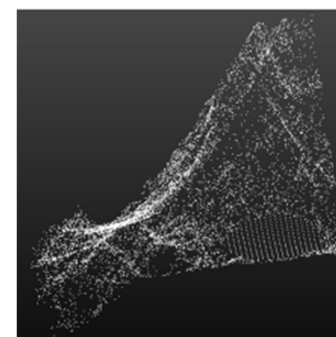
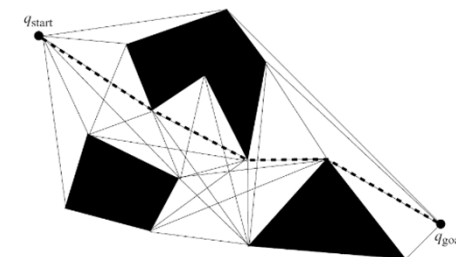
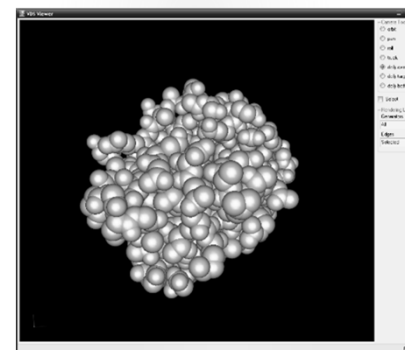


Geometrické algoritmy

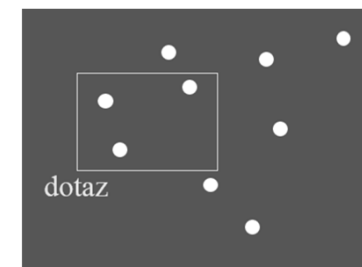
I. KOLINGEROVÁ

Popis

- ▶ Algoritmy pro geometricky formulované úlohy
- ▶ Typicky pro úlohy v počítačové grafice, geometrickém modelování, zpracování geodetických dat, plánování cesty pro robota/avatara/dron, vizualizace bio dat ...
- ▶ Řešený problém nemusí být původně geometrický (např. vyhledávání dat v databázích, optimalizace)
- ▶ Výběr pro tuto přednášku:
 - ▶ Test polohy bodu vůči nadrovině
 - ▶ Konstrukce konvexní obálky množiny bodů v 2D
 - ▶ Dělení prostoru pomocí Voroného diagramu v 2D
 - ▶ Hledání nejbližšího souseda v množině bodů
 - ▶ Zjednodušování polygonů



Discrete Coons Patches

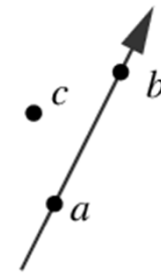


Test polohy bodu vůči nadrovině (test orientace)

- ▶ Vstup: tři body a , b , c
- ▶ Úkol: Leží bod c vlevo, na nebo vpravo od \overrightarrow{ab} ?
- ▶ Identická úloha: určení orientace trojice bodů (pravotočivost/levotočivost/kolinearita)
- ▶ Řešení: test orientace trojice bodů

$$\begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = \begin{vmatrix} a_x - c_x & a_y - c_y \\ b_x - c_x & b_y - c_y \end{vmatrix}$$

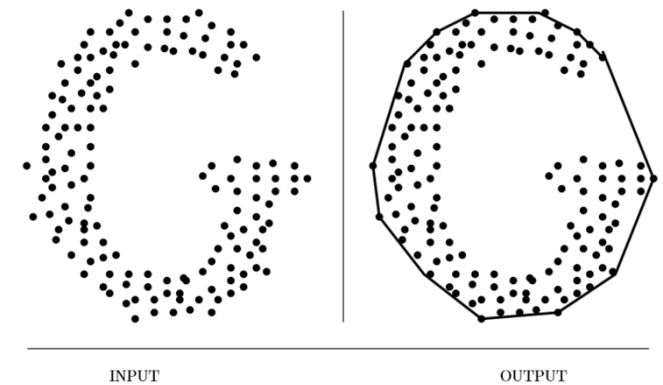
- ▶ Det. $< 0 \iff$ pravotočivost (a naopak), nula – leží v přímce
- ▶ Komplikace: nepřesnost výpočtů ve FPP v blízkosti nuly
- ▶ Lze zobecnit na libovolnou dimenzi



Obr. [Shew]

Konvexní obálka (CH(S))

- ▶ Vstup: množina S n bodů v d -rozměrném prostoru
- ▶ Úkol: Najít nejmenší konvexní polygon (nebo polyedr) obsahující všechny body S
- ▶ Význam: Předzpracování mnoha algoritmů –
 - CH(S) dává přibližnou představu o tvaru S
- ▶ V 2D a 3D snadné, v nejhorším případě $O(n \log n)$
- ▶ Algoritmy konstrukce:
 - ▶ buď $O(n \log n)$ v nejhorším případě
 - ▶ nebo $O(n)$ v očekávaném a $O(n^2)$ v nejhorším případě (podle konkrétních vstupních dat)



Obr. [Skie]

Algoritmy konstrukce CH(S)

- ▶ Důležitý problém – mnoho algoritmů
- ▶ Grahamovo prohledávání (jde se po vrcholech)
- ▶ Balení dárku (Jarvisův pochod) – jde se po hranách
- ▶ Inkrementální vkládání – přidávají se jednotlivé body
- ▶ Rozděl a panuj
- ▶

Grahamovo prohledávání

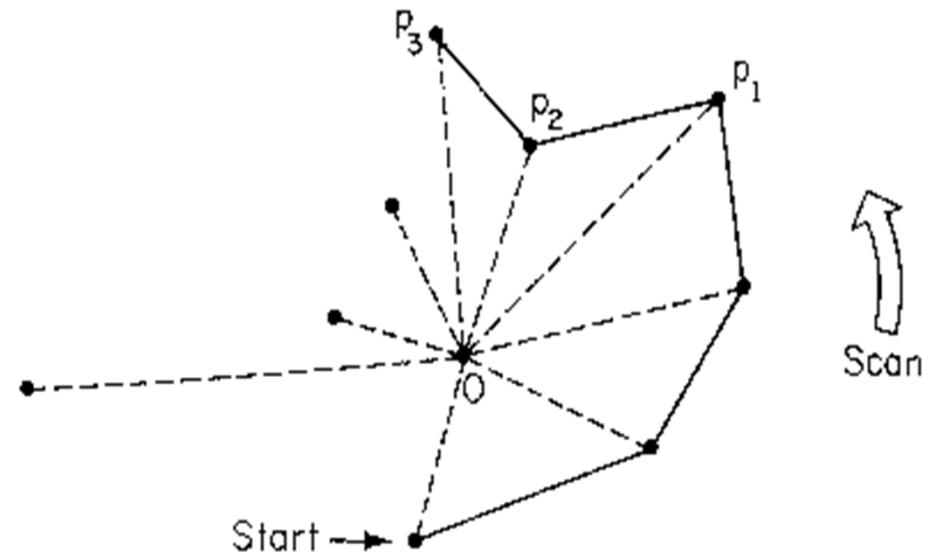
1. Úhlový sort bodů z vnitřního bodu
nebo z některého extrémního vrcholu
2. Průchod z extrémního bodu např. v y :
 - ▶ pokud $p_1p_2p_3$ je pravotočivý:
eliminuj vrchol p_2 a zkontroluj $p_0p_1p_3$
 - ▶ pokud $p_1p_2p_3$ je levotočivý:
zkontroluj $p_2p_3p_4$

▶ [Pre85, Oro94]

Kritérium vyřazení bodu:

Úhel $p_1p_2p_3 \geq \pi$ (pravotočivost)

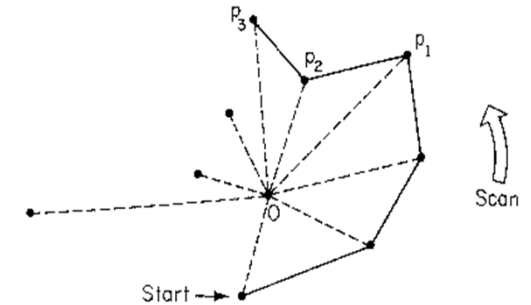
- ale nepočítat úhly, užít determinant. test



Obr. [Pre85]

Grahamovo prohledávání - vlastnosti

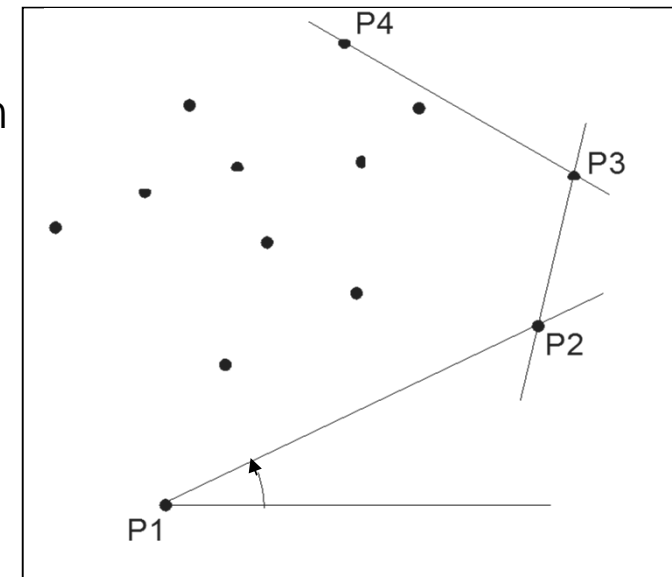
- ▶ Optimální v nejhorším případě: $O(n \log n)$
- ▶ Všechny vstupní body musí být na počátku k dispozici



Obr. [Pre85]

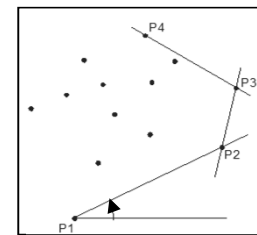
Balení dárku (Jarvisův pochod)

- ▶ Start: extrémní bod např. v y
a pomocná vodorovná přímka proložená tímto bodem
- ▶ Z přímek spojujících aktuální bod s ostatními
se vybere ta s nejmenším úhlem od předchozí hrany
- ▶ Stop: v počátečním bodě anebo v 2. extrému
ve stejné ose, pak druhá strana zrcadlově
- ▶ [Pre85]



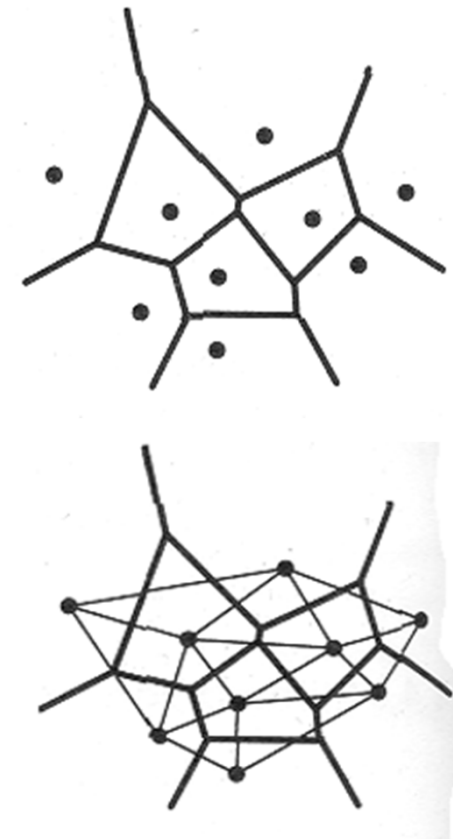
Balení dárku - vlastnosti

- ▶ $O(n^2)$ čas v nejhorším případě
- ▶ $O(k n)$ očekávaný (pro k bodů v $CH(S)$)
- ▶ Všechny vstupní body musí být na počátku k dispozici



Voroného diagram (VD(P))

- ▶ Vstup: množina P n bodů v d -rozměrném prostoru
- ▶ Úkol: Rozdělit prostor na oblasti tak, aby všechny body v oblasti kolem p_i měly blíže k p_i než ke kterémukoliv jinému bodu v P
- ▶ VD: graf, v 2D planární, uzly stupně 3 s výjimkou singularit
- ▶ Význam: dělení na oblasti vlivu, hledání nejbližšího souseda, plánování cest, kvalitní triangulace
- ▶ V 2D a 3D existujē mnoho algoritmů, v nejhorším případě $O(n \log n + n^{\lfloor d/2 \rfloor})$
- ▶ Algoritmy konstrukce v 2D:
 - ▶ buď $O(n \log n)$ v nejhorším případě
 - ▶ nebo $O(n \log n)$ v očekávaném a $O(n^2)$ v nejhorším případě (podle konkrétních vstupních dat)



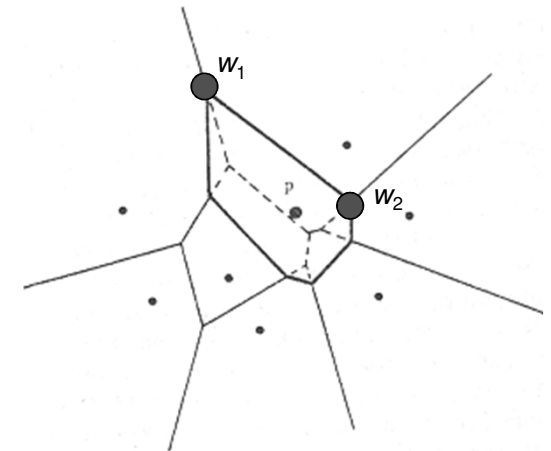
Obr. [Las96]

Algoritmy konstrukce VD(P)

- ▶ Důležitý problém – mnoho algoritmů v mnoha variantách (především různé dat. struktury pro rychlé hledání)
- ▶ Inkrementální vkládání
- ▶ Inkrementální konstrukce
- ▶ Rozděl a panuj
- ▶ Zametání (sweep)
- ▶ Převod do vyšší dimenze
- ▶ Duální konstrukce z triangulace

Inkrementální vkládání

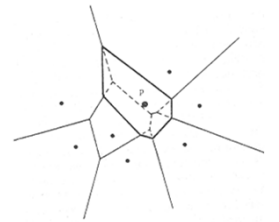
- ▶ Je dán $VD(p_0, \dots, p_{l-1})$
- ▶ Pro $i=l, l+1, \dots, n-1$ proved':
 - ▶ Nalézt Voroného oblast V_k , ve které leží p_i
 - ▶ Určit průsečíky w_1, w_2 kolmého bisektoru k úsečce p_i a p_k s hranami oblasti V_k
 - ▶ Z w_1 do w_2 postupovat přes hranice oblastí pomocí bisektorů, vytvořit tak novou Voroného oblast V_i
- ▶ Např. [Oro94]



Obr. [Oro94]

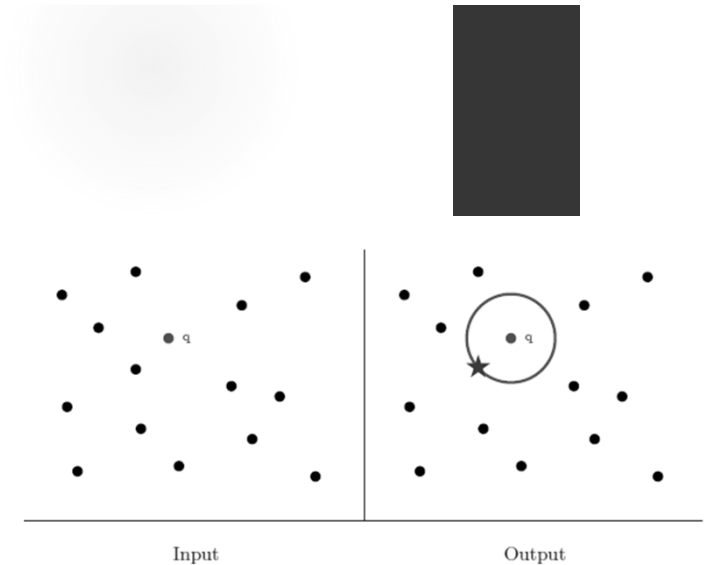
Inkrementální vkládání - vlastnosti

- ▶ $O(n \log n)$ v očekávaném a $O(n^2)$ v nejhorším případě (podle toho, jak dobře vyřešíme hledání oblasti, kde leží nově vkládaný bod)
- ▶ Všechny vstupní body nemusí být na počátku k dispozici



Hledání nejbližšího souseda v množině bodů

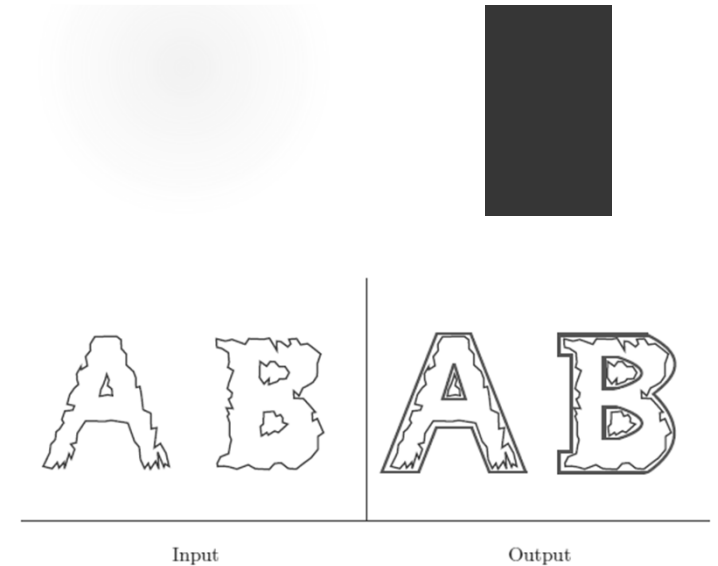
- ▶ Vstup: množina S n bodů v d -rozměrném prostoru, dotazovací bod q
- ▶ Úkol: Který bod S je nejbližší ke q ?
- ▶ Význam: jednoduchá klasifikace příslušnosti bodu ke skupině, hledání nejbližšího potřebného objektu, prevence kolizí ...
- ▶ Bližší specifikace: Kolik sousedů chceme? Jak velké je d ? Potřebujeme přesné řešení? Jsou data statická?
- ▶ Řešení:
 - ▶ VD a kd -stromy, grid – hlavně do $d=3$, projekce do nižší dimenze, hledání v (typicky náhodném) vzorku dat a pak sekvenční dohledání, NN graf



Obr. [Skie]

Zjednodušování polygonů

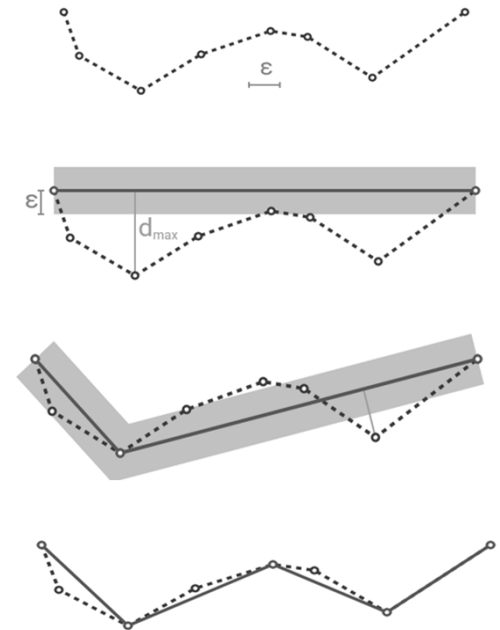
- ▶ Vstup: polygon P s n vrcholy
- ▶ Úkol: Zjednodušit na polygon P' s n' vrcholy,
 $n' \leq n$, tak, aby P' bylo co nejpodobnější P
- ▶ Význam: odstranění šumu, komprese dat
- ▶ Bližší specifikace:
 - ▶ Jak velké zjednodušení chceme? (CH(S) je fajn pro překážky, ale katastrofa pro OCR, CH(S) nepomůže pro konvexní polygony).
 - ▶ Smíme body jen rušit nebo i vkládat?
 - ▶ Pozor na vznik sebeprotínání!
- ▶ Řešení: mnoho metod, např. (Ramer-)Douglas-Peuckerův algoritmus



Obr. [Skie]

(Ramer-)Douglas-Peuckerův algoritmus

- ▶ Vstup: polygon P (i neuzavřený) = (p_1, \dots, p_n) , parametr ε – max. povolená vzdálenost mezi původními body a zjednodušeným polygonem ($\varepsilon > 0$)
- ▶ Výstup: polygon P' s n' vrcholy, $n' \leq n$
- ▶ Převzít body $p_1 p_n$ do P'
- ▶ Vztít přímkou $p_1 p_n$ a najít p_m , nejvzdálenější z bodů p_2, \dots, p_{n-1} od této přímky;
pokud tato vzdálenost nepřesahuje ε , konec,
jinak p_m přidáme do P' a rekurzivně zpracováváme části $p_1 \dots p_m$ a $p_m \dots p_n$
- ▶ Po skončení rekurze převzaté body tvoří P'
- ▶ $O(n' n)$
- ▶ Např. [Car]



Obr. [Car]

Literatura

- ▶ [Car] Cartography Playground, <https://cartography-playground.gitlab.io/playgrounds/douglas-peucker-algorithm/>
- ▶ [Las96] Laszlo M.J.: Computational Geometry and Computer Graphics in C++, Prentice Hall, 1996
- ▶ [Oka08] Okabe A. et al.: Spatial Tessellations: Concepts and Applications of Voronoi Diagrams, 2nd edition, Wiley Series, 2008
- ▶ [ORo94] O'Rourke J.: Computational Geometry in C, Cambridge University Press, 1994 (first edition), 1998 (second edition)
- ▶ [Pre85] Preparata F.P., Shamos M.I.: Computational Geometry: An Introduction, Springer Verlag New York Berlin Heidelberg Tokyo, 1985
- ▶ [Skie] Skiena S.S.: The Algorithm Design Manual, Springer, 3. vydání, 2020, <https://link.springer.com/book/10.1007/978-3-030-54256-6>
- ▶ [Shew] Shewchuk, JR: Adaptive Precision Floating-Point Arithmetic and Fast Robust Predicates for Computational Geometry, <https://www.cs.cmu.edu/~quake/robust.html>