nonlinear transformation is used.
- It includes self squaring fractals 

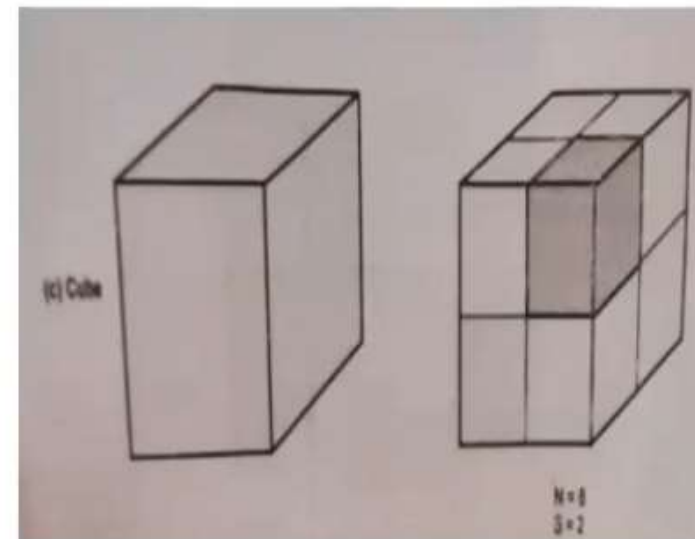
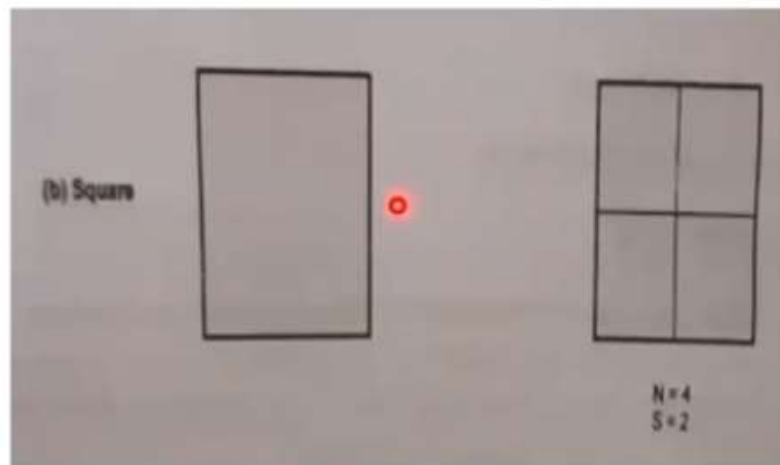
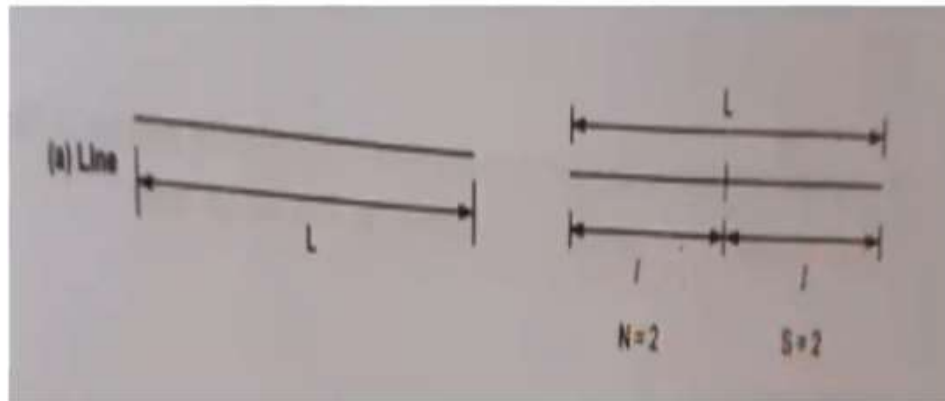
# Topological Dimension

- Consider an object composed of elastic or clay. If the object can be deformed into a line or line segment we assign its dimension  $D_t = 1$ . If object deforms into a plane or half plane or disk we assign its dimension  $D_t = 2$ . If object deforms into all space or half space or sphere, we assign its dimension  $D_t = 3$ . The dimension  $D_t$  is referred to as the topological dimension.

# Fractal Dimension

- It is the second measure of an object dimension. Imagine that a line segment of length  $L$  is divided into  $N$  identical pieces.
- The length of each line segment  $l$  can be given as  $l = L/N$
- The ratio of length of original line segment and the length of each part of the line segment is referred to as scaling factor and is given as  $s = L/l$

# Following figures show scaling of objects in various dimensions





- From above two equations we can write

$$N = s$$

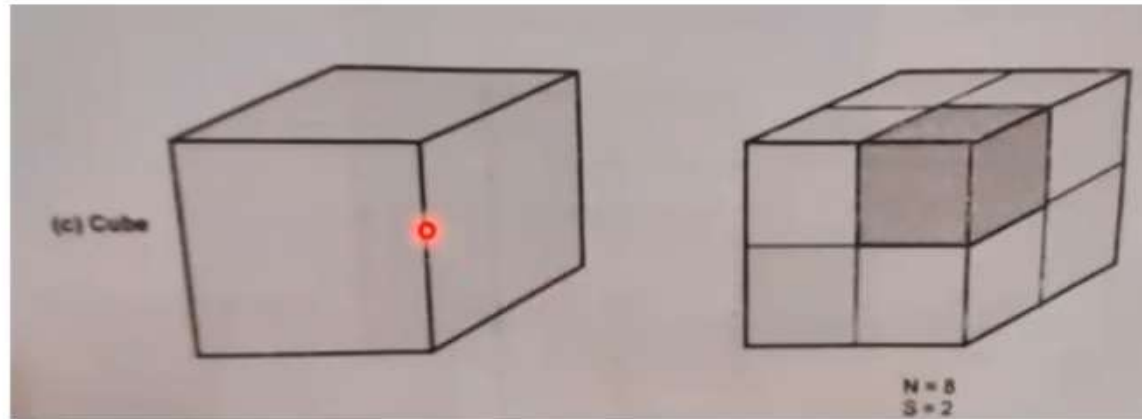
i.e  $N = s^1$

- In other words we can say that if we scale a line segment by a factor  $1/s$  then we have to add  $N$  pieces together to get the original line segment.
- If we scale square object by a factor  $1/s$  we will get a small square. In general we can write

$$N = s^2$$

- Similarly for cubical object we have, (refer the below diagram)

$$N = s^3$$



- We have seen that we can specify the dimension of the object by variable  $D$ . Here the exponent of  $s$  is a measure of object dimension. Thus we can write

$$N = s^D$$

solving for  $D$  we get,

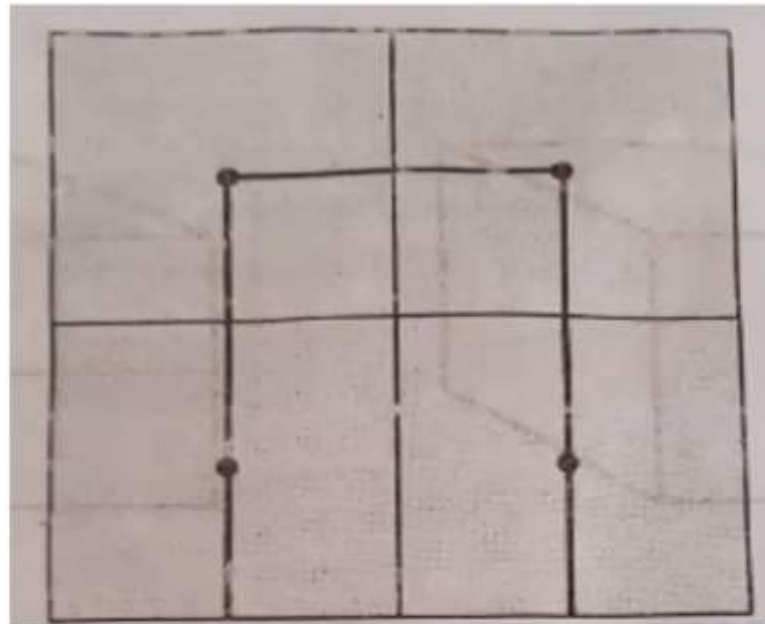
$$D = \log N / \log s$$

- This  $D$  is called fractal dimension

# Hilbert's Curve

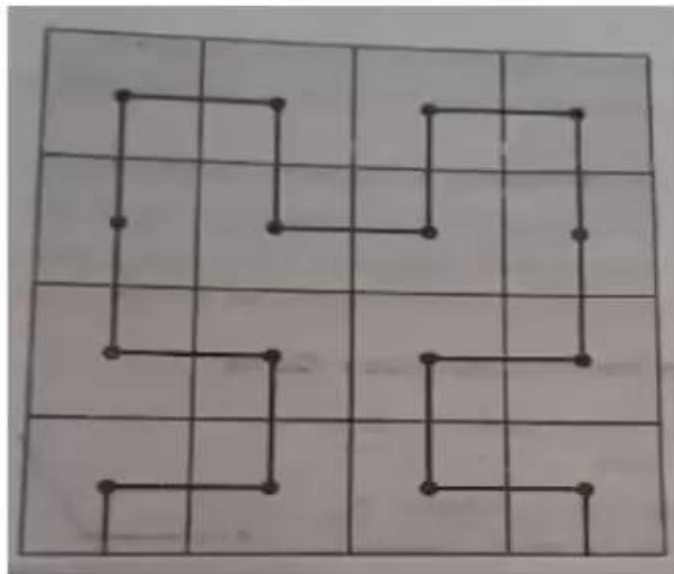
- The Hilbert curve is a space filling curve that visits every point in a square grid with a size of  $2 \times 2$ ,  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ , or any other power of 2.
- It was first described by David Hilbert in 1892. Applications of the Hilbert curve are in image processing: especially image compression.
- It has advantages in those operations where the coherence between neighboring pixels is important .

- The Hilbert's curve can be constructed by following successive approximations.
- If a square is divided into four quadrants we can draw the first approximation to the Hilbert's curve by connecting centre points of each quadrant as shown in figure.



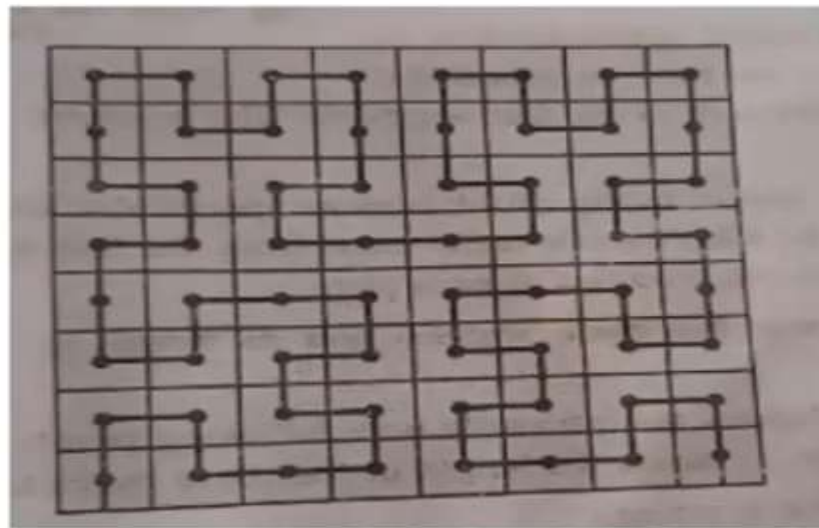
- The above figure shows the first approximation to Hilbert's curve

- The second approximation to the Hilbert's curve can be drawn by further subdividing each of the quadrants and connecting their centers before moving to next major quadrant.



- The above figure shows second approximation to Hilbert's curve.

- The third approximation subdivides the quadrants again.
- We can draw third approximation to Hilbert's curve by connecting the centers of finest level of quadrants before stepping to the next level of the quadrant.



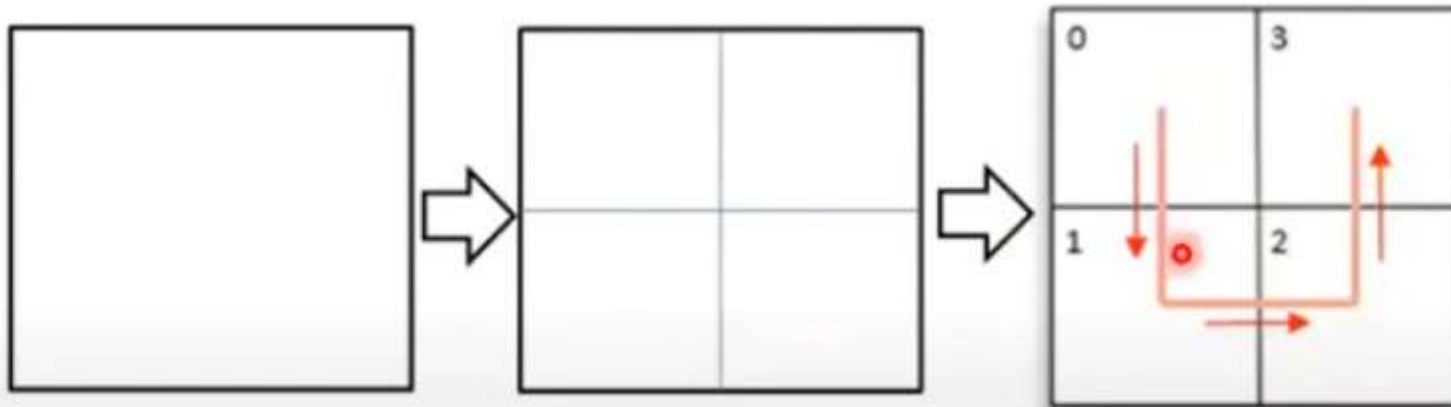
- The above figure shows the third approximation to the Hilbert's curve

From the above three figures we can easily note following points about Hilbert's curve:-

- If we infinitely extend the approximations to the Hilbert's curve, the curve fills the smaller quadrants but never crosses itself.
- The curve is arbitrarily close to every point in the square.
- The curve passes through a point on a grid, which becomes twice as fine with each subdivision.
- There is no limit to subdivision and therefore length of curve is infinite.
- With each subdivision length of curve increases by factor of 4.
- At each subdivision the scale changes by 2 but length changes by 4 therefore for Hilbert's curve topological dimension is one but the fractal dimension is 2.

# Hilbert Curve..

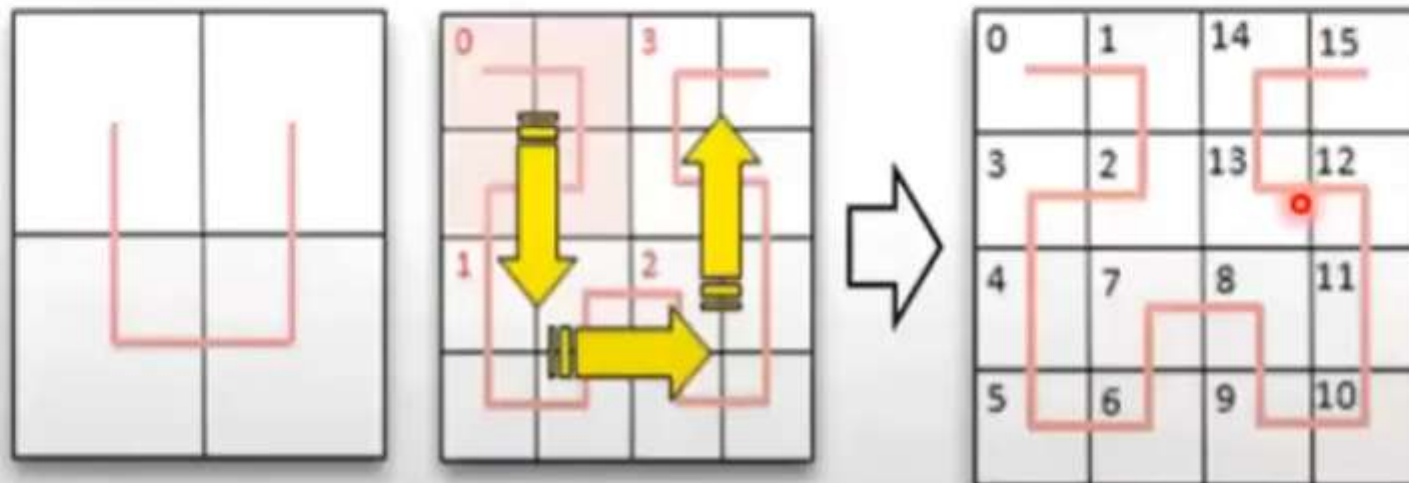
- In the first approximation, we are dividing the square into four quadrants.





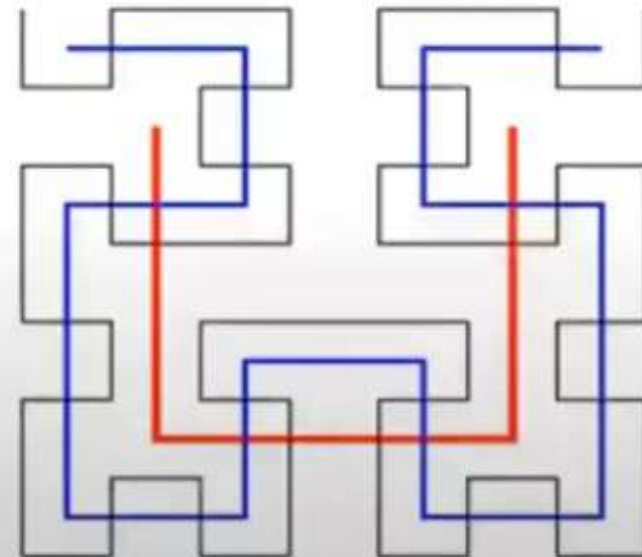
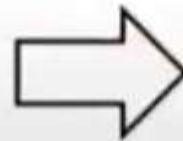
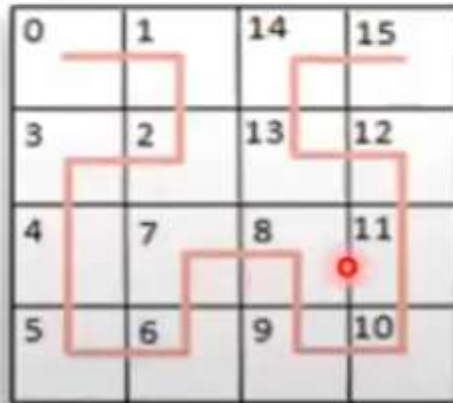
# Hilbert Curve..

- And draw the curve which connects the center points of each these finer subdivisions before moving the next major quadrant.



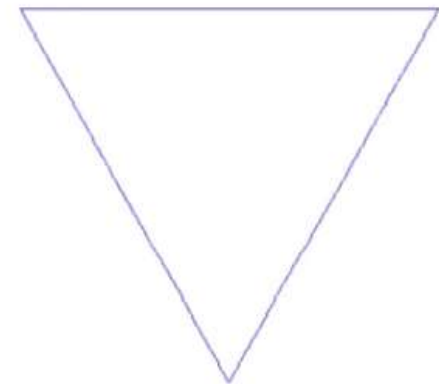
# Hilbert Curve..

- The third approximation again subdivides each quadrant and process continues.



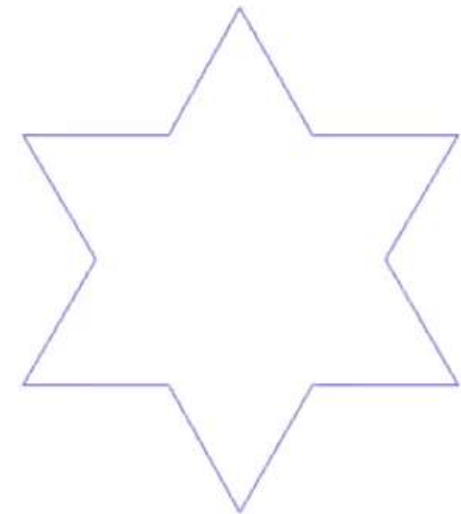
# What is Koch Curve?

- This is one of the earliest fractal curves to have been described
- It is based on the Koch curve, which appeared in a 1904 paper titled “On a continuous curve without tangents, constructible from elementary geometry” by the Swedish mathematician Helge von Koch.
- The progression for the snowflake’s perimeter diverges to infinity while the progression for the area of the snowflake converges to  $\frac{8}{5}$  times the area of the original triangle...
  - The snowflake has a finite area bounded by an infinitely long line



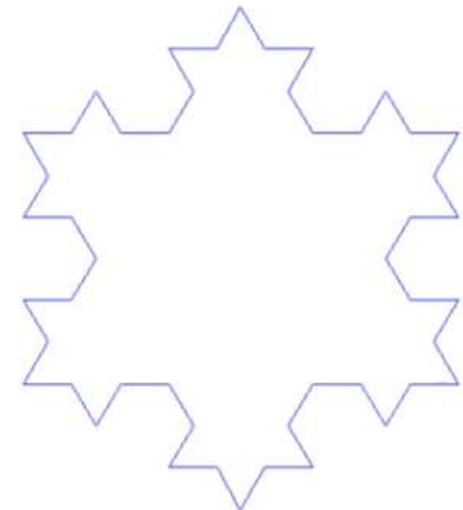
# What is Koch Curve?

- This is one of the earliest fractal curves to have been described
- It is based on the Koch curve, which appeared in a 1904 paper titled “On a continuous curve without tangents, constructible from elementary geometry” by the Swedish mathematician Helge von Koch.
- The progression for the snowflake’s perimeter diverges to infinity while the progression for the area of the snowflake converges to  $8/5$  times the area of the original triangle...
  - The snowflake has a finite area bounded by an infinitely long line



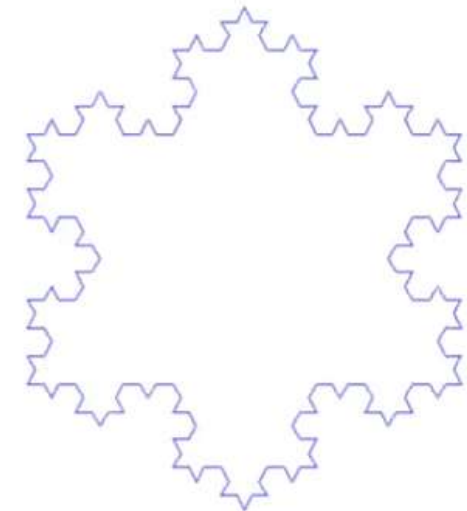
# What is Koch Curve?

- This is one of the earliest fractal curves to have been described
- It is based on the Koch curve, which appeared in a 1904 paper titled “On a continuous curve without tangents, constructible from elementary geometry” by the Swedish mathematician Helge von Koch.
- The progression for the snowflake’s perimeter diverges to infinity while the progression for the area of the snowflake converges to  $\frac{8}{5}$  times the area of the original triangle...
  - The snowflake has a finite area bounded by an infinitely long line



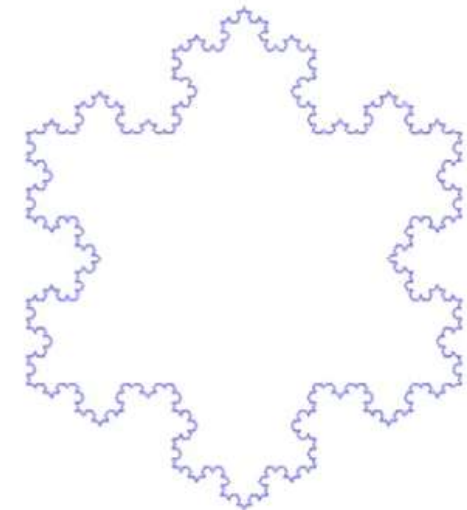
# What is Koch Curve?

- This is one of the earliest fractal curves to have been described
- It is based on the Koch curve, which appeared in a 1904 paper titled “On a continuous curve without tangents, constructible from elementary geometry” by the Swedish mathematician Helge von Koch.
- The progression for the snowflake’s perimeter diverges to infinity while the progression for the area of the snowflake converges to  $8/5$  times the area of the original triangle...
  - The snowflake has a finite area bounded by an infinitely long line



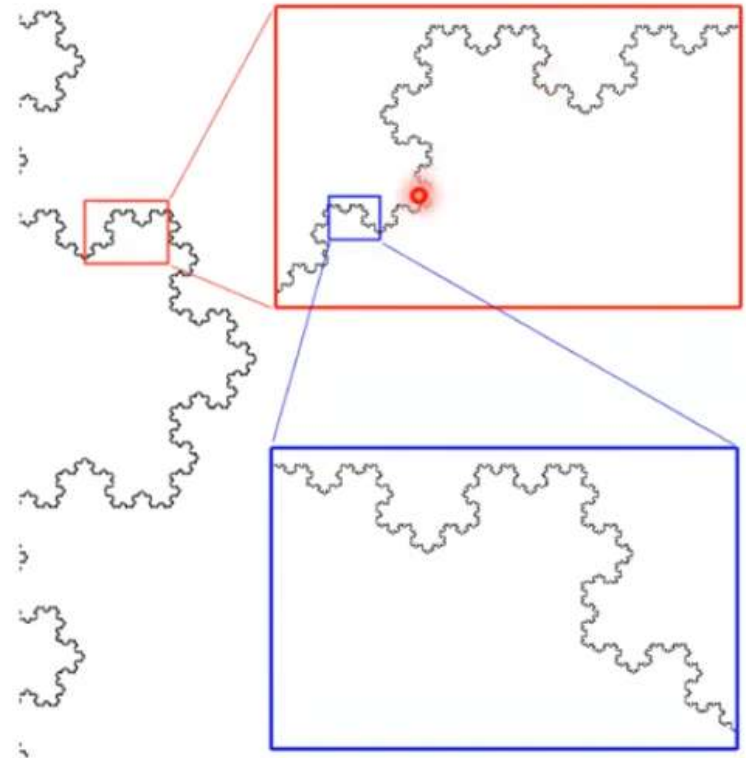
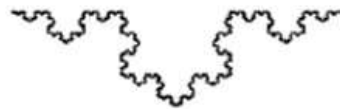
# What is Koch Curve?

- This is one of the earliest fractal curves to have been described
- It is based on the Koch curve, which appeared in a 1904 paper titled “On a continuous curve without tangents, constructible from elementary geometry” by the Swedish mathematician Helge von Koch.
- The progression for the snowflake’s perimeter diverges to infinity while the progression for the area of the snowflake converges to  $8/5$  times the area of the original triangle...
  - The snowflake has a finite area bounded by an infinitely long line



# Koch Curve

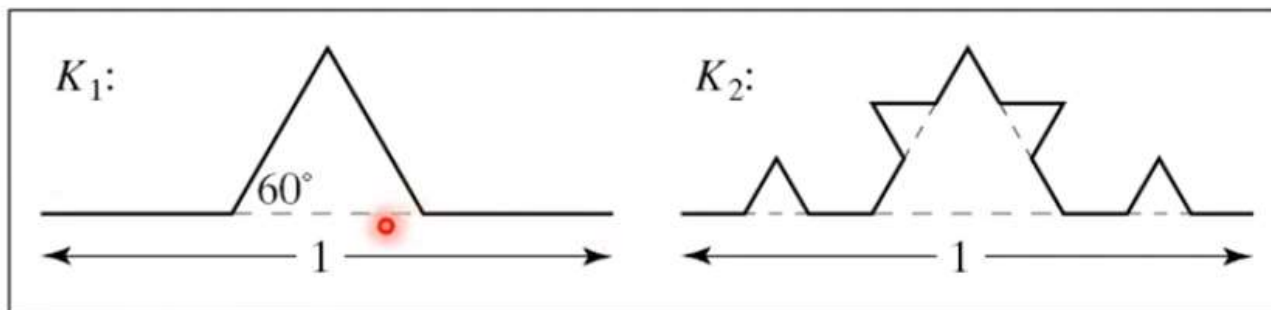
- Self-similar:
  - zoom in on any portion
  - If  $n$  is large enough, shape still same
  - On computer, smallest line segment  $>$  pixel spacing






# Koch Curve

- The Koch curve can be drawn by dividing line into 4 equal parts with scaling factor  $1/3$
- The middle two segments are adjusted that they form adjacent sides of an equilateral triangle – this is the first approximation to the Koch curve
- Repeat the process for each of the four segments



# Koch Curve

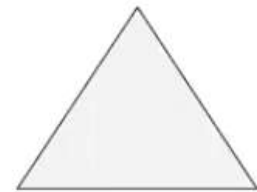
- The resultant curve is known as triadic Koch curve
- The name is derived from the fact that the middle thirds of the line segments are modified at each step 
- The length of the resultant curve is  $\frac{16}{9}$  times the original length!

# Some Observations....

- Each repetition increases the length of the curve by factor  $4/3$
- Length of the curve is infinite
- Unlike Hilbert's curve, this does not fill an area
- It does not deviate much from its original shape
- If we reduce the scale of the curve by 3, we find the curve that looks like the one; but we must assemble 4 such curves to make the original
  - $4 = 3^D$
  - $D = \log_3 4 = 1.2618$
  - Fractal dimension

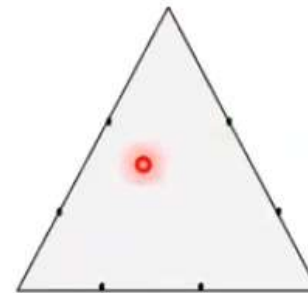
# Geometric Construction

- **Step1:**
- Draw an equilateral triangle.
- It's best if the length of the sides are divisible by 3, because of the nature of this fractal.



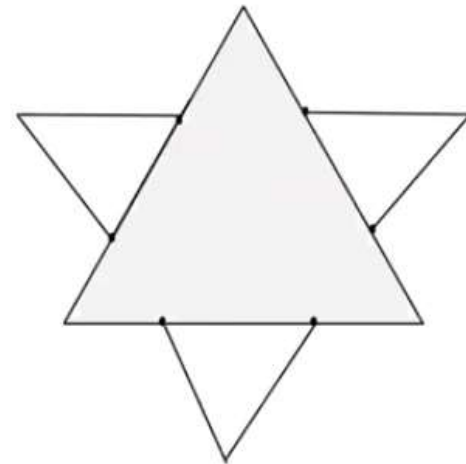
# Geometric Construction

- **Step 2:**
- Divide each side in three equal parts.



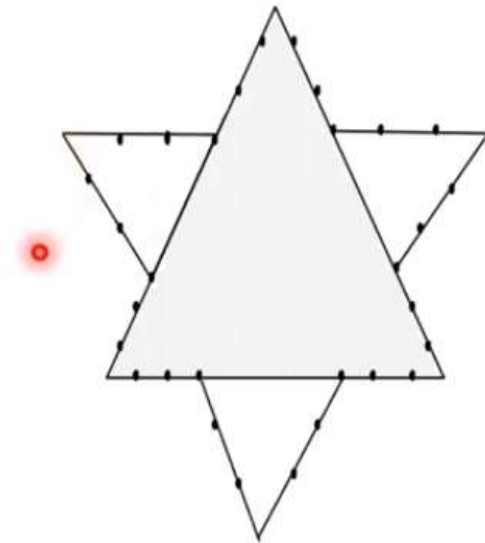
# Geometric Construction

- **Step 3:**
- Draw an equilateral triangle on each middle part.
- Measure the length of the middle third to know the length of the sides of these new triangles.

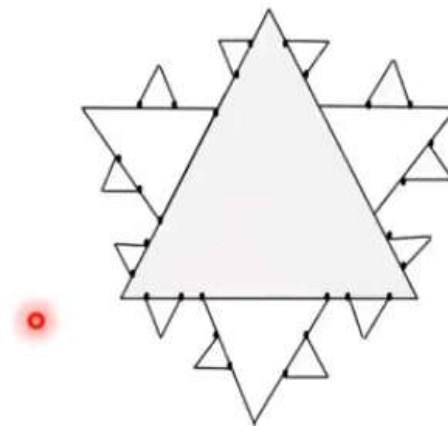


# Geometric Construction

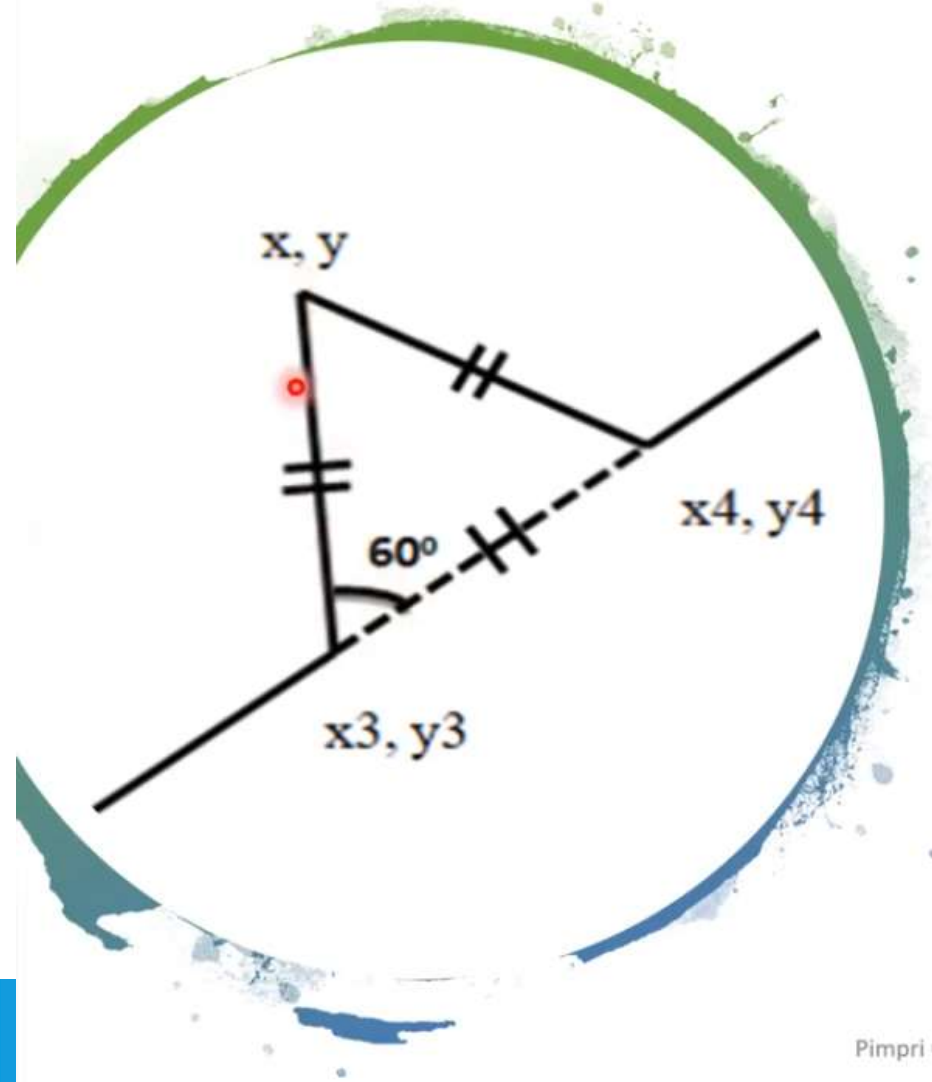
- **Step 4:**
- Divide each outer side into thirds. You can see the 2nd generation of triangles covers a bit of the first.
- These three line segments shouldn't be parted in three.



- **Step 5:**
- Draw an equilateral triangle on each middle part
- Continue...







- **Step 1:** In Iteration 0, we have a horizontal line.
- **Step 2:** In Iteration 1, line is divided into 3 equal parts. Middle part of a line is rotated in  $60^\circ$ , because it forms a perfect equilateral triangle
- Here,  $(x_1, y_1)$  and  $(x_2, y_2)$  is accepted from user.
- Now, we can see line is divided into 3 equal segments:  $\text{segment}((x_1, y_1), (x_3, y_3))$ ,  $\text{segment}((x_3, y_3), (x_4, y_4))$ ,  $\text{segment}((x_4, y_4), (x_2, y_2))$  in above figure.
- Coordinates of middle two points will be calculated as follows:
  - $x_3 = (2*x_1+x_2)/3$ ;  $y_3 = (2*y_1+y_2)/3$ ;
  - $x_4 = (x_1+2*x_2)/3$ ;  $y_4 = (y_1+2*y_2)/3$ ;
- In our curve, middle segment  $((x_3, y_3), (x_4, y_4))$  will not be drawn. Now, in order to find out coordinates of the top vertex  $(x, y)$  of equilateral triangle, we have rotated the point  $(x_4, y_4)$  with respect to arbitrary point  $(x_3, y_3)$  by angle of 60 degree in anticlockwise direction. After performing this rotation, we will get rotated coordinates  $(x, y)$  as:
  - $x = x_3 + (x_4 - x_3) * \cos \theta + (y_4 - y_3) * \sin \theta$
  - $y = y_3 - (x_4 - x_3) * \sin \theta + (y_4 - y_3) * \cos \theta$

- Step 3:

- In iteration 2, you will repeat step 2 for every segment obtained in iteration1.



