Module 5 - Three Dimensional Geometric Transformations, Curves and Fractals

By Poonam Pangarkar
Department CSE Data Science
APSIT

Curves

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

A curve is an infinitely large set of points. Each point has two neighbors except endpoints. Curves can be broadly classified into three categories −explicit, implicit, and parametric curves.

<u>Implicit Curves</u>

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

<u>Explicit Curves</u>

A mathematical function y =f(x), x can be plotted as a curve. Such a function is the 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.

Curves

<u>Parametric Curves</u>

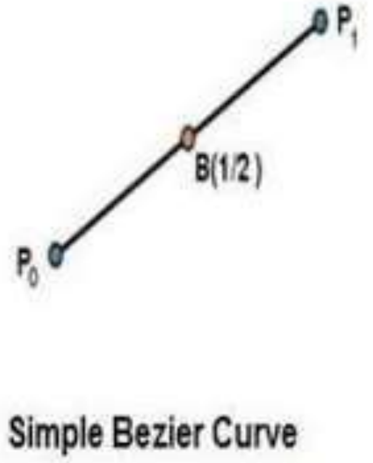
Curves having parametric form are called parametric curves. The explicit and implicit curve representations can be used only when the function is known. In practice the parametric curves are used. A two-dimensional parametric curve has the following form −

P t = f t, g t or P t = x t, y t

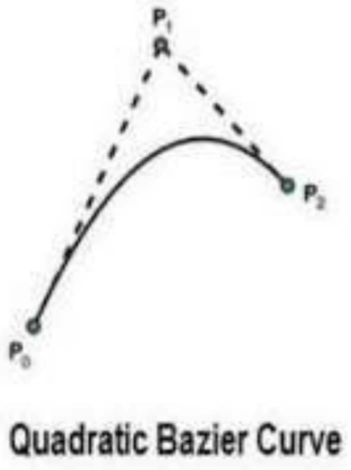
The functions f and g become the x,y coordinates of any point on the curve, and the points are obtained when the parameter t is varied over a certain interval [a, b], normally [0, 1].

Bezier Curve

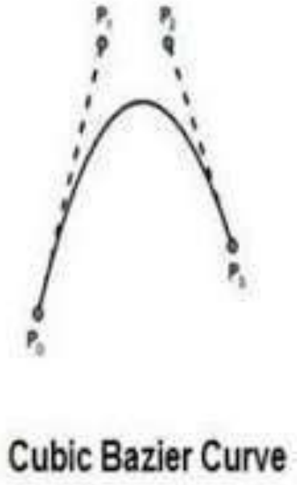
- A **Bezier curve** is a parametric **curve** used in computer graphics and related fields. The **curve**, which is related to the Bernstein polynomial, is named after Pierre **Bezier**, who used it in the 1960s for designing **curves** for the bodywork of Renault cars.



Simple Bezier Curve



Quadratic Bazier Curve



Cubic Bazier Curve

- Let suppose we are given (n+1) control points position then $P_i = (x_i, y_i, z_i)$ from 0 to n.
- These coordinate points can be blended to produce the following position vector P(u), which describes the path of an approximation.
- So Bezier polynomial function between P_0 to P_n is

$$\mathbf{P(u)} = \sum_{i=0}^{n} \mathbf{P_i\, B_{i,n}(u)} \qquad 0 \le u \le 1$$

where, P_i = control points

$B_{i,n}$ / $BEZ_{i,n}$ = Bezier function or Barstein Polynomials.

The Bernstein polynomial or the Bezier function is very imp function will dictate the smoothness of this curve & the weight will be dictated by boundary conditions.

$BEZ_{i,n}(u) = {}^{n}C_{i} . u^{i} (1-u)^{n-i}$

where ${}^{n}C_{i} =$ $\dfrac{n!}{i!(n-i)!}$ [Binomial Coefficient]

$$\mathbf{P(u) = \quad P_i\, B_{i,n}(u)}$$

$P(u) = \sum_{i=0}^{n} = P_0 B_{0,3}(u) + P_1 B_{1,3}(u) + P_2 B_{2,3}(u) + P_3 B_{3,3}(u)$ ----------- (1)

where, $B_{0,3}(u) = {}^{3}C_{0} . u^{0} (1-u)^{3}$

$= 3! / 0! (3-0)! .1 (1-u)^{3}$

$= (1-u)^{3}$

Similarly,

$B_{1,3} = 3u.\ (1-u)^2$

$B_{2,3} = 3u^2.\ (1-u)$

$B_{3,3} = u^3$

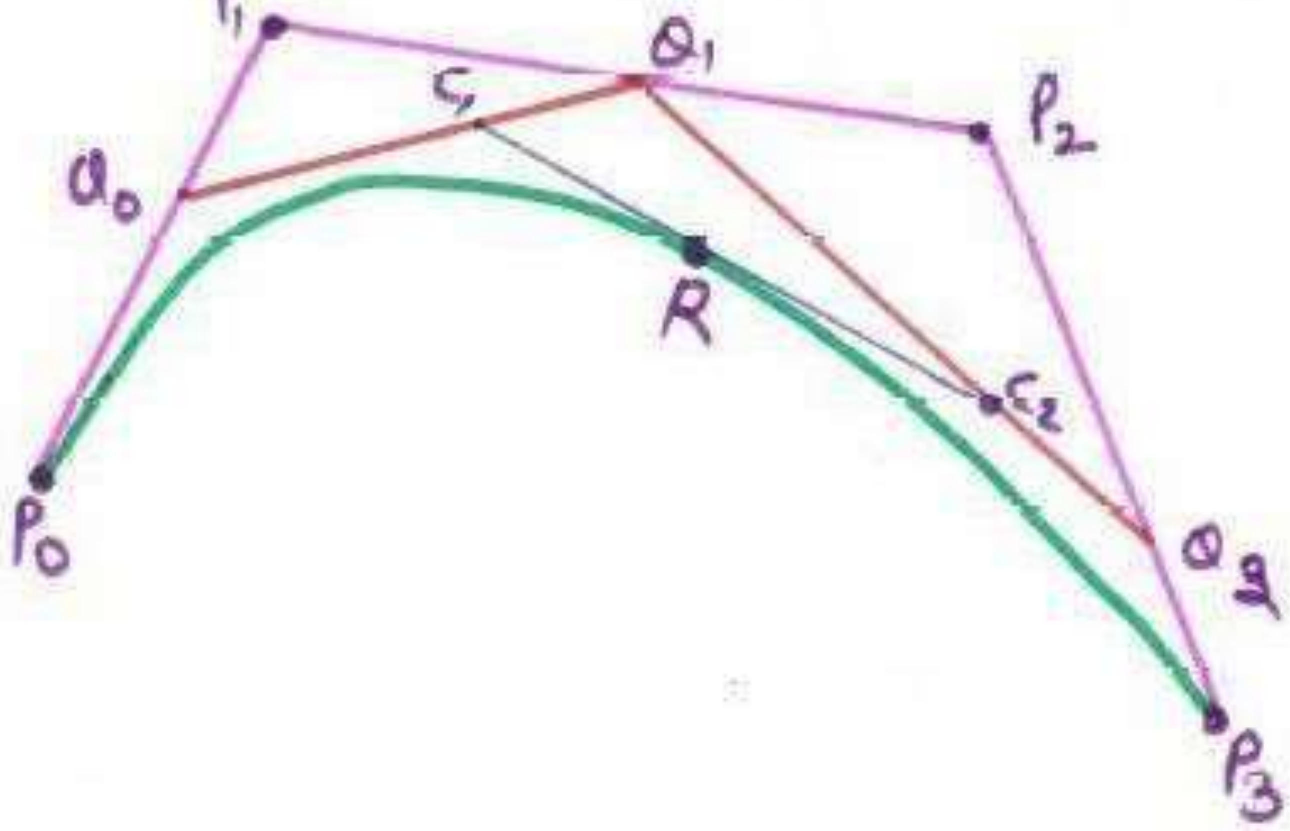
Now substituting these value in equation 1 we get

$\mathbf{P(u) = P_0\ (1-u)^3 + P_1\ 3u.\ (1-u)^2 + P_2\ 3u^2.\ (1-u) + P_3\ u^3}$

$x(u) = (1-u)^3\ x_0 + 3u(1-u)^2\ x_1 + 3u^2(1-u)\ x_2 + u^3\ x_3$

$y(u) = (1-u)^3\ y_0 + 3u(1-u)^2\ y_1 + 3u^2(1-u)\ y_2 + u^3\ y_3$

$z(u) = (1-u)^3\ z_0 + 3u(1-u)^2\ z_1 + 3u^2(1-u)\ z_2 + u^3\ z_3$



P_1
Q_1
C_1
P_2
Q_0
R
C_2
P_0
Q_2
P_3

By using a line parametric equation we can derive Bezier Curve equation for any no. of control points -

$Q_0 = (1-u)\ P_0 + u\ P_1$

$Q_1 = (1-u)\ P_1 + u\ P_2$

$Q_2 = (1-u)\ P_2 + u\ P_3$

$C_1 = (1-u)\ Q_0 + u\ Q_1$

$C_2 = (1-u)\ Q_1 + u\ Q_2$

$R = (1-u)\ C_1 + u\ C_2$

Substituting the values of C_1, C_2, Q_0, Q_1, Q_2 in R we get-

$R = (1-u)\,[(1-u)\,Q_0 + u\,Q_1] + u\,[(1-u)\,Q_1 + u\,Q_2]$

$R = (1-u)^2\,Q_0 + u\,(1-u)\,Q_1 + u\,(1-u)\,Q_1 + u^2\,Q_2$

$R = (1-u)^2\,Q_0 + 2\,u\,(1-u)\,Q_1 + u^2\,Q_2$

$R = (1-u)^2\,[(1-u)\,P_0 + u\,P_1] + 2\,u\,(1-u)\,[(1-u)\,P_1 + u\,P_2] + u^2\,[(1-u)\,P_2 + u\,P_3]$

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

$\mathbf{R = (1-u)^3\,P_0 + 3u\,(1-u)^2\,P_1 + 3\,u^2\,(1-u)\,P_2 + u^3\,P_3}$

Thus we have received the same equation which we get from Bernstein Polynomial Form

Matrix Representation

- The equation of Bezier Curve is -

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

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

$$P(u) = P_0\,(-u^3+3u^2-3u+1) + P_1\,(3u^3-6u^2+3u) + P_2\,(-3u^3+3u^2) + P_3\,u^3$$

$$P(u) = [u^3 \quad u^2 \quad u \quad 1]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

$P(u) = U.\,M_{BEZ}.\,G_{BEZ}$

where M_{BEZ} & G_{BEZ} represents Bezier basis matrix and Bezier geometric matrix respectively.

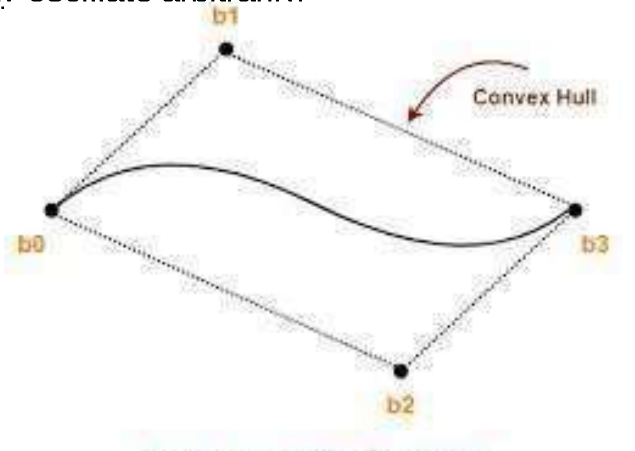
Properties of Bezier Curves

- Always interpolates first and last control points and approximate the remaining two
- The slope of the derivative at the beginning is along the line joining the first two points and slope of the derivative at the end is along the line joining the last two points
- The degree of the curve is 1 less than the number of control points
- Bezier curve always satisfies convex hull property
- Bezier curves do not have local control, repositioning one control point changes the entire curve
- Bezier curve can fit any number of control points
- Reversing the order of control points yields the same bezier curve
- The curve begins at $\mathbf{P}_0$ and ends at $\mathbf{P}_n$ this is the so-called *endpoint interpolation* property.
- The curve is a straight line if and only if all the control points are collinear.

Convex Hull Property

Convex hull is an enclosed polygon which passes through the outermost control points we say that the curve is satisfying convex hull property if the entire curve lies within the convex Hull.

Curves having convex hull property indicates that the curves grows smoothly within the convex hull is curves which do not satisfy the convex hull property may oscillate arbitrarily.



Bezier Curve With Convex Hull

B-Spline Curves

- The Bezier-curve produced by the Bernstein basis function has limited flexibility.
- First, the number of specified polygon vertices fixes the order of the resulting polynomial which defines the curve.
- The second limiting characteristic is that the value of the blending function is nonzero for all parameter values over the entire curve.
- The B-spline basis contains the Bernstein basis as the special case. The B-spline basis is non-global.

B-Spline Curves

A B-spline curve is defined as a linear combination of control points Pi and B-spline basis function $N_{i,}$ k (t) given by

$$C(t) = \sum_{i=0}^{n} P_i N_{i,k}(t), \qquad n \geq k-1,$$
$$t \in [tk-1, tn+1]$$

- Where,{pi: i=0, 1, 2....n} are the control points
- k is the order of the polynomial segments of the B-spline curve. Order k means that the curve is made up of piecewise polynomial segments of degree k - 1,
- the Ni,k(t) are the "normalized B-spline blending functions". They are described by the order k and by a non-decreasing sequence of real numbers normally called the "knot sequence".

B–Spline Curves

$$t_i : i = 0, \ldots n + K$$

The N$_i$, k functions are described as follows –

$$N_{i,1}(t) = \begin{cases} 1, & if\, u \,\epsilon\, [t_i, t_{i+1}) \\ 0, & Otherwise \end{cases}$$

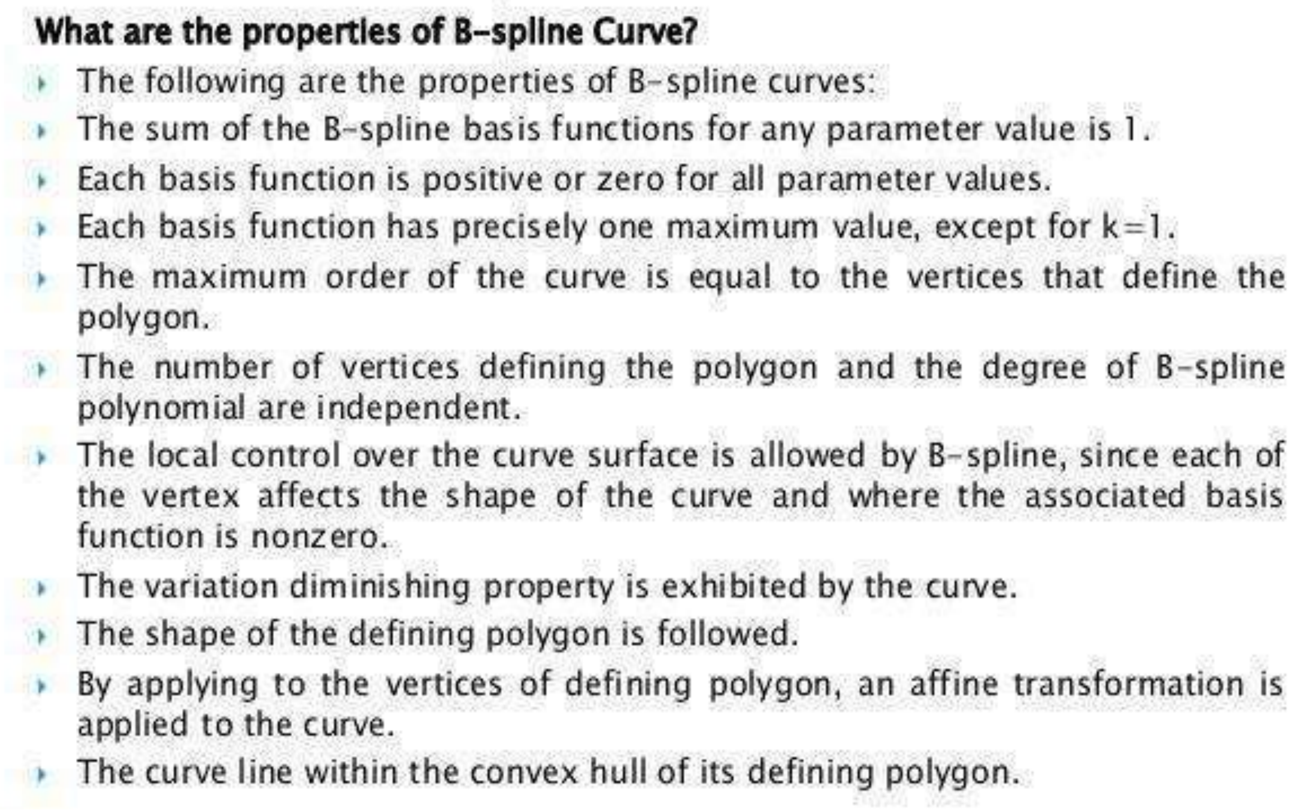
and if k > 1,

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1}} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t)$$

and

$$t \,\epsilon\, [t_{k-1}, t_{n+1})$$

B-Spline Curves



What are the properties of B-spline Curve?

- The following are the properties of B-spline curves:
- The sum of the B-spline basis functions for any parameter value is 1.
- Each basis function is positive or zero for all parameter values.
- Each basis function has precisely one maximum value, except for k=1.
- The maximum order of the curve is equal to the vertices that define the polygon.
- The number of vertices defining the polygon and the degree of B-spline polynomial are independent.
- The local control over the curve surface is allowed by B-spline, since each of the vertex affects the shape of the curve and where the associated basis function is nonzero.
- The variation diminishing property is exhibited by the curve.
- The shape of the defining polygon is followed.
- By applying to the vertices of defining polygon, an affine transformation is applied to the curve.
- The curve line within the convex hull of its defining polygon.

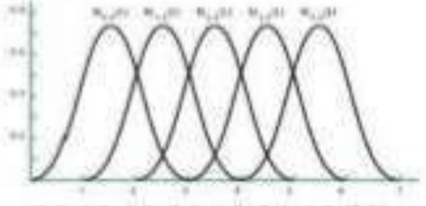
Properties of B-Spline Curve

- B-spline basis is non-global (local) effect.
- B-Spline Curve made up of n+1 control point.
- B-Spline Curve let us specify the order of basis (k) function and the degree of the resulting curve is independent on the no. of vertices.
- It is possible to change the no. of control points.
- B-Spline can be used to define both open & close curves.
- Curve generally follows the shape of defining polygon.
- The curve line with in the convex hull of its defining polygon.
- In B-Spline, we segment out the whole curve which is decided by the order(k) by formula **n-k+2** .

Uniform and non uniform B-spline curves

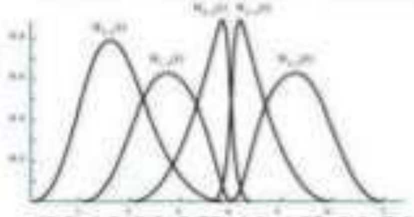
- When the spacing between knot values is constant, the resulting curve is called a uniform B-spline.
- The spacing between knot values is not constant and hence ,any values and intervals can be specified for the knot vector the curve is called a non uniform B-spline .
- Different intervals which can be used to adjust spline shapes .

• Unif. Knot Vector

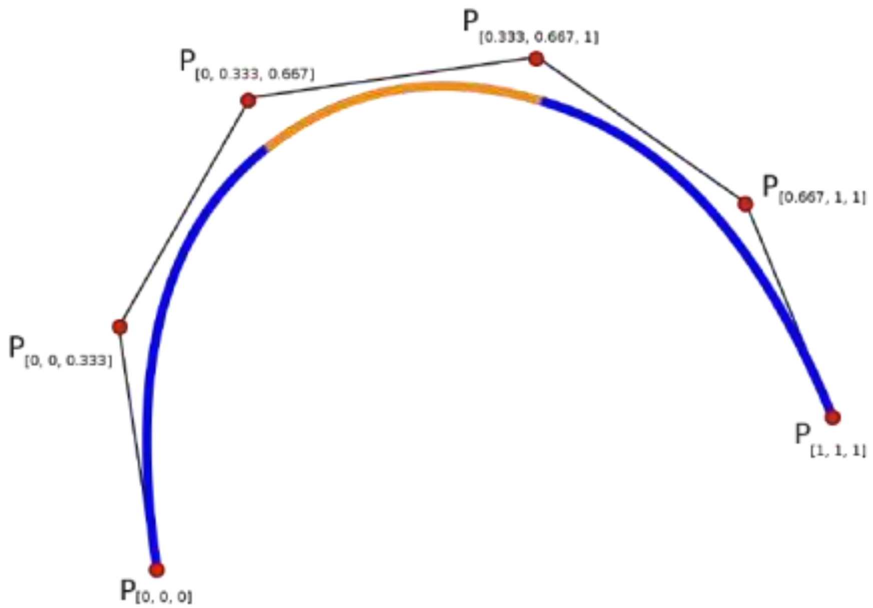


{0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0}

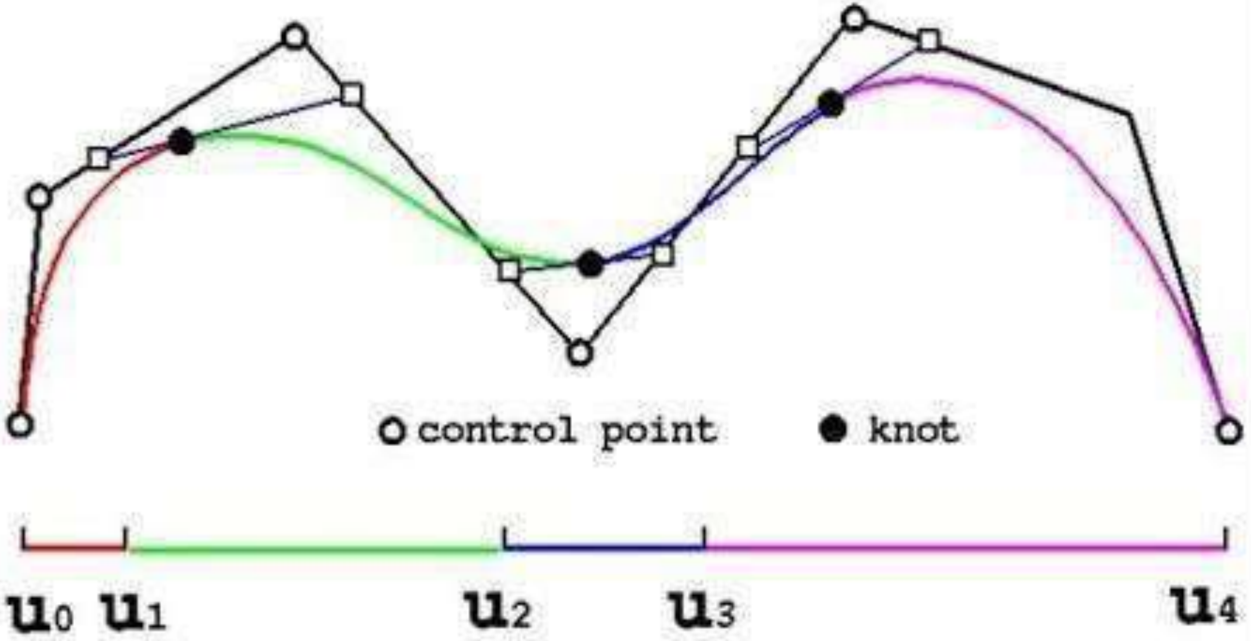
• Non-Unif. Knot Vector



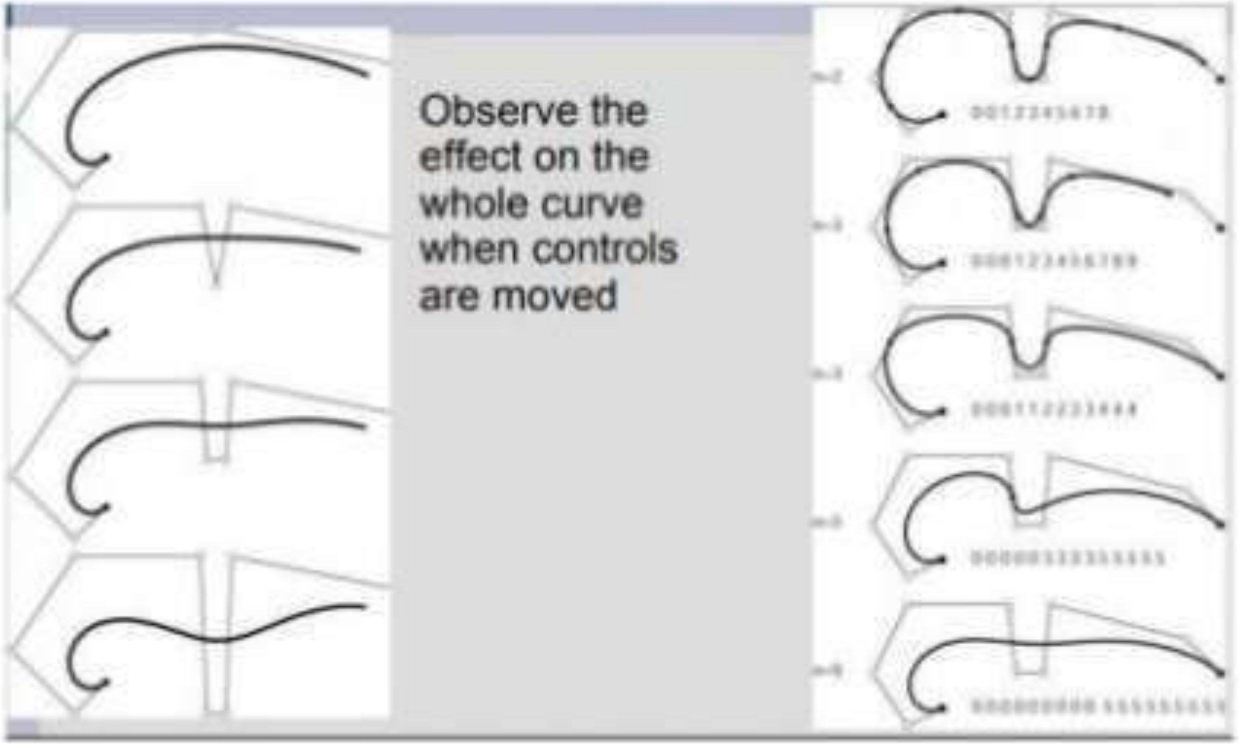
{0.0, 1.0, 2.0, 3.75, 4.0, 4.25, 6.0, 7.0}


$P_{[0.333, 0.667, 1]}$
$P_{[0, 0.333, 0.667]}$
$P_{[0.667, 1, 1]}$
$P_{[0, 0, 0.333]}$
$P_{[1, 1, 1]}$
$P_{[0, 0, 0]}$

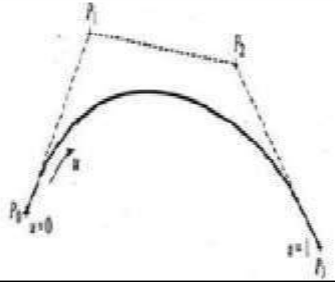
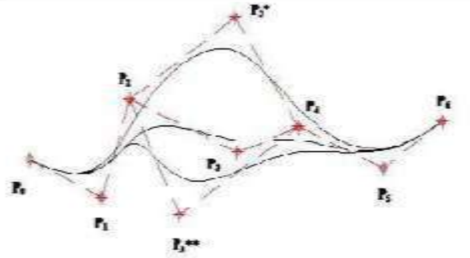
B-Spline



Bezier Curve vs B-Spline - Control

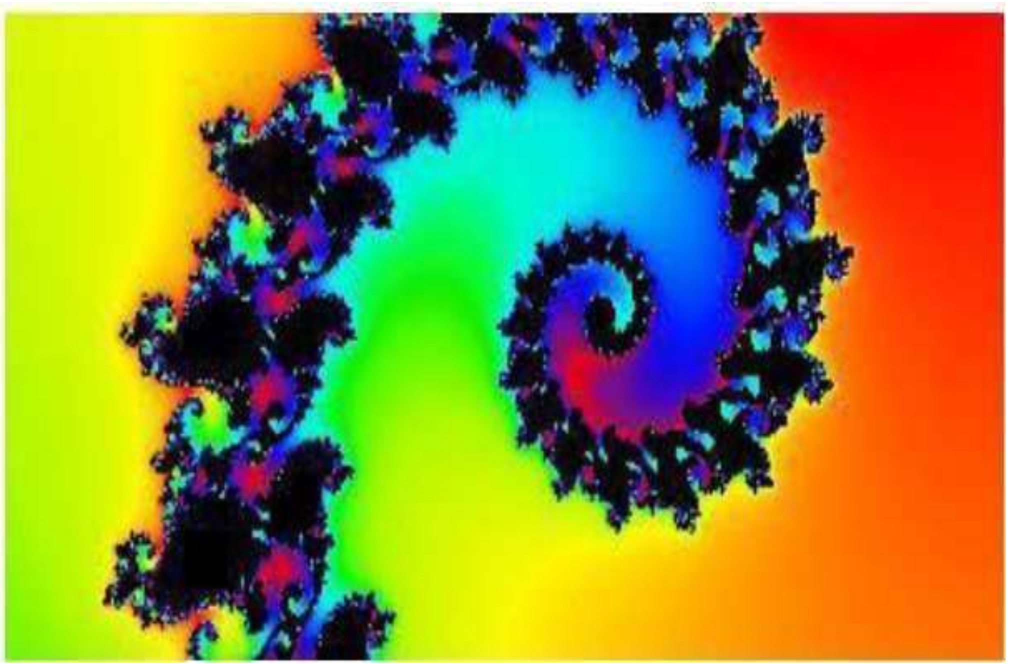
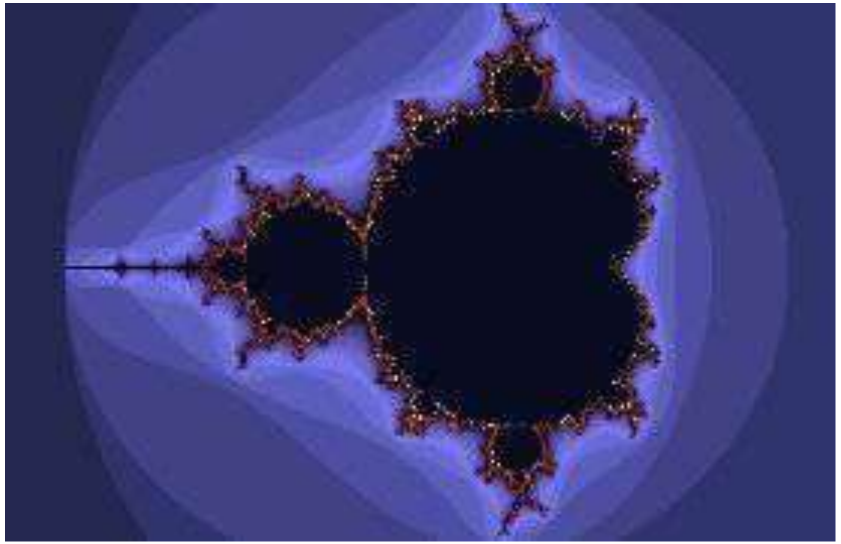


Observe the effect on the whole curve when controls are moved

Parameter	Bezier curve	B-spline curve
Definition	It is curve of n th degree polynomial with n+1 no. of data points.	It is Bezier curve with varying degree.
Formula	$P(u)=\sum_{i=0}^{n} P_i B_i,n(u), 0 \le u \le 1$	$P(u)=\sum_{i=0}^{n} P_i N_i,k(u), 0 \le u \le u_{max}$
Advantages	Curve can pass smoothly through number of data points. Smooth due to high degree continuity.	It has local control over shape of curve. Degree of curve is independent of data points.
Disadvantages	Computation required is more due to higher degree. As it has global control over shape of curve, movement of points will give different shape to the curve.	Computational time increases with complexity of curve.
Figures		 P_3^* P_2 P_4 P_6 P_3 P_0 P_1 P_5 P_3^{**}

Fractal-Geometry

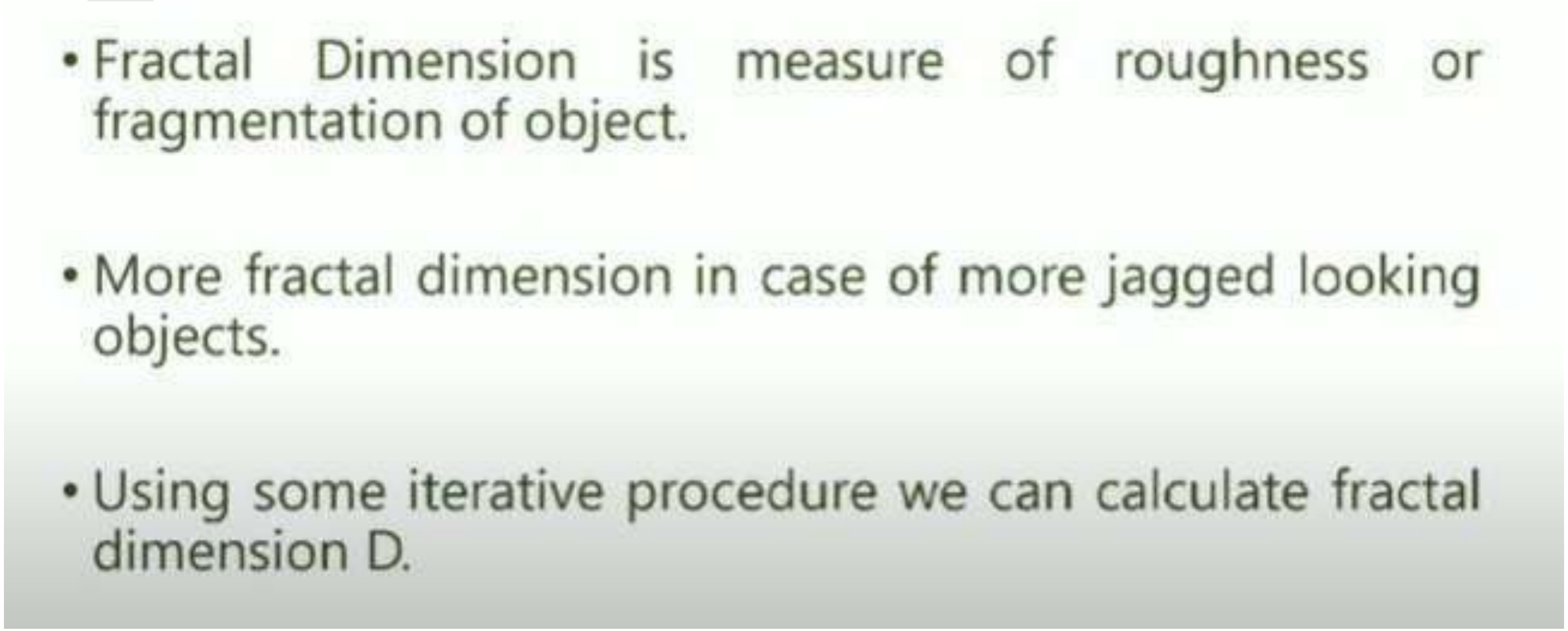
- **Fractals** are very complex pictures generated by a **computer** from a single formula. They are created using iterations.
- This means one formula is repeated with slightly different values over and over again, taking into account the results from the previous iteration.
- Fractals are used in many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

Types of Fractals

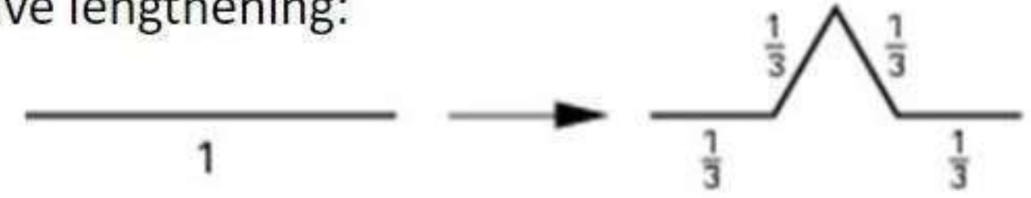
- Fractals can be classified into three groups
  - Self similar fractals
    - ⇨ These have parts that are scaled down versions of the entire object
    - ⇨ Commonly used to model trees, shrubs etc
  - Self affine fractals
    - ⇨ Have parts that are formed with different scaling parameters in each dimension
    - ⇨ Typically used for terrain, water and clouds
  - Invariant fractal sets
    - ⇨ Fractals formed with non-linear transformations
    - ⇨ Mandelbrot set, Julia set – generally not so useful

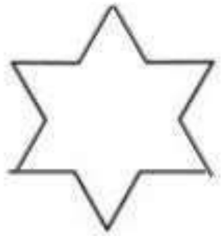
Fractal Dimension



- Fractal Dimension is measure of roughness or fragmentation of object.
- More fractal dimension in case of more jagged looking objects.
- Using some iterative procedure we can calculate fractal dimension D.

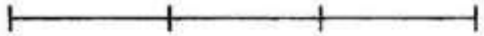
- Recursive lengthening:



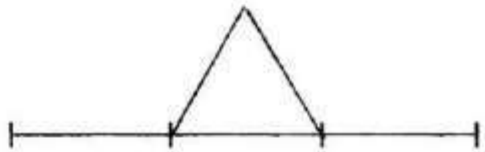


1. Start with a line.

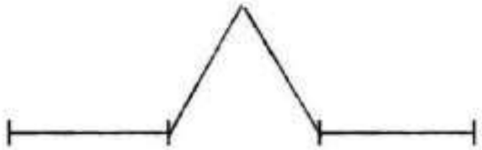
2. Divide the line into three equal parts.



3. Draw an equilateral triangle (a triangle where all the sides are equal) using the middle segment as its base.



4. Erase the base of the equilateral triangle (the middle segment from step 2).



5. Repeat steps 2 through 4 for the remaining lines again and again and again.

Calculating Fractal Dimension

- Let's look at the line on the right, when it is divided by 2, the number of self-similar pieces becomes 2. When divided by 3, the number of self-similar pieces becomes 3.

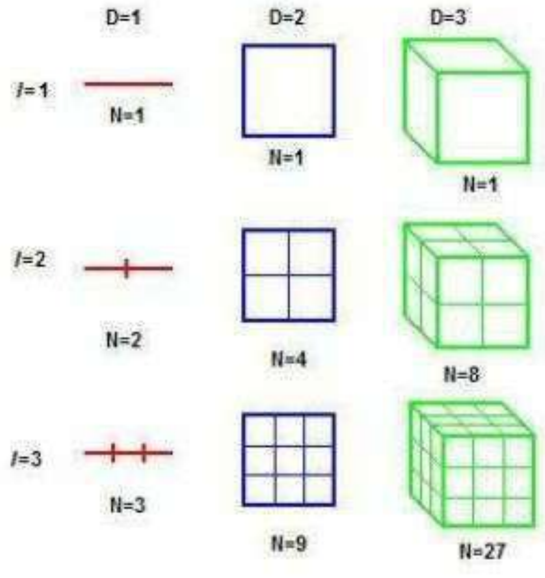
A formula is given to calculate the dimension of a given object:

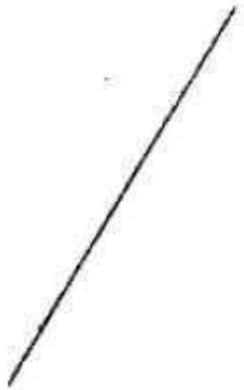
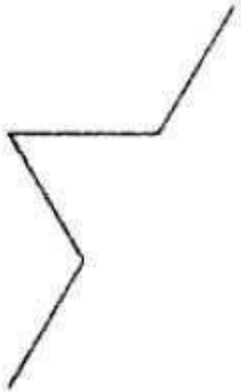
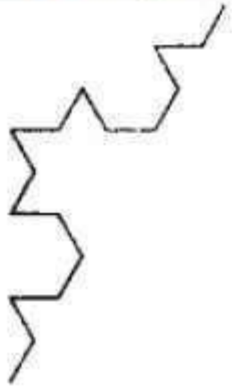
$$\frac{\log(N)}{\log(\epsilon)}$$

where N = number of self-similar pieces

ϵ = scaling factor

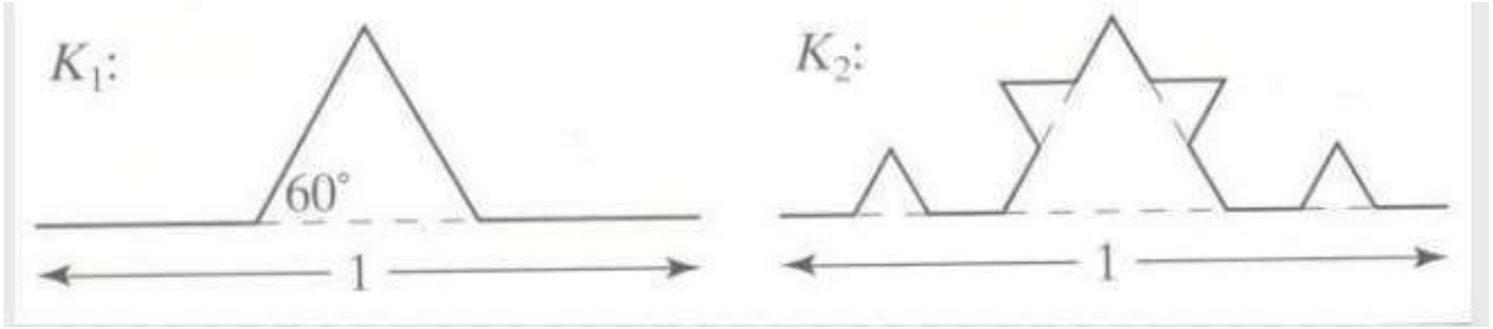
Scaling factor = 1/s

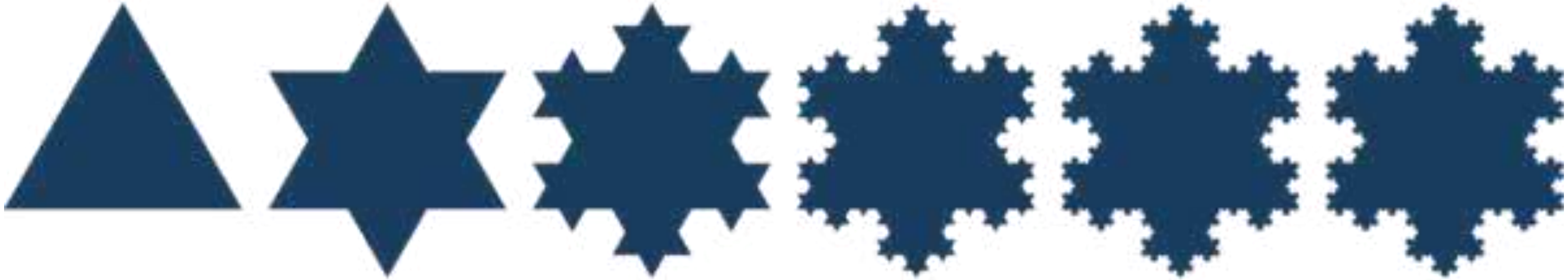


Segment Length = 1	Segment Length = 1/3	Segment Length = 1/9
		
Length = 1	Length = 4/3	Length = 16/9

Koch Curves

- It was discovered in 1904 by Helge von Koch
- It starts with straight line of length 1
- It works recursively by the following rules:
  - Divides line into 3 equal parts
  - Replace middle section with triangular bump with sides of length ⅓
  - New length = 4/3





References

- K. Weiler and P. Atherton. 1988. Hidden surface removal using polygon area sorting. In Tutorial: computer graphics; image synthesis, Kenneth I. Joy, Charles W. Grant, Nelson L. Max, and Lansing Hatfield (Eds.). Computer Science Press, Inc., New York, NY, USA 209-217
- Hearn & Baker, "Computer Graphics C version", 2nd Edition, Pearson Publication