Diskrétne geometrické štruktúry

9.

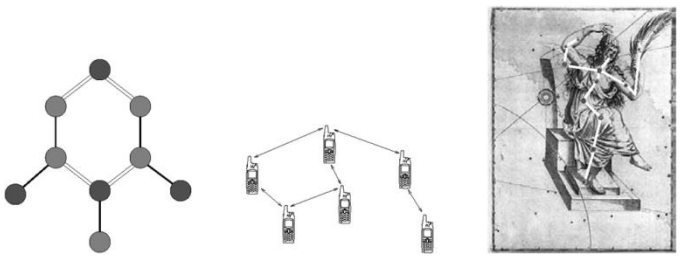
Martin Florek

florek@sccg.sk

www.sccg.sk/~florek

Proximity Graphs

- neighborhood graphs
- prináša susednosť, štruktúru, tvar, topológiu pre neštruktúrované dáta
- geometrické grafy – topológia je daná geometriou
- použitie vo výpočtovej geometrii, počítačové videnie, tvorba sietí, geografia, biológia,...

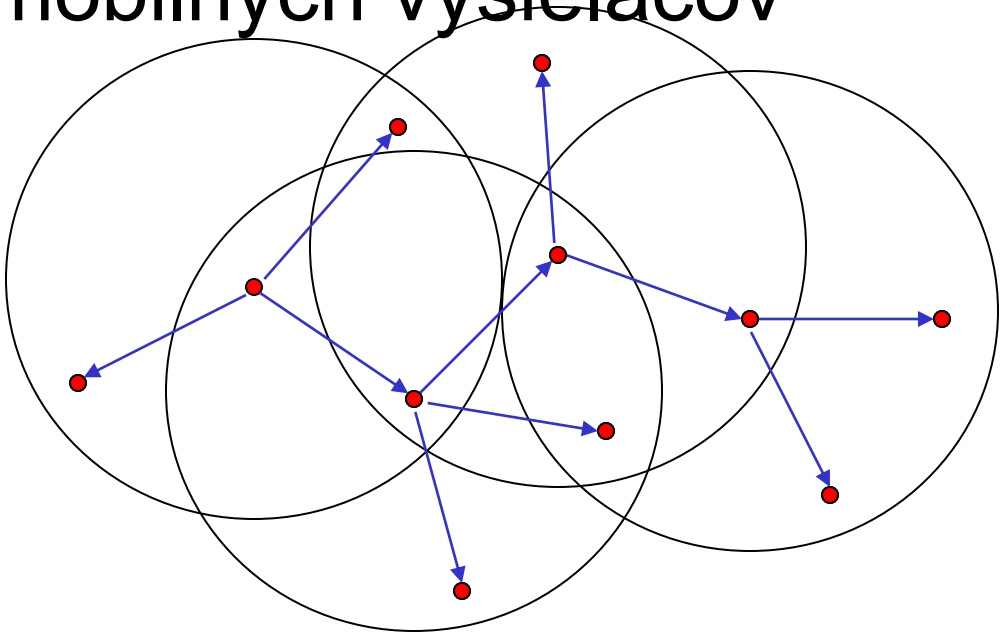


Definície

- pracujeme v nejakom metrickom priestore
- najčastejšie s Euklidovskou metrikou
- najčastejší vstup – množina bodov $V \in R^d$
- hrany spájajú vrcholy, ktoré sú si navzájom blízke (“proximity”)
- tým je definovaná topológia

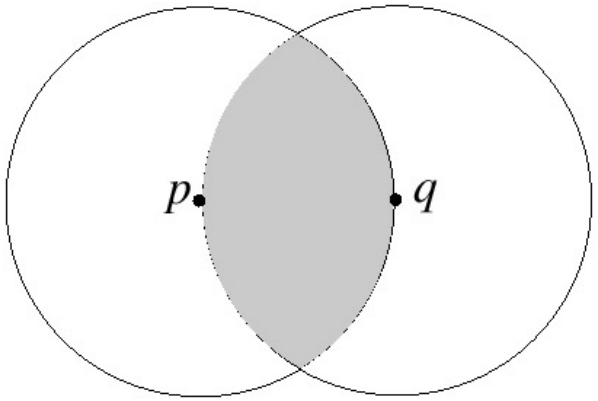
Jednoduché grafy

- Unit Disc Graph, UDG(V)
- graf sfér s jednotkovou dĺžkou
- množina hrán je $E := \{pq \mid d(p,q) \leq 1\};$
- vhodné pre siete mobilných vysielačov



Relatívny graf susednosti

- relative neighborhood graph *RNG*(*V*)
- $L(p, q) = B(p, d) \cap B(q, d)$, kde $d = ||p - q||$
- množina hrán $E := \{pq \parallel L(p, q) \cap V = \varnothing\}$.



$$pq \in E \Leftrightarrow \nexists v \in V : d(p, v) < d(p, q) \wedge d(q, v) < d(p, q).$$

$$pq \in E \Leftrightarrow \forall v \in V : d(p, q) \leq \max\{d(p, v), d(q, v)\}.$$

Gabrielov graf

- definované pomocou sféry

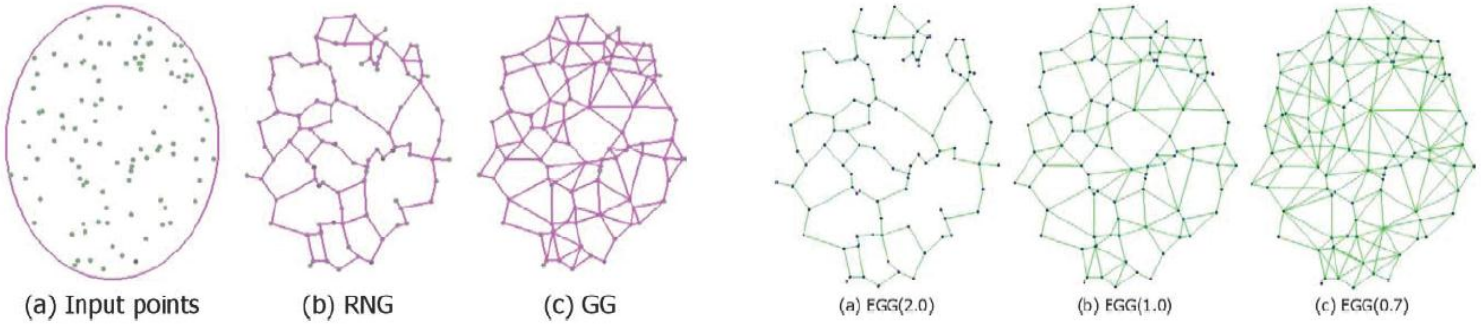
$G(p,q) := B\left(\frac{\mathbf{p}+\mathbf{q}}{2}, \frac{d}{2}\right)$ kde $d = ||p - q||$

- GG(V) má množinu hrán

$$E := \{pq \parallel G(p,q) \cap V = \varnothing\}.$$

$$pq \in E \Leftrightarrow \forall v \in V : d(p,q) \leq \sqrt{d^2(p,v) + d^2(q,v)}.$$

- rozšírenie v podobe elíps (EGG)

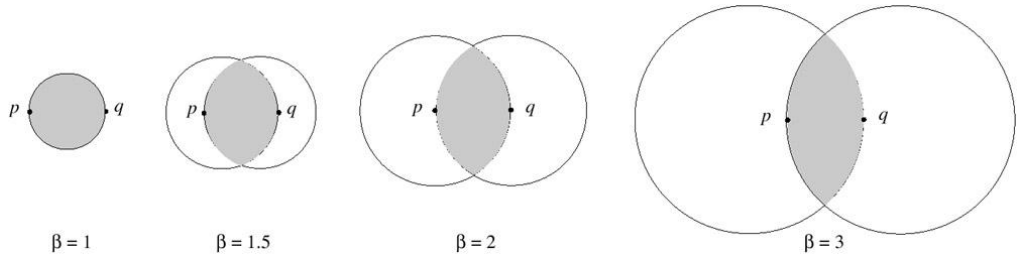


β-kostry

- grafy dané parametrom β, $1 \leq \beta < \infty$
- označenie $BG_\beta(V)$
- okolie bodov p a q je dané

$$U_\beta(p,q) := B\left((1-\frac{\beta}{2})\mathbf{p}+\frac{\beta}{2}\mathbf{q}, \frac{\beta}{2}d\right) \cap B\left((1-\frac{\beta}{2})\mathbf{q}+\frac{\beta}{2}\mathbf{p}, \frac{\beta}{2}d\right),$$

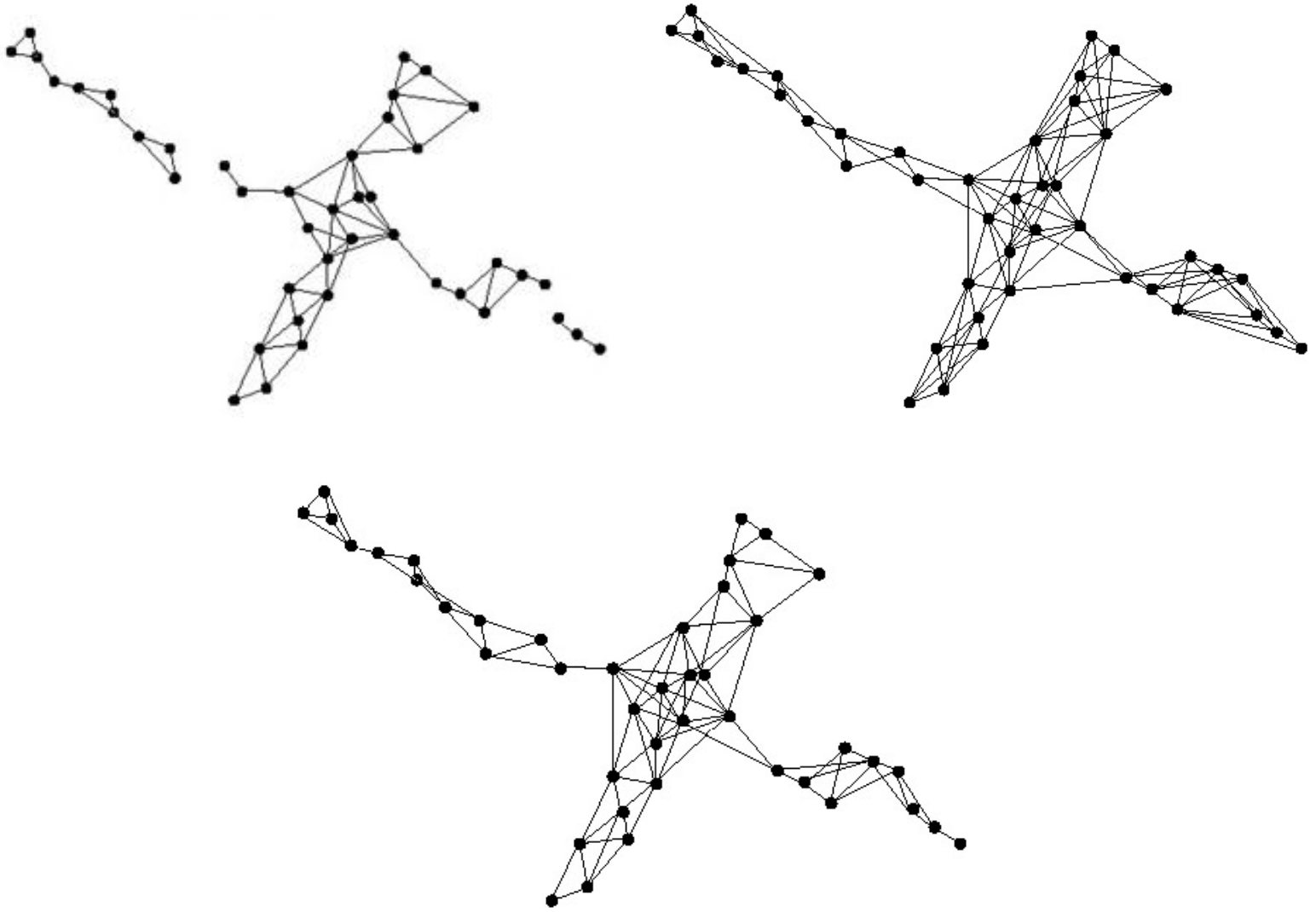
- množina hrán $E := \left\{pq \parallel U_\beta(p,q) \cap V = \varnothing\right\}$.
- $RNG(V) = BG_2(V)$, $GG(V) = BG_1(V)$
- ak $\beta_1 > \beta_2$, potom $BG_{\beta_1}(V) \subset BG_{\beta_2}(V)$



Sféry vplyvu

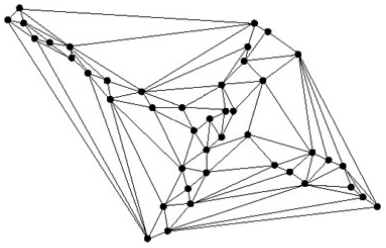
- sphere-of-influence graph, SIG(V)
- pre bod $p \in V$, r_p je vzdialenosť k najbližšiemu susedovi
- množina hrán $E := \{pq \parallel d(p,q) \leq r_p + r_q\}$.
- nemusí byť spojitý →
  - rozšírenie: *r*-SIG, r_p je vzdialenosť k *r*-tému najbližšiemu susedovi

SIG, 3-SIG, 3-SIG bez dlhých hrán

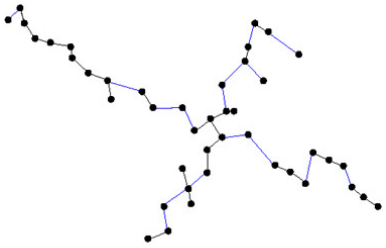


Geometrické grafy

- *Minimum Spanning Tree* (MST) spája body do stromu tak, aby súčet dľžok hrán bol minimálny (minimálna kostra)
- *Delaunay graph* (DG) spája dva body *p* a *q* vtedy, ak existuje kružnica *k*, ktorá prechádza bodmi *p*, *q* a žiadne iné body nie sú vnútri kružnice *k*



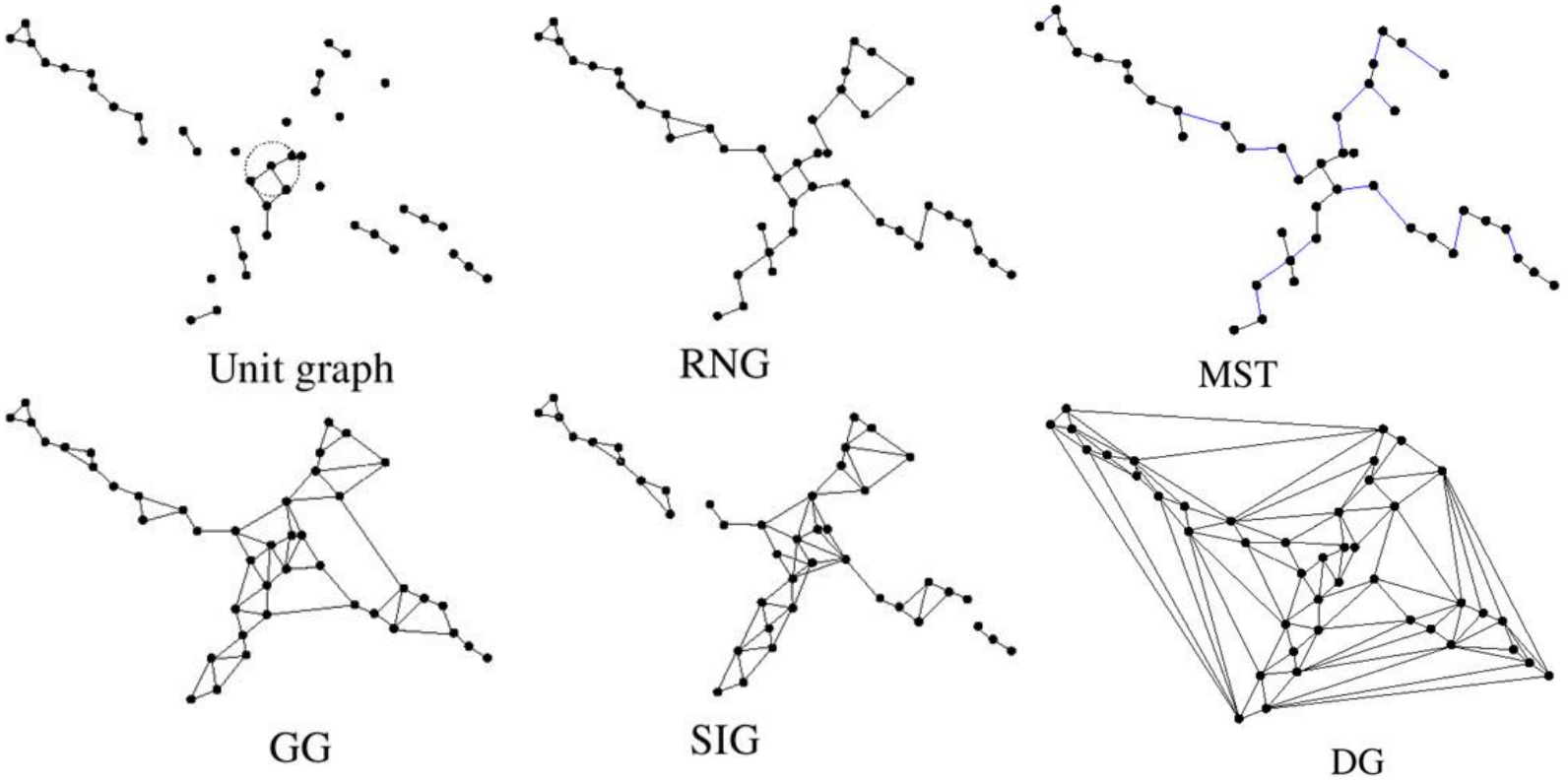
DG



MST

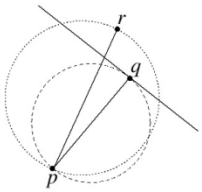
Porovnanie

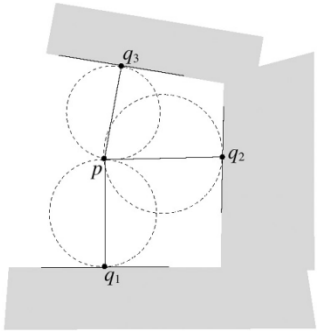
- $MST(V) \subseteq RNG(V) \subseteq GG(V) \subseteq DG(V).$
- $|MST(V)| \leq |RNG(V)| \leq |GG(V)| \leq |DG(V)|$



Konštrukcia GG

- hrubá sila pre vytvorenie GG → $O(n^3)$
  - všetky možné hrany $O(N^2)$ porovnať so všetkými vrcholmi $O(N)$
- zlepšenie-heuristika-odstránenie vrcholov napravo od skúmaného vrcholu → $O(n^2)$






```
CreateGG(V)
{
        for (all p from V)
        {
                N_p = V \ p;
                for (all q from N_p)
                {
                        for (all r in V)
                        {
                                if (d(p,r)^2+d(q,r)^2 < d(p,q)^2)
                                        remove q from N_p;
                                else if (r is to the right of H_qp)
                                        remove r from N_p;
                        }
                }
                Connect p with points from N_p;
        }
}
```

Konštrukcia r-SIG

- v priemere lineárna časová zložitosť
- pamäťová zložitosť je tiež lineárna

```
CreaterSIG(r, V)
{
        initialize grid with n cells;
        for (all p in V)
            assign p to its grid cell;
        for (all p in V)
            find r-th nearest neighbor to p by searching the grid cells in spiral order around p with increasing distance;
        for (all p in V)
        {
            for (all cells around p that intersect the sphere of influence around p (in spiral order))
                assign p to cell;
        }
        for (all cells in the grid)
        {
            for (all pairs p_i, p_j of points assigned to the current cell)
            {
                if (spheres of influence of p_i and p_j intersect)
                    create edge p_i p_j;
            }
        }
}
```

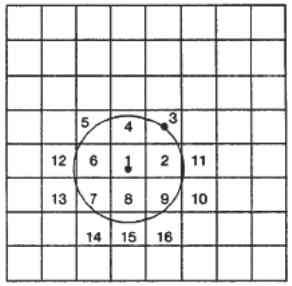


Fig. 1 Spiral nearest neighbor search using cells.

Voronoiov diagram

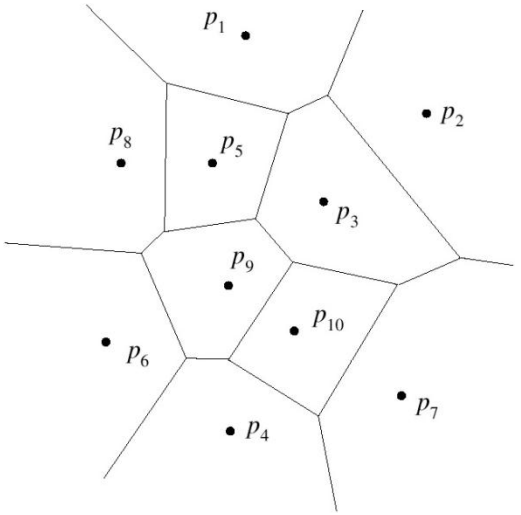
- geometrický graf
- pre danú množinu bodov S={p_1, p_2, …, p_n}, každá stena VD je priradená jednému bodu p_i a pre každý bod oblasti je tento bod bližšie k p_i ako ku inému bodu z S

$$\mathrm{Bis}(p_i, p_j) = \{x \mid d(p_i, x) = d(p_j, x)\}$$

$$\mathrm{H}(p_i, p_j) = \{x \mid d(p_i, x) < d(p_j, x)\}$$

$$\mathrm{VoR}(p_i, S) = \bigcap_{p_j \in S, p_j \neq p_i} H(p_i, p_j)\,.$$

$$\mathrm{VD}(S) := \bigcup_{p_i, p_j \in S, p_i \neq p_j} \overline{\mathrm{VoR}(p_i, S)} \cap \overline{\mathrm{VoR}(p_j, S)}.$$



Vlastnosti VD

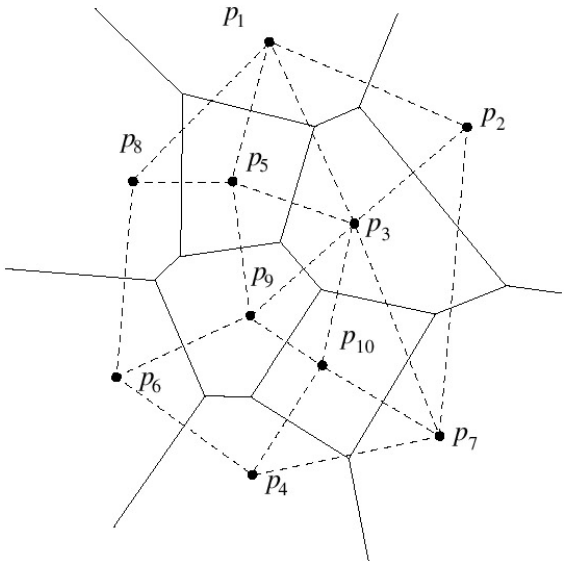
- VoR(p_i, S) je prienikom max n-1 polrovín, je to otvorená a konvexná množina
- VoR(p_i, S) je neohraničná práve vtedy keď p_i patrí do konvexného obalu S
- VD má O(n) hrán a vrcholov
- priemerný počet hrán na hranici VoR(p_i, S) je menší ako 6

Delaunayova Triangulácia

- pre množinu bodov S a VD(S), DT(S) je duálny graf k VD
- body p_i, p_j, p_k tvoria trojuholník v DT $\Leftrightarrow$ kružnica prechádzajúca tromi bodmi neobsahuje vnútri ani na hranici iný bod z S
- DT maximalizuje

  minimálny vnútorný

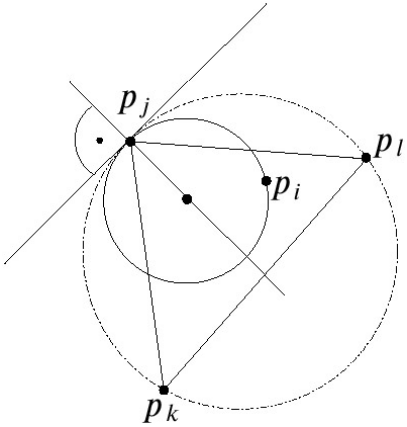
  uhol



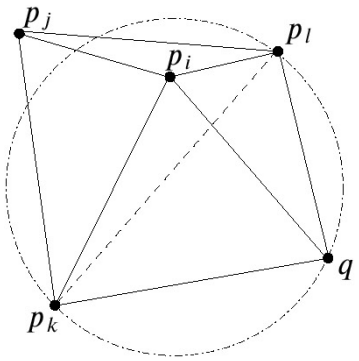
Konštrukcia DT

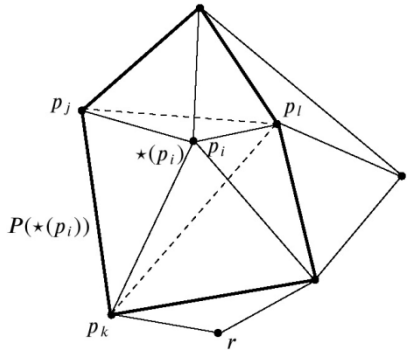
- algoritmus vkladania nového bodu *p* do DT
- dva prípady – bod vložíme do konvexného obalu S alebo mimo
- po vložení sa kontroluje, či sa neporušila vlastnosť DT
- pri porušení sa vykonáva otočenie hrany v „zlom“ štvoruholníku

1. prípad



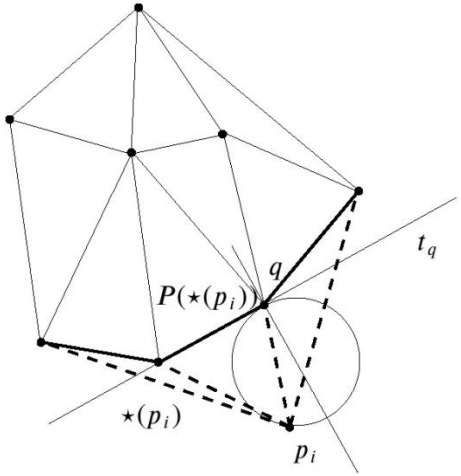
- p_i leží v $T = \Delta(p_j, p_k, p_l)$
- hrany p_ip_j, p_ip_k, p_ip_l patria novej DT
- konflikt môže nastať v Δ ktoré sú susedné s T
- vtedy sa otočí hrana T
- konflikt aj v ďalších Δ
- ak hrany P(*(p_i)) sú z DT, koniec



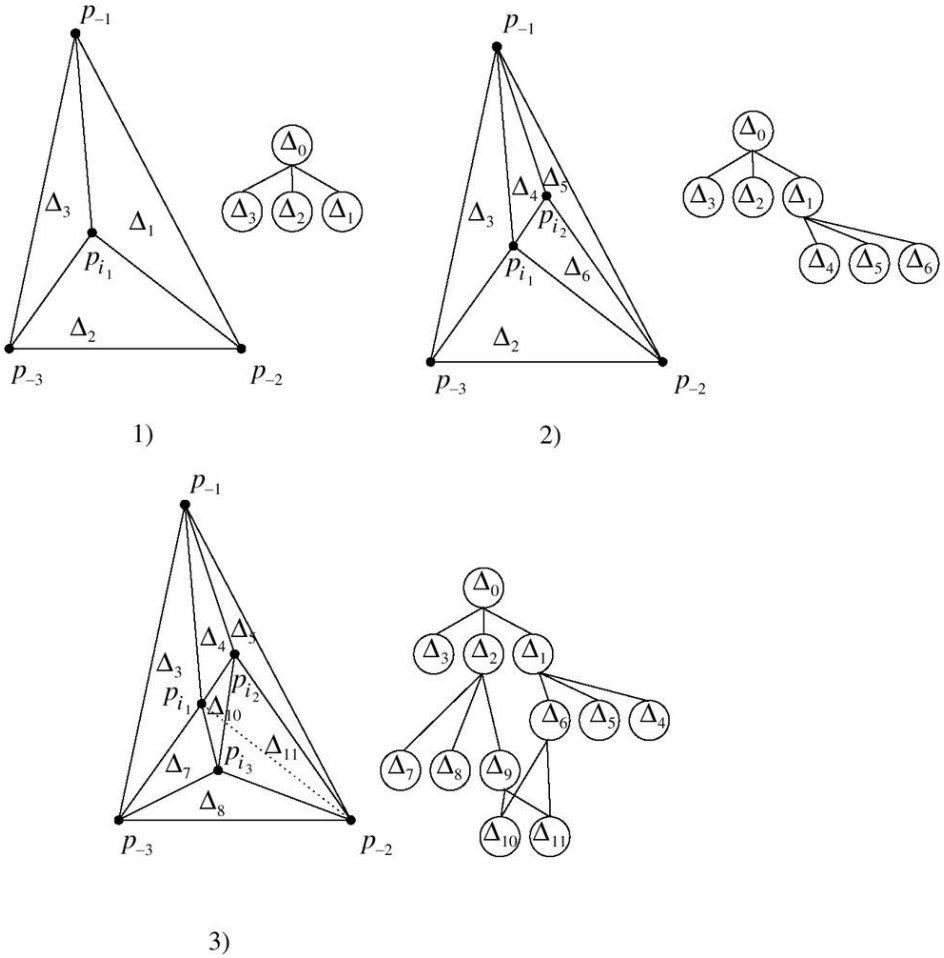


2. prípad

- ak p_i leží z vonku konvexného obalu pôvodnej DT
- pre všetky body q z S, ktoré sú „viditeľné“ z p_i, p_iq bude patriť novej DT
- niektoré hrany na hranici bude treba otočiť



nájdenie trojuholníka



Algoritmus

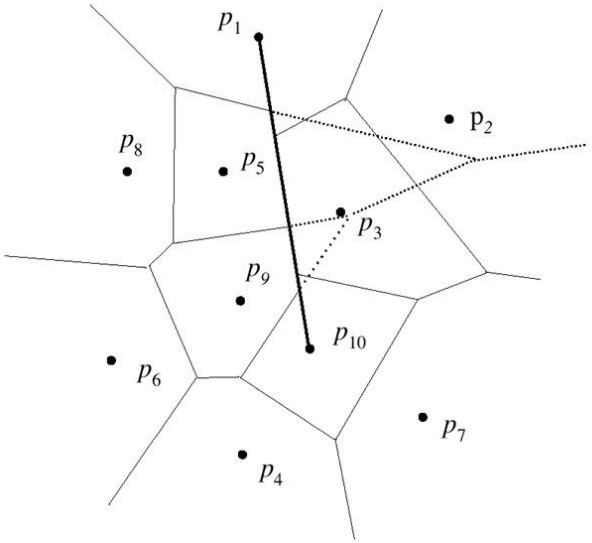
```
Delaunay(S) (S is set of sites)
{
        T = new array;
        while (S.size() > 0)
        {
                p = S.First;
                S.DeleteFirst;
                T.InsertSite(p);
        }
}
```

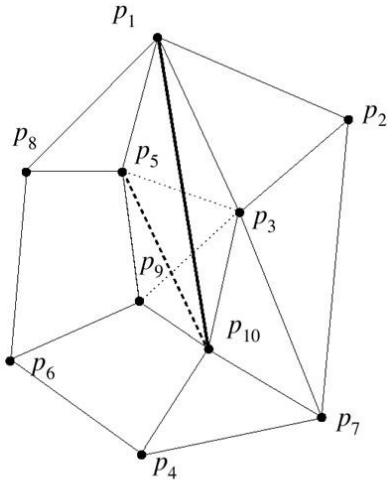
```
InsertSite(T, p) (T represents the current Delaunay triangulation, p is a new site)
{
        t = T.FindTriangle(p);
        if (t == NULL)
                Star(p) = T.OuterTriangle(p);
        else
                Star(p) := T.Edges(t, p);
        T.Insert(Star(p), t);
        StarPoly = t.Edges();
        while (StarPoly.size() > 0)
        {
                e = StarPoly.First();
                StarPoly.DeleteFirst();
                q = p.Opposite(e);
                if (q ≠ NULL)
                {
                        (r, s) = e.EndPoints();
                        if (InCircleTest(p, r, s, q))
                        {
                                T.Remove(e);
                                T.Add((p, q));
                                StarPoly.Add((r,q));
                                StarPoly.Add((s,q));
                        }
                }
        }
}
```

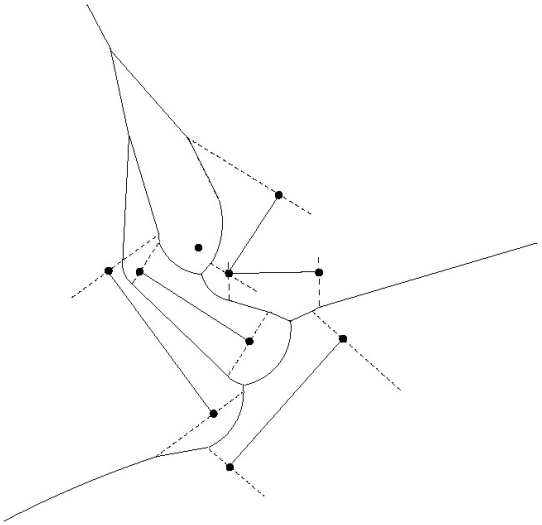
Zovšeobecnenia VD, DT

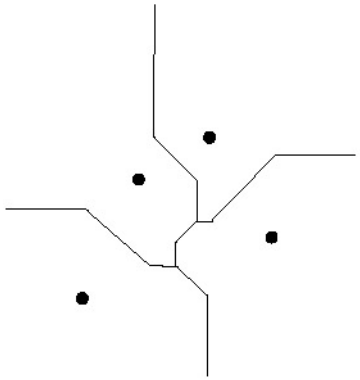
- rozšírenie do vyšších dimenzií, 3D – tetrahedralizácia
- použitie obmedzení – sú dané časti (hrany, …), ktoré triangulácia obsahuje (Constrained DT), obmedzujúca metrika pre VD
- použitie iných metrík
- váhovanie $w(p_i)d(p_i, q) = w(p_j)d(p_j, q)$
- rozšírenie bodov na zložitejšie objekty

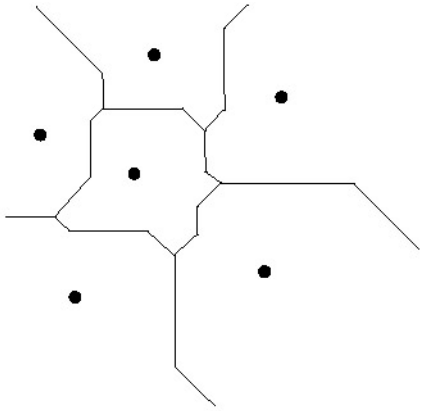
Zovšeobecnenia

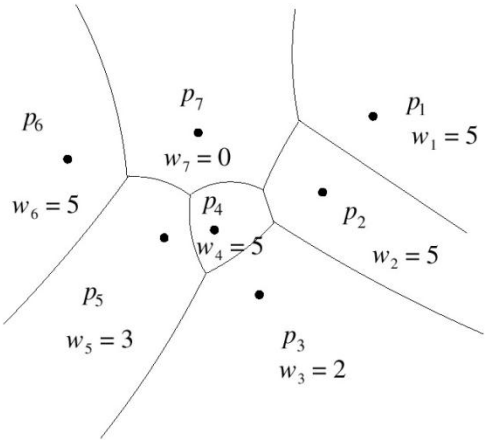












Mračná bodov

- jednoduchá množina bodov
- výstup z mnohých meracích zariadení
- rozvíjajúca sa oblasť v modelovaní a vizualizácii
- potreba definovania povrchu



Generovanie povrchu

- priamo nie sú dobre použiteľné
- nutnosť vygenerovať povrch – polygonálny model
  - vygenerovanie implicitného povrchu
  - marching cubes/triangles
  - ball-pivoting
- priame použitie pri porovnávaní súčiastok

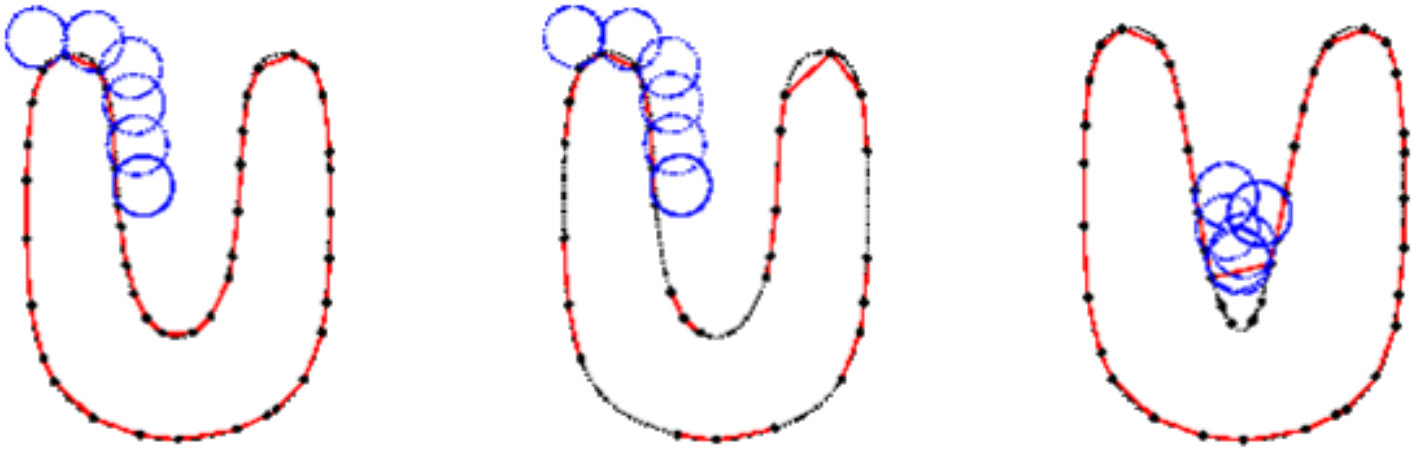
Marching cubes/triangles

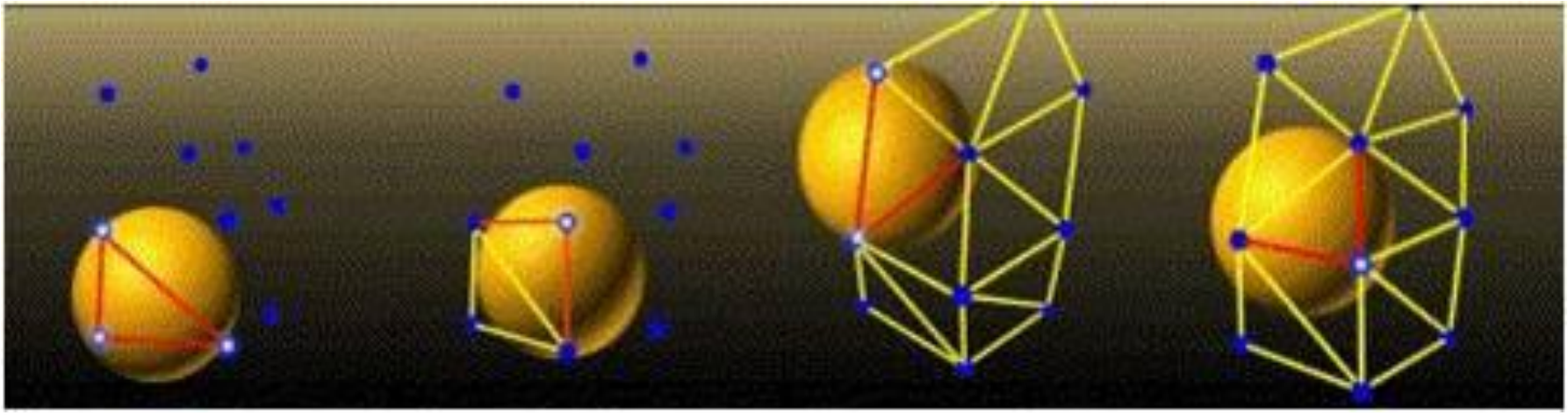
- ako pri iso povrchoch
- marching triangles rešia nejednoznačnosť prípadov z marching cubes algoritmu
- adaptívne marching triangles
  - lokálne spĺňa podmienku DT
  - zo štartovacieho trojuholníka pridávame ďalšie

Ball-pivoting

- „gúľanie lopty po povrchu“
- najskôr sa vyrátajú normály bodov
- voxelová štruktúra
- zvolíme si priemer sféry
- štartovací trojuholník
- gúľame sa po hranách
- viac komponent

Ball-pivoting 2





Problémy

- chýbajúce body
- merania z viacerých strán a spájanie do jednej množiny
- vela bodov (nedostatok pamäte) – riešenie po pásoch

Distance fields

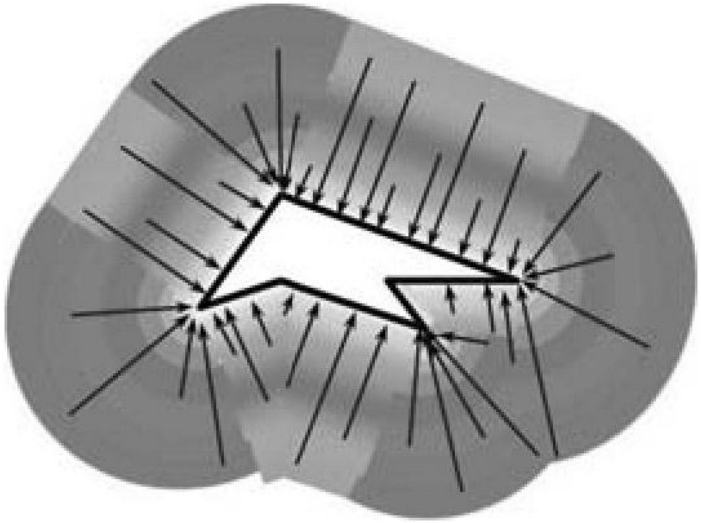
- Funkcia určujúca vzdialenosť k danému objektu
- Pre množinu Σ, unsigned distance function je $\operatorname{dist}_\Sigma(\mathbf{p}) = \inf_{\mathbf{x}\in\Sigma} \|\mathbf{x}-\mathbf{p}\|.$
- Rozšírenie – vzdialenostné vektory
- Vzdialenosť ku objektu S so znamienkom

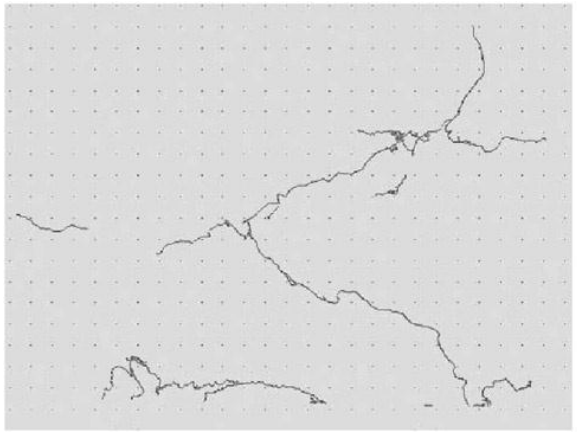
$$\mathrm{d}_S(\mathbf{p}) = \operatorname{sgn}(\mathbf{p}) \inf_{\mathbf{x}\in\partial S} \|\mathbf{x}-\mathbf{p}\|,$$

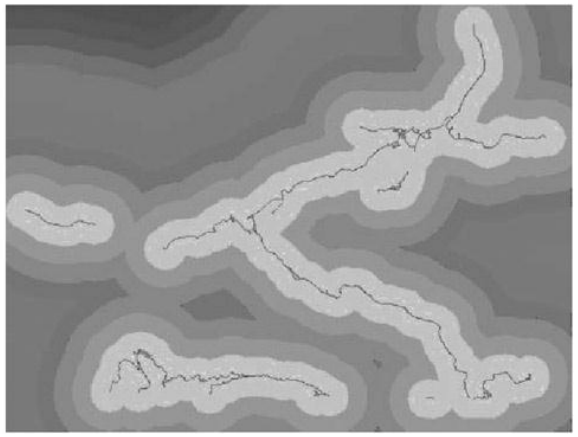
where

$$\operatorname{sgn}(\mathbf{p}) = \begin{cases} -1 & \text{if } \mathbf{p} \in S \\ 1 & \text{otherwise.} \end{cases}$$

Príklady

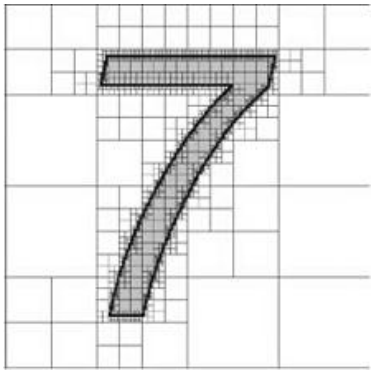


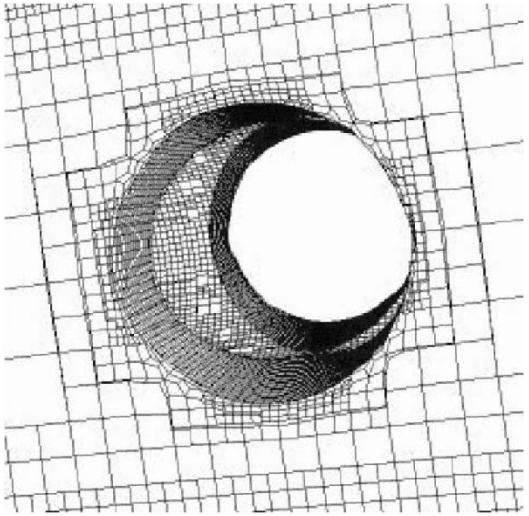




Reprezentácie

- Regulárna mriežka
- Hierarchická mriežka
- Adaptívne pole – quadtree, octree, bsp…





Vlastnosti

- Isopovrch pre isohodnotu τ $\{\mathbf{p}|\mathrm{d}(\mathbf{p}) = \tau\}$
- Gradient $\|\nabla \mathrm{d}\| = 1$ pre takmer všetky body, smer gradientu kolmý na isopovrch
- Hesián $$H = \begin{pmatrix} \mathrm{d}_{xx} & \mathrm{d}_{xy} & \mathrm{d}_{xz} \\ \mathrm{d}_{yx} & \mathrm{d}_{yy} & \mathrm{d}_{yz} \\ \mathrm{d}_{zx} & \mathrm{d}_{zy} & \mathrm{d}_{zz} \end{pmatrix}$$
- Stredná krivosť $\kappa_M = \frac{1}{2}\left(\mathrm{d}_{xx} + \mathrm{d}_{yy} + \mathrm{d}_{zz}\right)$
- Gaussova krivosť $$\kappa_G = \begin{vmatrix} \mathrm{d}_{xx} & \mathrm{d}_{xy} \\ \mathrm{d}_{yx} & \mathrm{d}_{yy} \end{vmatrix} + \begin{vmatrix} \mathrm{d}_{xx} & \mathrm{d}_{xz} \\ \mathrm{d}_{zx} & \mathrm{d}_{zz} \end{vmatrix} + \begin{vmatrix} \mathrm{d}_{yy} & \mathrm{d}_{yz} \\ \mathrm{d}_{zy} & \mathrm{d}_{zz} \end{vmatrix}$$

Vlastnosti 2

- Vzdialenostná funkcia je spojitá
- Problémové body – body rovnako vzdialené od aspoň dvoch bodov na povrchu – cut locus
- Pre C^k povrch, funkcia je C^k v nejakom okolí bodu na povrchu
- Funkcia je diferencovateľná až na body z cut locus

Diskretizácia

- Vzorkovanie vzdialenostnej funkcie
- Rôzne formy vzoriek – mriežky, grid
- Je potrebné zachytiť detaily a určiť problémové miesta – cut locus
- Potreba aproximácie gradientu

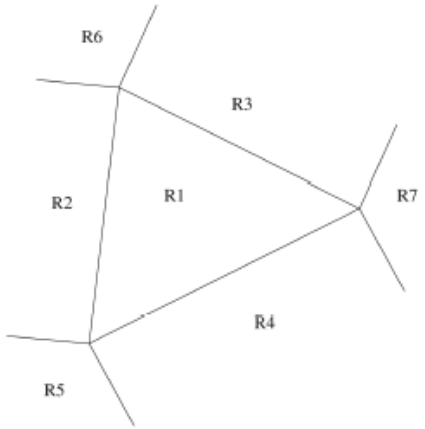
$$
\begin{aligned}
g^x_{i,j,k} &= d_{i+1,j,k} - d_{i-1,j,k} \\
g^y_{i,j,k} &= d_{i,j+1,k} - d_{i,j-1,k} \\
g^z_{i,j,k} &= d_{i,j,k+1} - d_{i,j,k-1} \\
n_{i,j,k} &= \frac{g_{i,j,k}}{\|g_{i,j,k}\|} .
\end{aligned}
$$

Výpočet DF (Voxelization)

- Brute-force – pre každý mrežový bod (voxel) sa určí najmenšia vzdialenosť k objektu
- Časovo náročné
- Zlepšenia:
  - Na základe priestorového usporiadania častí objektu sa kontrolujú iba najbližšie objekty
  - Výpočet iba v niektorých častiach a následná propagácia do ďalších častí pomocou transformácií

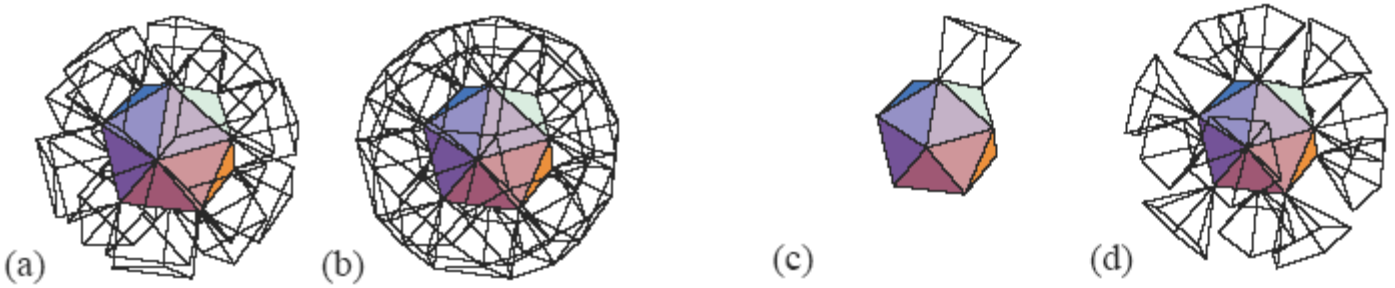
Voxelizácia z trojuholníkov

- Trojuholníky tvoria uzatvorené, orientovateľné 2-manifoldy
- Vzdialenosť bodu a trojuholníka – 7 prípadov pri projekcii bodu do roviny trojuholníka
- Urýchlenie prehľadávania – ohraničujúce objemy, octrees



Lokálne metódy

- Pre výpočet vzdialenosti do určitej veľkosti
- Rozšírenie ohraničujúceho objemu trojuholníka o danú hodnotu
- Identifikácia oblastí vzdialených od povrchu o max. Danú hodnotu



Distance Transforms

- Výpočet vzdialeností na základe informácií blízko povrchu
- Spôsob prechodu mriežkou:
  - Zametanie – po jednotlivých rezoch a riadkoch
  - Wavefront – od povrchu ku stále väčším vzdialenostiam

Výpočet pre voxel

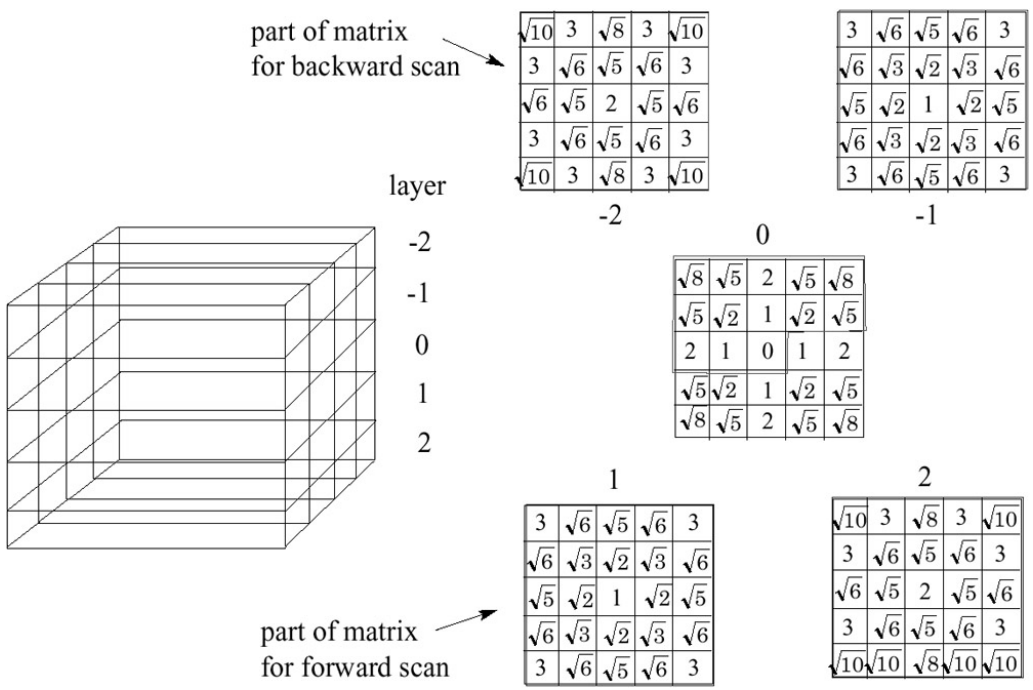
- Chamfer:
  - Nová vzdialenosť pre voxel je vypočítaná zo vzdialeností okolitých voxelov
- Vector:
  - Nový vektor vzdialenosti je vypočítaný z vektorov okolitých voxelov

Chamfer

Transform	a	b	c	d	e	f
City Block (Manhattan)	1					
Chessboard	1	1				
Quasi-Euclidean $3 \times 3 \times 3$	1	$\sqrt{2}$				
Complete Euclidean $3 \times 3 \times 3$	1	$\sqrt{2}$	$\sqrt{3}$			
$< a, b, c >_{opt}$ $3 \times 3 \times 3$[102]	0.92644	1.34065	1.65849			
Quasi-Euclidean $5 \times 5 \times 5$	1	$\sqrt{2}$	$\sqrt{3}$	$\sqrt{5}$	$\sqrt{6}$	3

- Sweeping:

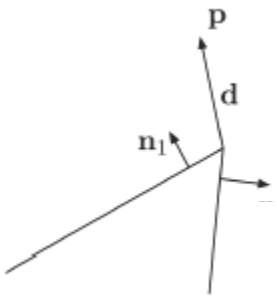
```
/* Forward Pass */
FOR(z = 0; z < fz; z++)
  FOR(y = 0; y < fy; y++)
    FOR(x = 0; x < fx; x++)
      F[x,y,z] =
 inf∀i,j,k∈fp(F[x+i,y+j,z+k]+m[i,j,k])

/* Backward Pass */
FOR(z = fz-1; z ≥ 0; z--)
  FOR(y = fy-1; y ≥ 0; y--)
    FOR(x = fx-1; x ≥ 0; x--)
      F[x,y,z] =
 inf∀i,j,k∈bp(F[x+i,y+j,z+k]+m[i,j,k])
```

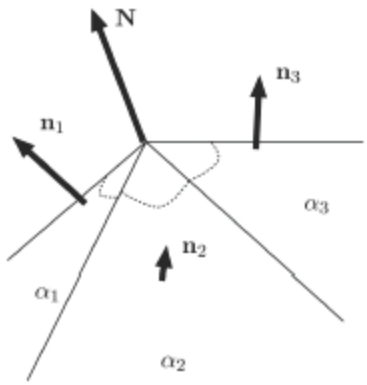


- Wavefront:
  - Prioritná fronta pre voxely s najmenšou vzdialenosťou

Výpočet znamienka

- Pre C^1 povrchy, stačí nájsť skalárny súčin vektora vzdialenosti a normály
- Množina trojuholníkov nie je C1
- Iný výpočet:
  - Pseudonormály
  - Vysielanie lúča
  - Konverzia rezov



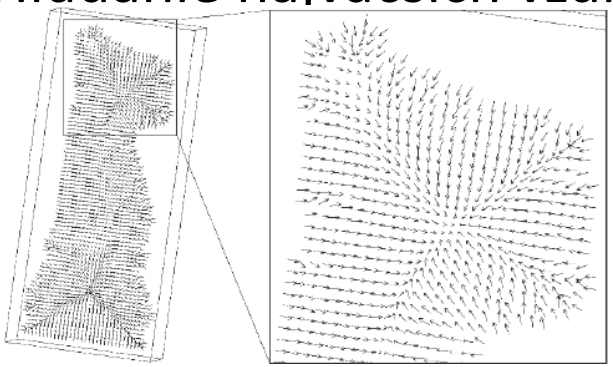


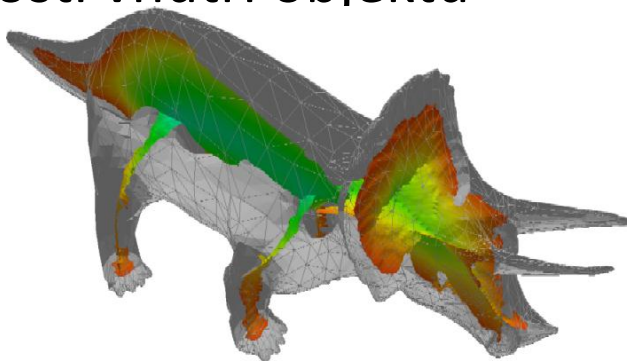
Aplikácie

- Reprezentácia objektov, výhoda oproti binárnym voxelom, napr. approximačné určenie najbližšieho bodu $\mathbf{p}_f = \mathbf{p} - \nabla d_S(\mathbf{p}) d_S(\mathbf{p})$
- Modelovacie schopnosti
- Animácie, morfing
- Image processing
- Fyzikálne simulácie

Kostra, stredná os

- Vytvorenie jednoduchého objektu ktorý aproximuje daný objekt
- Využitie v kinematike,
- Analýza DF
  - Hľadanie nespojitostí v derivácii DF
  - Porovnávanie smerových vektorov k najbližším bodom na povrchu
  - Hľadanie najväčších vzdialeností vnútri objektu



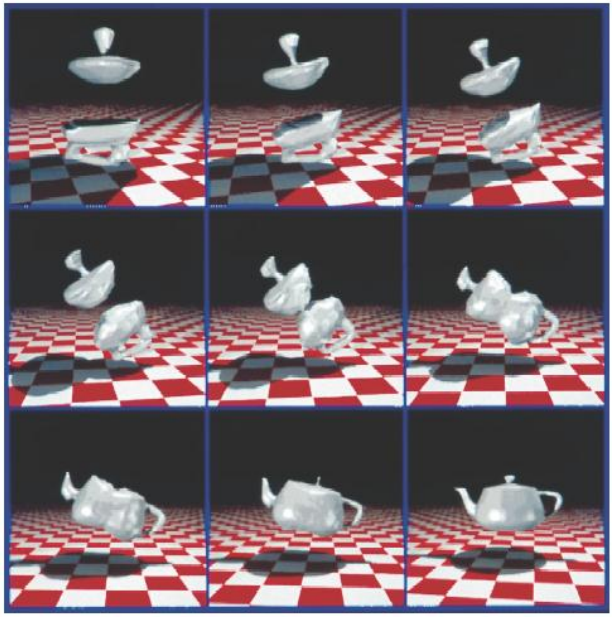


Morfing

- Prechod medzi dvoma objektami v čase

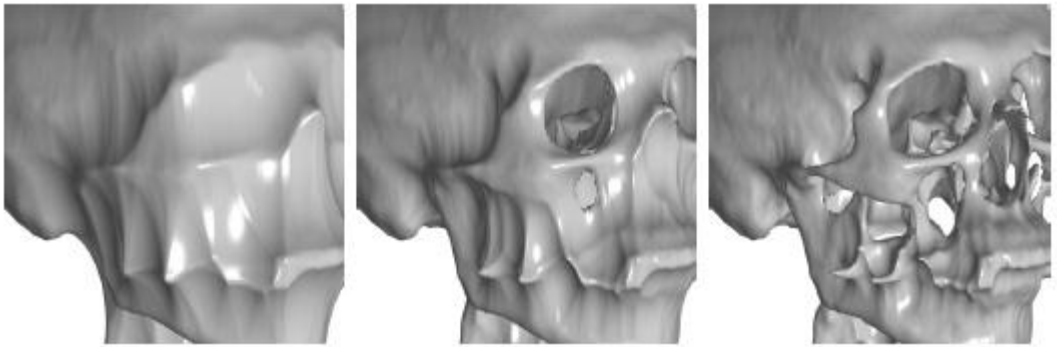
$$M(t, S) = J(t, S, T) = t D_S + (1 - t) D_T.$$

- Nezáleží na rode objektu
- Potrebné zjednotiť aproximácie oboch funkcií
- Priloženie objektov



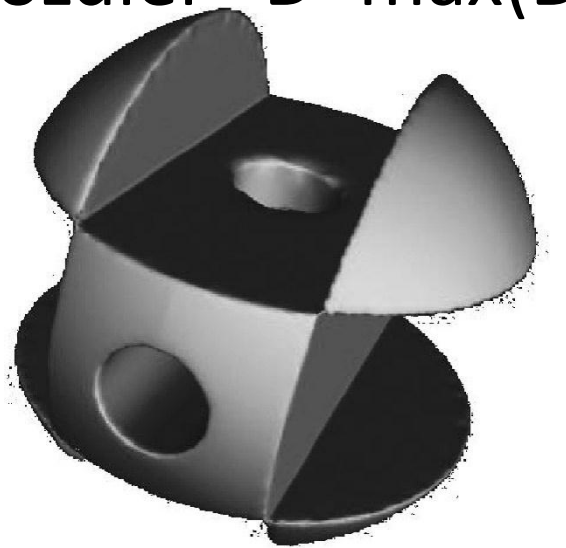
Morfológia

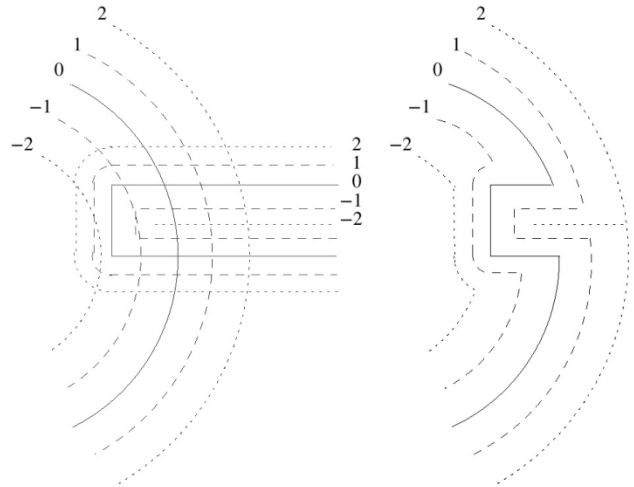
- Operácie pre úpravu diskrétneho signálu
- Erózia $X \ominus B = \{\mathbf{p} | B_{\mathbf{p}} \subset X\}$
- Dilatácia $X \oplus B = \{\mathbf{p} | B_{\mathbf{p}} \cap X \neq \emptyset\}$
- Uzavretie $X \bullet B = (X \oplus B) \ominus B$ $S^{\bullet_r} = \{\mathbf{q} : d'(\mathbf{q}) = -r\}$ $S^{\oplus_r} = \{\mathbf{q} : d(\mathbf{q}) = r\}$
- Otvorenie $X \circ B = (X \ominus B) \oplus B.$



CSG operácie

- Jednoduché a rýchle výpočty prienikov, zjednotení a rozdielov
- Zjednotenie - $D=\min(D_1,D_2)$
- Prienik - $D=\max(D_1,D_2)$
- Rozdiel - $D=\max(D_1,-D_2)$



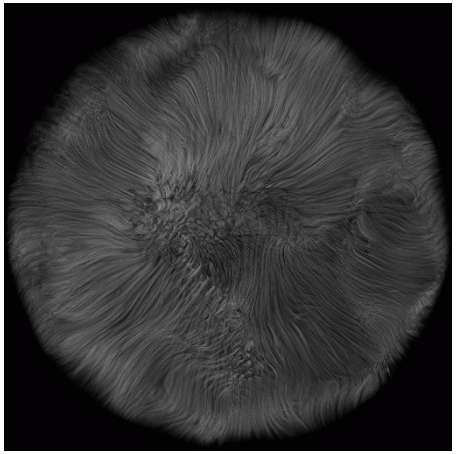


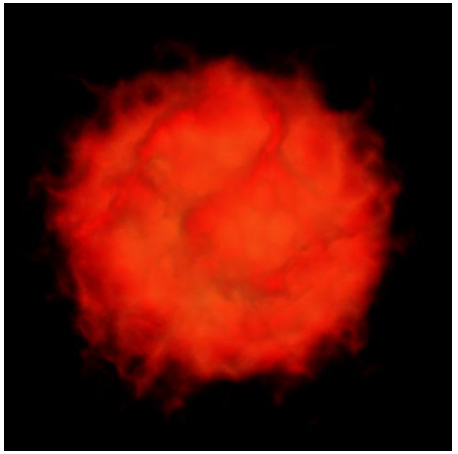
Hypertextúry

- Pridanie detailu nad povrch – srsť, oheň, dym
- Určenie regiónu pre namapovanie textúry

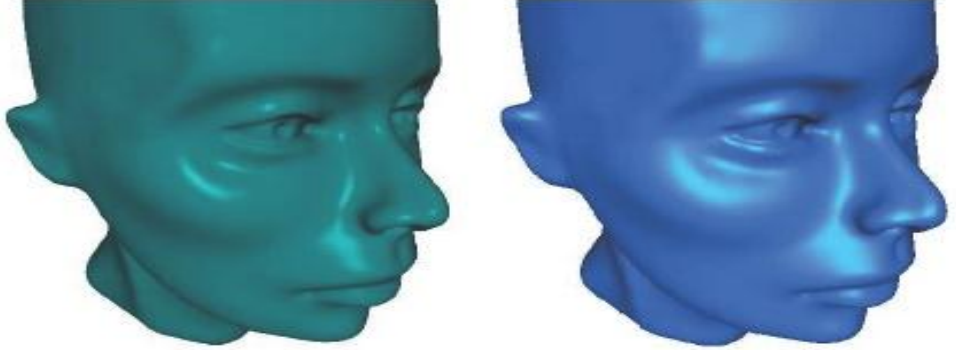
$$D(p) = \begin{cases} 1 & \text{if } \mathrm{d}(p)^2 \leq r_i^2 \\ 0 & \text{if } \mathrm{d}(p)^2 \geq r_o^2 \\ \frac{r_o^2 - \mathrm{d}(p)^2}{r_o^2 - r_i^2} & \text{otherwise,} \end{cases}$$

- Pomocou d(p) sa určia ďalšie vlastnosti ako smer, dotyková plocha, tvorba šumu, ...





Vizualizácia



- Prevod do iných reprezentácií:
  - Polyhedrálna – marching cubes
  - Mračno bodov – priemety voxelov na povrch
- Vizualizácia 3D objemov
- Raytracing:
  - Pre voxle pozadia sa určujú regióny kde sa povrch nenachádza
  - Podobne pre prvý nájdený voxel pozdĺž lúča

koniec (-:

florek@sccg.sk