Koherencia, Interpolácia…
(x,y,z,t,r,g,b,alfa)

Andrej FERKO

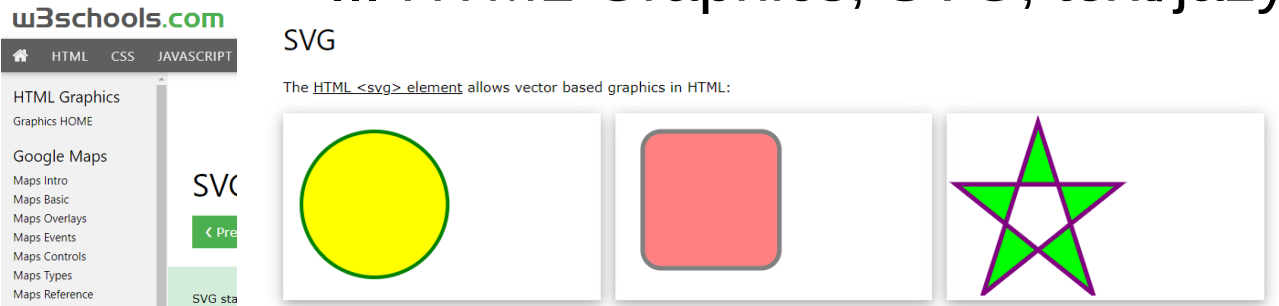
Comenius University Bratislava

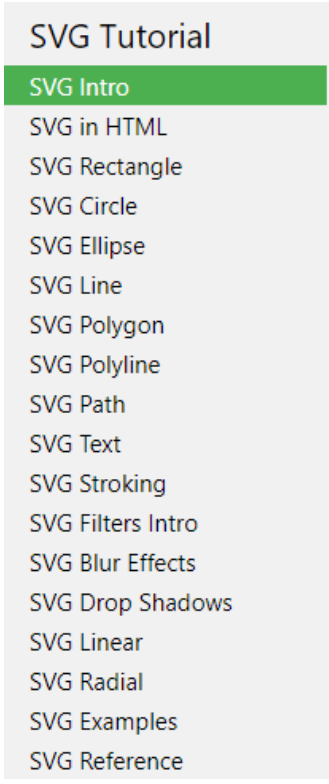
PG1, 26. 10. 2020, FMFI UK

Tri metodiky, funkčnosť

- Matematické modelovanie, model >> obraz
- Web (alebo PPTX) design, nápad >> obraz
- Vizualizácia, alfanumerické dáta >> obraz

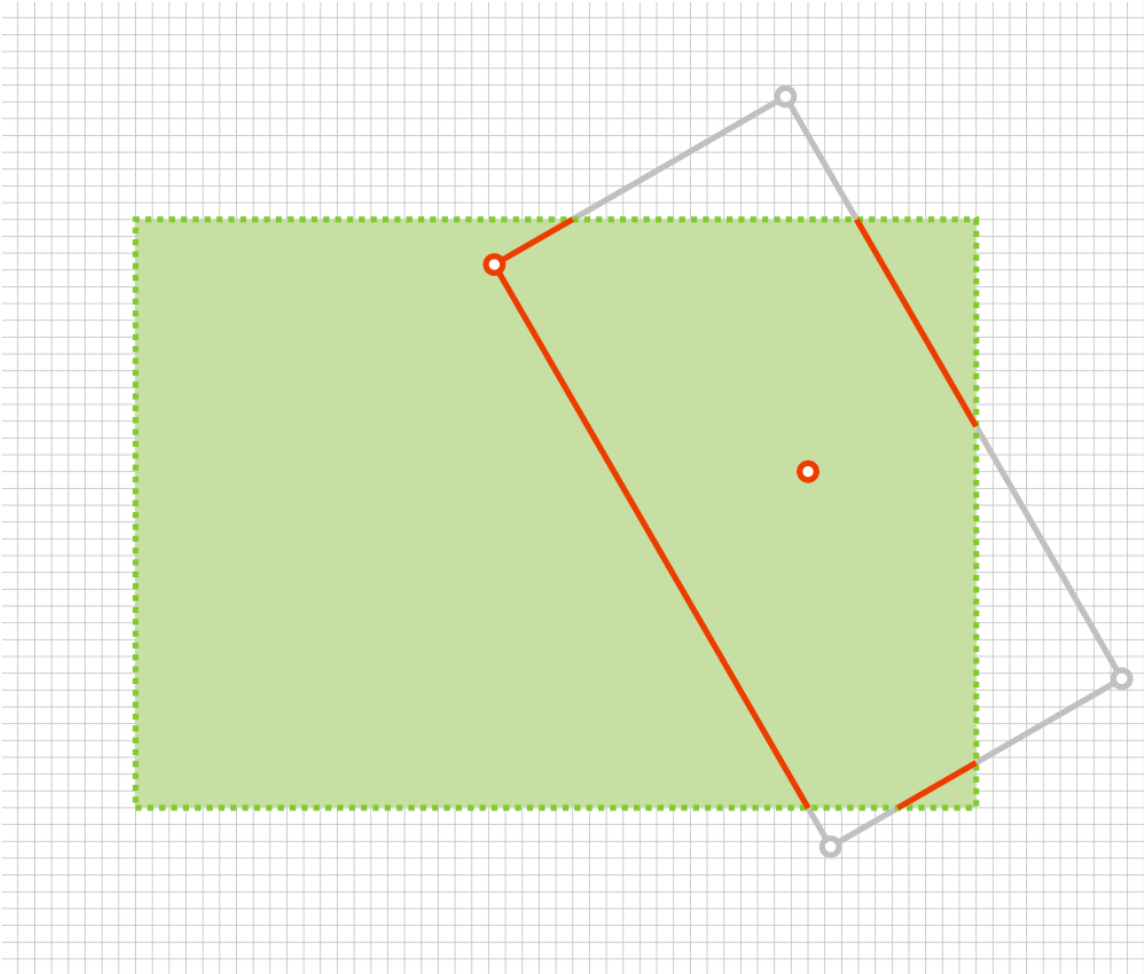
- Ako vytvoriť obraz (image, picture)?
- Procedurálne, grafický systém (jazyk)
- Deklaratívne, grafický formát, napr. SVG
- ... HTML Graphics, SVG, text/jazyk

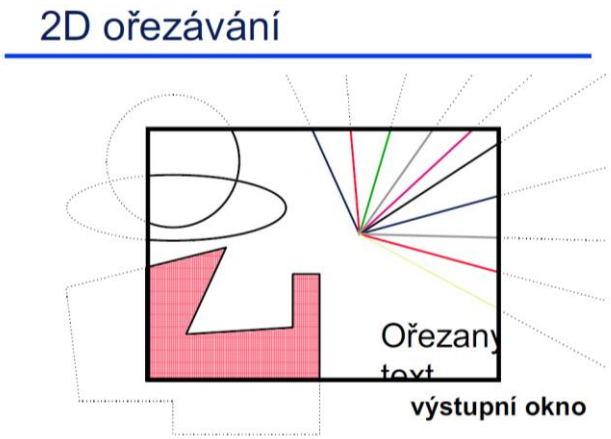


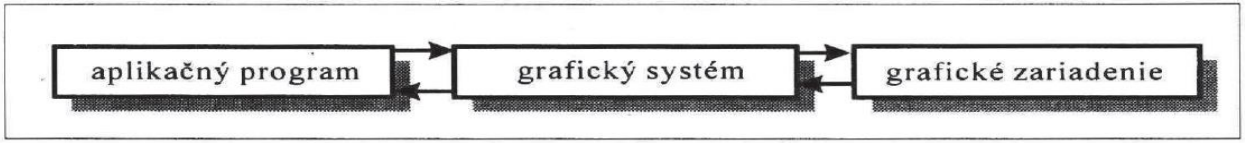


Viditeľnosť,

obr. J. Kučerová ZPGSO´15, J. Pelikán PG´11







Obr. 1.8 Základná štruktúra interaktívnej grafickej aplikácie

Algoritmické stratégie

- Do okien a záberov treba orezať body, úsečky či výplňové oblasti, aby sme nepočítali ani nezobrazovali zbytočné.
- Tak ako sme pridali ku 2D ďalší rozmer, pridáme k oknu v rovine ďalšie jej časti, každý bod v Euklidovej rovine patrí do jedinej z deviatich.
- Zatiaľ sme využili algoritmickú stratégiu **Duality**, orezávanie realizuje Odsekni a hľadaj (**Prune & Search**). Rasterizácia uplatní **Iteration, Divide & Conquer** a **Sweeping Technique**.
- Zmena hľadiska: **Čo nepočítať?**
- Viac o algoritmických stratégiách: Chalmovianský, P. et al. 2001. ***Zložitosť geometrických algoritmov.***

Okno, záber, orezanie

- **Set Window (id, ...)**
- **Set Viewport (id, ...)**
- **Set Clipping Indicator (Clip/Noclip)**

Obrázky zostrojíme z častí, ktoré nazývame **grafické výstupné prvky**:

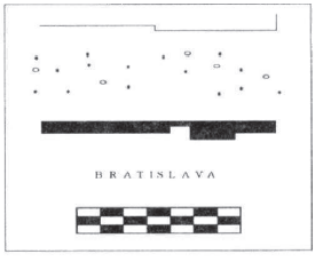
1. lomená čiara: kreslí postupnosť spojených úsečiek

2. sled značiek: označí postupnosť bodov značkou

3. výplňová oblasť: kreslí plochu danú bodmi hranice

4. text: vykreslí znakový reťazec

5. pole buniek: vytvorí obraz z buniek rôznych farieb



Obr. 17.2 Výstupné prvky

6. zovšeobecnený grafický výstupný prvok:

vykreslí napr. elipsu, ak to vie zariadenie.

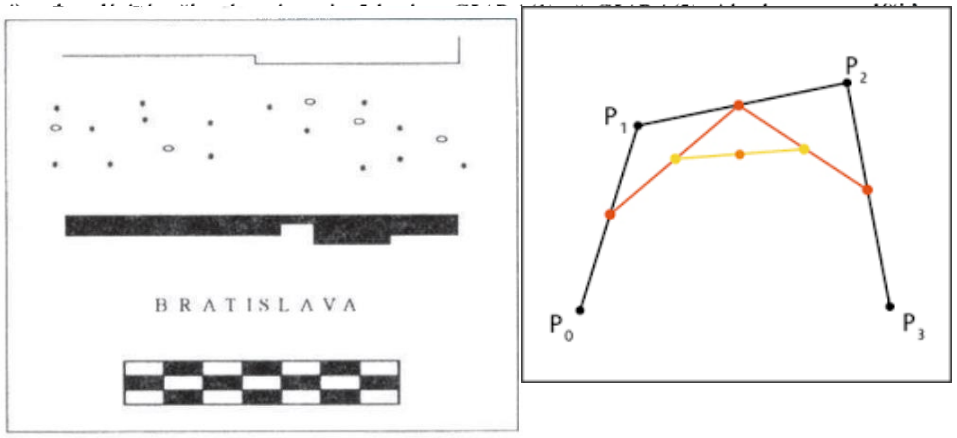
Grafický prvok má množinu **parametrov**, ktoré určujú jeho konkrétny tvar. Napr. TEXT má dva parametre - pozíciu a reťazec, príkaz TEXT (*WHERE, "ABC"*) vykreslí znaky *ABC* na pozícii *WHERE*. Kým parametrami riadime tvar grafických prvkov, ich **aspekty vzhľadu** (napr. farbu, font) riadime **atribútmi**. Atribút môže mať rovnakú hodnotu pre viac prvkov, platí **modálne**. Napr. s daným fontom možno vykresliť niekoľko reťazcov.

Atribúty v GKS patria do 2 skupín - **1. globálne**, ktoré možno použiť na všetkých staniciach, a **2. závislé na stanici**. Globálne atribúty možno nastaviť **individuálne**, napr. priamo výšku znaku. "Lokálne" závislé atribúty združujeme do tzv. **zväzkov** (*bundles*) a riadime indexom v tabuľke zväzku pre danú stanicu. Index určuje **reprezentáciu prvku na stanici.**

Lomenú čiaru vykreslíme volaním funkcie

POLYLINE (*N, POINTS*),

kde *POINTS* je zoznam *N* bodov - od bodu *POINTS(1)* po bod *POINTS(N)*. Lomená čiara sa skladá z N-1 úsečiek, spájajúcich po sebe nasledujúce body. *POLYLINE* (*5, CIA-*



Obr. 17.2 Výstupné prvky

16.3 Spracovanie výstupu v oknovom systéme

Oknový systém musí mať na spracovanie výstupu aspoň tieto základné funkcie:

Create Window (name) - vytvorí okno s daným menom

Set Position (xmin, ymin) - nastaví pozíciu aktuálneho okna

Set Size (height, width) - nastaví veľkosť aktuálneho okna

Select Window (name) - určí aktuálne okno

Show Window - zobraz aktuálne okno

Hide Window - skry aktuálne okno

Set Title (name) - nastav meno aktuálneho okna

Get Position (xmin, ymin) - zisti pozíciu aktuálneho okna

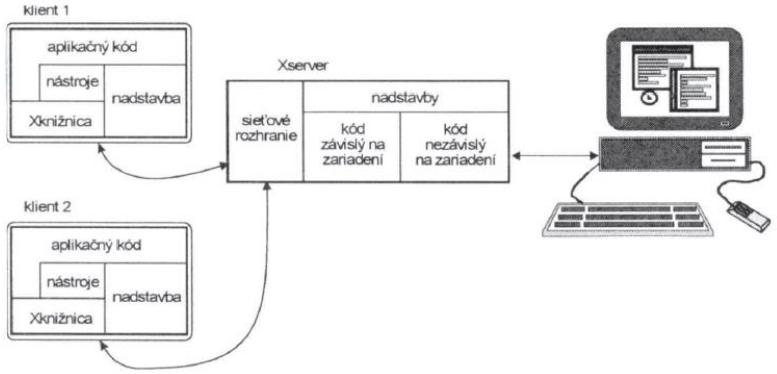
Get Size (height, width) - zisti veľkosť aktuálneho okna

Bring To Top - pošli aktuálne okno na vrch všetkých okien

Send To Bottom - pošli aktuálne okno na dno, za všetky okná

Delete Window - zruš aktuálne okno

Tým sme popísali minimálnu funkčnosť oknového systému pri spracovaní výstupu, pričom algoritmické riešenia tejto funkčnej špecifikácie nás na tejto úrovni nezaujímajú, hoci niektoré úvahy môžeme naznačiť. Napr. uvedené funkcie predpokladajú obdĺžnikové okno s menom, rozmermi a pozíciou na obrazovke, súbor takýchto okien s



Obr. 16.3 Architektúra X window system

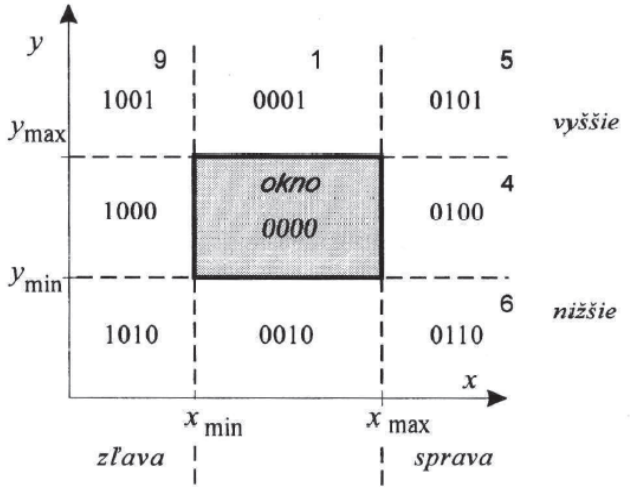
Orezávanie, *J. Pelikán PG´01*

3.3.1 Algoritmus orezávania Cohena-Sutherlanda

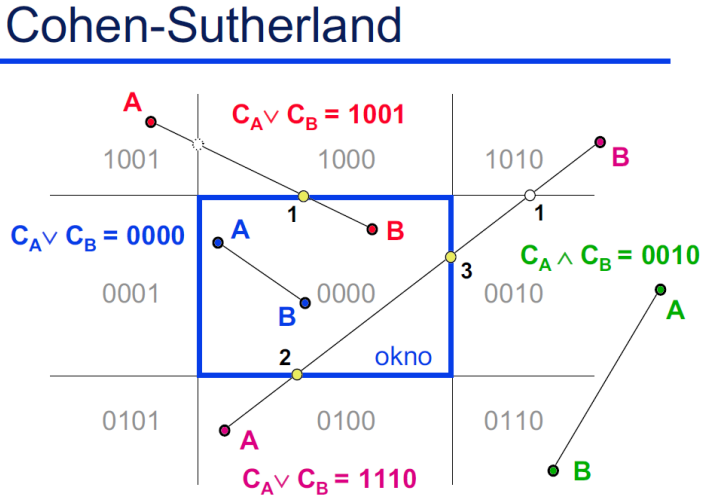
Tento algoritmus rýchlo vylúči vyššie spomenuté prípady. Zvlášť rýchly je v prípade okna, ak obsahuje veľa úsečiek vo vnútri a taktiež pri takom okne, ak väčšina úsečiek je mimo okna. V týchto prípadoch sa úsečka podľa polohy buď celá zobrazí alebo nezobrazí.

Na začiatku algoritmu nastavíme 4-bitové hodnoty (kód) pre obidva koncové body úsečky podľa polohy vzhľadom na okno. Pre každú hraničnú priamku okna nastavíme jeden bit podľa toho, či leží bod v danej polrovine (pozri obr. 3.1). Pravidlo upresníme podľa polohy bodu vzhľadom na okno:

1. bit - bod leží vyššie od okna ($x_{max} < x$);
2. bit - bod leží nižšie od okna ($x_{min} < x$);
3. bit - bod leží sprava od okna ($y_{max} < y$);
4. bit - bod leží zľava od okna ($y_{min} < y$).



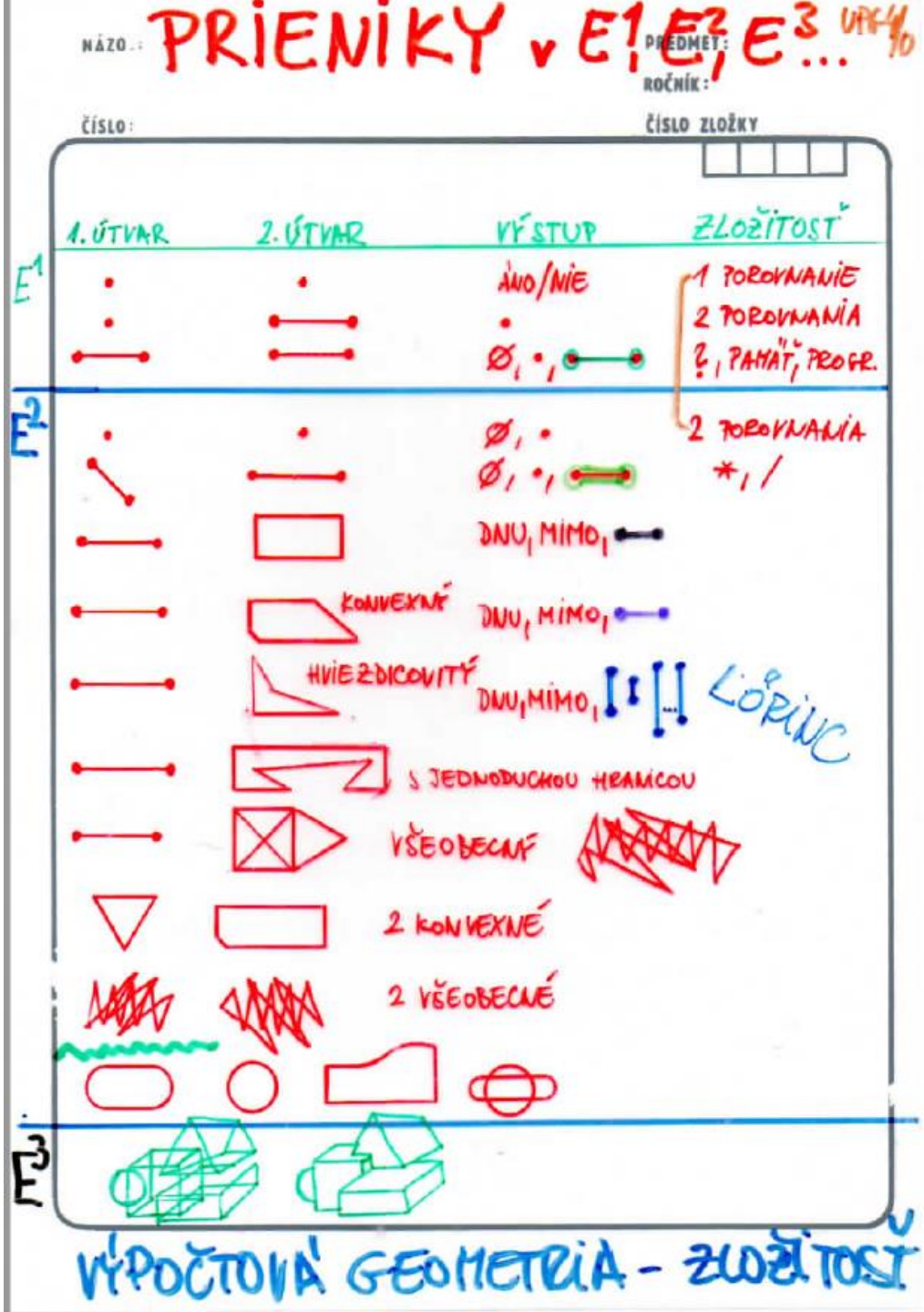
Obr. 3.1 Kódovanie jednotlivých častí roviny



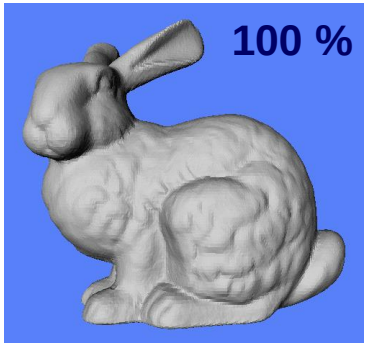
Ružický, E. et al. 2012. ***Počítačová grafika a spracovanie obrazu.*** http://www.sccg.sk/~ferko/PGASO2012-bookmarks.pdf

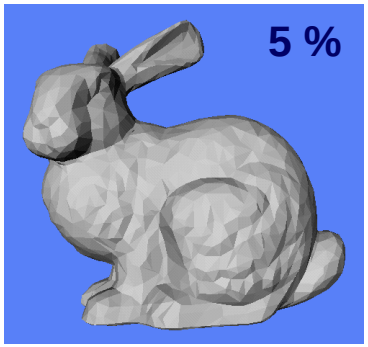
Algoritmus Cohen-Sutherlanda

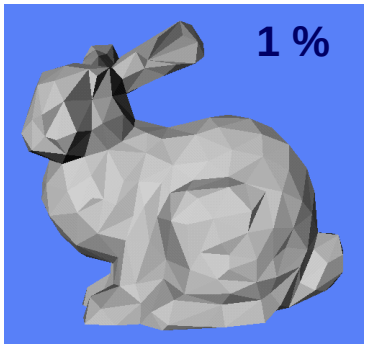
```
Procedure Clipping;
begin
1.  accept:= false;                                   { nastav - P1P2 sa nevykresľuje }
    outcod (x2, y2, cd2);                             { zistíme kód bodu P2 }
2.  repeat
        outcod (x1, y1, cd1);                         { zistíme kód bodu P1 }
3.      if and (cd1, cd2) ≠ 0 then done:= true        { 1. jednoduchý test - mimo okna}
        else
4.          begin
            if (cd1=0 and cd2=0) then                 { 2. jednoduchý test - vnútri okna}
                              begin accept:= true;    { žiadaj vykreslenie v kroku 10. }
                                    done:= true end
            else
5.              begin
                if cd1=0 then swap(P1, P2);           { zameníme body, aby 1. bol von}
6.              if cd1 ∈ (1, 5, 9) then               { orezávanie zhora }
                    begin
                    x1:= x1 + (x2-x1)*(ymax-y1)/(y2-y1);
                    y1:= ymax ;
                    end
7.              else if cd1 ∈ (2, 6, 10) then         { orež zdola }
                    begin
                    x1:= x1 + (x2-x1)*(ymin-y1)/(y2-y1);
                    y1:= ymin ;
                    end
8.              else if cd1 ∈ (4, 5, 6) then          { orež sprava }
                    begin
                    y1:= y1 + (y2-y1)*(xmax-x1)/(x2-x1);
                    x1:= xmax ;
                    end
9.              else if cd1 ∈ (8, 9, 10) then         { orež zľava }
                    begin
                    y1:= y1 + (y2-y1)*(xmin-x1)/(x2-x1);
                    x1:= xmin ;
                    end
                end             { od kroku 5 }
            end                 { od kroku 4 }
    until done
10. if accept then draw(P1, P2);                      { vykresli úsečku }
end.
```


NÁZOV:
PRIENIKY v E1, E2, E3 ...
PREDMET:
ROČNÍK:
ČÍSLO:
ČÍSLO ZLOŽKY
1. ÚTVAR
2. ÚTVAR
VÝSTUP
ZLOŽITOSŤ
E1
ÁNO/NIE
1 POROVNANIE
2 POROVNANIA
?, PAMÄŤ, PROGR.
E2
2 POROVNANIA
DNU, MIMO,
KONVEXNÝ
DNU, MIMO,
HVIEZDICOVITÝ
DNU, MIMO,
LOPTIVÉ
S JEDNODUCHOU HRANICOU
VŠEOBECNÝ
2 KONVEXNÉ
2 VŠEOBECNÉ
E3
VÝPOČTOVÁ GEOMETRIA - ZLOŽITOSŤ

Prune!!!


100 %


5 %


1 %

Rasterizácia, DDA

5.2 Rastrový rozklad úsečky

Uvedieme dva algoritmy generovania úsečiek do rastrovej formy. Pri rozklade úsečky budeme predpokladať, že oba koncové body majú celočíselné súradnice $(x1, y1)$, $(x2, y2)$.

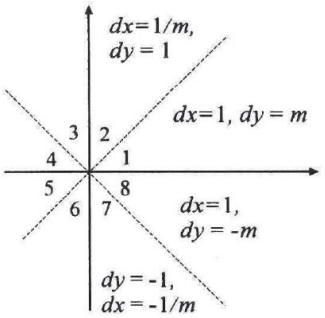
5.2.1 Jednoduchý prírastkový algoritmus

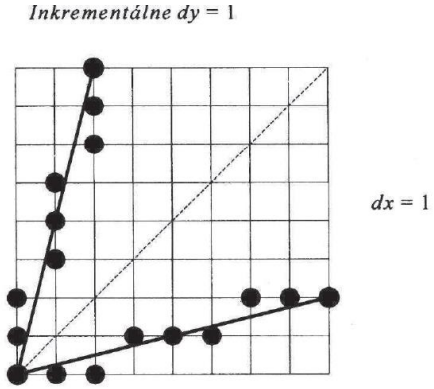
Analytické vyjadrenie priamky, ktorá nie je rovnobežná s osou y, vyjadrujeme v tvare:

$$y = m.x + b,$$

kde m je smernica priamky a b posun na osi y. Koncové body úsečky určujú priamku s parametrami m a b:

$$m = \frac{y2-y1}{x2-x1} \quad \text{a} \quad b = \frac{x2 \cdot y1 - x1 \cdot y2}{x2-x1}.$$





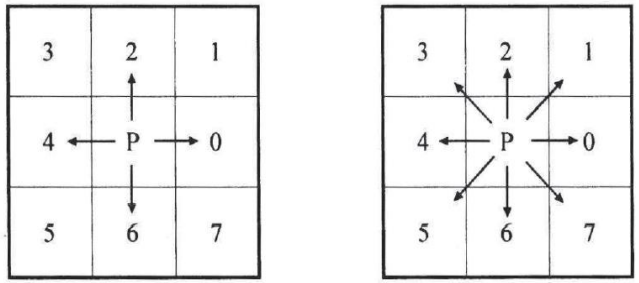
Obr. 5.1 Prírastky pre niektoré oktanty a dve úsečky v 1. a 2. oktante

Prírastkový algoritmus úsečky DDA (Digital Differential Analyzer)

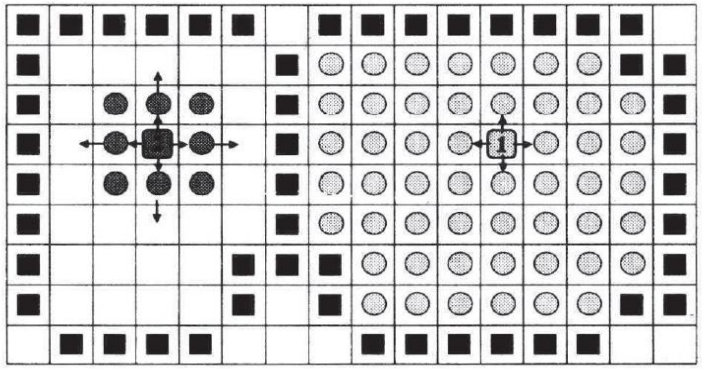
```
procedure DDA_line ( x1, y1,                    { začiatočný bod P }
                     x2, y2,                    { koncový bod Q }
                     colour :integer );         { farba vykreslenej úsečky }

var dx, dy, x, y, m: real;
begin
    if x1 > x2 then swap;                       { zameň body, aby prvý bod bol ľavý }
    if x1 < x2 then
        begin
        dx:= x2 - x1;
        dy:= y2 - y1;
        m:= dy/dx;
        y:=y1;
        for x:= x1 to x2 do
            begin
            write_pixel (x, round(y), colour);      { zobraz bod (x, y) }
            y:= y + m                               { inkrementuj súradnicu y }
            end
        end
    else if y1 = y2 then write_pixel (x1, y1, colour)   { zapíš len jeden bod }
    else error ;
end.
```

Rasterizácia oblastí



Obr. 5.7 Susednosť (4 a 8) v štvorcovom rastri pre bod P

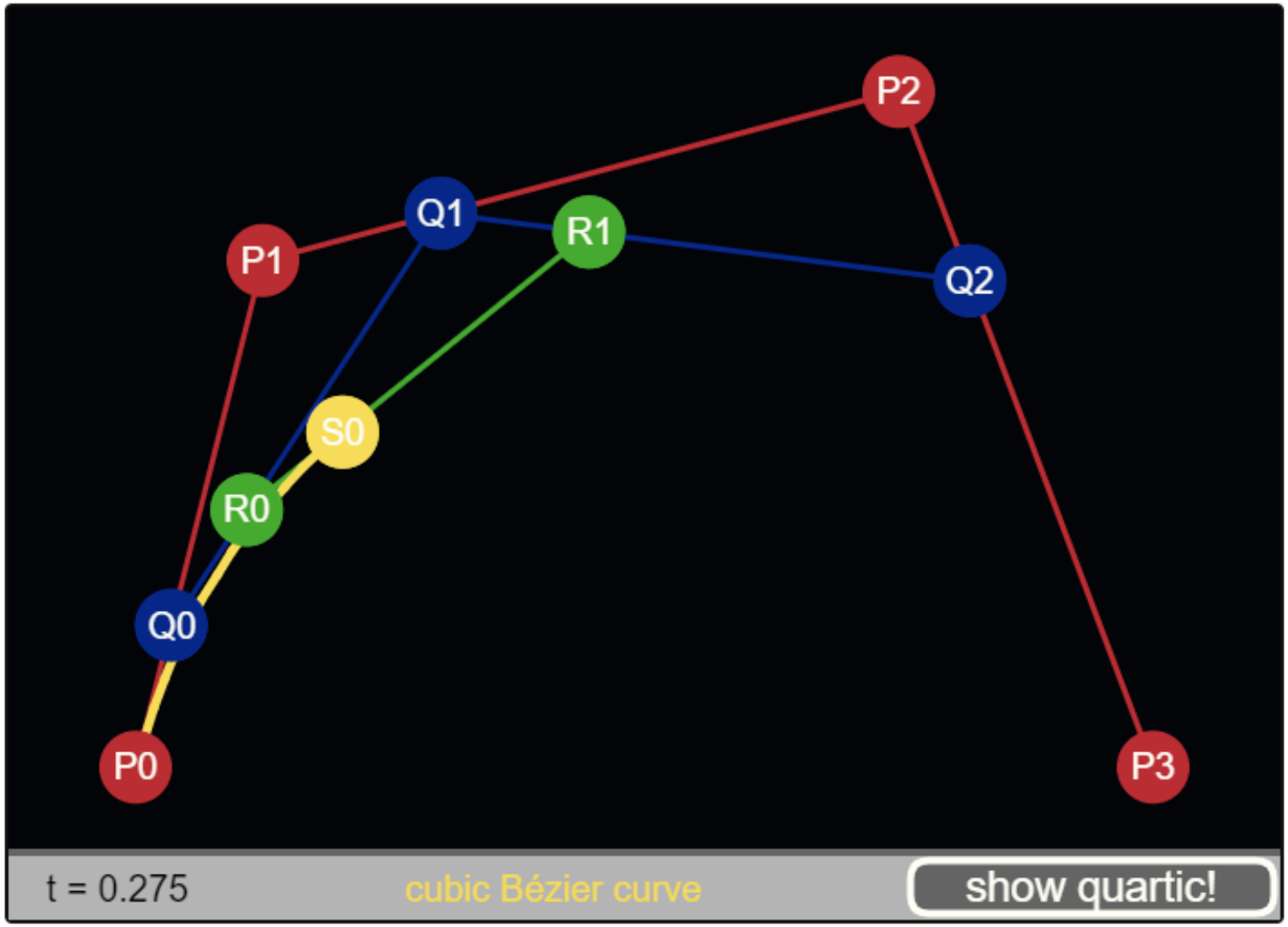


Obr. 5.8 Vypĺňanie oblastí farbou zadaného vnútorného bodu

Algoritmus vlnového vypĺňania *Flood_fill*

```
procedure Flood_fill_4 ( x, y,              { začiatočný vnútorný bod vypĺňania oblasti }
                         old_colour,        { stará farba oblasti }
                         new_colour : integer); { nová farba oblasti }
begin
  if read_pixel (x,y) = old_colour then
    begin
        write_pixel (x, y, new_colour);
        Flood_fill_4 (x, y-1, old_colour, new_colour);
        Flood_fill_4 (x, y+1, old_colour, new_colour);
        Flood_fill_4 (x-1, y, old_colour, new_colour);
        Flood_fill_4 (x+1, y, old_colour, new_colour);
    end
end.
```

http://www.malinc.se/m/DeCasteljauAndBezier.php



Geometric construction showing a linear, quadratic, cubic, and quartic Bézier curve.
Move the red points!

Koherencia geom. i radiom.

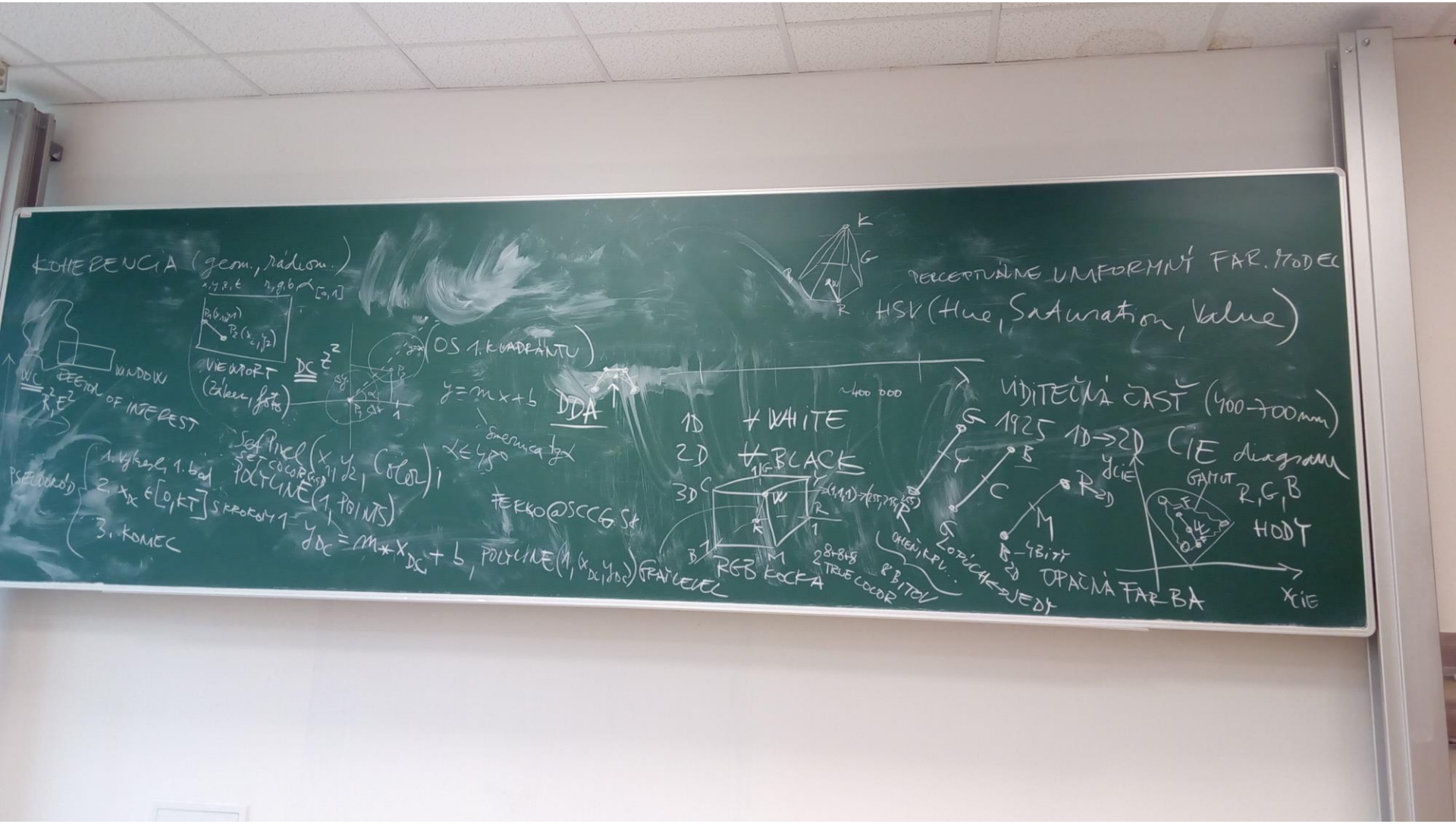
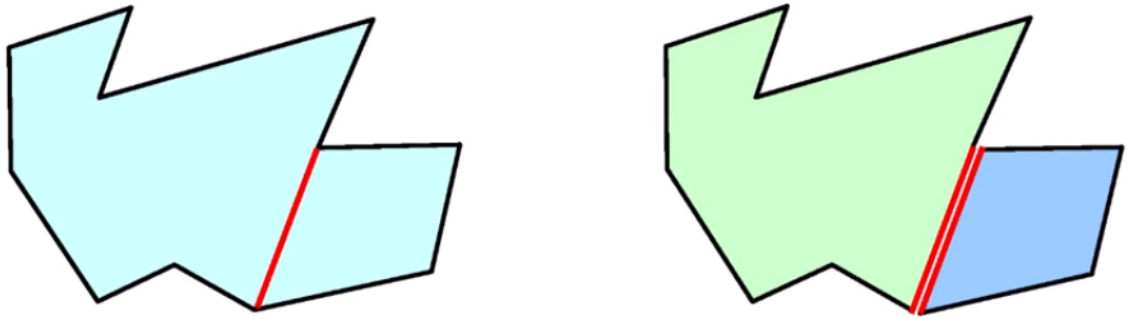


Photo Nikola GIBASOVÁ, 2016

Triangulácia

Vnútorná uhlopriečka umožňuje rozdeliť mnohouholník na dva mnohouholníky s menším počtom vrcholov:



Veta

1. Každý mnohouholník má jednoduchú triangluláciu.
2. Každá jednoduchá triangulácia n-uholníka má práve $n-2$ trojuholníkov.

Ukážka využitia jednoduchej triangulácie:

Veta

Súčet veľkostí vnútorných uhlov každého n-uholníka je $(n-2)\pi$.

Koherencia, Interpolácia…
(x,y,z,t,r,g,b,alfa)

Andrej FERKO

Comenius University Bratislava

PG1, 26. 10. 2020, FMFI UK