

Algoritmy v počítačové kartografii

Úloha č. 3

Digitální model terénu

*Čelonk Marek, Sýkora Matúš*

# Zadání

# Popis a rozbor problému + vzorce

Triangulácia T nad množinou bodov P={p1,p2,..,pn} predstavuje také planárne rozdelenie, ktoré vytvorí súbor m trojuholníkov t ={t1,t2,..,tm} a hrán, aby platilo:

* Ľubovoľne dva trojuholníky ti, tj ∈ T ,(i 6= j), majú spoločnú najviac hranu.
* Zjednotenie všetkých trojuholníkov t ∈ T tvorí H(p).
* Vnútri žiadneho trojuholníka neleží žiadny ďalší bod z P .

Medzi najčastejšie aplikácie triangulácie v kartografii a GIS patrí kartografická generalizácia, tvorba digitálnych modelov terénu, segmentácia obrazu a rozpoznávanie vzorov. V oblasti DPZ tvorba priestorových modelov z dát laserového skenovania. Triangulácia sa ďalej využíva v počítačovej grafike modelovaní prírodných javov v biometrii.

Medzi požiadavky na triangulačný algoritmov patria:

* Jednoduchosť algoritmu a jednoduchá implementácia
* Dostatočná rýchlosť pre veľké množiny bodov a zložitosť O(n.log(n))
* Malá citlivosť na singulárne prípady
* Prevod do vyšších dimenzií
* Schopnosť paralelizácie algoritmu
* Optimálny tvar trojuholníkovej siete

Tirangulácia by mala vytvárať pravidelné trojuholníky vhodného tvaru. Táto trojuholníková sieť by sa mala čo najviac primykať k terénu. Ďalšou dôležitou súčasťou triangulácie je vkladanie povinných hrán a modifikovanie tvaru triangulácie a schopnosť tirangulácie nekonvexných oblasti či oblasti obsahujúce dieru. A vynechať plochy ktoré nie sú triangularizované ako napr. vodné plochy a budovy.

Najznámejšie triangulácie

* Greedy triangulácia.
* Delaunay triangulácia.
* MWT (Minimum Weight Triangulation).
* Constrained triangulácia.
* Dátovo závislá triangulácia.

Metódy pre konštrukciu delaunay triangulácie:

* Lokálne prehadzovanie
* Inkrementálna konštrukcia
* Inkrementálne vkladanie
* Divide and conquer
* Sweep line

# Popisy algoritmů formálním jazykem

Triangulace je v naší aplikaci vytvořena metodou inkrementální konstrukce. V prvním kroku je náhodně z množiny vybrán bod *p1* a k němu nejbližší bod *p2*. Z těchto dvou bodů je vytvořena hrana e. V následujícím kroku je nalezen bod *p*, který s body *p1* a *p2* tvoří trojúhelník s nejmenší opsanou kružnicí. Dále dojde ke kontrole, zda nebyl bod hledán v prázdné polorovině. Pokud nebyl bod nalezen, je prohozena orientace hrany *e*. V 5. kroku jsou vytvořeny hrany *e2*a *e3*. V následujících dvou krocích jsou všechny tři hrany přidány do AEL (active edge list) a DT (delaunay triangulation).

Následuje while cyklus, který běží do té doby, než je AEL prázdný. V prvním kroku se vyjme z AEL hrana *e*. Následuje prohození její orientace. Dále je opět hledán bod *p*, který s vyjmutou hranou vytvoří trojúhelník s nejmenší opsanou kružnicí. Při každém opakování cyklu se pracuje s jinou hranou. V dalším kroku je podmínka, zda existuje takový bod *p*, pokud ne, tak cyklus končí. V případě, že je podmínka splněna dojde k vytvoření hran *e2*a *e3*. Hrana *e* je přidána do DT. V posledním kroku cyklu dochází k volání funkce *add* pro hrany *e2*a *e3*.

Funkce *add* v prvním kroku vytvoří hranu *e′*. Dále je kontrolováno, jestli tato hrana existuje v AEL, pokud ano, tak je z listu vyjmuta. V opačném případě je do listu přidána. V posledním kroku je vždy hrana přidána do DT.

**Delaunayho triangulace inkrementální konstrukce:**

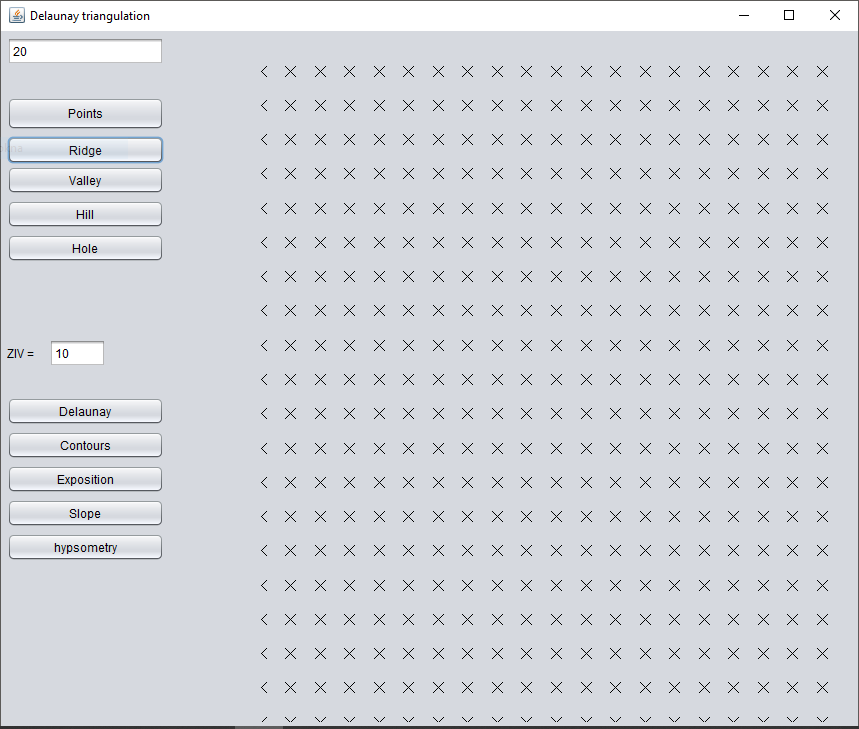
1. Point p1 = rand(P), ‖p2 – p1‖2 = min
2. Edge e = (p1p2)
3. p = arg min∀pi ∈ σL(e) r′(ki), ki = (a, b, pi), e = (a, b)
4. Pokud ~~Ǝ~~p, prohodit orientaci e ← (p1p2). Jdi na 3.
5. e2 = (p2,p), e3 = (p,p1)
6. AEL ← e, AEL ← e2, AEL ← e3
7. DT ← e, DT ← e2, DT ← e3
8. while AEL not empty:
   1. AEL → e, e = (p1p2)
   2. e = (p2p1)
   3. p = arg min∀pi ∈ σL(e) r′(ki), ki = (a, b, pi), e = (a, b)
   4. if Ǝp:
      1. e2 = (p2,p), e3 = (p,p1)
      2. DT ← e
      3. add(e2, AEL, DT), add(e3, AEL, DT)

**Funkce add(e, AEL, DT):**

1. e′ = (b, a)
2. if (e′ ∈ AEL):
3. AEL → e′
4. else:
5. AEL ← e
6. DT ← (a, b)

# Problematické situace a jejich rozbor

Pri vytváraní algoritmu pre generovanie terénnych tvarov sme narazili na problém, kde pre body usporiadané v mriežke nebolo možne vytvoriť delaunay trianguláciu. Algoritmus pre tento druh vstupu funguje maximálne pre veľkosť mriežky 8x8 bodov, pre viac bodov sa algoritmus zacykli a padne. Spočiatku sme mysleli že to bude kvôli kolineárnosti bodov, ale to nám vyvratil fakt že algorimus funguje pre 64 bodov kde sú body rovnako rozmiestnené. Situácia je zobrazená na obrázku č.x.



Obr. č.x Body usporiadané v mriežke

# Vstupní data, formát vstupních dat, popis

V aplikácií ma užívateľ možnosť náhodne vygenerovať body, alebo si môže vybrať generovanie náhodných bodov v štyroch terénnych tvarov. Medzi terénne tvary patria Ridge (chrbát), Valley (údolie), Hill (kopec) a Hole (jama). Užívateľ ma taktiež možnosť zadať pre aký počet bodov sa majú dané útvary alebo náhodne body vygenerovať a taktiež zadať interval rozostupu vrstevníc.

# Výstupní data, formát výstupních dat, popis

Výstupné dáta sú zobrazené vo forme grafiky v grafickom okne aplikácie. Aplikácia umožňuje vykresliť body v rôznych tvaroch, ktoré boli popísane v predchádzajúcej kapitole. Tieto body sú zobrazené ako čierne krížiky. Následne si užívateľ môže vybrať rôzne analýzy medzi ktoré patri Delaunay triangulácia ktorá vykresli jednotlivé delaunay trojuholníky, následne môže spočítať sklon alebo orientáciu delaunay trojuholníkov voči svetovým stranám alebo môže využiť hypsometriu pre zobrazenie výšok digitálneho modelu terénu. Aplikácia ponuka možnosť aj vykreslenia vrstevníc kde si užívateľ zadáva interval rozostupu

# Printscreen vytvořené aplikace

# Dokumentaci: popis tříd, datových položek a jednotlivých metod

Aplikácia sa skladá zo šiestich tried. Jedna reprezentuje grafické rozhranie, druhá vykresľovanie grafiky, ďalšia algoritmy a posledné tri reprezentujú dátové štruktúry.

Trieda Triangle reprezentuje trojuholník a obsahuje 5 premenných medzi ktoré patria tri body typu Point3D a dve premenné typu double ktoré sa používajú pri výpočte sklonu a hypsometrie. Trieda obsahuje tri metódy. Metóda getNormalVec ma za úlohu výpočet normálový vektor trojuholníka. Táto metóda sa následne využíva v ďalších dvoch metódach getSlope ktorá vypočíta sklon a getExp ktorá vypočíta orientáciu trojuholníka voči svetovým stranám. Trieda Point3D obsahuje tri premenné typu double, ktoré reprezentujú súradnice bodov x,y,z. Obsahuje metódu toPoint ktorá vráti nový Point2D. Obsahuje 3 metódy pre dopyt súradníc a tri pre nastavenie týchto hodnôt. Trieda Edge obsahuje dve premenné typu Point3D ktoré reprezentuje začiatok a koniec hrany. Trieda ma päť metód. Metóda equals porovnáva objekty, metóda hasCode vráti hodnotu pre objekt, metóda swapp mení orientáciu hrany, swappedEdge vytvorí hranu s opačnou orientáciou, a metóda toString nastaví formát v tvare string.

Trieda Algorithms obsahuje tri premenne dve typu enum a jedna typu double. OrientationEnum– metóda na určenia orientácie (clockwise, counter clockwise a colinear), getOrientation metóda na určenie orientácie pre Point3D, dotProd – výpočet skalárneho súčinu, len – výpočet dĺžky skalárneho súčinu, angle – výpočet uhlu medzi vektormi, dotProdNorm – výpočet normovaného uhla, splitPoints – pridá body do listov podľa strany na ktorej ležia, distanceFromLine – výpočet vzdialenosti bodu od línie, dist – výpočet vzdialenosti medzi dvoma bodmi, circleRadius – pre výpočet kružnice a stredového bodu, minimalBoundingCircle – pre nájdenie s opísaním minimálnej kružnice, addToAel – metóda, ktorá kontroluje či daná hraná sa nachádza v liste, delaunay – metóda ktorá zabezpečuje výpočet delaunay triangulácie, calcContours – ma na starosti výpočet vrstevníc, ďalšia metóda calcContours – vytvára list v ktorom sú uložené línie reprezentujúce vrstevnice a metóda det3 vypočíta determinant.

Trieda Drawpanel ma na starosti okno na vykreslenie bodov a jednotlivých analýz v aplikácií. Obsahuje sedem premenných, ktoré sú typu Point3D, Path2D, List a boolean. Obsahuje jednu metódu predefinovanú metódu paintComponent, ktorá zabezpečuje vizualizáciu bodov, delaunay triangulácie, vrstevníc, sklonu, expozície a hypsometrie.

Gui je trieda, ktorá reprezentuje grafické rozhranie celej aplikácie. Obsahuje viacero premenných, ktoré reprezentujú jednotlivé tlačidlá v aplikácii. V tejto triede sa nachádzajú metódy zabezpečujúce jednotlivé úkony po stlačení jednotlivých tlačidiel. Ďalej sa tam nachádza metóda generateRandom3D pre generovanie náhodne vygenerovaných bodov, generateRidge metóda pre generovanie bodov v tvare chrbátnice, generateValley metóda pre generovanie bodov v tvare údolia. Metóda main, ktorá tvorí celkové grafické rozhranie.

# Závěr, možné či neřešené problémy, náměty na vylepšení

Aplikácia je funkčná pre všetky zadané algoritmy čiže pre delaunay trianguláciu sklon, expozíciu a pre vykreslenie vrstevníc. V aplikácia sme ešte implementovali generovanie štyroch terénnych tvarov a taktiež hypsometriu.

Nedostatok aplikácie je tvorba delaunay triangulácie pre body usporiadané v mriežke, čo bolo spomenuté v predchádzajúcej kapitole.

Medzi vylepšenia aplikácie by sme ešte zaradili generovanie popis vrstevníc, ktorý zatiaľ chýba. Taktiež by bolo možne vyriešiť elegantnejšie orientáciu voči svetovým stranám jednotlivých trojuholníkov, kde by sa dalo vyriešiť automatické generovanie farieb. Taktiež by bolo možne aplikáciu rozšíriť o možnosť výberu farebných škál pre sklon a hypsometriu.

# Seznam literatury

BAYER, T. 2018. Rovinné triangulace a jejich využití: Greedy Triangulation. Delaunay Triangulation. Constrained Delaunay Triangulation. Data Dependent Triangulation. DMT. [elektronický zdroj]. Praha.

Dostupné na: https://web.natur.cuni.cz/~bayertom/index.php/teaching/algoritmy-pro-tvorbu-digit-map