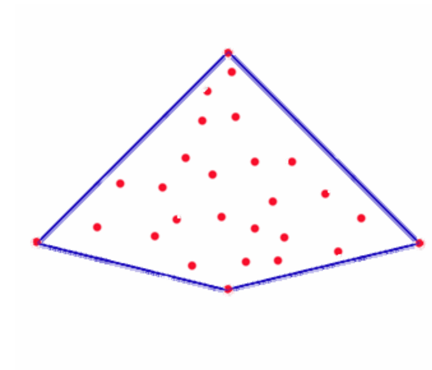
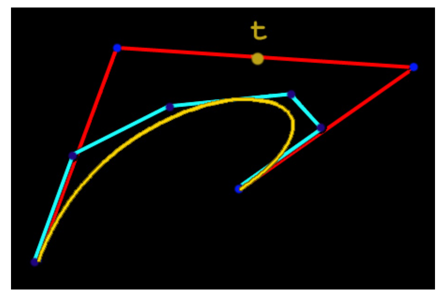
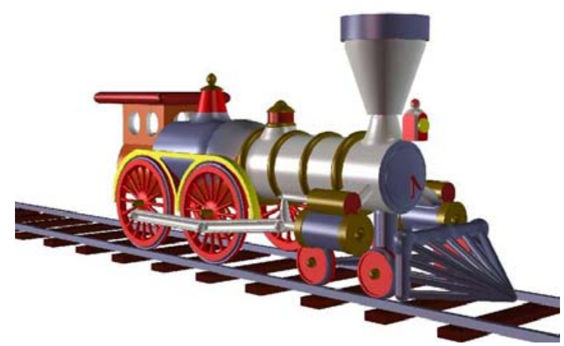


CSC 305
More Modelling

Faramarz Samavati
& Brian Wyvill



The University of Victoria Graphics Group



Bernstein Polynomials – any degree

$$\boxed{J_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}}$$

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

Bezier curve defined as $p(t) = \sum_{i=0}^{n} B_i J_{n,i}(t)$

$$J_{3,i}(t) = \binom{3}{i} t^i (1-t)^{3-i}$$

$$\binom{3}{i} = \frac{6}{i!(3-i)!}$$

$$J_{3,0}(t) = 1t^0(1-t)^3 = (1-t)^3$$

$$J_{3,1}(t) = 3t^1(1-t)^2$$

$$J_{3,2}(t) = 3t^2(1-t)^1$$

$$J_{3,3}(t) = t^3$$

$$\boxed{Q(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t)P_3 + t^3 P_4}$$

Implementation (brute force)

```
Input  P[i], d
//P[i]: control point, d : degree , u : parameter
  for u=0 to 1 step 0.01
        for i=0 to d
              b = Berenstein( i , d , u )
              q = q + b * P[i]
        end
    plot(q);
end
```

- This program isn't efficient
  Redundant computations

  High number of control points

  High degree polynomial

  High degree polynomial

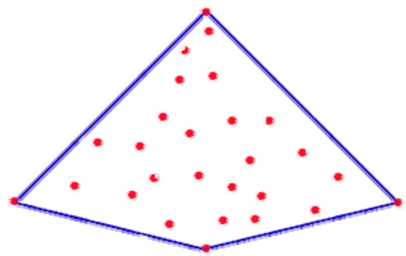
  Unstable computation

Convex Hull

What is the Convex Hull?

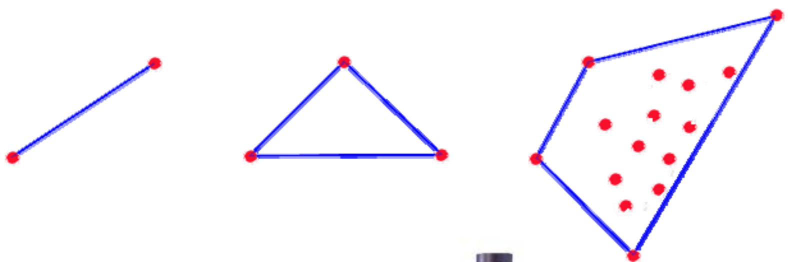
Given a set of points (in R^n)

The smallest convex polyhedral that includes all the points



given $\mathrm{P} = \{\mathrm{p}_0, \mathrm{p}_1, \ldots, \mathrm{p}_d\}$

$$\mathrm{C(P)} = \left\{ \mathrm{p} \mid \mathrm{p} = \sum_{i=0}^{d} \mathrm{c_i p_i} \;;\; \mathrm{c_i} \geq 0, \sum \mathrm{c_i} = 1 \right\}$$

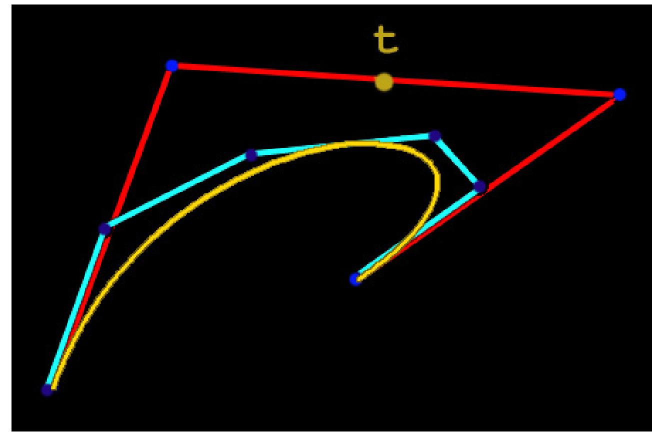


de Casteljau Algorithm

Avoid direct evaluating of polynomials

Geometric interpretation

Consider a planer cubic at $u = \frac{1}{2}$



$$
\begin{array}{ccccccc}
j=3 & \begin{bmatrix}0\\0\end{bmatrix} & & & & & \\
j=2 & \begin{bmatrix}0\\2\end{bmatrix} & \begin{bmatrix}0\\1\end{bmatrix} & & & & \\
j=1 & \begin{bmatrix}8\\2\end{bmatrix} & \begin{bmatrix}4\\2\end{bmatrix} & \begin{bmatrix}2\\1.5\end{bmatrix} & & & \\
j=0 & \begin{bmatrix}4\\0\end{bmatrix} & \begin{bmatrix}6\\1\end{bmatrix} & \begin{bmatrix}5\\1.5\end{bmatrix} & \begin{bmatrix}3.5\\1.5\end{bmatrix} & Q(\frac{1}{2}) &
\end{array}
$$

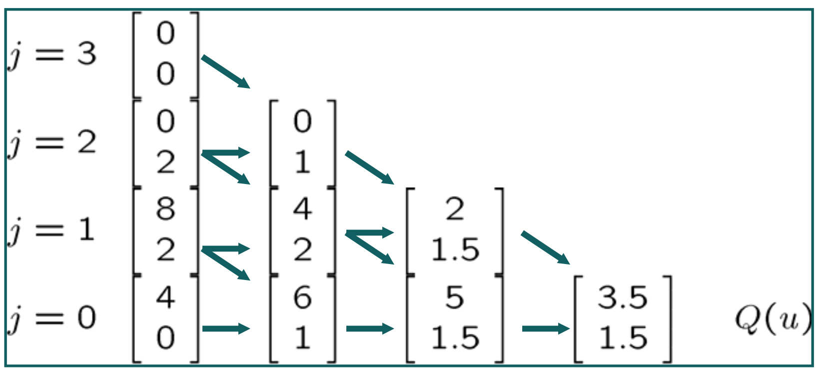
u

1-u

❖column by column updating rule

Pseudo Code

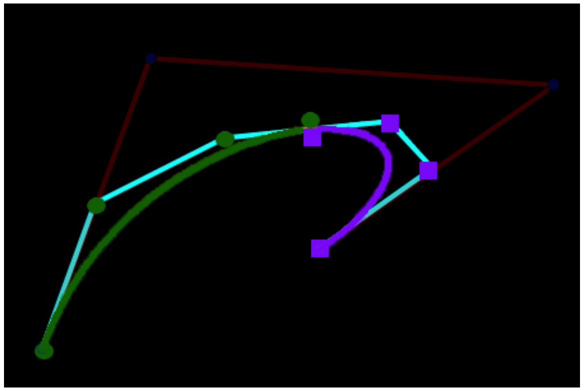
```
Input  P[j] , d , u
//P[j] : control point, d : degree , u : parameter
//output will be Q(u)
   for  i = 1 to d
      for j = 0 to d-i
          P[j] =(1-u)*P[j] + u*P[j+1]
       end
   end
Output   ???
```



$$(1-u)^3 \begin{bmatrix} 4 \\ 0 \end{bmatrix} = B_{0,3}(u) \begin{bmatrix} 4 \\ 0 \end{bmatrix}$$

Some Observations on Cubic Case

- After one step of deCastlejeau algorithm for $u = \frac{1}{2}$
- Obtaining $Q(\frac{1}{2})$
- 4 green points L_0, L_1, L_2, L_3
- 4 blue points R_0, R_1, R_2, R_3
- Where $L_3 = R_0 = Q(\frac{1}{2})$



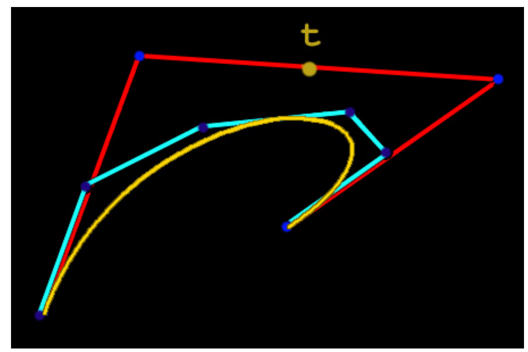
- Small left Bezier curve:
- Small right Bezier curve:

Original Bezier curve

- Divided and conquer method

Matrix Relation

$$\begin{bmatrix} L_0 \\ L_1 \\ L_2 \\ L_3 \\ R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$



- Unit summation of any row.
- Non negative elements.
- Banded structure.

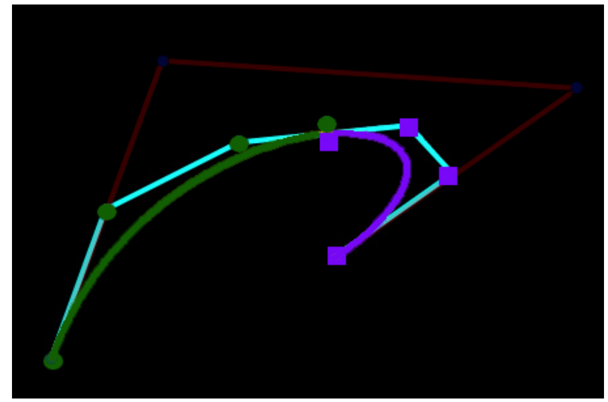
Another View

We start with 4 points (P_0, P_1, P_2, P_3)

We will obtain 8 new points

7 new points

- ❖ New points: $L_0, L_1, L_2, L_3, R_1, R_2, R_3$



- ❖ New points are closer to the curve
- ❖ We can repeat for any 4 points in the new points sets
- ❖And initial polyline is replaced by a finer polyline

- ❖ Subdivision method

Pseudo Code for Subdivision Method

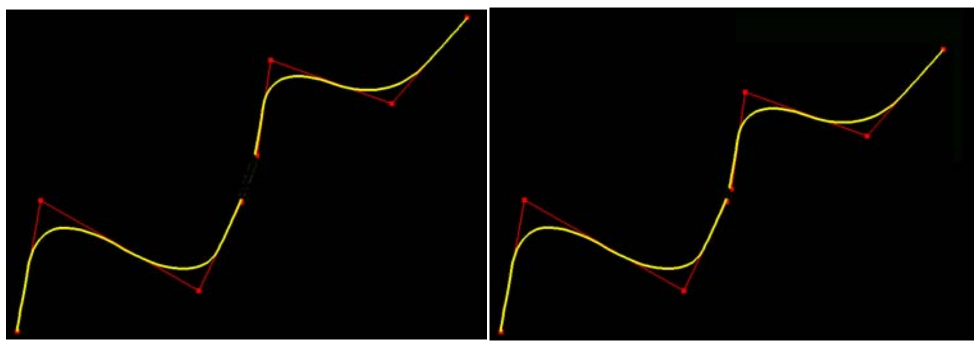
Polyline subBezier (polyline P)

```
//cubic Bezier, d=3,
//uniform subdivision, u=1/2
//Input is polyline P
// n is number of P points (coarse polyline)
// O is output polyline (fine polyline)
    j=0;
   for ( i=0 ; i<n ; i=+3)
       O[j]=P[i];
       O[j+1]=(P[i] + P[i+1])/2
       O[j+5]=(P[i+2] + P[i+3])/2
        t = (P[i+1] + P[i+2])/2
       O[j+2]=(t + O[j+1])/2
       O[j+4]=(t + O[j+5])/2
       O[j+3]=(O[j+2] + O[j+4])/2
        j=j+6;
   endfor
Return O
```

Weakness of Bezier curves

- **High degree polynomial**

- **How can we obtain better control of the curve without adding control points?**

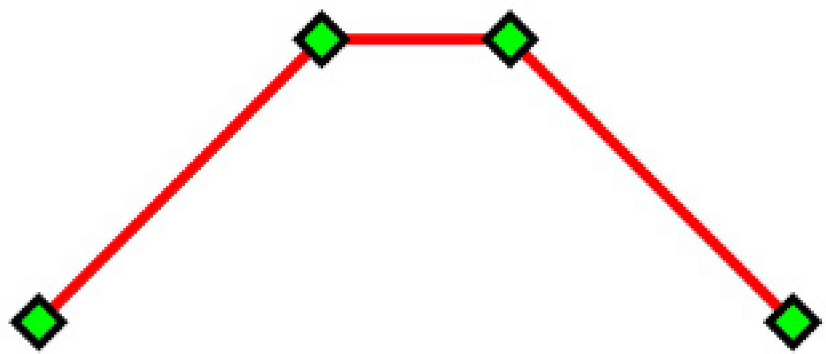


- ❖ Composite Bezier curve: join Bezier curve segments.
- ❖ Apply some constraints at the connection points.
- ❖ What are these constraints?
  - o Positional Continuity = zero degree continuity
  - o Same direction tangents = first degree continuity

Any other weakness?

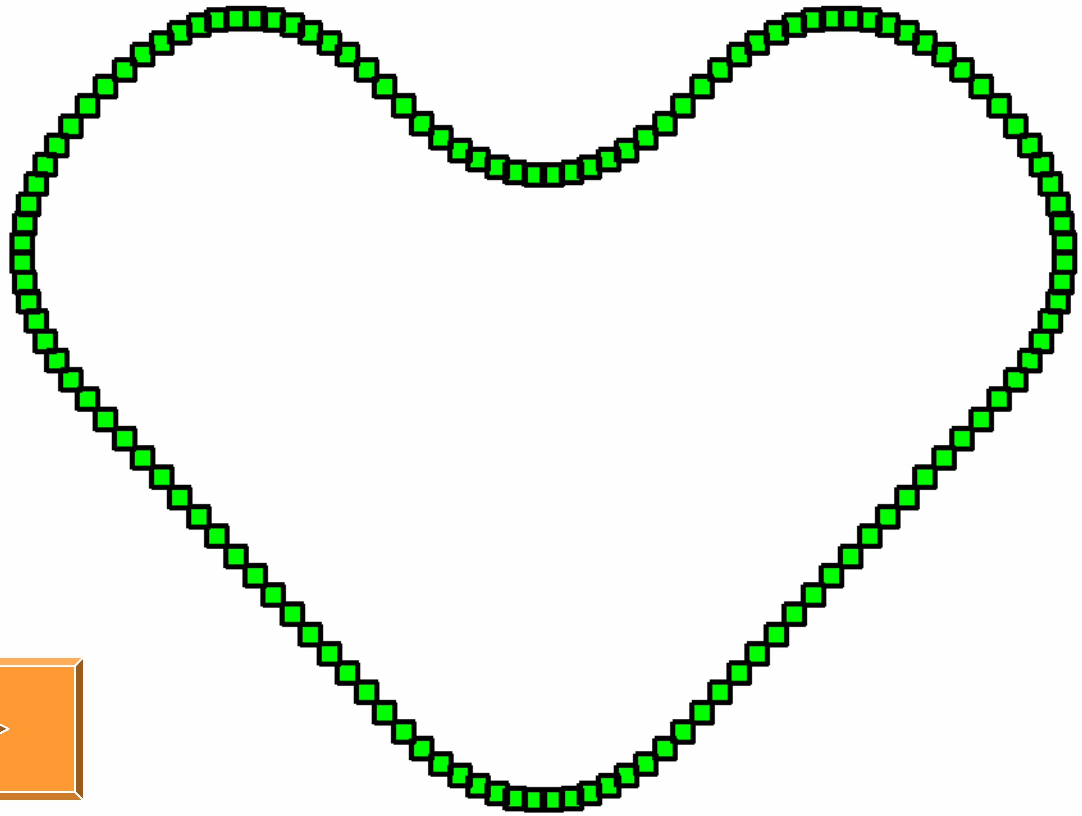
Piecewise quadratic polynomials: Chaikin Algorithm

- **Easy and local algorithm**
- **Two magic! numbers** $\frac{3}{4}$ **&** $\frac{1}{4}$
- **Corner cutting**



Convergence

- smooth limit curve (no corner), quadratic B-spline!



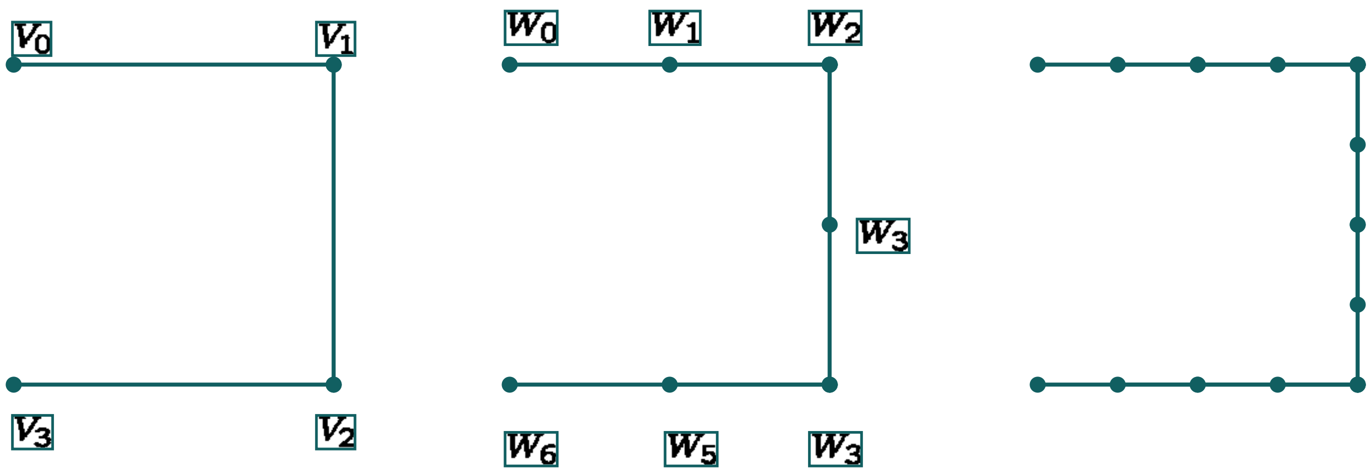
Subdivision Matrix

- **Coarse points** $\xrightarrow{\text{subdivision}}$ **fine points**
- ***F*= *PC***
- **Iterative process**
- ***P* has a regular banded structure**

$$P = \begin{bmatrix} & \frac{1}{4} & 0 & 0 & \\ & \frac{3}{4} & 0 & 0 & \\ & \frac{3}{4} & \frac{1}{4} & 0 & \\ \cdots & \frac{1}{4} & \frac{3}{4} & 0 & \cdots \\ & 0 & \frac{3}{4} & \frac{1}{4} & \\ & 0 & \frac{1}{4} & \frac{3}{4} & \\ & 0 & 0 & \frac{3}{4} & \\ & 0 & 0 & \frac{1}{4} & \end{bmatrix}$$

Faber Subdivision

❖ $$\begin{cases} W_{2i} = V_i \\ W_{2i+1} = \frac{1}{2}(V_i + V_{i+1}) \end{cases}$$



Subdivision Matrix

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- What is the limit curve?
- Obvious interpretation in the resolution enhancement of images
- Periodic case

Image Example

Lena (Lenna) from the November 1972 issue of Playboy Magazine.

each row

each column

Image application

- Increasing the resolution of image
- Traditional method(Faber)
- ½ left + ½ right
- Chaikin rule

Repeating algorithm for every row

Comparing the results

Lena (Lenna) a famous early digitized image.
Photo stolen from the November 1972 issue of Playboy Magazine.