Intro to Computer graphics

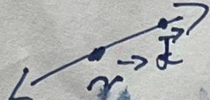
MIT 6.837

- How to make a pic?

- Model
  - geometry
  - materials
  - lights
- Animate
  - make it move
- Render
  - lighting, shadows, textures ..

- Parameterised line / Circle [Parametric Geometry]

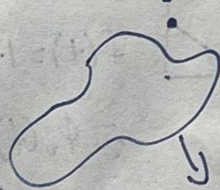
consider it as 1D on a 2D plane ∴ time defined progression



$$\vec{r}(t) = \vec{r}_0 + t\vec{d} \rightarrow \text{unit vec}$$

$$\vec{r}(t) = (\cos t, \sin t)$$

- Implicit Models

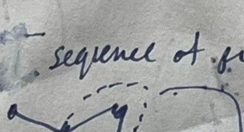


level set [isocontour]

(x, y) such that $f(x,y) = 0 / 1 / 2$ ↓ any constant

for closed loops

- Polylines [sequence of points connected by line segments]



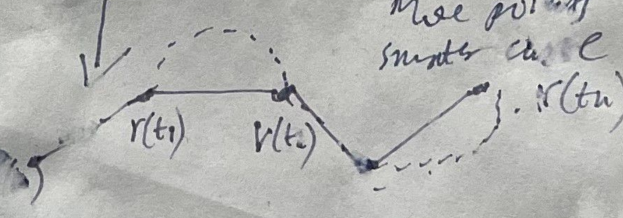
hard to smooth since too many edits

Pro	Con
Easy to store	Hard to smooth
" " render	" " edit

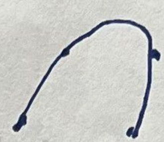
- Tessellation [approximation of geometry]

e.g drawing curves

since we can't draw curves on computer



·) Splines [type of smooth 2D/3D curve]   since polylines are computationally expensive



- User specifies control points
- interpolate control points by a smooth curve
- $x(t), y(t), z(t)$ piecewise polynomials → parametric

·) space of curves is too big

$$\{ r(t) : [0,1] \to R^3 \}$$

·) Basis functions

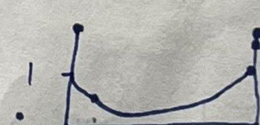
→ set of functions $\phi_i(t)$ spanning set of curves

- expressible as $\Sigma_i a_i \phi_i(t)$

$\therefore r(t) = \Sigma a_i \phi_i(t)$ ↳ More ϕ_i more expressive

·)

$\underbrace{\phi_0(t) = 1 \quad \phi_1(t) = t \quad \phi_2(t) = t^2}_{\text{Monomial Basis}}$



$f(t) = 1 - t + t^2$

$= \phi_0(t) - \phi_1(t) + \phi_2(t)$

$\Updownarrow$

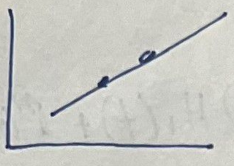
$\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$ Storing well in vectors

$$\begin{array}{c} x \\ y \\ z \\ w \end{array} \begin{pmatrix} 1 \\ 1+t \\ 1+t+t^2 \\ 1+t-t^2+t^3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

↳ change of basis matrix

↳ "canonical monomial basis"

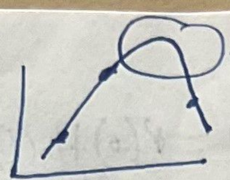
.)



$\phi_0(t) = 1$

$\phi_1 = t$

$\phi(t) = at + b = b\phi_0(t) + a\phi_1(t)$



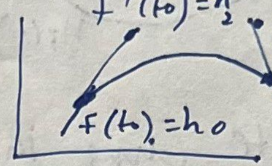
→ Like we thought how to define curveness

$\phi_0(t) = 1$

$\phi_1(t) = t$

$\phi_2(t) = t^2$

∴

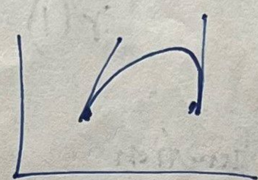


define tangents by $\frac{dy}{dx}$

∴ 4 DOF = 2 points, 2 slopes

$= a + bt + ct^2 + dt^3$ [Monomial basis] 4 DOF, 3 degree

.) Cubic Hermite interpolation [Monomial Basis]



we consider using $[0,1]$ as before to define two points like a car travelling on a path making a line appear ① ①

$P(t) = at^3 + bt^2 + ct + d$

$P(t) = h_0(2t^3 - 3t^2 + 1) + h_1(-2t^3 + 3t^2) + h_2(t^3 - 2t^2 + t) + h_3(t^3 - t^2)$ → Cubic Hermite basis

Did this cuz it gives curves with 2 points, 2 tangents rather than solve this is to change basis

$P(t) = at^3 + bt^2 + ct + d$

$P'(t) = 3at^2 + 2bt + c$

∴ $P(0) = d = h_0$

$P(1) = a + b + c + d = h_1$

$P'(0) = c = h_2$

$P'(1) = 3a + 2b + c = h_3$

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

↓

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}^{-1} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

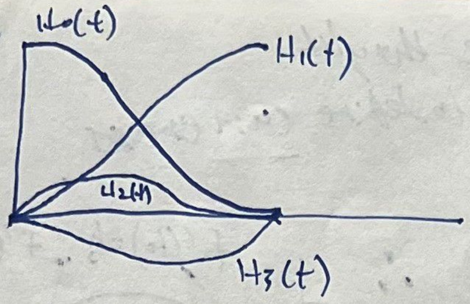
∴ $a = 2h_0 - 2h_1 + h_2 + h_3$

⋮

d

a, b, c, d solve will give unexpected changes to curve ∴ we use cubic hermite cuz are can control at H_0, H_1

1)



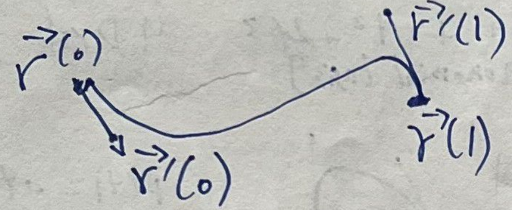
$$P(t) = P(0)H_0(t) + P(1)H_1(t) + P'(0)H_2(t) + P'(1)H_3(t)$$

1) Hermite Curve in plane

$$\vec{r}(t) = (r_0(t), r_1(t))$$

r_0: x-axis, r_1: y-axis — both are cubic function of t

$$\vec{r}'(t) = (r_0'(t), r_1'(t))$$



1) Cubic blossom of function

$F(t_1,t_2,t_3) = F(t_1,t_3,t_2) = F(t_2,t_1,t_3) = \dots$ symmetric

$F(au+(1-a)v, t_2, t_3) = aF(u,t_2,t_3) + (1-a)F(v,t_2,t_3)$ Affine

$f(t) = F(t,t,t)$ Diagonal

if $f(t) = t^3$ it satisfies all

$F(t_1,t_2,t_3) = t_1 t_2 t_3$ → Cubic blossom

if $f(t) = t^2$

$F(t_1,t_2,t_3) = \frac{1}{3}(t_1t_2 + t_1t_3 + t_2t_3)$ → cubic blossom

if $f(t) = t$

⋮

if $f(t) = 1$

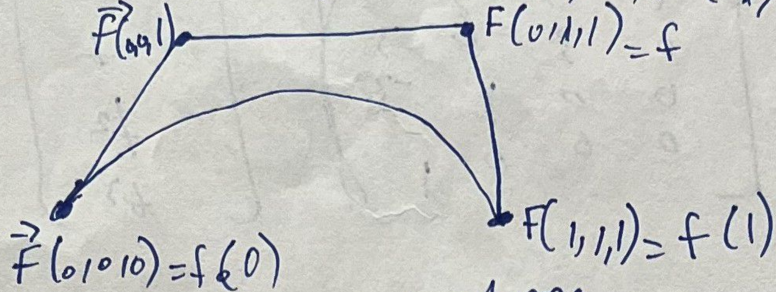
1) Cubic blossom of a Curve

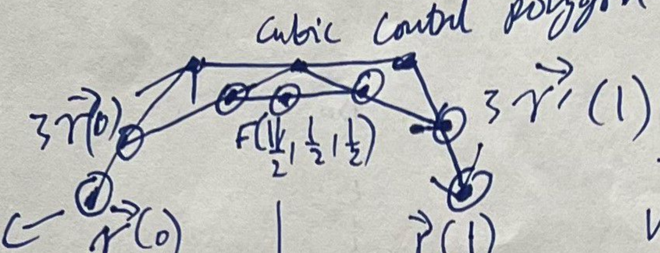
$\vec{F}(t_1,t_2,t_3)$ blossom each coordinate function seperately

$\vec{F}: R^3 \to R^2$

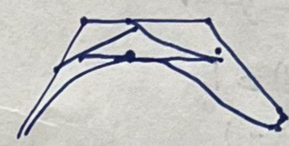
1) To obtain the curve
- start with a cubic curve and compute its blossom
- specify 4 points

$$\vec{F}(0,0,0),\ \vec{F}(0,0,1),\ \vec{F}(0,1,1),\ \vec{F}(1,1,1)$$





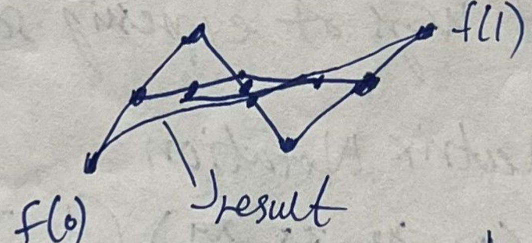
voila!



control points

Affine property

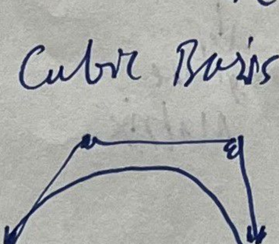
recursive technique to give more points more control more definition



$f(1/2) = ?$

This 4 point Method is known as "De Casteljau's Algo"

Alternate Cubic Basis



$F(0,0,0) = f(0)$ $\quad$ $f(1,1,1) = f(1)$

$$B_0(t) = (1-t)^3 = 1-3t+3t^2-t^3$$
$$B_1(t) = 3t(1-t)^2$$
$$B_2(t) = 3t^2(1-t)$$
$$B_3(t) = t^3$$

$$f(t) = F(0,0,0)B_0(t) + F(0,0,1)B_1(t) + F(0,1,1)B_2(t) + F(1,1,1)B_3(t)$$

1) Change of Basis

Monomial to Bernstein

$$\begin{bmatrix} B_0(t) \\ B_1(t) \\ B_2(t) \\ B_3(t) \end{bmatrix} = \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

→ coeff of B_0, B_1, B_2, B_3 expanded

- Summy

Monomial	Hermite	Bernstein
$\varphi_0(t) = 1$	$H_0(t)$	B_0
$\varphi_1(t) = t$	⋮	⋮
$\varphi_2(t) = t^2$		
$\varphi_3(t) = t^3$	$H_3(t)$	$B_3(t)$

Different ways of expressing same curve

- Cubic Bezier in Matrix Notation

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

$P(t)$ ↓ point on curve (2x1 vector)

"geometry matrix" of control points $P_1 \ldots P_4$ (2×4)

Spline Matrix

Canonical monomial Basis

∴ General Spline formulation

$$\vec{r}(t) = \text{Geometry } G \cdot \text{Spline basis } B \cdot \text{Monomial basis } T(t)$$

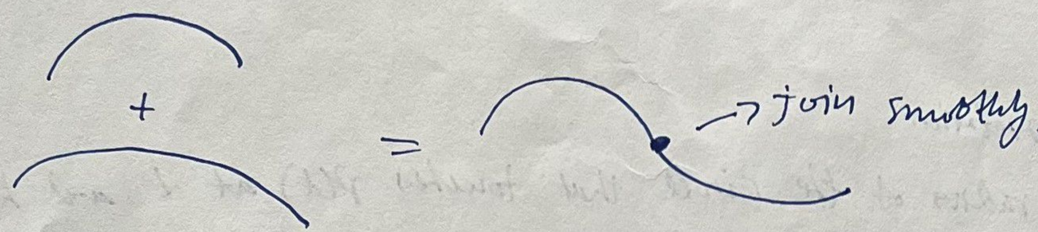
☆ Bernstein Basis are canonical for Bezier

☆ Hermite basis " " " points and tangents

- **spline**

piecewise polynomial with a high degree of smoothness where pieces meet

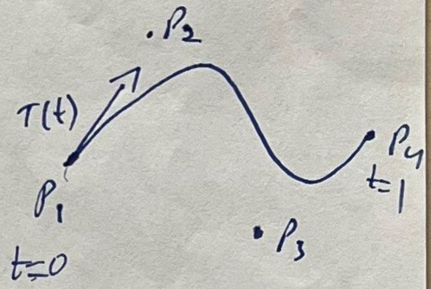
- Long Curves



- Differential properties of Curves:
  - Join together cubic curves
  - Compute normal for surfaces
  - compute velocity for animation.
  - Analyse smoothness

- Velocity

$P(t) = (1-t)^3 P_1 + 3t(1-t^2)P_2 + 3t^2(1-t)P_3 + t^3 P_4$



$P'(t) = \dots\dots$ velocity

but we need something else to define curve

$\therefore$ we need unit vector $\vec{T}$

$\therefore$ tangent vec $T(t) = \dfrac{P'(t)}{\|P'(t)\|}$ in direction of velocity

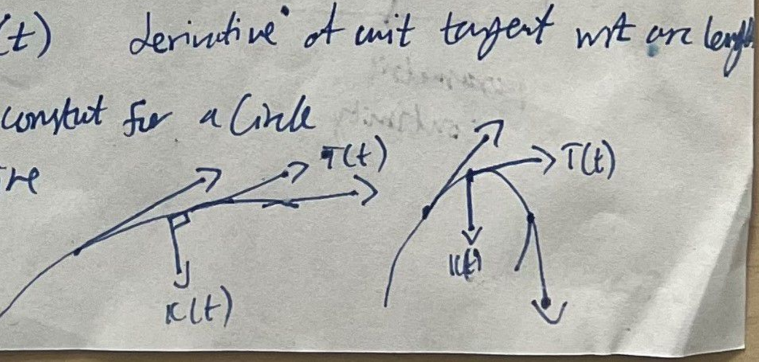
★ Regular Curve: Curve for which $P'(t)$ is always non zero

- Curvature Vector

$K(t) := \dot{T}(t) \propto T'(t)$ derivative of unit tangent wrt arc length

- Magnitude $\|K(t)\|$ is constant for a Circle
- Zero for a straight line

- Always orthogonal to tangent $K \cdot T = 0$



$\| T(t) \|_2 = 1$

$1 = T(t) \cdot T(t)$ dot product / Norm

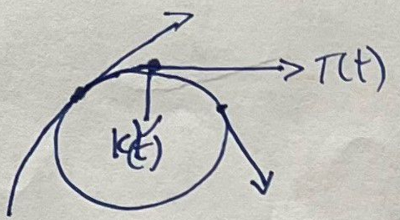
differentiating both sides

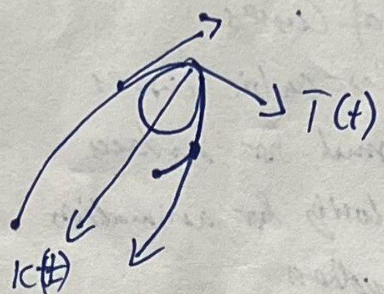
$0 = 2T(t) \cdot T'(t)$

$\propto 2T(t) \cdot K(t)$

- Geometric Interpretation

$\frac{1}{\|K(t)\|}$ is the radius of the circle that touches $p(t)$ at t and has the same curvature as the curve



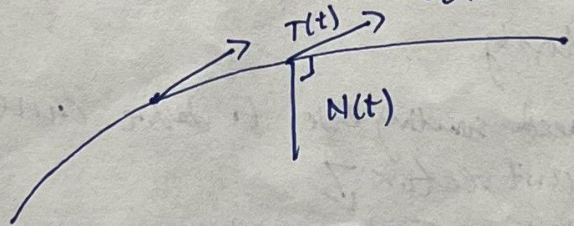


$r = \frac{1}{K}$

$$N(t) := \frac{K(t)}{\|K(t)\|} = \frac{T'(t)}{\|T'(t)\|}$$

↓

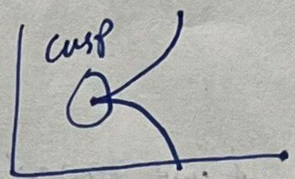
Normalised curvature vector



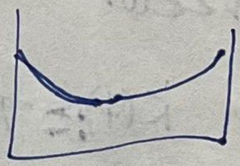
1) Notions of smoothness

$r(t) = (t^2, t^3) \rightarrow x(t), y(t)$

$$r(t) = \begin{cases} (t, t^2) & t < 0 \\ (t^2, t^4) & t \geq 0 \end{cases}$$



parametric continuity



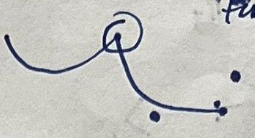
geometric continuity

[segments meeting up at the same tangent vector]

even if velocity not same

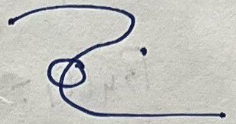
i) Order of Continuity [We check by looking previous and next segment from control points]

C^0 = continuous
- The seam can be a sharp kink



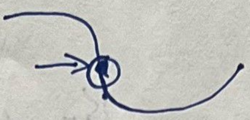
Tangent vec changes

G^1 = geometric continuity
- tangents align at seam



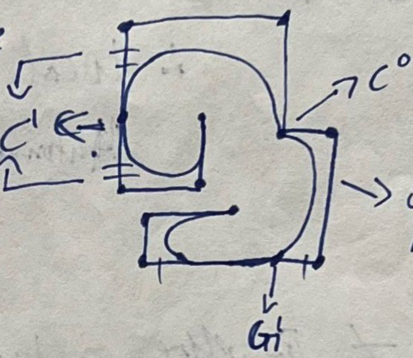
Good enough for Modely

C^1 = parametric continuity
- same velocity at seam



Same length segment before, after

C^2 = curvature continuity
- tangents and their derivatives are same

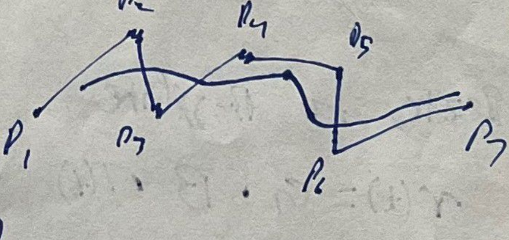


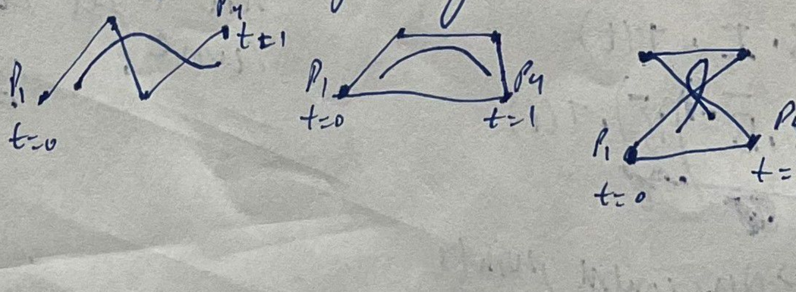
→ Observe control point segments

i) Bezier Curve Major drawback is is that tangent continuity requires constraints linking control points, Move one and then you have to maintain other $C^0 C^1 G^1$ .... etc

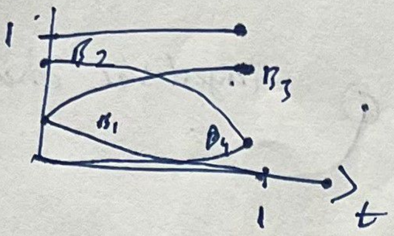
∴ We use cubic B-splines

i) Cubic B-splines
- More then 4 control points but defined by atleast 4 control points
- Locally Cubic [cubics chained together]
- Curve is not constrained to pass through any control points





·) B spline curve is bounded by the convex hull of its control points



$$B_1(t) = \frac{1}{6}(1-t)^3$$

$$B_3(t) = \frac{1}{6}(-3t^3+3t^2+3t+1)$$

$$B_2(t) = \frac{1}{6}(3t^3-6t^2+4)$$

$$B_4(t) = \frac{1}{6}t^3$$

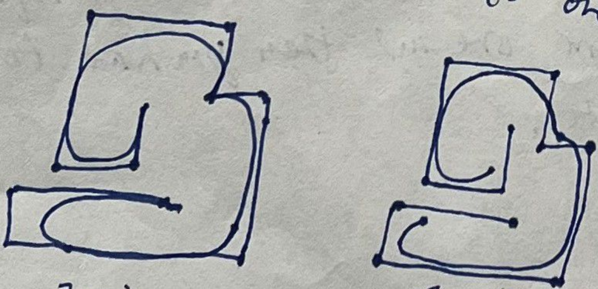
$$Q(t) = \underbrace{G}_{\text{Geo Matrix}}\ \underbrace{B}_{\text{Basis Matrix}}\ \underbrace{T(t)}_{\text{Monomial Basis}}$$

$$B_{B\text{-spline}} = \frac{1}{6}\begin{bmatrix} 1 & -3 & 3 & -1 \\ 4 & 0 & -6 & 3 \\ 1 & 3 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

∴ Local control [windowing]

Automatically C^2, no need to match tangents

* Bezier $\neq$ B-spline   but both are cubics so can be converted to one another



Bezier   B-spline

·) Bezier to B-spline

$$r(t) = G \cdot B \cdot T(t)$$

$$= G \cdot B \cdot I \cdot T(t)$$

$$= \underbrace{(G \cdot B \cdot \bar{B}^{-1})}_{\bar{G}} \bar{B} \cdot T(t)$$

$\bar{G}$ → New control points

consider we have a new basis $\bar{B}$ how to get control points for that curve. $\bar{B} \cdot \bar{B}^{-1} = I$

1) Representing Surfaces
- Triangle Meshes : Surface ~~[illegible]~~ analog of polylines, what GPU's draw
- Tensor product splines : Surface analog of spline Curves
- Subdivision Surfaces
- implicit Surfaces : $f(x,y,z) = 0$
- Procedural : surfaces of revolution, generalized cylinders

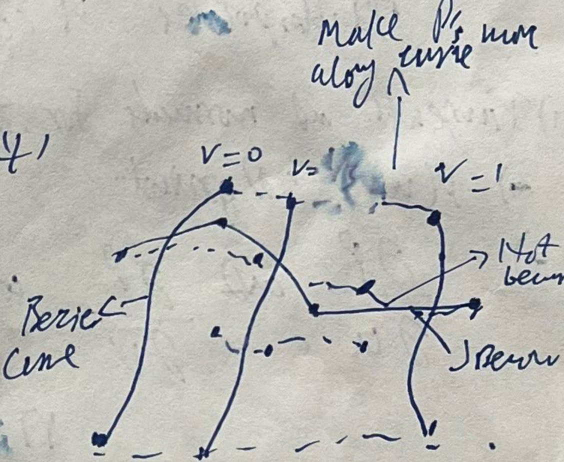
2) Tessellation
- Tiling or approximating geometry with a collection of primitive shapes

3) Curves to Surfaces

$$P(u) = (1-u)^3 P_1 + 3u(1-u)^2 P_2 + 3u^2(1-u) P_3 + u^3 P_4$$

'u' relabled from 't'

→ Bezier basis



$$P(u,v) = (1-u)^3 P_1(v) + 3u(1-u)^2 P_2(v) + 3u^2(1-u) P_3(v) + u^3 P_4(v)$$

when $v=0$ we get left boundry when $v=1$ we get right boundry

But if we use Bezier Curve for both u and v we call it "Tensor product Bezier patch" or "Bicubic Bezier Surface"
keeping either 'u' or 'v' constant
∴ control mesh is basically composed of '16' points

·) Bicubic Tensor Product

$$P(u,v) = B_1(u)P_1(v) + B_2(u)P_2(v) + B_3(u)P_3(v) + B_4(u)P_4(v)$$

$$P_i(v) = B_1(v)P_{i,1} + B_2(v)P_{i,2} + B_3(v)P_{i,3} + B_4(v)P_{i,4}$$

16 control points $P_{i,j}$

16 2D basis functions $B_{i,j}$

$$P(u,v) = \overbrace{\sum_{i=1}^{4} B_i(u) \underbrace{\left[\sum_{j=1}^{4} P_{i,j} B_j(v)\right]}_{v\text{ curves}}}^{u\text{-curves}} = \sum_{i=1}^{4}\sum_{j=1}^{4} P_{i,j} B_{i,j}(u,v)$$

$$B_{i,j}(u,v) = B_i(u)B_j(v)$$

·) Interpolates 4 corners, approximates others

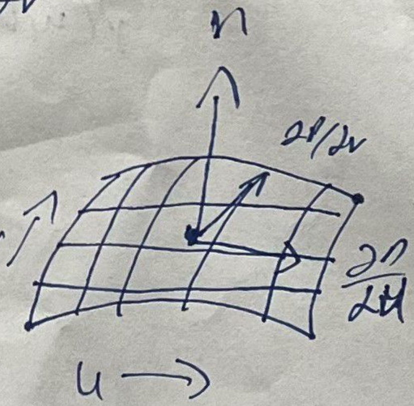
·) Tangents and normals for patches

-) $P(u,v)$ 3D point

-) $\frac{\partial P}{\partial u}$ and $\frac{\partial P}{\partial v}$ are tangent vectors = partial derivatives

$$n = \frac{\partial P}{\partial u} \times \frac{\partial P}{\partial v}$$

normal is perpendicular to both



·) Matrix Notation for patches

$$P^x(u,v) = \begin{bmatrix} B_1(u) & \cdots & B_4(u) \end{bmatrix} \begin{bmatrix} P^x_{1,1} & \cdots & \\ & \ddots & \\ \cdots & & P^x_{4,4} \end{bmatrix} \begin{bmatrix} B_1(v) \\ \vdots \\ B_4(v) \end{bmatrix}$$

x coordinate of surface at (u,v)

4x4 Mat of x coordinates of control points

Column Vec of basis funct (v)

$$\therefore\ P(u,v) = \sum_{i=1}^{4} B_i(u) \left[\sum_{j=1}^{4} P_{i,j}\, B_j(v) \right]$$ 4x4 Mat representatn

•) Mat Notation of patches

- Curves

$$P(t) = G \cdot B \cdot T(t)$$

- Surfaces

$$P^x(u,v) = T(u)^T\, B^T\, G^x\, B\, T(v)$$

↓

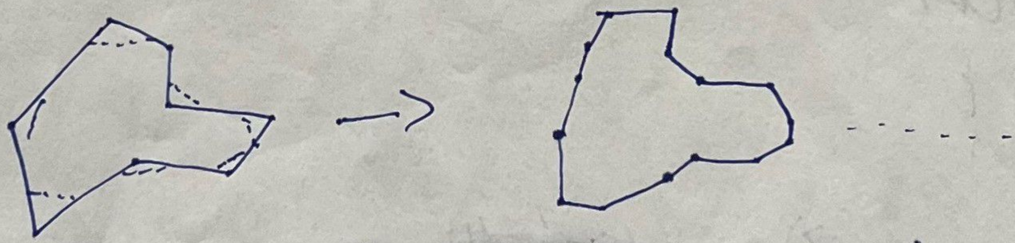
Seperate 4x4 geometry matrix for x,y,z

T= power basis

B= spline Matrix

G= geometry matrix

• Corner Cutting Algo



Produces a quadratic B-spline! •[Chaikins Algo]

- Subdividing

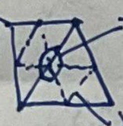
•) 2 techniques

Triangles

↓

Loop Subdivision

Old

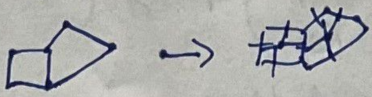


→ Valence 6

→ outgoing vertices

→ New

Quads



based on centroids

↓

Catmull-Clark Subdivision

Semi-regular Mesh.

full regular when most vertices have valence 6

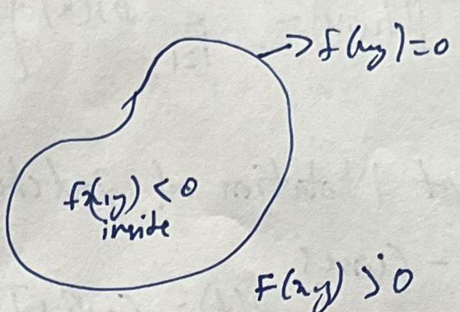
· Implicit Surfaces

- Surface defined implicitly as a function

  $f(x,y,z)=0$ on surface

  $f(x,y,z)<0$ inside

  $\quad$ " $\;>0$ outside



imp in <u>deep learning</u>

- Super useful in slides.
- Handles weird topology really well

·) Coordinates and transformations

-) Hierarchical Modelling

1) Camera
2) Scene
3) Car

·) $\vec{v} \mapsto L(\vec{v})$ for ~~transformations~~ transformations

maps to

$$L(\vec{v}) = L\left(\sum_i c_i \vec{b_i}\right) = \sum_i c_i L(\vec{b_i})$$

$$\vec{v} = \underbrace{(\vec{b_1}\ \vec{b_2}\ \vec{b_3})}_{\text{basis}}\underbrace{\begin{pmatrix}c_1\\c_2\\c_3\end{pmatrix}}_{\text{coordinate}} \mapsto (L(\vec{b_1})\ L(\vec{b_2})\ L(\vec{b_3})) \cdot \begin{pmatrix}c_1\\c_2\\c_3\end{pmatrix}$$

·) L are also vectors in same space

$$\therefore\ L(\vec{b_i}) = (\vec{b_1}\ \vec{b_2}\ \vec{b_3})\begin{pmatrix}M_{1,i}\\M_{2,i}\\M_{3,i}\end{pmatrix}$$

$$(L(\vec{b_1})\ L(\vec{b_2})\ L(\vec{b_3})) = (\vec{b_1}\ \vec{b_2}\ \vec{b_3})\begin{bmatrix}M_{1,1} & & \\ & \ddots & \\ & & M_{3,3}\end{bmatrix}$$

·) All together

$$(\vec{b_1}\ \vec{b_2}\ \vec{b_3})\begin{pmatrix}c_1\\c_2\\c_3\end{pmatrix} \xmapsto{h} (\vec{b_1}\ \vec{b_2}\ \vec{b_3})\begin{pmatrix}M_{11} & & \\ & & M_{3,3}\end{pmatrix}\begin{pmatrix}c_1\\c_2\\c_3\end{pmatrix}$$

$\therefore$

$$B\mathbf{c} \mapsto BM\mathbf{c}$$

2 interpretations from above

$B \mapsto BM$ transforming Basis

$\mathbf{c} \mapsto M\mathbf{c}$ " coordinates

·) given $\vec{v} = B\mathbf{c}$ in basis B we have $A = BM$

coordinates of $\vec{v}$ in A $AM^{-1} = B$

$\vec{v} = B\mathbf{c} = \cancel{AM^{-1}B\mathbf{c}}$

$A(M^{-1}\mathbf{c})$

keeping track of coordinate system using basis on left

·) of all types of translations "translation" is not linear

·) Vectors are members of vector space $\vec{v}$

points " " " Affine space $\tilde{p}$

$$\tilde{p} = \tilde{o} + \sum_i c_i\vec{b_i} = (\vec{b_1}\ \vec{b_2}\ \vec{b_3}\ \tilde{o})\begin{pmatrix}c_1\\c_2\\c_3\\1\end{pmatrix} := F\mathbf{c}$$

origin

4D

but 4th is '1'

$\tilde{p} - \tilde{q} = \vec{v}$ subtracting points gives vectors

$\tilde{q} + \vec{v} = \tilde{p}$ vector + point = point

☆ Affine spaces to keep track of frame

- Affine frame

$\tilde{o}$ + basis $F = (\vec{b_1}\ \vec{b_2}\ \vec{b_3}\ \tilde{o})$

$\tilde{p} = \tilde{o} + \sum_i c_i \vec{b_i}$ ☆ using coordinates $\mathbf{c}$ of vector in B

i) Affine allows linear transformation for "translation" as well

$$\tilde{p} = F\mathbf{c}$$

$$(\vec{b}_1\ \vec{b}_2\ \vec{b}_3\ \bar{o})\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{pmatrix} \mapsto (\vec{b}_1\ \vec{b}_2\ \vec{b}_3\ \bar{o})\begin{pmatrix} M_{11} & & & \\ & & & M_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{pmatrix}$$

$$F\mathbf{c} \mapsto FA\mathbf{c}$$

$$\tilde{p} = \bar{o} + \sum_i c_i L(\vec{b}_i) = (\vec{b}_1\ \vec{b}_2\ \vec{b}_3\ \bar{o})\begin{bmatrix} M_{11} & & & 0 \\ & & M_{33} & \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{bmatrix}$$

$$\tilde{p} = \bar{o} + \vec{t} + \sum_i c_i \vec{b}_i = \quad '' \quad \begin{bmatrix} 1 & & & M_{14} \\ & 1 & & M_{24} \\ & & 1 & M_{34} \\ & & 0 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{bmatrix}$$

☆ $AB \neq BA$

- Affine point

$(x, y; 1)$ $(x, y, z; 1)$

☆ from lecture '5' slides snipped in PC, only imp info on paper

i) $a_1 = M_{12} a_2$

↳ going from coordinate system of 2 to 1

projection$_w$ $v = (v \cdot w)\, w$ → sort of scaling by this amount

↓ v onto w — ↓ dot product

↓ measures how much of v in w direction

w = unit vec

v = vector