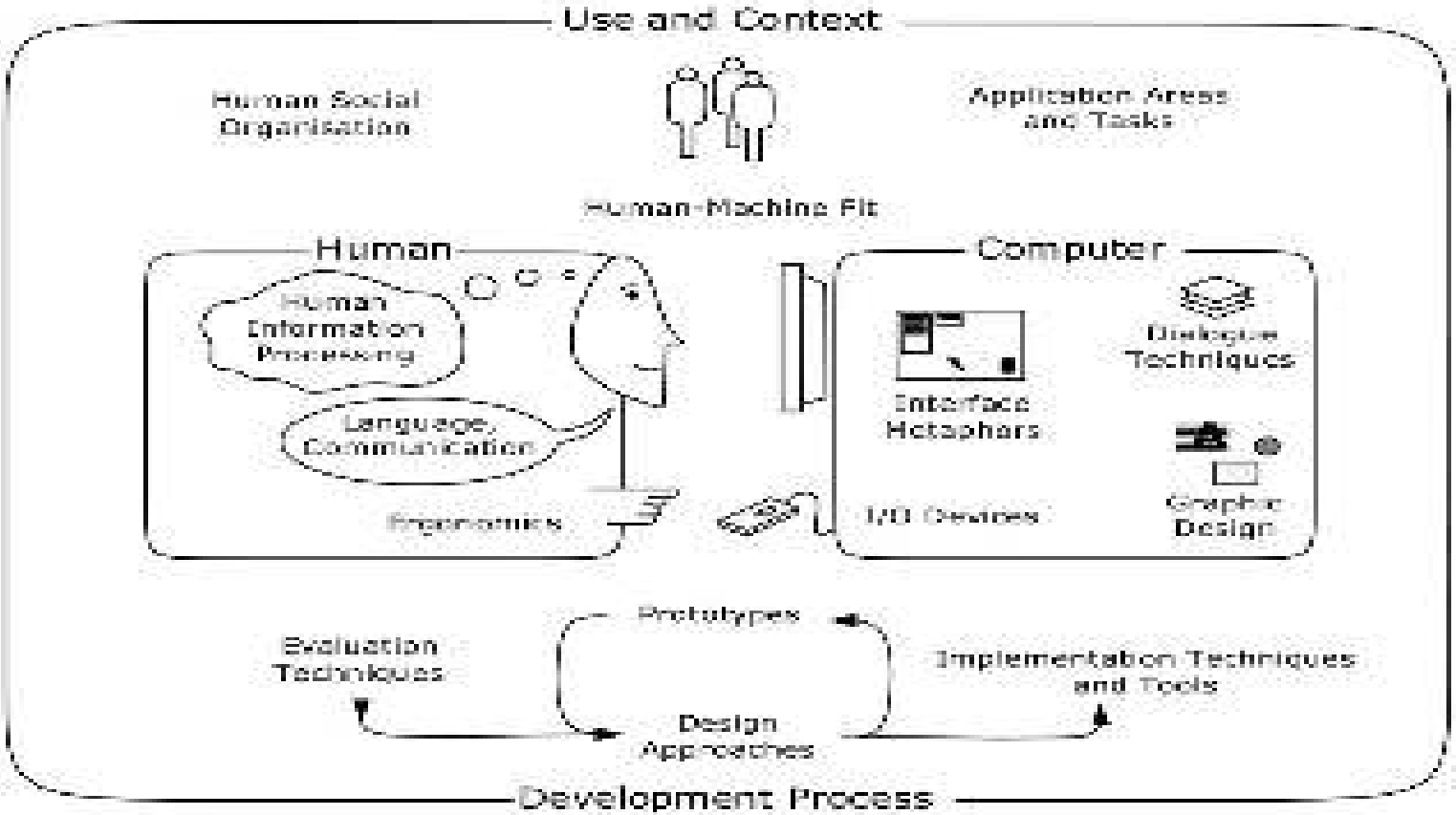
Izhodni modeli

Vsebina

- izhodni modeli
- risanje
- barvni modeli
- rasterizacija
- barvanje poligonov

Interakcije

- narava interakcije človek računalnik



Miller, MIT OpenCourseWare, 6.831, UI Design and Implementation

Interakcije

- trije izhodni modeli
  - komponente (objekti)
    - grafični objekti v drevesni hierarhiji z avtomatskim ponovnim izrisom
    - primeri: meniji, seznami, oznake, gumbi
  - osnovni grafični gradniki
    - visoko-nivojski osnovni grafični gradniki: črte, oblike, krivulje, besedilo
    - primer: *drawText(), drawLine()*
    - poznano kot: vektorska grafika, strukturna grafika
  - piksli
    - dvodimenzionalno polje pikslov
    - poznano kot: raster, slika, bitna slika, polje pikslov

Trije izhodni modeli

komponente (objekti)

risanje komponent

osnovni grafični gradniki

rasterizacija

piksli

Model komponent

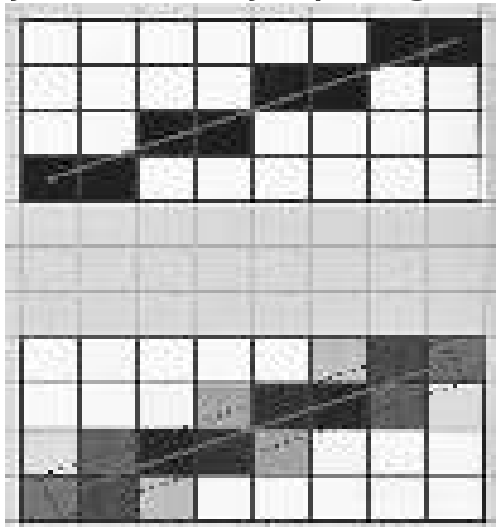
- komponente ali objekti
  - poznano kot:
    - pogledi
    - interaktorji
    - podobe (widget)
    - osnovni gradniki
  - primeri
    - meniji
    - seznami
    - oznake
    - .....

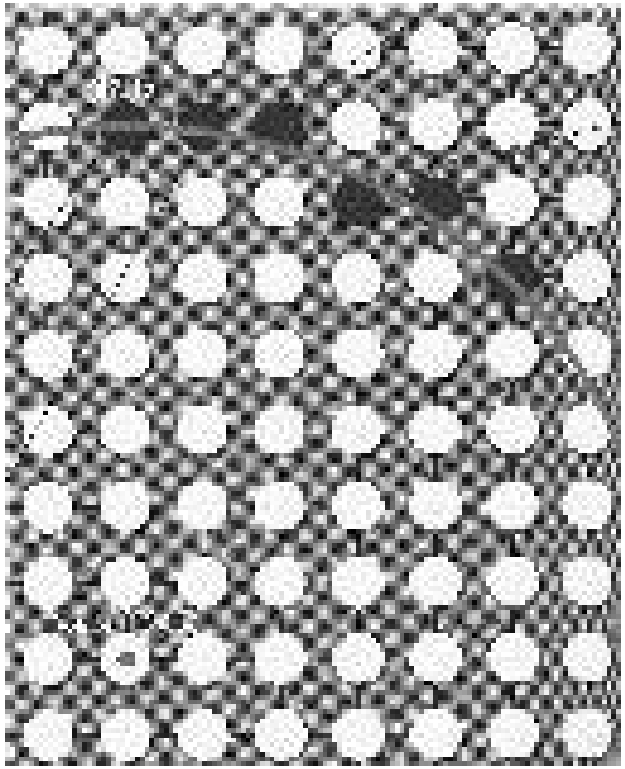
Model osnovnih grafičnih gradnikov

- risalna površina
  - struktura za risanje: »drawable« (»Pixmap«, »Window«)
  - zaslon, del pomnilnika, gonilnik za tiskalnik, oddaljen zaslon
- grafični kontekst
  - hrani grafične lastnosti, uporabljene pri izrisu, ni potrebno ob vsakem klicu pošiljati gradniku
  - pisava, barve, debelina črte, polnitev, robovi,...
- koordinatni sistem
  - izhodišče, skala, rotacija
- področje obrezovanja
- osnovni grafični gradniki
  - črta, krog (lok, elipsa), pravokotnik, besedilo, poligon, oblike

Pretvorba osnovnih gradnikov v piksle

- rasterizacijski algoritmi
  - za risanje črte
  - za risanje kroga
  - za glajenje osnovnih gradnikov
  - za polnjenje/barvanje poligonov





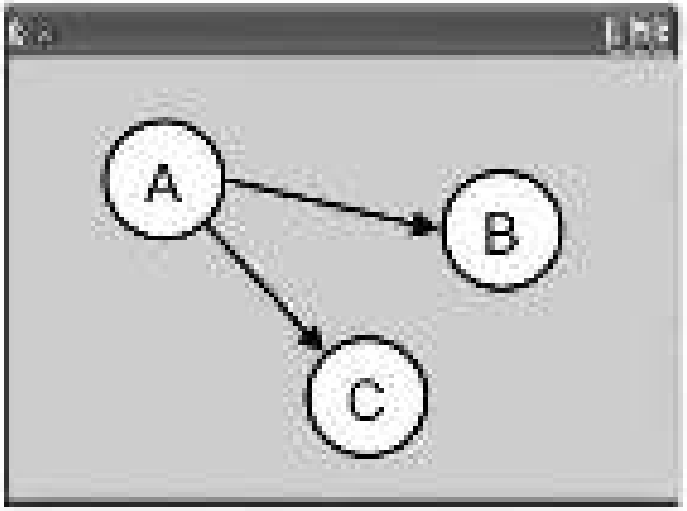
Miller, MIT OpenCourseWare, 6.831, UI Design and Implementation

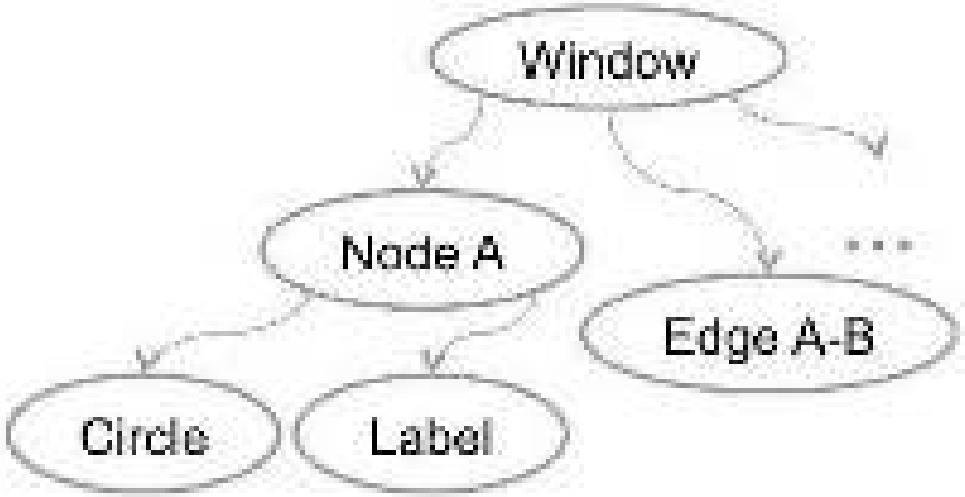
Modeli pikslov

- piksli so v pravokotnem polju
  - piksel je vektor (npr. RGB), polje je tridimenzionalno
- RGB barvne komponente imenujemo tudi kanali
- biti na piksel
  - 1 bit na piksel: črno/bel
  - 4-8 bitov na piksel: vsak piksel je indeks v iskalni tabeli
  - 24 bitov na piksel: 8 bitov za vsako barvo (npr. RGB)
  - 32 bitov na piksel: 8 bitov za vsako barvo, 8 bitov za alfa kanal
- piksli lahko organizirani na več načinov
  - v besede (RGBRGB...RGB) ali z izgubami (RGB- RGB-), soležno
  - v ločene ravnine (samo R, samo G, samo B)
  - od začetka proti koncu ali od konca proti začetku (npr. BMP)

Risanje komponent

- risanje poteka od vrha navzdol
  - gradnik nariše samega sebe z uporabo grafičnih osnovnih gradnikov ali pikslov
  - gradnik naroči neposredno vsebovanim gradnikom, da se narišejo



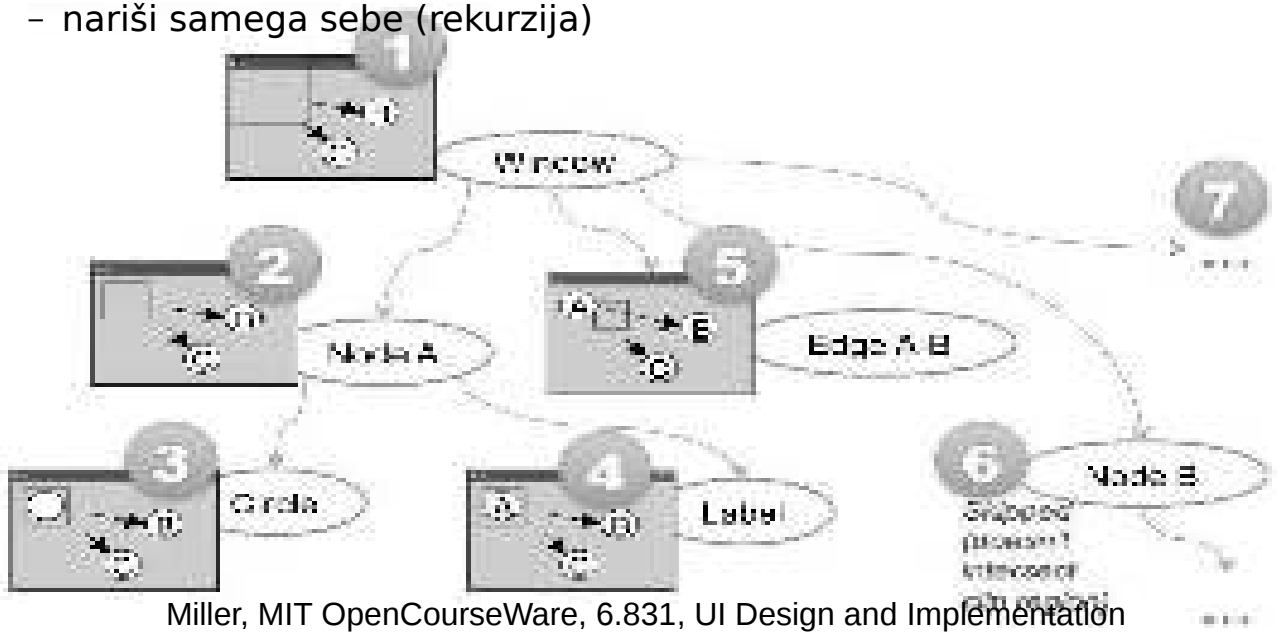


Risanje komponent

- določi področje rezanja kot presečišče med najvišjo podobo in področjem obrezovanja
- nariši samega sebe
  - nariši sebe v področju rezanja
  - za vsakega naslednika
    - če obstaja presečišče med področjem rezanja in naslednikom
      - določi področje rezanja kot presečišče med naslednikom in področjem obrezovanja -
      - nariši samega sebe (rekurzija)

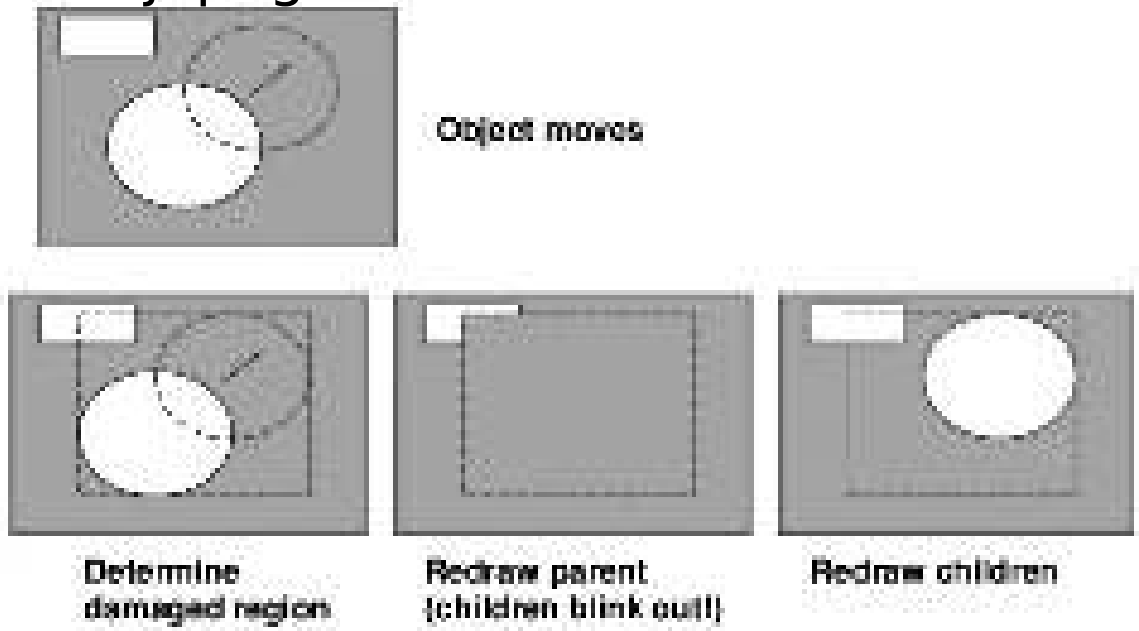
Risanje komponent-primer

- določi področje rezanja kot presečišče med najvišjo podobo in področjem obrezovanja
- nariši samega sebe
  - nariši sebe v področju rezanja
  - za vsakega naslednika
    - če obstaja presečišče med področjem rezanja in naslednikom
      - določi področje rezanja kot presečišče med naslednikom in področjem obrezovanja
      - nariši samega sebe (rekurzija)



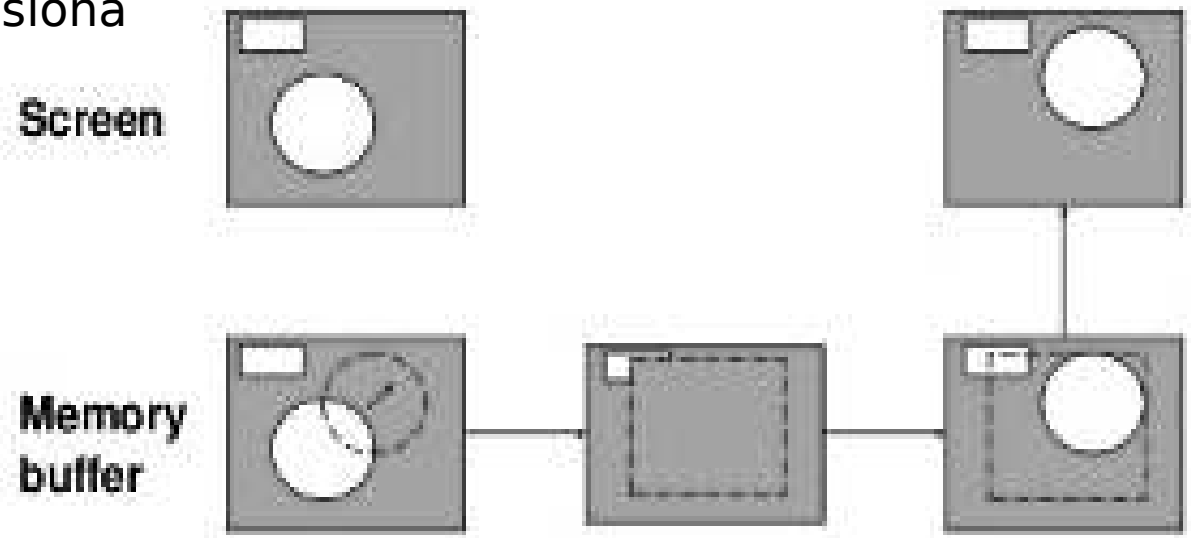
Naivni ponovni izris

- naivni ponovni izris ni primeren, saj povzroči učinek utripanja
- potrebno je risanje od vrha navzdol vseh objektov v hierarhiji pogleda



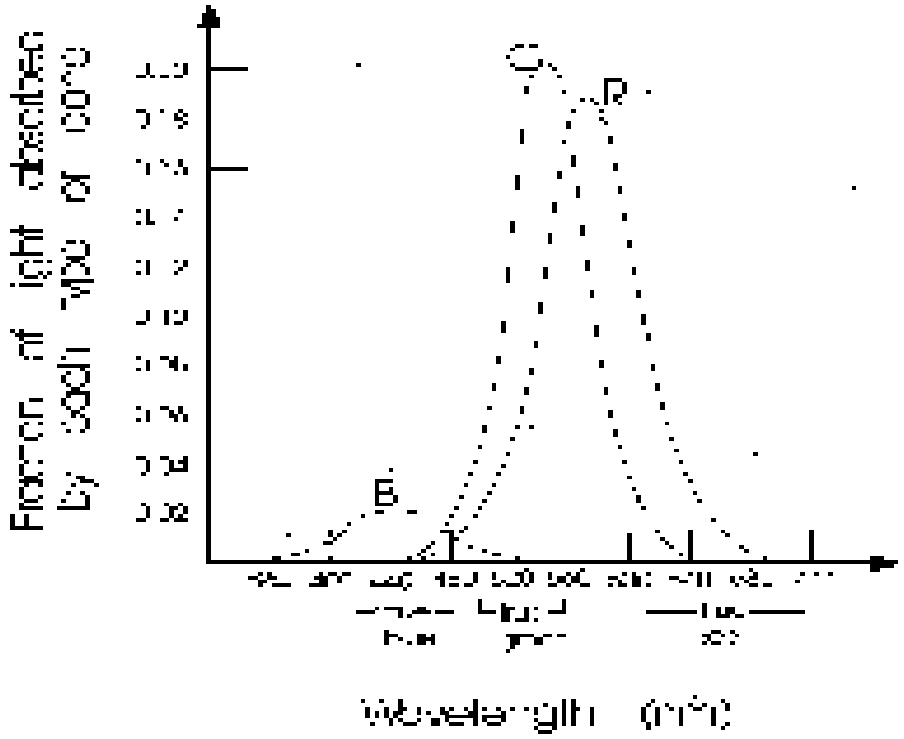
Dvoji pomnilnik

- dvojni pomnilnik reši problem utripanja
  - risanje v pomnilnik je hitrejše kot regeneracija zaslona
  - regeneracija pomeni, da zbrišemo gradnike
  - kopiranje je hitrejše od regeneriranje drevesne strukture
  - »poškodovani« objekti so zbrisani samo v pomnilniku
  - statični gradniki, ki morajo biti regenerirani, niso nikoli zbrisani z zaslona



Vizualno zaznavanje

- človekovo oko sestavljeno iz fotoreceptorjev: paličnice (svetlo/temno) in čepnice (barve)
- frekvenca: 430 – 750 * 10^12 Hz



Crane, 1997. Slika 1.7

Fotoreceptorji

- paličnice (vlakna): ~ 125 milijonov
  - občutljiva na spremembe svetlobe
  - bolj občutljiva
- čepnice (stožci): ~ 7 milijonov
  - občutljivi na barve
  - zelena in rdeča bolj občutljivo
  - modra manj občutljivo

Brezbarvna svetloba

- atribut v fizikalnem smisli
  - intenziteta: gostota moči alli energijskega toka, ki se izseva v prostorski kot
- atributi v fiziološkem smislu
  - svetlost (»lightness«): sprejeta intenziteta odbite brezbarvne svetlobe
  - svetlost (»brightness«): sprejeta intenziteta brezbarvne svetlobe lastno svetlečih objektov

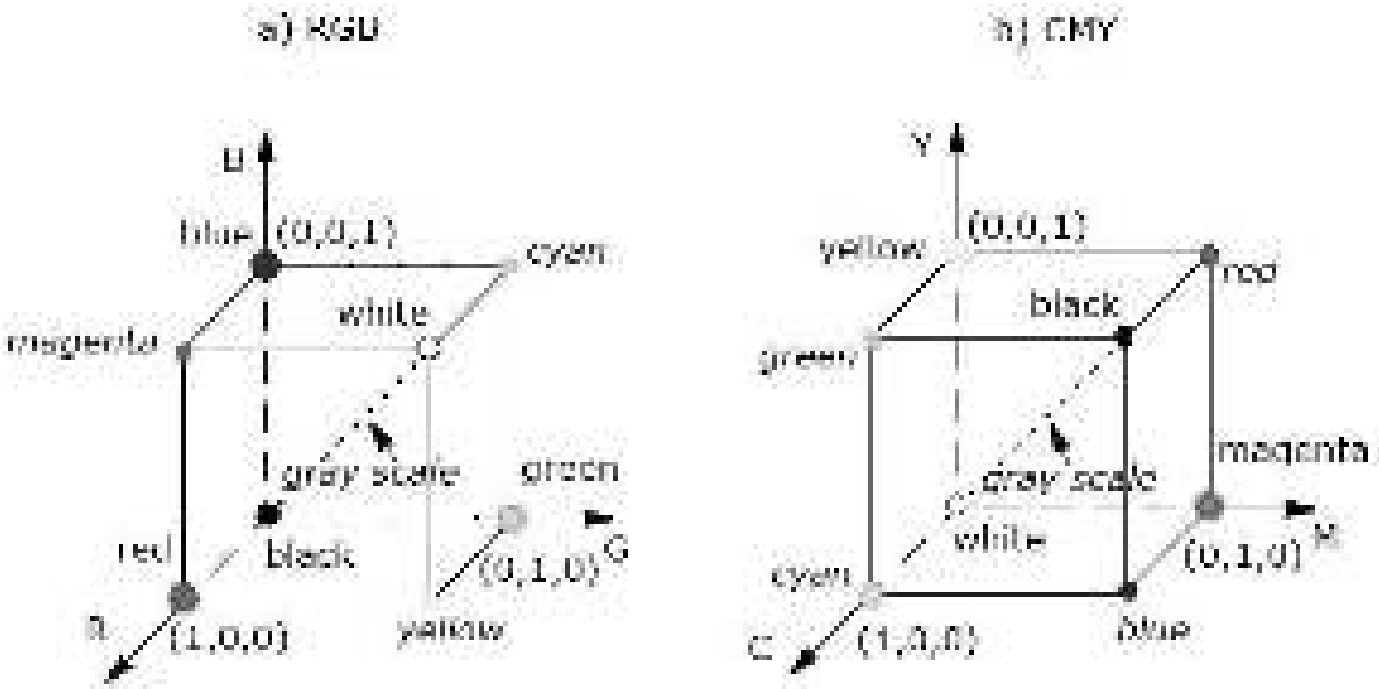
Barvna svetloba

- atributi v fizikalnem smisli
  - valovna dolžina (ali frekvenca, valovna dolžina*frekvenca=svetlobna hitrost)
  - čistost: 100% čista barva je nenasičena – brez primešane bele
  - intenziteta: gostota moči alli energijskega toka, ki se izseva v prostorski kot
- atributi v fiziološkem smislu
  - barva
  - nasičenje
  - svetlost (»lightness«): sprejeta intenziteta odbite brezbarvne svetlobe
  - svetlost (»brightness«): sprejeta intenziteta brezbarvne svetlobe lastno svetlečih objektov

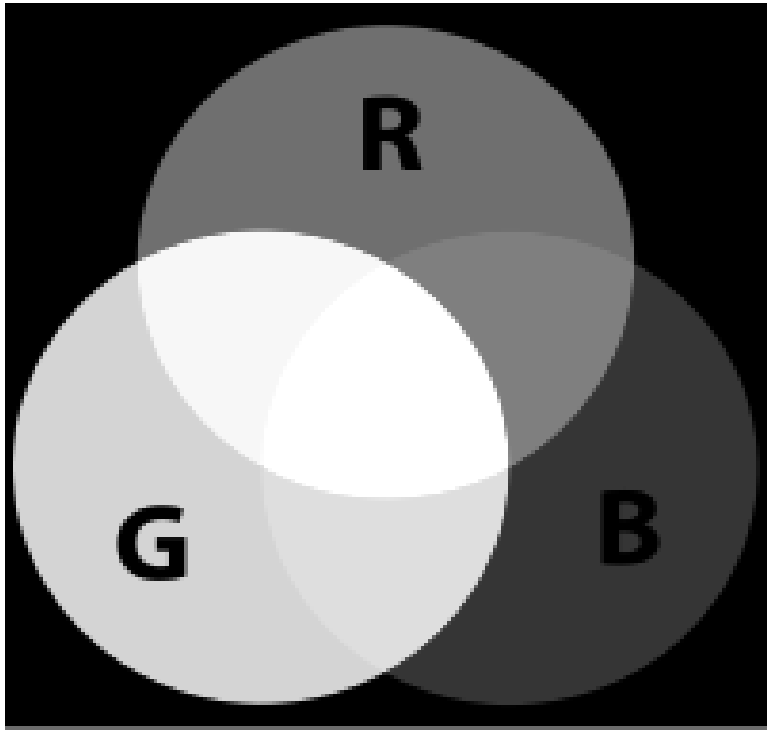
Barvni modeli

- RGB kocka: rdeča (R), zelena (G), modra (B)
  - pretežno uporabljen v računalništvu
- CMY: cianova (C), magenta (M), rumena (Y)
  - uporabljen v tiskarstvu: pigmenti absorbirajo valovne dolžine

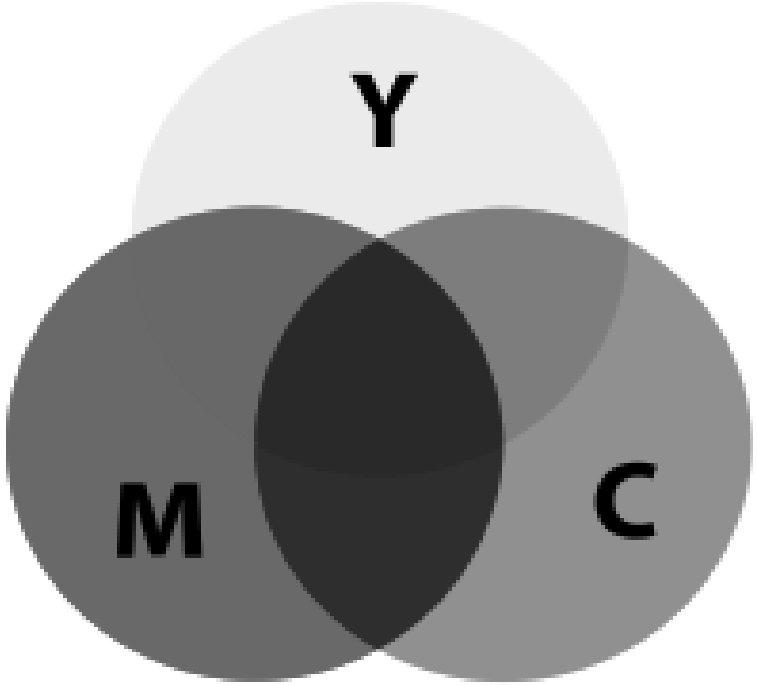
Barvni modeli



Barvni modeli



Additive RGB Color

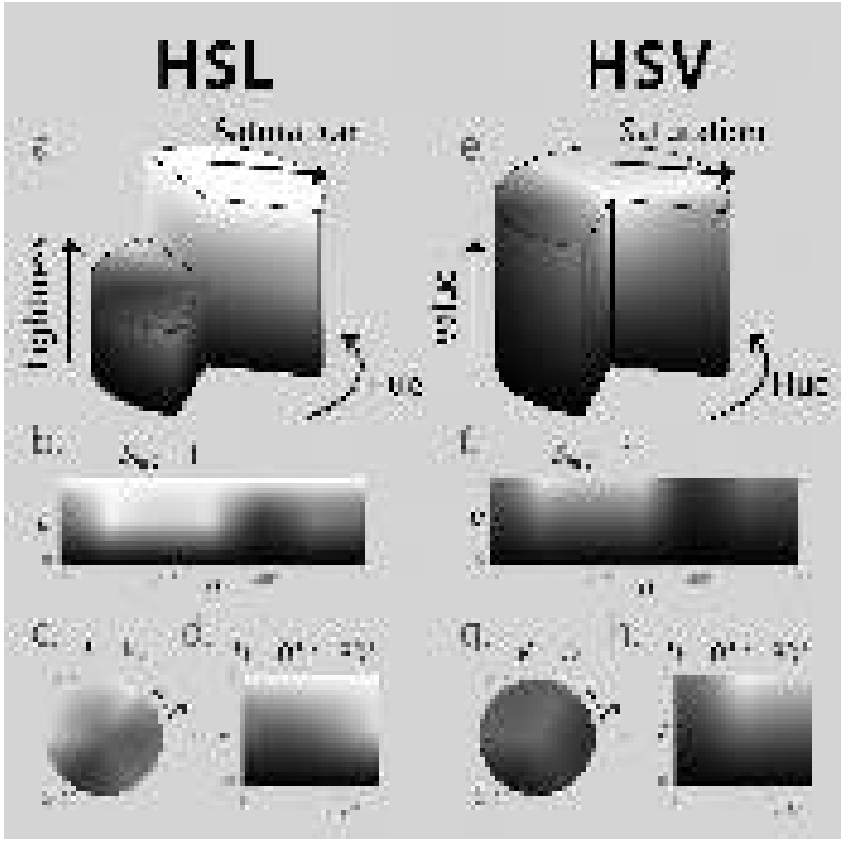


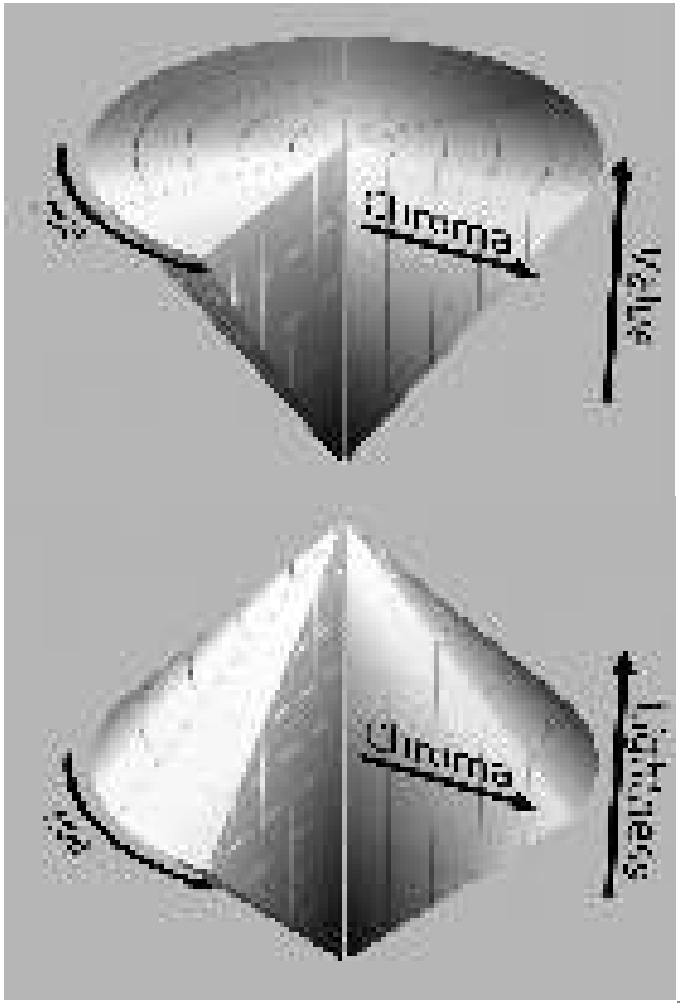
Subtractive CMY(K) Color

Barvni modeli

- HSV (ali HSB) stožec
  - odtenek (»hue«): vrsta barve, predstavlja kot okoli stožca
  - nasičenje (»saturation«): količina barve
    0 % (siva) – 100 % (barva)
  - intenziteta (»value«): 0 % (temna) – 100 % (svetla)
- HSL (ali HSL) dvojni stožec
  - odtenek (»hue«): vrsta barve, predstavlja kot okoli stožca
  - nasičenje (»saturation«): količina barve
    0 % (siva) – 100 % (barva)
  - intenziteta (»value«): 0 % (temna) – 100 % (svetla)
  - dvigne center HSV modela tako, da ima samo bela intenziteto 1.0, čiste barve pa imajo intenziteto 0.5

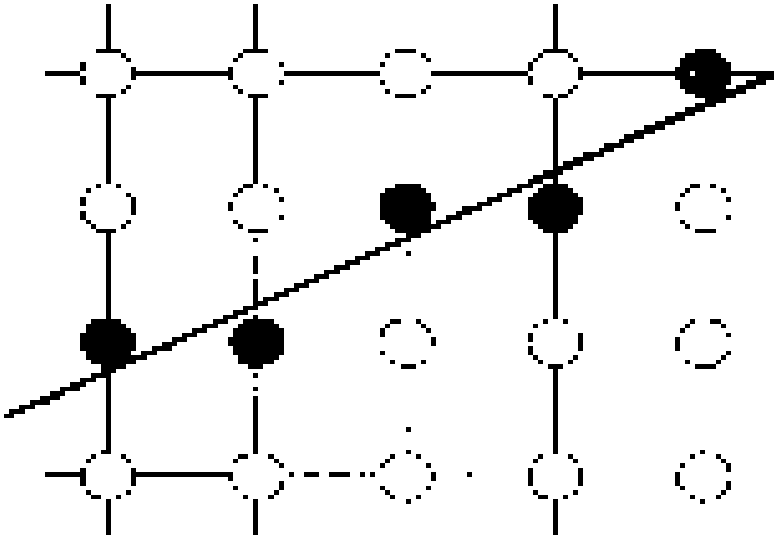
Barvni modeli

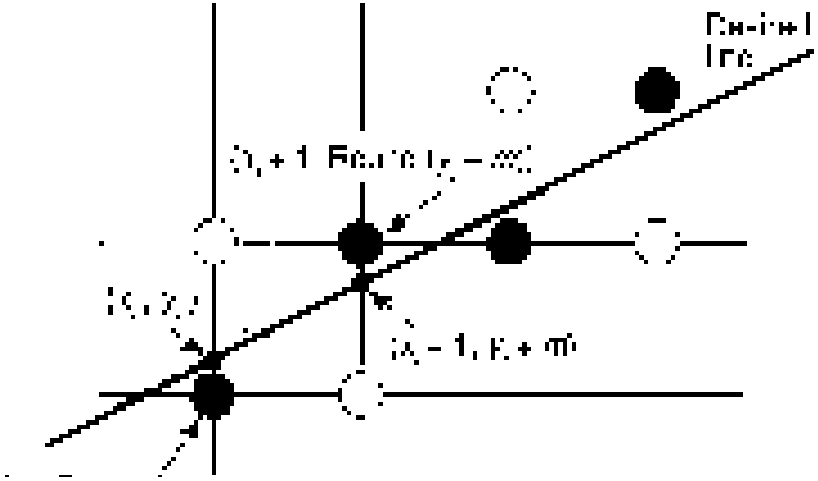




Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt: linearna enačba
    - osnovni algoritem

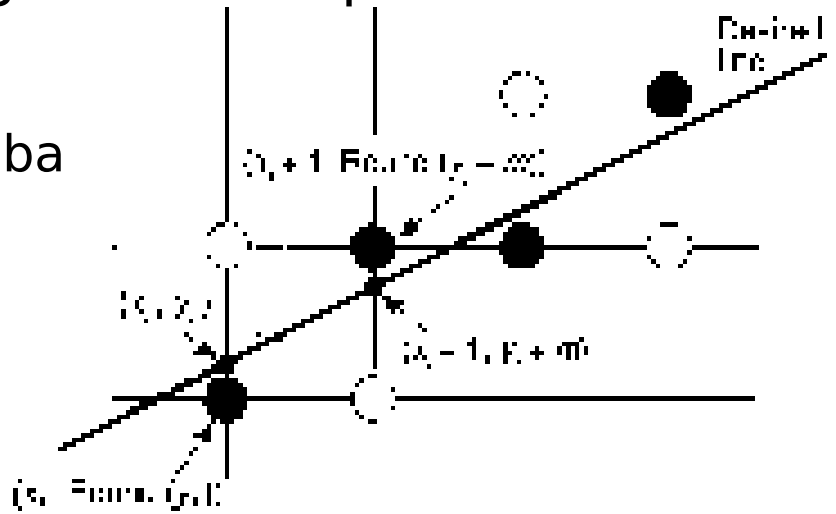




Foley et al, Slika 3.4 in 3.5

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt: linearna enačba
    - osnovni algoritem



$$y_i = m.x_i + B \qquad m = \Delta y / \Delta x$$

$$y_{i+1} = m.x_{i+1} + B \quad = \quad m.(x_i + \Delta x) + B \quad = \quad y_i + m.\Delta x$$

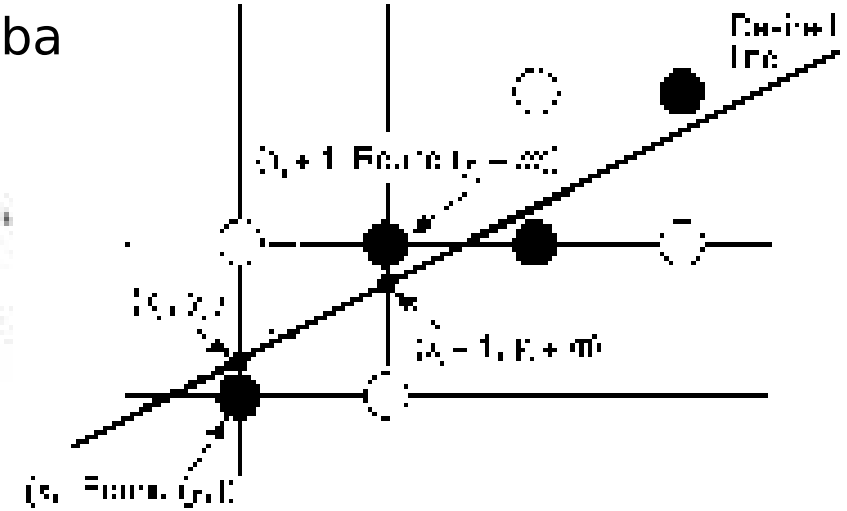
$$Ce \quad \Delta x = 1, \quad potem: \quad y_{i+1} = y_i + m$$

$$Ce \quad |m| > 1, \quad potem: \quad x_i \Leftrightarrow y_i, \quad \Delta x = \Delta y / m = 1/m$$

Foley et al, Slika 3.5

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt: linearna enačba
    - osnovni algoritem



```
void Line(                  /* Assumes  1 >= m >= 1,  x0 < x1 */
    int x0, int y0,         /* Left endpoint */
    int x1, int y1,         /* Right endpoint */
    int value)              /* Value to place in line's pixels */
{
  int x;
  double dy = y1 - y0;
  double dx = x1 - x0;
  double m = dy / dx;
  double y = y0;

  for (x = x0; x <= x1; x++) {
    WritePixel(x, Round (y), value); /* Set pixel to value */
    y += m;                          /* Step y by slope m */
  }
}    /* Line */
```

Foley et al, Slika 3.5

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt
    - algoritem najbližje točke

$$dx = x_1 - x_0, \qquad dy = y_1 - y_0, \qquad y = \frac{dy}{dx}x + B$$

$$F(x,y) = ax + by + c = 0$$

$$F(x,y) = dy\,x - dx\,y + dx\,B = 0$$

$$d = F(M) = F(x_p + 1, y_p + \frac{1}{2}) = a(x_p + 1) + b(y_p + \frac{1}{2}) + c$$

$$\text{Če} \quad d > 0: \quad NE, \quad \text{Če} \quad d <= 0: \quad E$$

$$NE: \quad d_{novo} = a(x_p + 2) + b(y_p + \frac{3}{2}) + c$$

$$d_{novo} = d_{stari} + a + b = d_{stari} + dy - dx, \qquad \Delta_{NE} = dy - dx$$

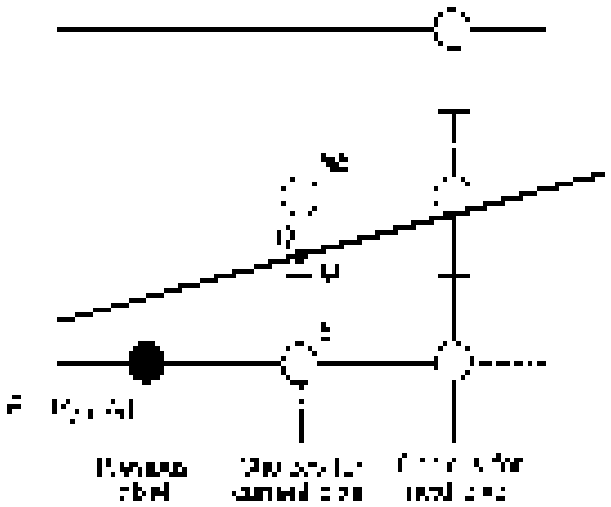
$$E: \quad d_{novo} = a(x_p + 2) + b(y_p + \frac{1}{2}) + c$$

$$d_{novo} = d_{stari} + a = d_{stari} + dy, \qquad \Delta_E = dy$$

$$d_{start} = a(x_0 + 1) + b(y_0 + \frac{1}{2}) + c$$

$$d_{start} = F(x_0, y_0) + a + \frac{b}{2} = 0 + dy - dx/2, \qquad d_{start} = dy - dx/2$$

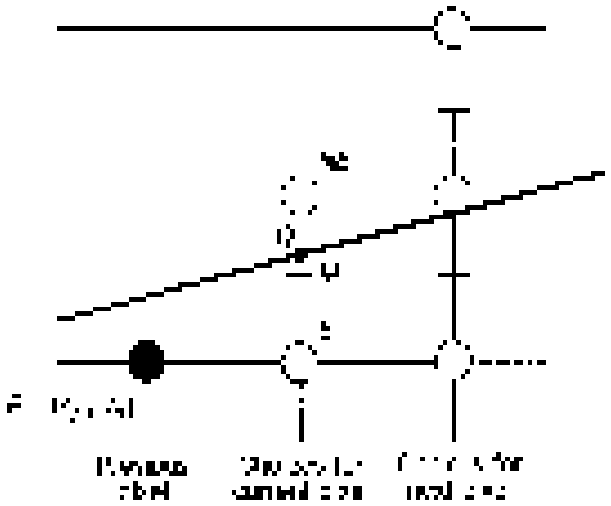
$$d_{start} = 2(dy - dx/2), \qquad \Delta_{NE} = 2(dy - dx), \qquad \Delta_E = 2\,dy$$



Foley et al, Slika 3.7

Rasterizacija

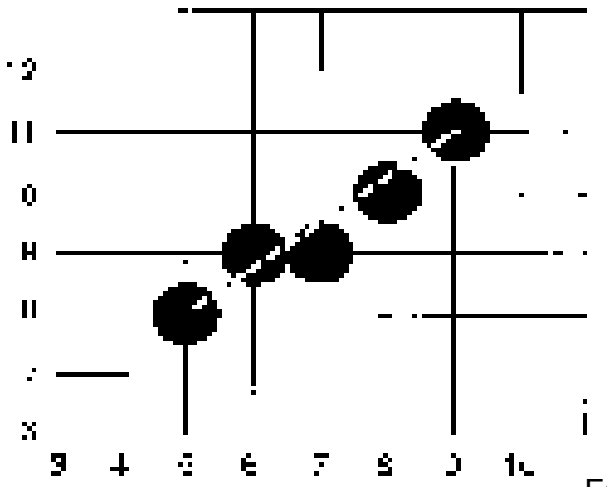
- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt
    - algoritem najbližje točke



Foley et al, Slika 3.7 in 3.8

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje črt
    - algoritem najbližje točke



Foley et al, Slika 3.8 in 3.9

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje kroga
    - algoritem najbližje točke

$$F(x,y) = x^2 + y^2 - R^2$$

$$d_{SE} = (x_p + 1)^2 + (y_p - \frac{1}{2})^2 - R^2$$

$$Če \quad d_{SE} >= 0: \quad SE; \quad Če \quad d_{SE} < 0: \quad E$$

$$SE: \quad d_{nov} = (x_p + 2)^2 + (y_p - \frac{3}{2})^2 - R^2$$

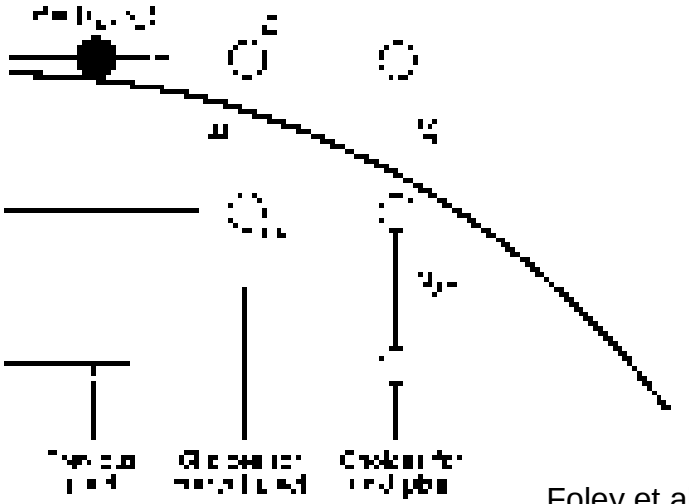
$$d_{nov} = d_{SE} + 2x_p - 2y_p + 5, \quad \Delta_{SE} = 2x_p - 2y_p + 5$$

$$E: \quad d_{nov} = (x_p + 2)^2 + (y_p - \frac{1}{2})^2 - R^2$$

$$d_{nov} = d_{SE} + 2x_p + 3, \quad \Delta_E = 2x_p + 3$$

$$d_{start} = F(1, R - \frac{1}{2}) = 1 + (R - \frac{1}{2})^2 - R^2 = \frac{5}{4} - R$$

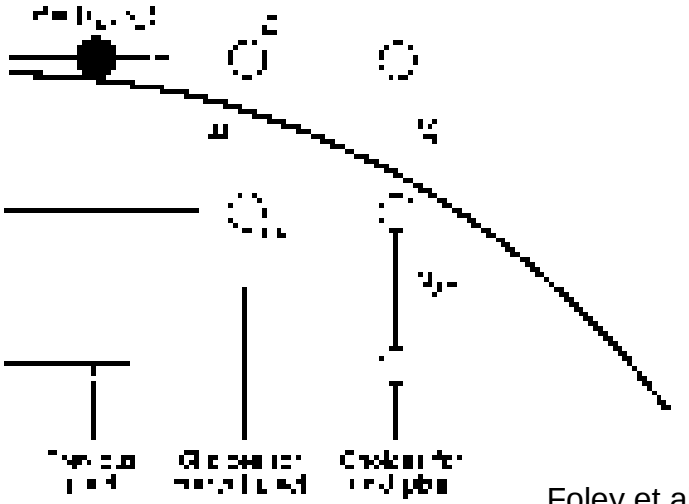
$$d_{start} = 1 - R$$



Foley et al, Slika 3.14

Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje kroga
    - algoritem najbližje točke



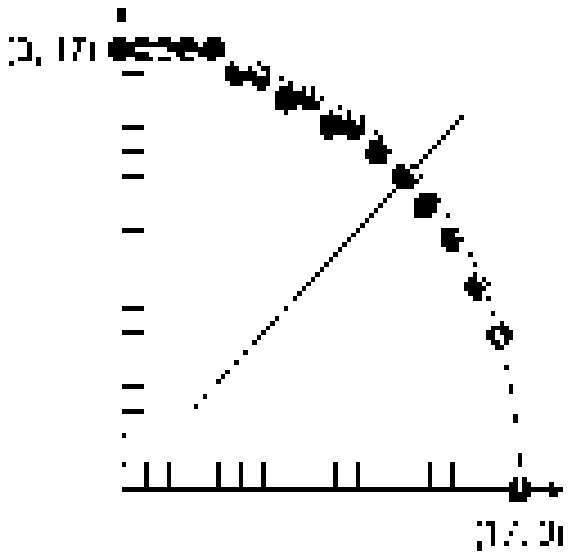
```
void MidpointCircle (int radius, int value)
{
    int x = 0;
    int y = radius;
    int d = 1 - radius;
    CirclePoints (x, y, value);

    while (y > x) {
        if (d < 0)
            d += 2 * x + 3;
        else {
            d += 2 * (x - y) + 5;
            y--;
        }
        x++;
        CirclePoints (x, y, value);
    }
}
```

Foley et al, Slika 3.14 in 3.15

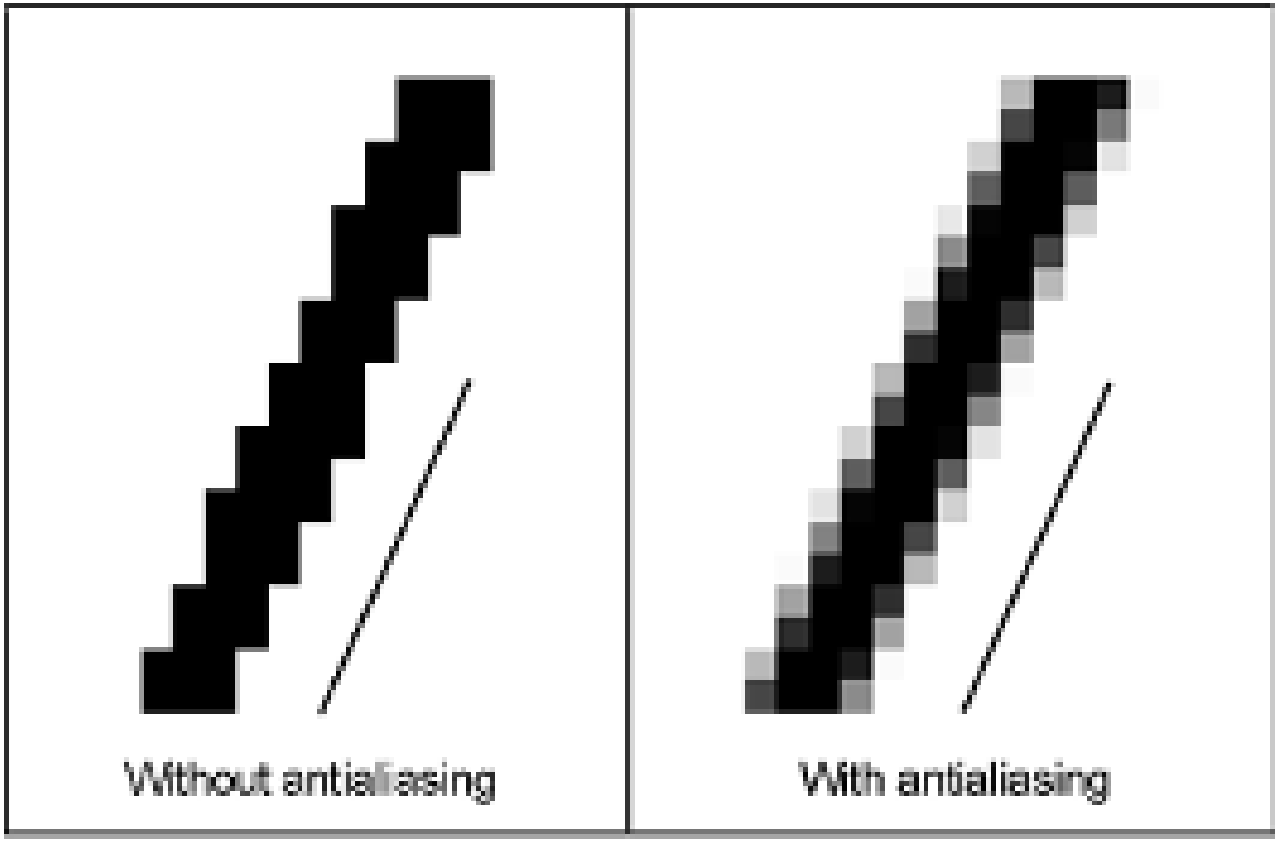
Rasterizacija

- pretvorba osnovnih grafičnih gradnikov v piksle
  - rasterizacijski algoritmi
    - za risanje kroga
    - algoritem najbližje točke



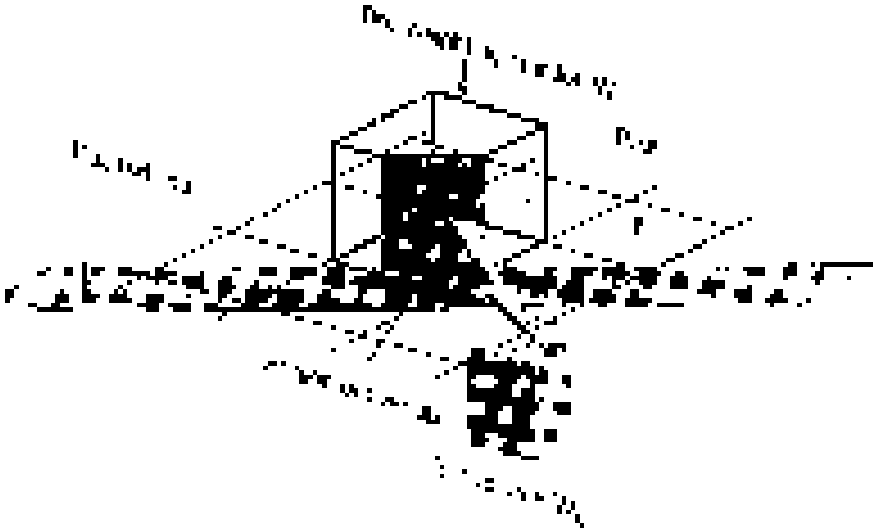
Foley et al, Slika 3.12 in 3.15

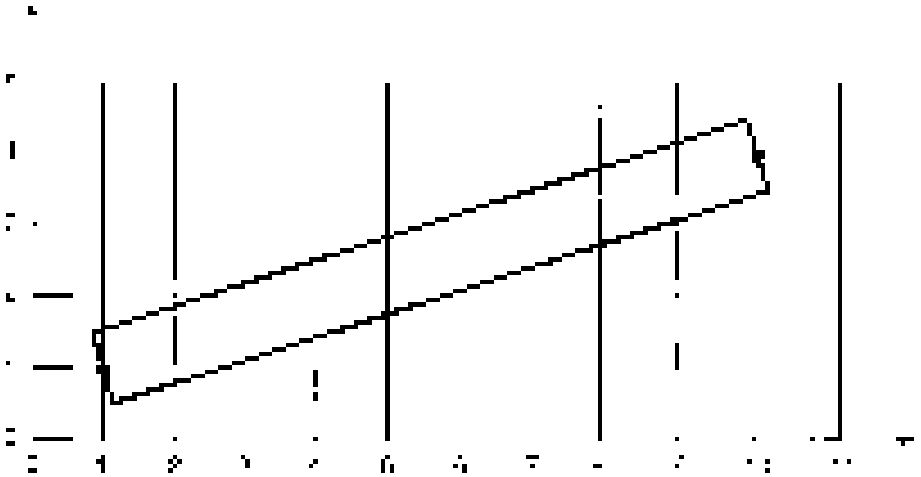
Glajenje osnovnih gradnikov



Glajenje osnovnih gradnikov

- glajenje z neuteženim prekrivanjem površine

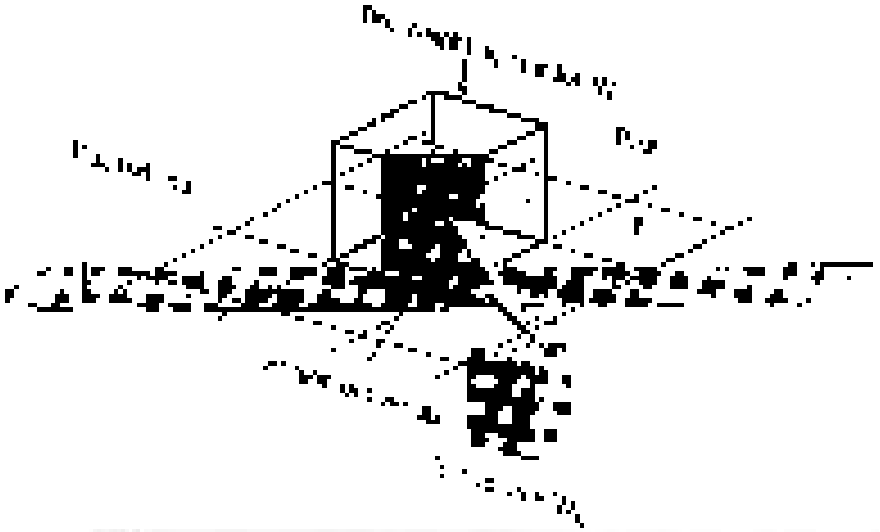


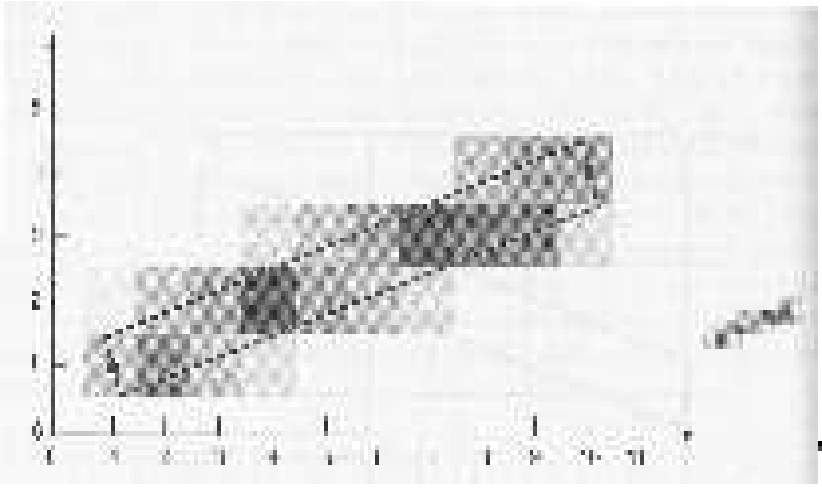


Foley et al, Slika 3.55 in 3.57

Glajenje osnovnih gradnikov

- glajenje z neuteženim prekrivanjem površine
  - bolj oddaljen piksel ima manjšo intenziteto
  - gradnik nima učinka na intenziteto če ni prekrivan
  - enako velika področja prekrivanja prispevajo enako intenziteto ne glede na oddaljenost

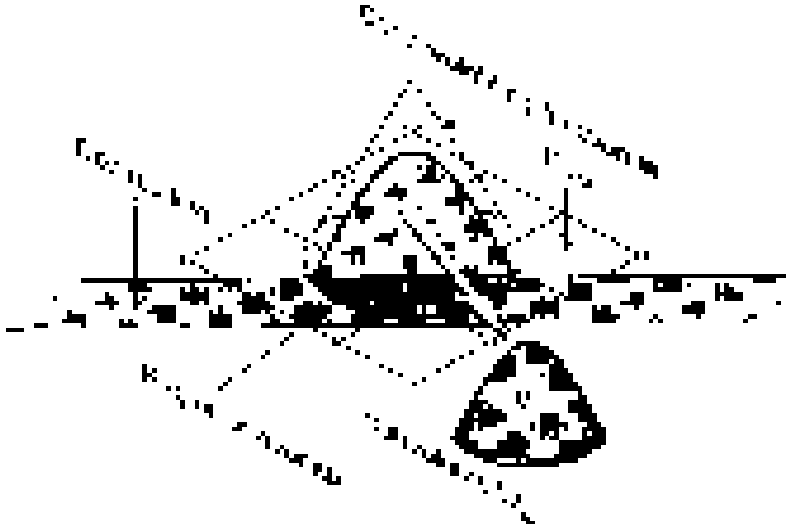


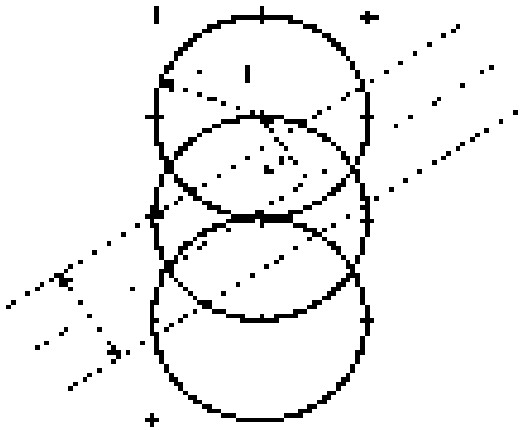


Foley et al, Slika 3.55 in 3.56

Glajenje osnovnih gradnikov

- glajenje z uteženim prekrivanjem površine
  - bolj oddaljen piksel ima manjšo intenziteto
  - gradnik ima učinen na intenziteto tudi če ni prekrivanja (krog z radijem ena)
  - enako velika področja prekrivanja imajo večji učinek, če so bližje središču piksla





Foley et al, Slika 3.58 in 3.60

Barvanje poligonov

- prva ideja: rekurzija
  - težave: hitro pride do porabe pomnilnika, počasno
- izboljšava: iterativni vrstični postopek barvanja
  1. naključno izberi piksel (ključni piksel) v poligonu
  2. barvaj levo in desno od piksla do robov poligona
  3. v neobarvanih pikslih v vrstici zgoraj in nato spodaj izbiraj ključne piksle, ki imajo na desni za soseda rob in jih shrani v sklad
  4. iz sklada vzemi naslednji ključni piksel
  5. če je ključni piksel obarvan skoči na točko 4, sicer pa na točko 2

Barvanje poligonov