Vidyalankar Institute of technology

Department of Computer Engineering

Introduction to Computer Graphics

Course : Computer Graphics (CG)

Sem-III

by

Sachin Deshpande

Bezier Curve

Bezier Curve may be defined as-

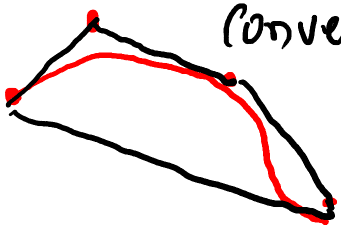
•Bezier Curve is parametric curve defined by a set of control points.

•Two points are ends of the curve.

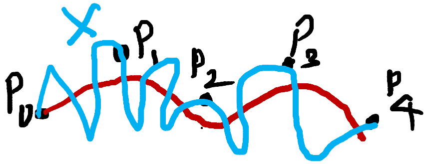
•Other points determine the shape of the curve.

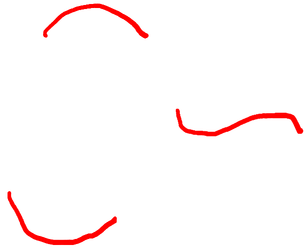
Input: set of points that dictates the shape of curve (control points)

Convex Hull

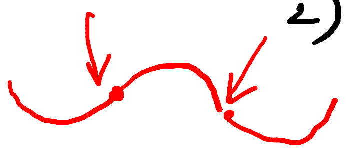


Variation diminishing property:



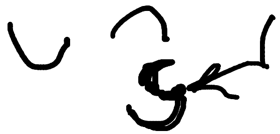


piecewise approximation process



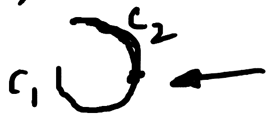
Order of continuity

1) zero order



2) first order

3) Second order



- **<u>Bezier Curve Equation-</u>**
- For (n+1) specified control points
- A bezier curve is parametrically represented by-

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

- Here,

P_i : i^{th} control pt.

- u is any parameter where 0 <= u <= 1
- P(u) = Any point lying on the bezier curve
- P_i $\cancel{B_i}$ = i^{th} control point of the bezier curve
- n = degree of the curve
- $B_{n,i}(u)$ = Blending function = C(n,i) u^i $(1-u)^{n-i}$

Binomial coeff

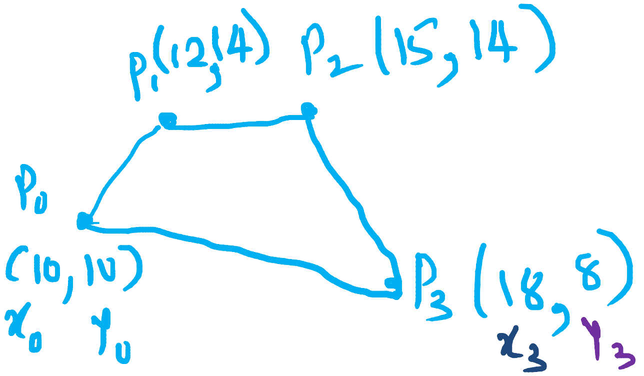
where C(n,i) = n! / i!(n-i)!

$$\boxed{{}^{n}C_i = \frac{n!}{i! * (n-i)!}}$$

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

$$x(u) = \sum_{i=0}^{n} x_i \cdot B_{n,i}(u)$$

$$y(u) = \sum_{i=0}^{n} y_i \, B_{n,i}(u)$$



4-control pts

$\Rightarrow n+1=4$

$\Rightarrow n=3$

$$B_{n,i}(u) = {}^nC_i\, u^i (1-u)^{n-i}$$

$$B_{3,0}(u) = {}^3C_0\, u^0 (1-u)^{3-0}$$

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

$$B_{3,1}(u) = {}^3C_1\, u^1 (1-u)^2$$

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

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

$${}^3C_0 = 1$$

$${}^3C_1 = 3$$

$${}^3C_2 = 3$$

$${}^3C_3 = 1$$

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

$$x(u) = \sum_{i=0}^{n} x_i B_{3,i}(u)$$

$$= x_0 B_{3,0}(u) + x_1 B_{3,1}(u) + x_2 B_{3,2}(u) + x_3 B_{3,3}(u)$$

$$\therefore x(u) = 10(1-u)^3 + 36u(1-u)^2 + 45u^2(1-u) + 18u^3$$

Similarly

$$y(u) = 10(1-u)^3 + 42u(1-u)^2 + 42u^2(1-u) + 8u^3$$

change u from 0 to 1

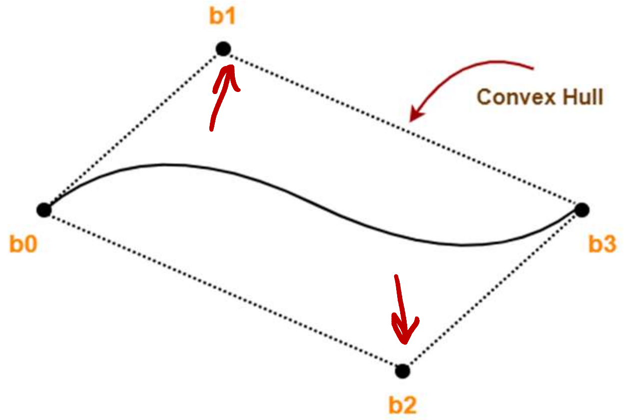
(smaller the change better the approximation)

finally connect all $x(u), y(u)$ using st. line segments in sequencial order of generated $x(u), y(u)$

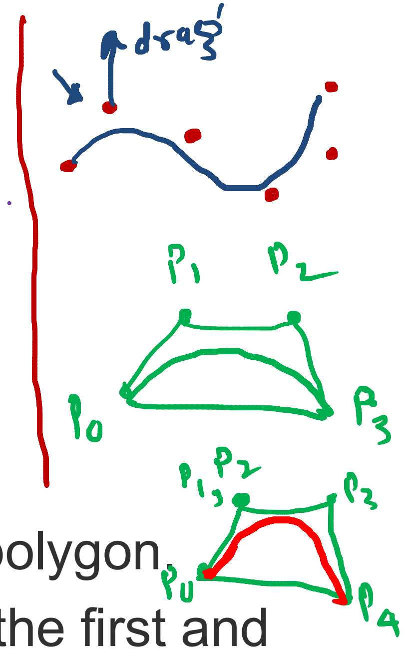
Properties :

- Bezier curve is always contained within a polygon called as convex hull of its control points.

Bezier curve supports global control but not local control



Bezier Curve With Convex Hull



- Bezier curve generally follows the shape of its defining polygon.
- The first and last points of the curve are coincident with the first and last points of the defining polygon.
- The degree of the polynomial defining the curve segment is one less than the total number of control points.

Degree = Number of Control Points – 1

- Bezier curve exhibits the variation diminishing property.

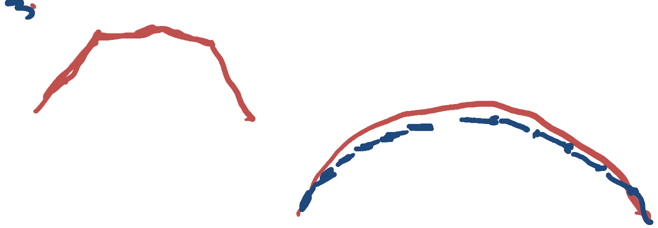
- Specifying multiple control points at same pos^n (loc^n) makes a curve to pass more closer to such loc^n.
- If first and last control pt. coincides → curve will be closed curve

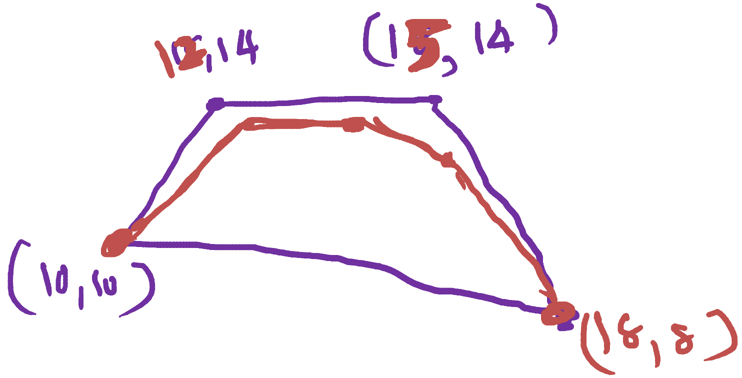
step size = 0.25

u=0	u=0.25	u=0.5	u=0.75	u=1
x(0) = 10 y(0) = 10	x(0.25)=13.13 y(0.25)=13.63	x(0.5)=19.74 y(0.5)=~~18.5~~ 12.75	x(0.75) =15.34 y(0.75) = 15.625	x(1)=18 y(1)=8

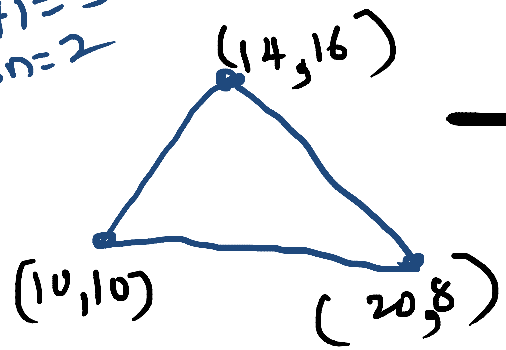
n=3

```
for(u=0; u<=1; u=u+0.1)
{ for(i=0; i<=n; ++i)
  { x(u)=
    y(u)=
  }
}
```





$n+1=3$
$\Rightarrow n=2$



$\rightarrow$ find the Bezier curve by calculating all $x(u)$ & $y(u)$

$$x(u) = x_0 B_{2,0}(u) + x_1 B_{2,1}(u) + x_2 B_{2,2}(u)$$

$$x(u) = 10\, B_{2,0}(u) + 14\, B_{2,1}(u) + 20\, B_{2,2}(u)$$

$$B_{2,0}(u) = {}^2C_0\, u^0 (1-u)^{2-0}$$
$$= (1-u)^2$$

$$B_{2,1}(u) = 2u(1-u)$$

$$B_{2,2}(u) = u^3$$

$$\boxed{x(u) = 10(1-u)^2 + 28u(1-u) + 20u^3}$$

Similarly

$$\boxed{y(u) = 10(1-u)^2 + 32u(1-u) + 8u^3}$$

take step size $= 0.2$

line fn

$\boxed{\begin{array}{l} x(0) = 10 \\ y(0) = 10 \end{array}}$ $\boxed{\begin{array}{l} x(0.2) = 11.04 \\ y(0.2) = 11.59 \end{array}}$ $\boxed{\begin{array}{l} x(0.4) = 11.6 \\ y(0.4) = 11.8 \end{array}}$ $\boxed{\begin{array}{l} x(0.6) = 12.64 \\ y(0.6) = 11.0 \end{array}}$ $\boxed{\begin{array}{l} x(0.8) = 15.12 \\ y(0.8) = 9.6 \end{array}}$

$\boxed{\begin{array}{l} x(1) = 20 \\ y(1) = 8 \end{array}}$

Fractals

- In Euclidean-geometry methods, object shapes are described with eq^n's.
- Natural objects can not be realistically modeled by using Euclidean-geometry methods.
- For natural objects, fractal geometry methods are more suitable, which uses procedures rather than equations to model objects
- A Fractal object has two basic characterstics :
  - Infinite details at every point
  - Certain self similarity
- Fractal dimension is used to describe the amount of variation in the object detail with a fractional number.

Applications : For modelling a wide variety of natural phenomena.

eg: To model terrain, clouds, water, trees, plants, feathers, fur.

eg: Distribution of stars, river islands, moon craters, stock market variation, music etc.

Fractal - Generation Procedures

- A Fractal Object is generated by repeatedly applying a specified transformation function to points within a region of space. If $P_0 = (x_0, y_0, z_0)$ is selected as initial point (entity), each iteration of a transformation function F generates successive levels of detail with the calculations $P_1 = F(P_0)$, $P_2 = F(P_1)$, $P_3 = F(P_2)$ ...

Fractal Dimension

- Detail variation in a fractal object can be described with a number **D**, called the fractional Dimension, which is a measure of the roughness, or fragmentation, of the object.
- More jagged-looking objects have larger fractal Dimensions.
- Relationship between the scaling factor s and the number of subparts n for subdivision of unit straight-line, a square & a cube is shown below :

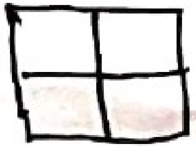
i.) 

1 $\Rightarrow$ $1/n$

Euclidean dimension $D_E = 1$

$\therefore$ if $n=2 \Rightarrow s = 1/2$

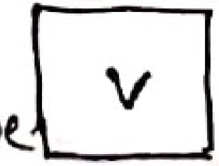
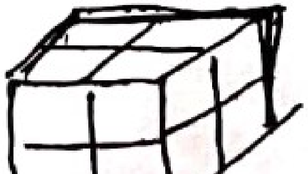
$s = 1/n \Rightarrow \boxed{ns^1 = 1}$

ii.)  $\Longrightarrow$ 

A

Area $D_E = 2$

$\therefore n=4 \Rightarrow s = \frac{1}{n^{1/2}} \Rightarrow \boxed{ns^2 = 1}$

iii.)  $\Longrightarrow$ 

V

volume $D_E = 3$

$\therefore n=8 \Rightarrow s = \frac{1}{n^{1/3}} \Rightarrow \boxed{ns^3 = 1}$

In analogy with the Euclidean object, the fractal dimension $\underline{D}$ for self similar objects can be obtained from

$$ns^D = 1$$

$$\Rightarrow \quad \cancel{D =} \quad s^D = \frac{1}{n}$$

$$\therefore \quad D\ln(s) = \ln\left(\frac{1}{n}\right)$$

$$\therefore \quad D = \frac{-\ln(n)}{\ln(s)} \qquad \therefore \quad D = \frac{-\ln(n)}{-\ln(1/s)}$$

$$\boxed{\therefore \quad D = \frac{\ln(n)}{\ln(1/s)}}$$

$\ln \rightarrow$ natural log.

Geometric Construction of Deterministic Self_Similar Fractals.

- To geometrically construct a deterministic (non random) 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.
- We can construct the snowflake pattern or koch curve by using the initiator & generator as given below

initiator

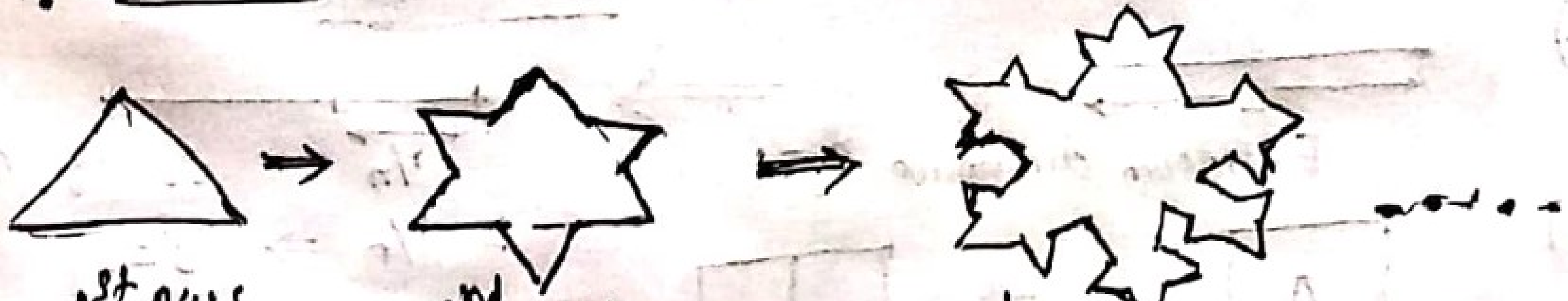
Generator

Each st. line segment in initiator is replaced with four equal line segments at each step.

the scaling factor is $\frac{1}{3}$

$$\therefore \quad \boxed{D = \frac{\ln(4)}{\ln(3)} = 1.2619}$$

Iterations :-



1st pass   2nd pass   3rd pass

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

B Spline Curve

for Bezier $\boxed{P(u) = \sum_{k=0}^{n} P_k B_{n,k}(u)}$

- It offers two advantages over Bezier curve
  - → degree of B-spline polynomial can be set independently of the no. of control points
  - → It allows local control
- ↓ It is more complex than Bezier curve.

- For $n+1$ given control points B-spline curve can be expressed as :

$$\boxed{P(u) = \sum_{k=0}^{n} P_k B_{k,d}(u)}$$

constraint
$u_{min} \leq u \leq u_{max}$
$2 \leq d \leq n+1$

where

P_k : $k+1^{th}$ control point

$B_{k,d}(u)$: B-spline blending f^n represented by polynomial of degree $d-1$

→ d → input parameter

$B_{k,d}(u)$ is expressed using a recursive formula:

k=0
d=3
0 1 2 3

k=1
u=1.3

u=0.9

$$\boxed{\begin{aligned} B_{k,1}(u) &= \begin{cases} 1 & \text{if } u_k \le u \le u_{k+1} \\ 0 & \text{otherwise} \end{cases} \\ B_{k,d}(u) &= \frac{u-u_k}{u_{k+d-1}-u_k} B_{k,d-1}(u) + \frac{u_{k+d}-u}{u_{k+d}-u_{k+1}} B_{k+1,d-1}(u) \end{aligned}}$$

- Each blending fn is defined over d subintervals of the total range of u.

eg: $n=\cancel{4}$ $d=3$

$u_0\ u_1\ u_2\ \ldots\ u_7$

[0, 1, 2, 3, 4, 5, 6, 7]

n+d

- u_k is a knot value taken from a knot vector

total (n+d+1) knot values

$[u_0, u_1, u_2, \ldots, u_{n+d}]$, The choice of knot vector depends on type of B-spline curve

eg: Uniform or Non Uniform B-spline curve

properties

- The polynomial curve has degree $d-1$ & C^{d-2} continuity over the range of u.
- for $n+1$ control points, the curve is described with $n+1$ blending functions.

eg: $d=3$ $[u_0, u_1, u_2, u_3, \ldots]$

- Each blending f^n $B_{k,d}$ is defined over d subintervals of the total range of u, starting at knot value u_k.
- The range of parameter u is divided into $n+d$ subintervals by $n+d+1$ values specified in knot vector.
- Each section of the spline curve between (two successive knot values) is influenced by d control points.
- Any one control point can affect the shape of at most d curve sections.

Assume
$d=3$
& 4 control pts
$n+1=4$
$\Rightarrow n=3$

$$p(u) = \sum_{k=0}^{n} P_k B_{k,d}(u)$$

$$x(u) = \sum_{k=0}^{n} x_k B_{k,d}(u)$$

$$x(u) = x_0 B_{0,3}(u) + x_1 B_{1,3}(u) + x_2 B_{2,3}(u) + x_3 B_{3,3}(u)$$

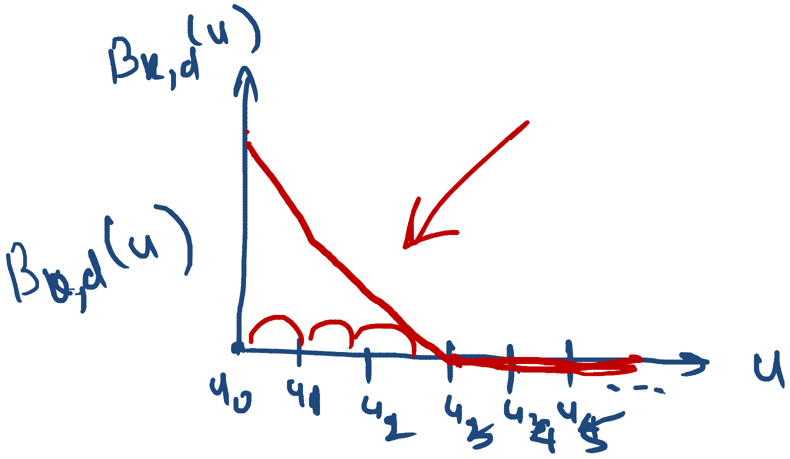
$$y(u) =$$

---

knot values

$d=3$ $n=4$

knot vector $\rightarrow [u_0 \; u_1 \; u_2 \; u_3 \; u_4 \; u_5 \; u_6 \; u_7]$



$B_{k,d}(u)$
$B_{0,d}(u)$
u
u_0
u_1
u_2
u_3
u_4
u_5