Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Language vs Action

Action → cut copy paste

Language → Smith at backend

(to delete file from original location)

(move to new location)

Programming is example by both action and language.

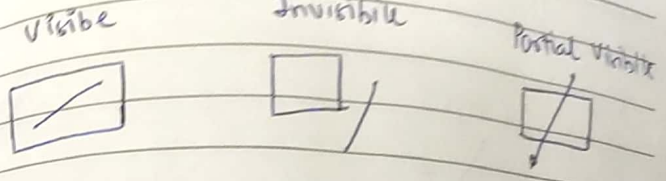
Hypertext : Non-linear browsing structure connected with each other.

memex machine → mechanic desk linked with external micro films and other accessories.

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

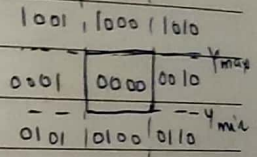
Line clipping: Process of removing lines or portions of lines that lie outside the clipping window



Algo minimize calculation in # of intersection.

COHEN - SUTHERLAND ALGO

Bit assigned to each region



TBRL : 4 bit (0/1)

Top Bottom Right Left

x_{min} x_{max}

For point assign code as well based on region it lies

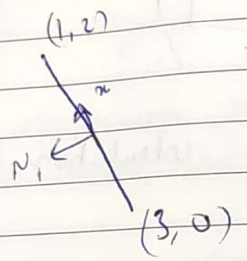
For both ~~code~~ end points.

If end points AND = 0 reject the line else find the intersection point

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Find intersection?
Set bit tells the direction / location of coordinate.



$\Rightarrow \vec{x} = -2\hat{i} + 2\hat{j}$ x y

$\vec{N_1} = -2\hat{j} - 2\hat{j}$ -y x

$N_1 = (-1, -1)$

NLL → 2 dimension Board 3

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Point clipping:

Clipping: A procedure by which we can select or reject a particular portion of an image in a specified region or space

Point clipping:
Inside: $x_{min} \le x \le x_{max}$
$y_{min} \le y \le y_{max}$

Line clipping:
Cohen - Sutherland 4 bit region code (TBRL)

→ If end points of line are (0000) line is completly inside

→ If AND is non zero → line is completly outside.

→ If AND = 0 partially overlapping with window

Find intersection point.

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Find intersection using slope intersection method :

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

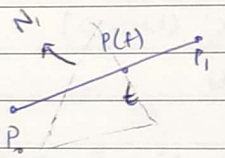
$x_{min/max}$, $y_{min/max}$ koi common hoga find another coordinate ez.

Cyrus - Beck line clipping

Clipping window with any polygon window.

(1) Dot Product

(2) Parametric eq of lines



$$P(t) = P_0 + t(P_1 - P_0)$$

Pick an edge :

→ Draw Normal vector N_i

→ Take a point on edge P_{ei}

→ Dot product : $N_i (P(t) - P_{ei})$

<0 inside clipping

>0 outside clipping

$=0$ intersection point

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

$$N_i (P(t) - P_{ei}) = 0$$

$$N_i (P_0 + t(P_1 - P_0) - P_{ei}) = 0$$

$$N_i P_0 + N_i t P_1 - N_i t P_0 - N_i P_{ei} = 0$$

$$t = \frac{N_i (P_{ei} - P_0)}{N_i (P_1 - P_0)} \quad \} D$$

Entring ↑ $D > 0$ leaving point

leaving ↓ $D < 0$ entering point

Substitue t value in line equation to Parametric equation to find intersection point.

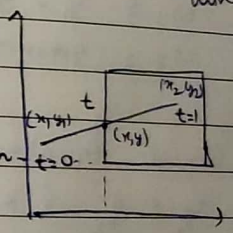
See Video (61) Numerical

Liang - Barsky clipping

Rectangular window

$$x = (1-t)x_1 + t x_2$$

$$y = (1-t)y_1 + t y_2$$



$$x = x_1 + t(x_2 - x_1)$$

$$\boxed{x = x_1 + t\Delta x}$$

$$\boxed{y = y_1 + t\Delta y}$$

As per pointing clipping

$$x_{min} \le x \le x_{max}$$

$$y_{min} \le y \le y_{max}$$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

$x_{min} \leq x_1 + t\Delta x \leq x_{max}$

$y_{min} \leq y_1 + t\Delta x \leq y_{max}$

4 equation here

$t\Delta x \geq x_{min} - x_1$ —①

$t\Delta x \leq x_{max} - x_1$

$t\Delta y \geq y_{min} - y_1$ —②

$t\Delta y \leq y_{max} - y$

Mult eq 1 and 2 by (-1)

$-t\Delta x \leq x_1 - x_{min}$

$t\Delta x \geq x_{max} - x_1$

$-t\Delta y \leq y_1 - y_{min}$

$t\Delta y \leq y_{max} - y_1$

General equation form

$t p_k \leq q_k \quad \{k = 1, 2, 3, 4\}$

$P_1 = -\Delta x$ $q_1 = x_1 - x_{min}$

$P_2 = \Delta x$ $q_2 =$ —

$P_3 = -\Delta y$ $q_3 =$ —

$P_4 = \Delta y$ $q_4 =$ —

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

If $P_k = 0$ Line is parallel with clipping window

$q_k < 0$ Line outside

$P_k != 0$

$P_k < 0$ Find t_1

$t_1 = max(0, q_k/P_k)$

$P_k > 0$ Find t_2

$t_2 = min(1, q_k/P_k)$

$t_1 > t_2$ Line outside reject it.

$t_1 < t_2$

$x = x_1 + t\Delta x$

$y = y_1 + t\Delta y.$

See video 63 . numerical.

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

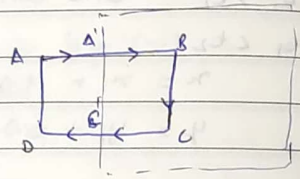
Polygon clipping:

→ Suther land - Hogeman
- Weiler - Atherton

(1) Suther land - Hodgeman Algo:

List → Vertices
↙ ↘
inside intersd.

4 Rules:



Outside → inside
Save A', B

(2) both points inside
Save last vertex

(3) inside → outside
Save intersd point.

(4) Both points outside : neglect

Save List → A' B C E'

Concave Polygon give Extra enous lines.

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

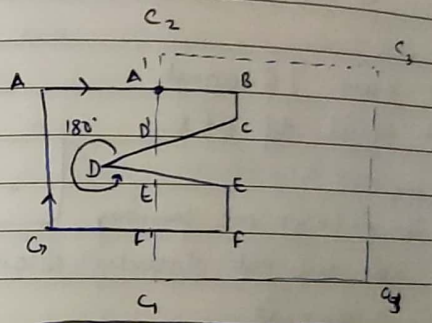
Notes

Weiler - Atherton

Previous gives best result in convex polygon
concave Polygon give Extraneous Lines

Polygon → ~~Subject~~ Polygon
clipping window: clip Polygon

make 2 List → Subject Polygon List,
Inserting Intersection points.



Subject Polygon List	clip polygon
A	C1
A' → start	F'
B	E' (end)
C	D'
D'	A' (End)
D	C2
E' → start	C3
E	C4
F	
F'	
G	
A	

List 1
A' B C D' A'

List 2
E' E F F' E'

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Polygon Filling

* Seed Fill
  - → Flood Fill
  - → Boundary fill
* Scan line

Video 5.6

Boundary Fill

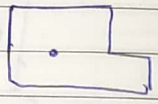
↳ one color boundary only

Flood Fill

↳ different color boundary

4 conn / 8 connet

→ typical ~~dfs~~ bfs.



Boundary fill:

if pix (x, y) not boundary color and not already colored

→ color it

↳ bfs on connected cells

Flood fill:

```
Flood_Fill ( x, y, old_col, new_col )
    if pix(x,y) == old_col
        put (x,y, new_color)
        dfs
```

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

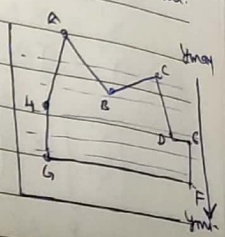
Notes

Scan line Algo

large polygon me large stack size change putting pixels etc ~~etc~~ flood fill not optimal.

steps:

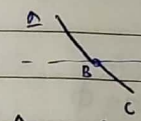
(1) locate Intersection points of scan line with the polygon edges.

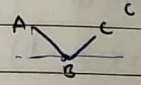


(2) Pairing intersection points.

(3) move down side as per scan line & sort all pairs.

Vertices ? → take B one





→ take B twice.

(4) All pairs are sorted from y_{max} to y_{min}

(5) Sides get sorted on Intersection point basis

(6) Area filling starts Now.

Video 5.8

Coherence Property:

$\Delta y = -1$ (one step down)

$m = \Delta y / \Delta x = -1/\Delta x = m$

$x_{k+1} = x_k - 1/m$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

2D Geometric Transformation:

→ **Translation**: change pos in a straight line path.

$x' = x + t_x$

$Y' = Y + t_y$

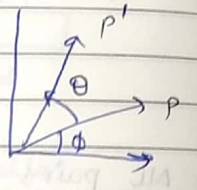
$$\begin{bmatrix} x' \\ Y' \end{bmatrix} = \begin{bmatrix} x \\ Y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$P' = P + T$

Rotation: Repositioning on circular path.

(x, y) : $r\cos\phi$, $r\sin\phi$

$x'y'$: $r\cos(\theta+\phi)$, $r\sin(\theta+\phi)$



$x' = r\cos\theta\cos\phi - r\sin\theta\sin\phi$

$y' = r\sin\theta\cos\phi + r\sin\phi\cos\theta$

$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

matrix: $[x' \; y'] = [x \; y] \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$

$\theta \rightarrow$ +ve for clockwise

Scaling: changes size of object. Multiply each coordinates with scaling factor.

$x' = x.S_x \qquad y' = y.S_y$

$$\begin{bmatrix} x' \\ Y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

$P' = P.S$

S_x & $S_y > 1$	object Enlarge
S_x & $S_y < 1$	object Decrease
S_x & $S_y = 1$	No change
S_x & S_y	Differencial scaling

Homogenous coordinates:

→ $[x'y'\;1] = [xy\;1] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix} = [x + t_x \;\; y + t_y \;\; 1]$

→ $[x'\;y'\;1] = [x\;y\;1] \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = [x\cos\theta - y\sin\theta \;\; x\sin\theta + y\cos\theta \;\; 1]$

$= [x'\;y'\;1] = [xy1] \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} = [xS_x \;\; yS_y \;\; 1]$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Reflection:

y-axis: $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

x-axis: $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

origin | 180°: $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$y = x$: $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$y = -x$: $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Shearing: Slant / Tilt

x-shear: Only x coord. changed

y-shr: Only y ——

x-shear

$$x.sh = \begin{bmatrix} 1 & 0 & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$x' = x + sh_x \cdot y$$

$$y' = y$$

Yshear: $y.sh_y = \begin{bmatrix} 1 & sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$x' = x$$

$$y' = y + sh_y . x$$

Windows to viewport is the process of trasforming 2D world coordinate object to devices coordinates. Objects inside the world or clipping are mapped to viewport to.

World coordinate → ~~Vie~~ Device Coordinate

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Normalized point of viewport are

$$\left(\frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}} , \frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}} \right)$$

Scaling factor

$$S_x = \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}}$$

$$S_y = \frac{y_{vmax} - y_{vmin}}{y_{wmax} - y_{wmin}}$$

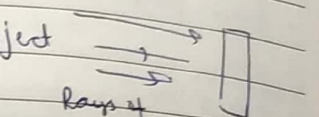
Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

plane.

Projection - process by which we can create an image of an object on a plane. (view plane)

Observer :) Object

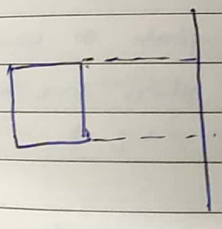


(Projector)

GFG

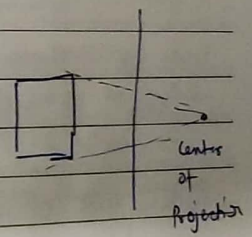
Types of Projection

Parallel
- true size and shape
- Projection rays are parallel.



not realistic view.

Perspective
Object appear bigger for while smaller



Distorted.

realistic view

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Types of design rule:
→ principles
→ standard
→ Guidlines

Principles to support usability.

→ Learnability
- Familiarity
- Synthesis
- Predictibility
- Consistency
- Generalizability

CG FPS

→ Flexibility
- Multithreadig
- Substitution
- Customisation
- Task migration

CSMM

→ Robustness
- Responsiveness
- Observe
- Recoverability
- Task conformance

RROT

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Curves

- What is curve: Explicit | Implicit | Par. repres.
- Parametric curves
- Continuity: Parametric | Geometric
- Spline curves: • Hermite curve
  • Bezier curve
  • Bspline curve

Curve and Representation:
Any figure which is continous with a set of points

Representation:

Explicit: $y = f(x)$
↑ dependent
↳ independent
eg $y = mx + c$

Implicit: $F(x, y) = 0$
$x^2 + y^2 - r^2 = 0$

Parametric: $x = f_x(t)$
↳ parameter

4 pts se curve = x^3
Cubic polynomial / parametric curve

1 2 3 (4 5) 6 7 8
← Line
→ complexity

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

$x = x(t) \quad y = y(t) \quad z = z(t)$

$Q(t) = [x(t) \quad y(t) \quad z(t)]$

$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$

$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$

$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$

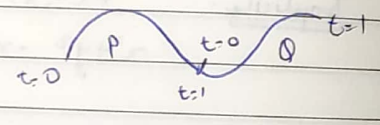
$$Q(T) = [t^3 \; t^2 \; t \; 1] \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

$Q(T) = T \, C$

Parametric | Geometric Continuity

C | G

Parametric continuity:



(1) Zero order (C^0) ~~P(0)~~

$P(1) = Q(0)$

(2) First order (C^1) $P'(1) = Q'(0)$

(3) 2nd order (C^2) $P'(1) = Q'(0)$

and $P''(1) = Q''(0)$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Geometric Continuity

(1) Zero Order : $P(1) = Q(0)$

(2) First Order : $P'(1) = k\,Q'(0)$ (slope same)

(3) Second Order : $P''(1) = k\,Q''(0)$

Zero : Positional Continuity

First : Tangential continuity

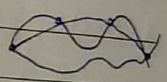
Seond : Curvature continuity

Spline Curve

Already C^1 and C^2 continuity.

control points / Coordinate

~~Q(u)~~ =



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

B_{in} : basis function

Cubic polynomial form

P_i : coordinates

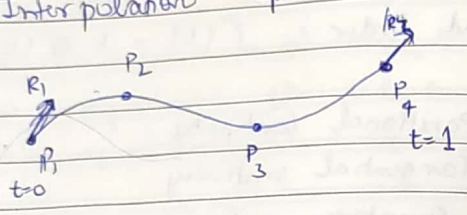
Interpolation curve : same coordinate se pass kare

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

• Hermite Spline Curve:

(i) Interpolation Spline Curve



(ii) Cubic Polynomial curve

End points

(iii) $R_1 = P_1'$ Tangent

$R_4 = P_4'$

$\Rightarrow Q(t) = T C$ $\quad M =$ Basis Matrix

$= T M G_H$ $\quad G =$ Geometric Vector Matrix

For Hermite curve

$Q(t) = T M_H G_H$

$= [t^3 \; t^2 \; t \; 1] \; M_H \; G_H$

For P_1

$Q(0) = [0 \; 0 \; 0 \; 1] \; M_H \; G_H$

For P_4 : $Q(1) = [1 \; 1 \; 1 \; 1] \; M_H \; G_H$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

$Q'(t) = [3t^2 \; 2t \; 1 \; 0] \; M_H \; G_H$

For R_1

$Q_1'(0) = [0 \; 0 \; 1 \; 0] \; M_H \; G_H = R_1(0)$

For R_4

$Q_4'(1) = [3 \; 2 \; 1 \; 0] \; M_H \; G_H = R_4(0)$

$$\Rightarrow \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} M_H G_{H_x}$$

$$M_H G_{H_x} = \text{Inve}\begin{pmatrix} \\ \\ \\ \\ \end{pmatrix} \begin{bmatrix} P_1 \\ R_4 \\ R_1 \\ R_4 \end{bmatrix}$$

$$M_H G_{H_x} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}$$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

$$Q(t) = T \cdot M_H \cdot G_{Hx}$$

$$Q(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}$$

H_1 H_2

$$Q(t) = (2t^3 - 3t^2 + 1)P_1 + (-2t^3 + 3t^2)P_4 + (t^3 - 2t^2 + t)R_1 + (t^3 - t^2)R_4$$

H_3 H_4

$H_1, H_2, H_3, H_4 \leftarrow$ Hermite Blending function

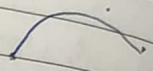
$$Q(t) = H_1 P_1 + H_2 P_4 + H_3 R_1 + H_4 R_4$$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

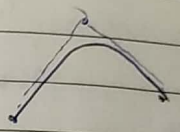
BEZEIR Spline Curve

- → Approximate spline curve.
- → Bernstein Polynomial fn
- → All control points
- → Touches first and last Point



2 points → Line

3 points →



Convex hull

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

↓ Position vector

↳ Bernstein func.

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

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

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

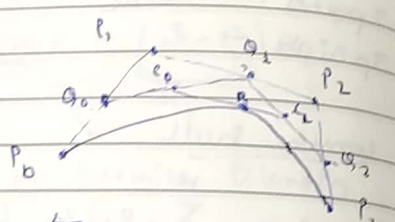
Notes

Eq. for 4 control Pts

$$Q(u) = \sum_{i=0}^{3} P_i \cdot B_{(i,3)}(u)$$

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

Derivation



Parametric eq of line

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

$$Q_i = (1-u) P_i + u P_{i+1}$$

$$C_i = (1-u) Q_i + u Q_{i+1}$$

$$R = (1-u) C_0 + u C_1$$

Substitute

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

$$= (1-u)\left[(1-u)^2 \left[(1-u) P_0 + u P_1\right]\right]$$

solve

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Properties: Bezier Curve:

(1) Always follow first and last control point.

(2) Polygon boundaries by C. Point.

(3) Bezier curve will always be inside convex hull of Polygon boundary.

(4) Poly eq degree is at most 1 less than control point

Drawback

(1) Degree = No of control points

(2) Global control: Ek coordinate ko change karne se puri shape change ho jaegi.

B-spline Curve (Basis)

→ Approximate spline Curve

→ local control point

→ blending function not necessary zero

→ We can specify the degree of order of curve. (k)

$k \leq n$

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Equation:

$$Q(u) = \sum_{i=0}^{n} P_i \cdot N_{i,k}(u)$$

$$N_{i,k}(u) = (u - x_i)\, N_{i,k-1}(u)$$

chord

Illumination Model

Shading	Illumination (Brightness)	Intensity (Quantity) (value)

Gouraud shading:

(1) Determine the Avg unit normal vector at each polygon vector.

$$N_V = \frac{N_1 + N_2 + N_3}{|N_1 + N_2 + N_3|}$$

N_i : no of surfaces sharing that vertex

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

(2) Apply Ill model to each polygon vector to determine polygon vertex intensity

Lambert's law.

(3) Linearly interpolate the vertex intensities over surface of polygon.

$$I_a = \frac{|y_a - y_1|}{|y_1 - y_2|} I_2 + \frac{|y_a - y_2|}{|y_1 - y_2|} I_1$$

$$I_b = \frac{|y_b - y_2|}{|y_2 - y_3|} I_3 + \frac{|y_b - y_3|}{|y_2 - y_3|} I_2$$

For point P

$$I_p = \frac{|x_p - x_a|}{|x_a - x_b|} I_b + \frac{|x_p - x_b|}{|x_a - x_b|} I_a$$

Advantage

$$I_c = I_a + \frac{|I_1 - I_2|}{|y_1 - y_2|}$$

y constant

less calculation.

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Advantage: Remove discontinuity by flat shading.

Disadvantage:

(1) Streak line
Mach band

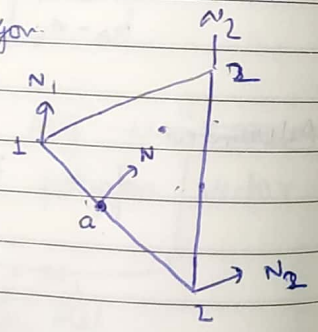
Phong Shading:

Normal vector Interpolation:

→ Determine the avg unit normal vector for vertex.

→ Linear interpolate the vertex normals over surface of polygon.

$$N_a = \frac{|y_a - y_1|}{|y_1 - y_2|} N_2 + \frac{|y_a - y_2|}{|y_1 - y_2|} N_1$$



Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Adv → hidden & visible / polygon filling
→ more realistic image
→ No - mach band
→ accurate

Dis → complex calculation.

Hidden and visible surfaces:

Algo use 2 approaches:
→ Object space method
→ Image space Method
→ Back face detection
→ z - Buffer Algo
→ Painters
→ Area - sub - Division.

Coherence: (similarity)

→ Object Coherence: Separate object.
→ Face coherence: High prob. that adjacent pixels belong to same phase
→ Edge coherence:
→ Area coherence: (Window) same properties and pixel in window.

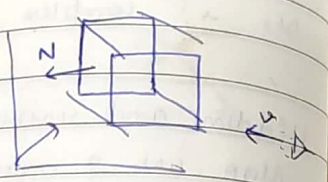
Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Back face Removal:

→ Non overlapping object.
→ Use object space Method.

(1)
$Ax + by + Cz + D < 0$
inside surface.
else back surface.



(2) $V \cdot N > 0$ hidden
i.e $\cos\theta \geq 0$
i.e $0 \leq \theta \leq \pi/2$

(3) $V \cdot N = V_z C$ $\quad V(0, 0, V_z)$

sign (C) ≤ 0 Back face
If we are viewing along -z dirn.
Otherwise visible sign(C) ≥ 0

Dis ad
→ Can't detect partially hidden surface.
→ Non overlapping for his has

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

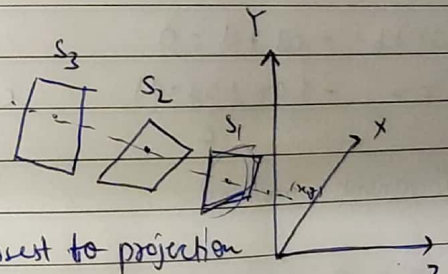
Notes

Z-Buffer Algorithm

Depth

→ Works well for overlapping surfaces
→ It compares depth at each pixel position.
→ Only used for OPAQUE surfaces
→ Image space method.
→ one frame buffer one depth buffer.

→ Algo compares surface depth at each pixel position on projection plane. Object depth is usually measured from the view plane along z-axis of a viewing system.



The surface closest to projection plane is visible surface

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Algo GFh..

depth (i,i) = infinite
color (i,i) = background

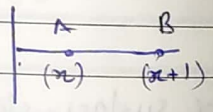
for (each pixel in polygon projection)
{
find depth (z value) at that (i,i)
if (z < d(i,j)){
d(i,j) = z
color(i,i) = dr
}
}

Calculation of depth

$ax + by + cd + d = 0$

$z = \frac{-(ax + by + d)}{c}$ $\quad c \neq 0$

increment method.



$z' = z - A/c$ $\quad (x' = x+1)$ scaba line.

Illy $z' = z - B/c$ $\quad (y' = y+1)$

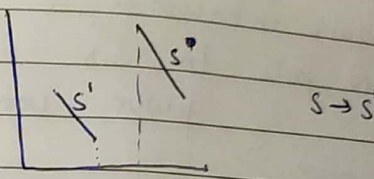
Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Painter's Algorithm Depth sort Algo.
Uses object & Image space method.
→ Sort surface → decreasing depth.

Test:

(1) Boundary Rectangle in 'x-y plane'
For the two surfaces don't overlap.



(2) Surface S is completly behind the over lapping surface wrt viewer.

GFh

Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes

Area Subdivision method.

→ Image space method

→ Area coherence

→ ~~Area~~ Divide and conquer.

NLN

calculation

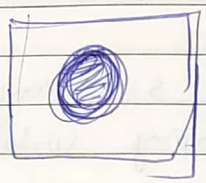
→ Multiplication is avoided in NLN

→ NLN fails in 3D

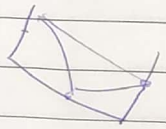
fewer comparisons

3 region

- window
- edge
- corner



3



Date:
Mon. Tue. Wed. Thu. Fri. Sat. Sun

Notes