Algoritmy v digitální kartografii

Množinové operace s polygony

Zimní semestr 2018/2019

Tereza Kulovaná
Markéta Pecenová

Obsah

1 Zadání

Zadání úlohy bylo staženo ze stránek předmětu 155ADKG.

Vstup: množina n polygonů P=$\{P_1, ..., P_n\}$.

Výstup: množina m polygonů $P' = \{P'_1, ..., P'_m\}$.

S využitím algoritmu pro množinové operace s polygony implementujte pro libovolné dva polygony $P_i, P_j \in P$ následující operace:

- Průnik polygonů $P_i \cap P_j$,
- Sjednocení polygonů $P_i \cup P_j$,
- Rozdíl polygonů: $P_i \cap \overline{P}_j$, resp. $P_j \cap \overline{P}_i$.

Jako vstupní data použijte existující kartografická data (např. konvertované shape fily) či syntetická data, která budou načítána z textového souboru ve Vámi zvoleném formátu.

Grafické rozhraní realizujte s využitím frameworku QT.

Při zpracování se snažte postihnout nejčastější singulární případy: společný vrchol, společná část segmentu, společný celý segment či více společných segmentů. Ošetřete situace, kdy výsledkem není 2D entita, ale 0D či 1D entita.

Pro výše uvedené účely je nutné mít řádně odladěny algoritmy z úlohy 1. Postup ošetření těchto případů diskutujte v technické zprávě, zamyslete se nad dalšími singularitami, které mohou nastat.

Hodnocení:

Krok	**Hodnocení**
Množinové operace: průnik, sjednocení, rozdíl	20b
Konstrukce offsetu (bufferu)	*+10b*
Výpočet průsečíků segmentů algoritmem Bentley & Ottman	*+8b*
Řešení pro polygony obsahující holes (otvory)	*+6b*
Max celkem:	**44b**

Čas zpracování: 2 týdny

V rámci této úlohy byly implementovány bonusové úlohy č. .

2 Popis a rozbor problému

Úloha **Množinové operace s polygony** se zabývá vytvořením aplikace, která Delaunayovou triangulací nad vstupní množinou bodů P vytvoří trojúhelníkovou síť, pro kterou se lineární interpolací vypočítají vrstevnice. Aplikace dále počítá a vhodně vizualizuje sklon a orientaci trojúhelníků ke světovým stranám.

Způsobů, jak geometricky zkonstruovat trojúhelníkovou síť, je více. Pro účely této úlohy byla vybrána Delaunayova triangulace, protože poskytuje optimální trojúhelníky z hlediska tvaru, což je zejména v kartografii velmi důležité. Delaunayova triangulace má čtyři základní vlastnosti:

1. Uvnitř kružnice opsané trojúhelníku $t_i \in DT$ neleží žádný jiný bod množiny P.
2. DT maximalizuje minimální úhel v $\forall t_i$, avšak DT neminimalizuje maximální úhel v t_i.
3. DT je lokálně optimální i globálně optimální vůči kritériu minimálního úhlu.
4. DT je jednoznačná, pokud žádné čtyři body neleží na kružnici.[1]

3 Algoritmy

Tato kapitola se zabývá popisem algoritmů, které byly v aplikaci implementovány.

3.1 Delanuayova triangulace

Delaunyova triangulace byla realizována metodou inkrementální konstrukce, která je založena na postupném přidávání bodů do již vytvořené triangulace. Během výpočtu je používaná struktura AEL (Active Edge List), která obsahuje všechny hrany, proto které ještě nebyl nalezen třetí bod trojúhelníku. Hrana, pro kterou byl bod nalezen, je vzápětí ze seznamu odstraněna. Před přidáním hran do seznamu je kontrolováno, zda se v něm již nenachází hrana s opačnou orientací. V takovém případě není hrana do seznamu přidána.

Mějme množinu bodů P a orientovanou hranu e_i. Hledáme takový bod $p_i \in P$, který se nachází v levé polorovině vymezené hranou e_i, pro který dále platí, že poloměr kružnice jemu a hraně opsané je minimální. Během výpočtu jsou upřednostňovány body, jejichž středy opsaných kružnic se nachází v pravé polorovině. Je-li bod splňující výše uvedená kritéria nalezen, vytvoří se dvě nové orientované hrany e_{i+1} a e_{i+2}, které se přidají do triangulace a do AEL. Původní hrana e_i je z AEL odstraněna. Není-li žádný vhodný bod nalezen, dochází k prohození orientace hrany e_i a postup je opakován. Celý proces je ukončen ve chvíli, kde se v AEL nenachází již žádná hrana.

Zjednodušený zápis algoritmu lze zapsat způsobem uvedeným níže:

[1] Zdroj: https://web.natur.cuni.cz, slide 22

1. Nalezení pivota a: $a = \min(x)$ a jemu nejbližší bod b
2. Vytvoření $e_1 = (a, b)$
3. Nalezení Delaunayova bodu: $r(k_i) = \min$, $k_i = (e_1, p_i)$
4. Podmínka: p_i nenalezen $\rightarrow e_1 = (b, a)$, opakuj krok 3
5. Vytvoř zbylé hrany trojúhelníku: $e_2 = (b, p_i)$, $e_3 = (p_i, a)$
6. Přidej hrany do AEL: $AEL \leftarrow e_1$, $AEL \leftarrow e_2$, $AEL \leftarrow e_3$
7. Přidej hrany do triangulace DT: $DT \leftarrow e_1$, $DT \leftarrow e_2$, $DT \leftarrow e_3$
8. Dokud $AEL \neq \emptyset$:

   Vezmi první hranu z $AEL \rightarrow e_1$

   Prohoď orientaci: $e_1 = (b, a)$

   Nalezení Delaunayova bodu: $r(k_i) = \min$, $k_i = (e_1, p_i)$

   Podmínka: p_i nalezen

   Vytvoř zbylé hrany trojúhelníku: $e_2 = (b, p_i)$, $e_3 = (p_i, a)$

   Přidej hranu do DT: $DT \leftarrow e_1$

   $add(e_2, AEL, DT), add(e_3, AEL, DT)$

Lokální algoritmus *add*:

1. Prohoď orientaci: $e' = (b, a)$
2. Podmínka: $e' \in AEL \rightarrow$ odstraň e' z AEL
3. Jinak: $AEL \leftarrow e$
4. $DT \leftarrow e$

3.2 Vrstevnice

Druhý algoritmus použitý v aplikaci slouží k výpočtu vrstevnic. Vrstevnice byly zkonstruovány metodou lineární interpolace, která je založena na předpokladu, že spád terénu mezi dvěma body p_i se mění stejně, tedy konstantně. Výpočet byl proveden postupně pro všechny trojúhelníky a vrstevnice byly ukládány jako seznam hran.

Mějme trojúhelník t_i tvořený třemi hranami $e_1(p_1, p_2)$, $e_2(p_2, p_3)$ a $e_3(p_3, p_1)$ a rovinu ρ o výšce Z. Hledáme průsečnici roviny trojúhelníku t_i s rovinou ρ. Pro kritérium $t = (z - z_i)(z - z_{i+1})$ mohou nastat tři základní situace:

1. $t > 0 \rightarrow e_i \notin \rho$
2. $t = 0 \rightarrow e_i \in \rho$

3. $t < 0 \rightarrow e_i \cap \rho$

Pro případy 1 a 2 nebyly vrstevnice řešeny. Nastane-li případ 3 ($e_i \cap \rho$), je pro hranu e_i a rovinu ρ níže uvedenými vzorci vypočten průsečík a o výšce z_a: (pro přehlednost uvedeno pro hranu e_1)

$$x_a = \frac{(x_2 - x_1)}{(z_2 - z_1)}(z_a - z_1) + x_1$$

$$y_a = \frac{(y_2 - y_1)}{(z_2 - z_1)}(z_a - z_1) + y_1$$

3.3 Sklon

Algoritmus pro výpočet sklonu počítá sklon jednotlivých trojúhelníků t_i. Sklon je úhel φ mezi svislicí n a normálou trojúhelníku n_t. Rovina trojúhelníku t_i je určena vektory u, v. Sklon nabývá hodnot <0°;90°> a v aplikaci je zobrazen v odstínech šedi.

$$n = (0, 0, 1)$$

$$n_t = \vec{u} \times \vec{v}$$

$$\varphi = \arccos\left(\frac{n_t \cdot n}{|n_t||n|}\right)$$

3.4 Orientace

Orientace terénu A je definována jako azimut průmětu gradientu normálového vektoru roviny trojúhelníku do roviny x,y. Nabývá hodnot <0°;360°> a v aplikaci je zobrazen barevnou škálou.

$$n_t = \vec{u} \times \vec{v}$$

$$A = \arctan 2\left(\frac{n_x}{n_y}\right)$$

4 Vstupní data

Pro účely této úlohy byla použita data, která byla naměřena v rámci geodetické výuky v terénu v Mariánské u Jáchymova. Souřadnice X a Y byly pro tuto úlohu zredukovány na rozumnou velikost, souřadnice Z byla zachována. Body byly zaměřeny metodou GNSS a totální stanicí a znázorňují tamní louku a část silnice. Seznam vstupních bodů je uložen v textovém souboru testing_data.txt. Soubor je nutné do aplikace nahrát pomocí tlačítka *Load points*. Struktura textového souboru je následující:

Sloupec 1: souřadnice X [m]
Sloupec 2: souřadnice Y [m]

Sloupec 3: souřadnice Z [m]

Po úspěšném/neúspěšném nahrání souboru je uživatel upozorněn hláškou. Uživatel nemůže kliknout na žádné jiné tlačítko pro výpočty, nejsou-li nahrána data (tlačítka jsou zašedivělá). Aplikace dále nedovoluje spustit výpočty, jejichž fungování je závislé na vygenerované trojúhelníkové síti, nebyla-li předtím vytvořena. Uživatel má dále možnost zvolit krok, v jakém se budou vykreslovat vrstevnice. Hodnoty lze měnit šipkami nahoru/dolů po 5 m nebo ručně vepsat hodnotu celého čísla v rozmezí 1 m až 100 m. Delaunayova triangulace, vrstevnice, sklon a orientace se generují stisknutím příslušných tlačítek.

5 Výstupní data

Vstupní množina bodů a nad ní vygenerovaná trojúhelníková síť je zobrazena ve grafickém okně aplikace. Vykreslování vrstevnic, sklonu a orientace je odděleno. U vrstevnic je každá pátá (hlavní) zvýrazněna. Sklon je v odstínech šedi (čím vyšší sklon, tím tmavší barva). Pro zobrazení orientace trojúhelníků ke světovým stranám byla využita prostřední kružnice barvené škály ze stránek společnosti *ESRI*, viz níže. Aplikace je uvedena do výchozího stavu stisknutím tlačítka *Clear*.

6 Aplikace

V následují kapitole je představen vizuální vzhled vytvořené aplikace tak, jak ji vidí prostý uživatel.

7 Dokumentace

Tato kapitola obsahuje dokumentaci k jednotlivým třídám.

7.1 !Algorithms

Třída *Algorithms* obsahuje metody pro výpočet Delaunayovy triangulace a analýzu DTM.

getPositionWinding

Metoda **delaunayTriangulation** vytváří nad množinou bodů Delaunayovu triangulaci. Na vstupu je vektor bodů typu `QPoint3D`, metoda vrací uspořádaný vektor hran `Edge`, které tvoří jednotlivé trojúhelníky.

Input:

- *vector* <`QPoint3D`> *points*

Output:

- *vector* <`Edge`>

createContours

Metoda **createContours** vytváří nad vstupní množinou hran vrstevnice dle zadaného kroku, po kterém se vrstevnice budou vykreslovat. Metoda vrací vektor hran, které představují vrstevnice.

Input:

- *vector* <`Edge`> *dt*
- `double` $z\_min \rightarrow$ minimální výška
- `double` $z\_max \rightarrow$ maximální výška
- `double` $dz \rightarrow$ krok vrstevnic

Output:

- *vector* <`Edge`>

getSlope

Metoda **getSlope** počítá sklon trojúhelníku, který je tvořen třemi body. Návratová hodnota typu `double` nabývá hodnot <0°;90°> a vrací sklon trojúhelníku.

Input:

- `QPoint3D` p_1
- `QPoint3D` p_2
- `QPoint3D` p_3

getAspect

Metoda **getAspect** počítá orientaci trojúhelníku, který je tvořen třemi body, ke světovým stranám. Návratová hodnota typu `double` vrací orientaci trojúhelníku ve stupních. Orientace je pravotočivá a nabývá hodnot <-180°;180°>.

Input:

- `QPoint3D` p_1
- `QPoint3D` p_2
- `QPoint3D` p_3

analyzeDTM

Metoda **analyzeDTM** vytváří z vektoru hran trojúhelníky a počítá pro ně sklon a orientaci. Vypočtené hodnoty ukládá do datového typu `Triangle`. Návratová hodnota metody je vektor trojúhelníků typu `Triangle`.

Input:

- *vector* <`Edge`> *dt*

Output:

- *vector* <`Triangle`>

getPointLinePosition

Metoda **getPointLinePosition** určuje polohu bodu q vzhledem k přímce tvořené dvěma body. Na vstupu jsou 3 body typu `QPoint3D`, návratová hodnota je nově definovaný typ `TPosition`.

Input:

- `QPoint3D` q
- `QPoint3D` a
- `QPoint3D` b

Output:

- `LEFT` → bod se nachází vlevo od přímky
- `RIGHT` → bod se nachází vpravo od přímky
- `ON` → bod se nachází na přímce

getCircleRadius

Metoda **getCircleRadius** počítá poloměr kružnice, která je tvořena 3 body. Na vstupu jsou 4 body typu `QPoint3D`, návratová hodnota typu `double` vrací velikost poloměru kružnice.

Input:

- `QPoint3D` p_1
- `QPoint3D` p_2
- `QPoint3D` p_3
- `QPoint3D` $c \rightarrow$ střed kružnice

getDistance

Metoda **getDistance** počítá vzdálenost mezi dvěma body. Na vstupu jsou 2 body typu `QPoint3D`, návratová hodnota typu `double` vrací vzdálenost mezi dvěma body.

Input:

- `QPoint3D` p_1
- `QPoint3D` p_2

getNearestPoint

Metoda **getNearestPoint** slouží k nalezení nejbližšího bodu z množiny bodů vzhledem k danému bodu p. Na vstupu je daný bod p a vektor bodů typu `QPoint3D`. Návratová hodnota typu `int` vrací index nejbližšího bodu.

Input:

- `QPoint3D` p
- *vector* <`QPoint3D`> *points*

getDelaunayPoint

Metoda **getDelaunayPoint** slouží k nalezení třetího bodu trojúhelníku, který splňuje Delaunayovo kritérium nejmenší opsané kružnice. Na vstupu jsou dva body typu `QPoint3D`, které představují orientovanou hranu, a vektor bodů typu `QPoint3D`. Návratová hodnota typu `int` vrací index hledaného bodu.

Input:

- `QPoint3D` $s \rightarrow$ počáteční bod hrany
- `QPoint3D` $e \rightarrow$ koncový bod hrany
- *vector* <`QPoint3D`> *points*

getContourPoint

Metoda **getContourPoint** počítá průsečík hrany trojúhelníku tvořené dvěma body typu `QPoint3D` s rovinou o dané výšce Z. Návratová hodnota je typu `QPoint3D`.

Input:

- `QPoint3D` p_1
- `QPoint3D` p_2
- `double` z

7.2 Draw

Třída *Draw* obsahuje metody, které nahrávají a vykreslují vstupní množinu bodů. Dále zajišťuje vykreslení a smazání všech operací, kterou jsou nad množinou prováděny.

paintEvent

Metoda **paintEvent** vykresluje vstupní množinu bodů, Delaunayovu triangulaci, vrstevnice a sklon a orientaci trojúhelníků.

clearDT

Metoda **clearDT** slouží k vymazání všech vykreslených dat.

getPoints

Metoda **getPoints** slouží k získání vektoru bodů z kreslící plochy. Metoda vrací vektor bodů typu `QPoint3D`.

getDT

Metoda **getDT** slouží k získání vektoru hran z kreslící plochy. Metoda vrací vektor hran typu `Edge`.

setDT

Metoda **setDT** slouží k převedení Delaunayovy triangulace do kreslícího okna.

setContours

Metoda **setContours** slouží k převedení vrstevnic do kreslícího okna.

setDTM

Metoda **setDTM** slouží k převedení digitálního modelu terénu do kreslícího okna.

loadDTM

Metoda **loadDTM** slouží k načtení vstupních dat do aplikace. Součástí metody je i kontrola, zda se soubor úspěšně nahrál. Návratová hodnota je typu *QString* vrací hlášku, zda byly polygony úspěšně nahrány či nikoli.

drawSlope

Metoda **drawSlope** slouží k vykreslení sklonu trojúhelníků.

drawAspect

Metoda **drawAspect** slouží k vykreslení orientace trojúhelníků.

7.3 QPointFB

Třída *QPointFB* slouží k definování nového datového typu `QPointFB`, který je odvozen od typu `QPointF` a který navíc obsahuje směrnice přímek *alfa* a *beta*, informaci, zda bod je průsečíkem, a polohu bodu.

getAlfa

Metoda **getAlfa** slouží k získání směrnice *alfa*.

setAlfa

Metoda **setAlfa** slouží k nastavení směrnice *alfa*.

getBeta

Metoda **getBeta** slouží k získání směrnice *beta*.

setBeta

Metoda **setBeta** slouží k nastavení směrnice *beta*.

getInters

Metoda **getInters** slouží k získání informace, zda bod je průsečík či nikoli.

setInters

Metoda **setInters** slouží k nastavení informace, zda bod je průsečík či nikoli.

getPosition

Metoda **getPosition** slouží k získání polohy bodu.

setPosition

Metoda **setPosition** slouží k nastavení polohy bodu.

7.4 Types

Třída *Types* slouží k definování nových datových typů výčtového typu.

TPointPolygon

Datový typ **TPointPolygon** definuje polohu bodu q vůči polygonu P.

- `INSIDE` $\rightarrow q \in P$
- `OUTSIDE` $\rightarrow q \notin P$
- `ON` $\rightarrow q$ leží na P

TBooleanOperation

Datový typ **TBooleanOperation** definuje množinovou operaci, která je nad polygony A a B prováděna.

- `INTERSECTION` $\rightarrow A \cap B$
- `UNION` $\rightarrow A \cup B$
- `DIFFAB` $\rightarrow A \backslash B$
- `DIFFBA` $\rightarrow B \backslash A$

T2LinesPosition

Datový typ **T2LinesPosition** definuje polohu dvou přímek a a b.

- `PARALLEL` $\rightarrow a \parallel b$
- `COLINEAR` $\rightarrow a = b$
- `INTERSECTING` $\rightarrow a \cap b \neq \emptyset$
- `NONINTERSECTING` $\rightarrow a \cap b = \emptyset$

TPointLinePosition

Datový typ **TPointLinePosition** definuje polohu bodu q a přímky a.

- `LEFT` $\rightarrow$ bod q leží vlevo od přímky a
- `RIGHT` $\rightarrow$ bod q leží vpravo od přímky a
- `COL` $\rightarrow$ bod q leží na přímce a

7.5 Widget

Metody třídy *Widget* slouží pro práci uživatele s aplikací. Metody na vstupu nemají žádné parametry a návratové hodnoty jsou typu `void`.

on_delaunay_button_clicked

Metoda **on_delaunay_button_clicked** nad vstupní množinou bodů zobrazí Delaunayovu triangulaci.

on_clear_button_clicked

Metoda **on_clear_button_clicked** vrací aplikaci do výchozí polohy smazáním všeho, co bylo vykresleno.

on_contours_button_clicked

Metoda **on_contours_button_clicked** nad vygenerovanou trojúhelníkovou sítí z Delaunayovy triangulace vykreslí vrstevnice.

on_slope_button_clicked

Metoda **on_slope_button_clicked** obarví trojúhelníky vygenerované Delaunayovou triangulací do odstínů šedi podle hodnoty sklonu daného trojúhelníku.

on_aspect_button_clicked

Metoda **on_aspect_button_clicked** obarví trojúhelníky vygenerované Delaunayovou triangulací na základě jejich orientace ke světové straně.

on_load_button_clicked

Metoda **on_load_button_clicked** načítá data z textového souboru. Uživatel sám vyhledává cestu k požadovanému souboru.

8 Závěr

V rámci úlohy *Digitální model terénu a jeho analýzy* byla vytvořena aplikace, která ze vstupní množiny bodů vytváří digitální model terénu. Implementace některých algoritmů byla náročná, avšak výsledek je obstojný. Z kartografického hlediska by aplikace mohla být vylepšena. Jedná se zejména o přidání možnosti navolení povinných a lomových hran, které by zpřesnily výsledný DMT. Algoritmus generuje přijatelné výsledky pro terén, který neobsahuje příliš výrazné terénní hrany. Je také nevhodný pro vstupní data, která jsou rozmístěna pravidelně na mřížce, jelikož hledaná kružnice s nejmenším poloměrem je pro tyto body nejednoznačná. Dále by bylo vhodné zajistit aspoň zevrubní vyhlazení vrstevnic, jelikož působí kostrbatým dojmem.

Do budoucna by bylo vhodné přidat aspoň popis hlavních vrstevnic, případně barevnou hypsometrii. Autorky jsou s výslednou podobou aplikace spokojené.

9 Zdroje

1. *BAYER, Tomáš.* Množinové operace s polygony [online][cit. 4. 1. 2019]. Dostupné z: https://web.natur.cuni.cz

2. *ArcGIS Blog - New Aspect-Slope Raster Function Now Available* [online] [cit. 5. 12. 2018]. Dostupné z: https://www.esri.com/