České vysoké učení technické v Praze
Fakulta stavební

Algoritmy v digitální kartografii

Geometrické vyhledávání bodu

Bc. Petra Pasovská
Bc. David Zahradník

Obsah

1 Zadání

Vstup: Souvislá polygonová mapa n polygonů $\{P_1, ..., P_n\}$*, analyzovaný bod* q*.*

Výstup: P_i, $q \in P_i$.

Nad polygonovou mapou implementujete následující algoritmy pro geometrické vyhledávání:

- Ray Crossing Algorithm (varianta s posunem těžiště polygonu).
- Winding Number Algorithm.

Nalezený polygon obsahující zadaný bod q graficky zvýrazněte vhodným způsobem (např. vyplněním, šrafováním, blikáním). Grafické rozhraní vytvořte s využitím frameworku QT.

Pro generování nekonvexních polygonů můžete navrhnout vlastní algoritmus či použít existující geografická data (např. mapa evropských států).

Polygony budou načítány z textového souboru ve Vámi zvoleném formátu. Pro datovou reprezentaci jednotlivých polygonů použijte špagetový model.

Hodnocení:

Krok	**Hodnocení**
Detekce polohy bodu rozlišující stavy uvnitř, vně na hranici polygonu.	10b
Ošetření singulárního případu u Winding Number Algorithm: bod leží na hraně polygonu.	*+2b*
Ošetření singulárního případu u obou algoritmů: bod je totožný s vrcholem jednoho či více polygonů.	*+2b*
Zvýraznění všech polygonů pro oba výše uvedené singulární případy.	*+2b*
Algoritmus pro automatické generování nekonvexních polygonů.	*+5b*
Max celkem:	**21b**

Čas zpracování: 2 týdny.

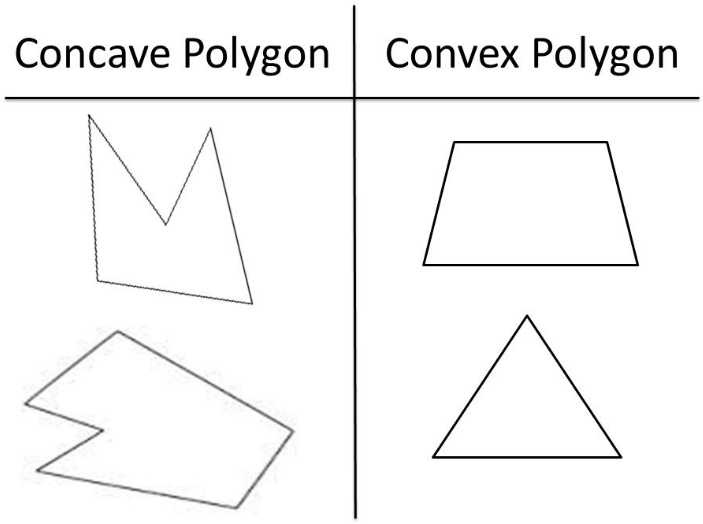
1.1 Údaje o bonusových úlohách

2 Popis a rozbor problému

Hlavním cílem této úlohy je tvorba aplikace, která uživateli určí pozici zvoleného bodu q. Termínem pozice je myšlen polygon, kterému zadaný bod přísluší.

V závislosti na tvaru polygonu rozlišujeme dva základní typy. Jedná se o konvexní a nekonvexní polygon. Polygon můžeme označit za konvexní právě tehdy, pokud jsou všechny vnitřní úhly konvexní, tedy v případě, že úhly jsou menší nebo rovny hodnotě 180 °. Zároveň pro takový polygon platí, že všechny přímky, jejichž oba krajní body leží uvnitř polygonu, mají s tímto polygonem všechny body společné. Takový polygon, který není konvexní, lze označit jako nekonvexní či konkávní.

Porovnání konvexního a nekonvexního objektu lze vidět na následujícím obrázku.



Obrázek 1: Porovnání konvexního a konkávního polygonu [zdroj: 1]

Bod q může mít vůči polygonu P jednu z těchto poloh:

1. Bod q leží uvnitř polygonu P.
2. Bod q leží vně polygonu P.
3. Bod q leží na hraně polygonu P.
4. Bod q je totožný s některým z vrcholu polygonu P.

Pro určení pozice bodu q vůči polygonu existuje několik metod. V této aplikaci jsou implementován metody Ray Crossing Algorithm (varianta s posunem těžiště polygonu) a metoda Winding Number Algorithm.

3 Popis použitých algoritmů

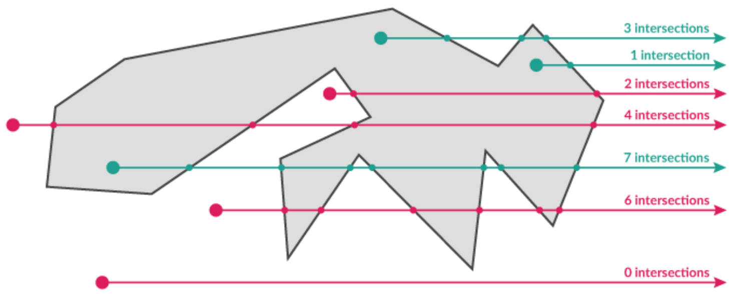
Existuje mnoho metod pro určení pozice bodu q vůči polygonu P. Při volbě metod je vždy potřeba zhodnotit několik důležitých bodů, například požadavky vstupních/výstupních dat, časová náročnost či zda typ problému nespadá mezi NP problémy. V aplikaci byly použity algoritmy Ray Crossing Algorithm a Winding Number Algorithm. Mezi další známé metody pro určení pozice bodů patří třeba Line Sweep Algorithm (Zametací přímka), Divide and Conquer (Rozděl a panuj) či lze pozice určit i metodou hrubé síly (Brute Force Algorithm).

3.1 Ray Crossing Algorithm

Ray Crossing Algorithm lze do češtiny přeložit jako paprskový algoritmus. Primárně slouží k určení polohy bodu v konvexních mnohoúhelnících. Lze jej však zobecnit i pro nekonvexní. Obecně si lze metodu představit tak, že z libovolného bodu vedeme polopřímky a hodnotíme průsečíky přímky s hranami polygonu.

Označme si určovaný bod q. Z tohoto bodu je veden paprsek r (ray). Pokud si průsečík přímky r s hranami polygonu P označíme jako k, pak platí:

1. Pokud je k liché: Bod náleží polygonu P. ($q \in P$)
2. Pokud je k sudé: Bod nenáleží polygonu P. ($q \notin P$)



Obrázek 2: Princip Ray Crossing Algorithm [zdroj: 2]

3.1.1 Problematické situace

Při použití Ray Crossing Algorithm může nastat několik problematických situací, které nelze opomenout. Tímto problémem jsou singularity. K singularitě v této metodě může dojít tehdy, pokud bod leží na hraně polygonu či pokud je bod totožný s některým z vrcholů polygonu. Z tohoto důvodu se využívá upravená varianta Ray Crossing Algorithm, kdy je provedena redukce souřadnic bodů

3.1.2 Implementace metody

1. Nastavení počtu průsečíků rovno nule: $inters = 0$
2. Redukce souřadnic x všech bodů polygonu vůči x-ové souřadnici bodu q: $x_i' = x_i - x_q$
3. Redukce souřadnic y všech bodů polygonu vůči y-ové souřadnici bodu q: $y_i' = y_i - y_q$
4. Volba podmínky: $if(y_i' > 0)\&\&(y_{i-1}' <= 0)\|(y_{i-1}' > 0)\&\&(y_i' <= 0)$

5. Při splnění podmínky: $x'_m = (x'_i y'_{i-1} - x'_{i-1} y'_i)/(y'_i - y'_{i-1})$

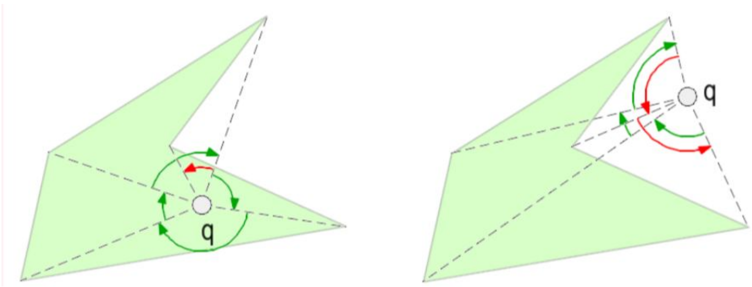
6. Pokud $x'_m > 0$, zvýšení počtu průsečíků o jeden: $inters = inters + 1$

7. Určení zda počet průsečíků sudý či lichý: $if(inters\%2) = 0$, pak: $q \in P$ - počet průsečíků je sudý

8. V opačném případě: $q \notin P$

3.2 Winding Number Algorithm

Metoda ovíjení, či známá jako Winding Number Algorithm, je často používána pro určení pozice bodu vůči nekonvexnímu mnohoúhelníku. Algoritmus si lze představit tak, že se z určovaného bodu otáčíme postupně ke každému bodu polygonu a pokud se otáčíme po směru hodinových ručiček, úhel sčítáme, v opačném případě odčítáme. Pokud je výsledný úhel roven 2π, lze říci, že bod náleží polygonu. V opačném případě nenáleží.



Obrázek 3: Princip Winding Number Algorithm [zdroj: 3]

Při této metodě je zapotřebí si implementovat Winding Number Ω. Pro Ω platí, že je rovna sumě všech rotací ω proti směru hodinových ručiček, které průvodič opíše nad všemi body: $\Omega = \frac{1}{2\pi} \sum_{i=1}^{n} \omega_i^2$
Orientace úhlů je dána:

1. Pokud je úhel $\sphericalangle p_i, q, p_{i+1}$ orientován ve směru hodinových ručiček, pak $\omega_i > 0$

2. Pokud je úhel $\sphericalangle p_i, q, p_{i+1}$ orientován proti směru hodinových ručiček, pak $\omega_i < 0$

V závislosti na výsledné hodnotě Ω lze vyvodit následující závěry:

1. Pokud je $\Omega = 1$, pak platí $q \in P$

2. Pokud je $\Omega = 0$, pak platí $q \notin P$

3.2.1 Problematické situace

Pro Winding Number Algorithm je snadnější řešení singulárních případů. K těm dochází pouze v případě, že $q \approx p_i$.

3.2.2 Implementace metody

1. Nastavení výchozího úhlu ω rovno 0, volba tolerance $\epsilon : \omega = 0, \epsilon = 1e - 10$
2. Určení orientace o_i bodu q ke straně p_i, p_{i+1}
3. Určení úhlu: $\omega_i = \sphericalangle p_i, q, p_{i+1}$
4. Volba podmínky - pokud pod vlevo: $\omega = \omega + \omega_i$
5. V opačném případě: $\omega = \omega - \omega_i$
6. Volba podmínky - pokud rozdíl: $(\omega - 2\pi) < \epsilon$, pak platí: $q \in P$
7. V opačném případě: $q \notin P$

4 Vstupní data

5 Výstupní data

6 Aplikace

7 Dokumentace

8 Závěr

9 Reference

1. Presentation about convex and concave polygons [online][cit. 21.10.2018].
   Dostupné z: https://slideplayer.com/slide/6161031/

2. Introducing Wherewolf - A serverless boundary service from WNYC [online][cit. 21.10.2018].
   Dostupné z: https://source.opennews.org/articles/introducing-wherewolf/

3. BAYER, Tomáš. Geometrické vyhledávání [online][cit. 21.10.2018].
   Dostupné z: https://web.natur.cuni.cz/ bayertom/images/courses/Adk/adk3.pdf