Technická zpráva úkol č. 3: Digitální model terénu

Miroslav Hrubeš, Lucie Peterková

Praha 2023

1 Zadání

Uloha č. 3: Digirální model terénu
Vstup: množina $P = \{p_1, \ldots, p_n\}, p_i = \{x_i, y_i, z_i\}$
Výstup: polyedrický DMT nad množinou P představovaný vrstevnicemi doplněný vizualizací sklonu trojúhelníku a jejich expozicí.

Metodou inkrementální konstrukce vytvořte nad množinou P vstupních bodů 2D Delaunay triangulaci. Jako vstupní data použijte existující geodetická data (alespoň 300 bodů) popř. navrhněte algoritmus pro generování syntetických vstupních dat představujících významné terénní tvary (kupa, údolí, spočinek, hřbet, ...).

Vstupní množiny bodů včetně níže uvedených výstupů vhodně vizualizujte. Grafické rozhraní realizujte s využitím frameworku QT. Dynamické datové struktury implementujte s využitím STL.

Nad takto vzniklou triangulací vygenerujte polyedrický digitální model terénu. Dále proveďte tyto analýzy:

- S využitím lineární interpolace vygenerujte vrstevnice se zadaným krokem a v zadaném intervalu, provedte jejich vizualizaci s rozlišením zvýrazněných vrstevnic.
- Analyzujte sklon digitálního modelu terénu, jednotlivé trojúhelníky vizualizujte v závislosti na jejich sklonu.
- Analyzujte expozici digitálního modelu terénu, jednotlivé trojúhelníky vizualizujte v závislosti na jejich expozici ke světové straně.

Zhodnoťte výsledný digitální model terénu z kartografického hlediska, zamyslete se nad slabými místy algoritmu založeného na 2D Delaunay triangulaci. Ve kterých situacích (různé terénní tvary) nebude dávat vhodné výsledky? Tyto situace graficky znázorněte.

Zhodnocení činnosti algoritmu včetně ukázek proveďte alespoň na 3 strany formátu A4.

2 Bonusové úlohy

Nebyly řešeny žádné bonusové úlohy.

3 Popis a rozbor problému

3.1 DMT

Matematické zobrazení povrchu země, včetně terénních rysů jako jsou kopce, údolí a jiné topografické prvky, se označuje zkratkou DMT (digitální model terénu). Data pro DMT se získávají pomocí LiDARu nebo fotogrammetrie, kdy je možné získat velké množství výškových bodů, tedy že pro libovolný bod modelu je možné získat jeho nadmořskou výšku.

Jednou z metod pro vytváření digitálních terénních modelů je TIN (Triangular Irregular Network). TIN model vytváří plochy terénu pomocí trojúhelníků, které se vytvoří spojením nespojitých bodů. Tyto polyedrické modely terénu zachycují nejen výškovou informaci, ale také údaje o sklonu terénu.

Obecně se polyedrické modely nejčastěji využívány pro přibližný výpočet průběhu vrstevnic, konstrukci profilů či řezů a další analýzy a to z toho důvodu, že výpočty nad polyedrickým digitálním modelem terénu jsou v porovnání s jinými modely poměrně jednoduché. Oproti jiným modelům, které využívají pravidelné sítě nebo vrstevnice, TIN vypočítává výšku pro každý bod pomocí interpolace výšek okolních bodů, což umožňuje přesnější odhad výšky.

3.2 Delaunayho triangulace

Metoda Delaunayho triangulace (DT) je široce používanou metodou triangulace v GIS. Tato metoda maximalizuje minimální vnitřní úhel vytvořených trojúhelníků, což je kritérium globální optimalizace geometrických parametrů všech trojúhelníků v síti. Princip této metody spočívá v tom, že v kružnici opsané libovolnému trojúhelníku z DT neleží žádný další bod ze vstupní množiny.

Metoda, využívající inkrementální vkládání, je založena na postupném přidávání bodů do již vytvořeného DT. Nejprve je nutné vybrat libovolný bod z množiny a najít jeho nejbližší bod (v euklidovském prostoru). Tento krok dává vzniknout orientované hraně $e = (P_1, P_2)$. Následně se hledá bod P, který musí ležet v levé polorovině od hrany e a zároveň musí minimalizovat poloměr r kružnice k, která je opsaná tomuto bodu a této hraně. Tuto podmínku lze zapsat vzorcem:

$$P = \arg, \min_{r'(k_i)} \forall P_i \in \sigma_L(e), k_i = (e, P_i), P_i(e) \in \sigma_L$$

Po nalezení nejvhodnějšího bodu vzniknou nové hrany $e_1 = (P_2, P), e_2 = (P, P_1)$. Pokud bod P nebyl nalezen, otočíme orientaci hrany a vyhledání bodu se opakuje, opět v levé polorovině hrany e. Tím je dokončena základní inicializace algoritmu, která se poté opakuje.

Po nalezení vhodného Delaunayho bodu jsou dvě nové hrany společně s první definovanou hranou e vloženy do seznamu aktivních hran (AEL). Následně se vezme první hrana, otočí se její orientace a nalezne se pro ni bod P. Dvě nově vzniklé hrany poté přidáme do AEL za předpokladu, že se nově vytvořené hrany v AEL již nenachází, byť s opačnou orientací. V tomto případě je hrana z AEL odstraněna a přidána do výsledné triangulace. Pokud pro aktuální hranu nebyl nalezen žádný bod, znamená to, že se jedná o součást konvexního obalu, tudíž se taky přidá do výsledné triangulace. Když nedojde k nalezení bodu P, hledá se v pravé polorovině.

Daný algoritmus je popsán pseudokódem na následujících řádcích.

Algorithm 1 Delaunayho triangulace

Inicializuj DT a AEL jako prázdné seznamy
Najdi bod P_1 s nejmenší souřadnicí x
Najdi bod P_2, který bude Euklidovskou vzdáleností k P_1 nejbližší
Z nalezených bodů P_1 a P_2 vytvoř hranu e
DokudAEL není prázdné:
    Vezmi první hranu e_1 z AEL a otoč její orientaci
    Najdi Delaunayův bod P
    Jestli P existuje:
        Vytvoř zbývající strany trojúhelníku $e_2 = (P_2, P)$ a $e_3 = (P, P_1)$
        Přidej vzniklé hrany do DT
        Aktualizuj AEL

3.3 Konstrukce vrstevnic

Problémem této úlohy je nalézt průsečnice roviny určené trojúhelníkem, který je součástí Delaunayovy triangulace (DT) a leží vodorovné rovině ρ o výšce h. Tento proces se opakuje pro všechny trojúhelníky v DT. Výpočet souřadnic bodů průsečnic AB se provádí pouze v případě, že průsečnice je úsečkou. Vzorec pro výpočet souřadnic má tento tvar:

$$x_a = \frac{x_3 - x_1}{z_3 - z_1}(z - z_1) + x_1$$
$$x_b = \frac{x_2 - x_1}{z_2 - z_1}(z - z_1) + x_1$$
$$y_a = \frac{y_3 - y_1}{z_3 - z_1}(z - z_1) + y_1$$
$$y_b = \frac{y_2 - y_1}{z_2 - z_1}(z - z_1) + y_1$$

Kontrolu, zda rovina ρ prochází hranou (p_i, p_{i+1}), lze provést pomocí rovnice

$$(z - z_i)(z - z_{i+1}) < 0$$

Daný algoritmus je popsán pseudokódem na následujících řádcích.

Algorithm 2 Konstrukce vrstevnic

1: Vytvoř prázdný seznam hran cl
2: Pro každou po sobě jdoucí trojici hran $e_1 = (p_1p_2)$, $e_2 = (p_2p_3)$, $e_3 = (p_3p_1)$ Delaunayho triangulace:
3: $p_i = [x_i, y_i, z_i]$
4: Pro každou hodnotu z z intervalu (z_{min}, z_{max}) s krokem dz:
5: $dz_i = z_i - z$
6: $t_{ij} = dz_i dz_j$
7: **Jestli** $dz_i = dz_{i+1} = dz_{i+2} = 0$:
8: Jdi na 4)
9: **Jinak Jestli** $dz_i = dz_{i+1} = 0$:
10: Přidej hranu e_1 do cl
11: **Jinak Jestli** $dz_{i+1} = dz_{i+2} = 0$:
12: Přidej hranu e_2 do cl
13: **Jinak Jestli** $dz_{i+2} = dz_i = 0$:
14: Přidej hranu e_3 do cl
15: **Jinak Jestli** $t_{12} \leq$ a $t_{23} < 0$ **nebo** $t_{12} < 0$ a $t_{23} \leq 0$:
16: A=bod ležící na hraně e_1 ve výšce z
17: B=bod ležící na hraně e_2 ve výšce z
18: $e = (A, B)$
19: přidání hrany e do cl
20: **Jinak Jestli** $t_{23} \leq$ a $t_{31} < 0$ **nebo** $t_{23} < 0$ a $t_{31} \leq 0$:
21: B=bod ležící na hraně e_2 ve výšce z
22: C=bod ležící na hraně e_3 ve výšce z
23: $e = (B, A)$
24: přidání hrany e do cl
25: **Jinak Jestli** $t_{12} \leq$ a $t_{31} < 0$ **nebo** $t_{12} < 0$ a $t_{31} \leq 0$:
26: C=bod ležící na hraně e_3 ve výšce z
27: A=bod ležící na hraně e_1 ve výšce z
28: $e = (C, A)$
29: přidej hranu e do cl

3.4 Analýza sklonu

Digitální model terénu umožňuje provádět různé analytické úlohy, například analýzu sklonu terénu. Výpočet sklonu se aplikuje na každý trojúhelník v digitálním modelu terénu.

Rovnice roviny ρ, pro kterou se sklon počítá, je

$$\rho = ax + by + cz + d = 0$$

,

pro níž je počítán gradient ($\nabla\rho$) pomocí vzorce

$$\nabla\rho(x_0, y_0, z_0) = (\frac{\sigma_\rho}{\sigma_x}(x_0), \frac{\sigma_\rho}{\sigma_y}(y_0), (\frac{\sigma_\rho}{\sigma_z}(z_0)) = (a, b, c).$$

Rovinu ρ lze definovat touto maticí

$$\begin{vmatrix} x - x_1 & y - x_1 & z - x_1 \\ x_2 - x_1 & y_2 - x_1 & z_2 - x_1 \\ x_3 - x_1 & y_3 - x_1 & z_3 - x_1 \end{vmatrix} = 0,$$

z které se následně výpočítá odchylka φ od roviny π vzorcem

$$\varphi = arccos \left| \frac{n_1 n_2}{||n_1||||n_2||} \right|$$

.

Daný algoritmus je popsán pseudokódem na následujících řádcích.

Algorithm 3 Sklon trojúhelníku

1: $n_x, n_y, n_z \leftarrow P_1, P_2, P_3$
2: $n \leftarrow \sqrt{nx^2 + ny^2 + nz^2}$
3: $\rho = \frac{nz}{n}$

3.5 Analýza orientace terénu

Dále je možné analyzovat orientaci terénu. Je definována jako azimut průmětu gradientu $\nabla\rho$ roviny trojúhelníku do roviny x, y. Pro vektor gradientu v platí, že

$$v = \frac{\partial p}{\partial x}(x_0), \frac{\partial p}{\partial y}(y_0) = (a, b, 0)$$

Azimut vektoru v je možné spočítat dle vzorce

$$A = arctan\left(\frac{b}{a}\right)$$

Algorithm 4 Orientace svahu

1: $n_x, n_y, n_z \leftarrow P_1, P_2, P_3$
2: aspect = arctan $\frac{n_y}{n_x}$
3: **Pokud** aspect < 0
4: aspect = aspect + 2π

4 Vstupní a výstupní data

Vstupní data jsou v podobě množiny bodů uložené v textovém souboru, kde jsou souřadnice x, y a z oddělené tabulátorem. Jako cvičná data bylo použito bodové mračno z online kurzu od *Trimble: Working With LiDAR (point cloud) Files* (viz zdroje). Dataset byl zmenšen na 552 bodů a některé hodnoty z byly cíleně pozměněny pro názornější funkčnost aplikace.

Výstupem je polyedrický DMT vytvořený nad vstupní bodovou množinou doplněn o vrstevnice a o vizualizace sklone a expozice terénu. V obecné rovině můžeme oblast vstupních data považovat za jedno z klíčových míst, kdy bude polyedrický model narážet na své hranice možností a nemusí dávat dobré výsledky. Například funkčním formátem vstupních dat je

zde pouze textový soubor s jednotlivými souřadnicemi bodů. V případě alternativy ve formě rasterového souboru by nebylo možné algoritmus spustit.

Další samostatnou kapitolou limitů polyedrického modelu je velikost, resp. obsah vstupních dat. Jak už bylo naznačeno výše, terén je obecně matematicky spojitý. Nejedná se o soubor samostatně oddělených diskrétních hodnot jednotlivých bodů. K absolutně přesnému popisu terénu by těchto bodů muselo být nekonečně mnoho, což z praktického hlediska není možné. Snažíme se tedy o jistou formu aproximace na základě menšího počtu bodů (triangulace). Chceme-li mít přesnější výsledky, je nutné aby vstupní data obsahovala větší množství bodů, resp. jejich větší zastoupení na jednotku plochy v dané oblasti. S rostoucím počtem vstupních bodů však narůstá velikost vstupního souboru a celkově větší zatížení algoritmu. Ten se bude k výsledku propracovávat mnohem delší dobu a přitom bude mnohem více hrozit, že dojde k nějakému problému a tím i k pádu programu.

Navíc zde stále mluvíme pouze o hustotě rozmístění bodů v oblasti a ne o velikosti, resp. o oblasti zobrazovaného území samotné. To poskytuje další rozměr problémů vstupních dat. Čím větší oblast, tím větší je potřeba vyššího objemu dat. Oblast zobrazovaného území je navíc z definice vždy uzavřená (tj. není možné zobrazit terén celé Země najednou) a proto problém může nastat na hranici této oblasti. Interpolační algoritmus nebude schopen krajní oblasti vykreslit správně, neboť kvůli definici uzavřeného území zde nebude mít dostatek vstupních dat pro adekvátní výsledky.

Samotný interpolační algoritmus je také oblastí možných nedostatků polyedrického modelu. Různé algoritmy fungují na různých principech a mají tudíž i různé (byť třeba podobné) výsledky. Vždy závisí na typu, možnostech a požadovaných výsledků u dané úlohy.

5 Aplikace

Pomocí skriptu *mainform.py* se spustí aplikace. Na widgetu (obr. 1) se nachází panel nástrojů, kde je možné pomocí ikon v tomto pořadí provést následující akce: *Open file*: otevření vstupního bodového souboru, *Generate Delaunay triangulation*: vygenerování Delaunayho triangulace (je nutné provést tento krok před spuštěním dalších analýz), *Create contour lines*: vygenerování vrstevnic, *Analyze slope of DTM*: analýza sklonu, *Analyze aspect of DTM*: analýza orientace svahu, *Set contour lines parameters*: nastavení parametrů vrstevnic, *Clear results*: odstranění provedených analýz, *Clear all*: odstranění provedených analýz a datového souboru z aplikace,*Exit apllication*: ukončení aplikace.

6 Dokumentace

Aplikace je tvořena šesti skripty. Uživatelské rozhraní je tvořeno skriptem *mainform.py*, jehož základ byl vytvořen pomocí softwaru QT Creator. Vizualizaci a vykreslování objektů zajišťuje skript *draw.py*. Skript *algorithms.py* definuje matematické metody pro správné provedení zvolených analýz. Skript *QPoint3DF* definuje třídu QPoint3DF, která reprezentuje bod v 3D prostoru pomocí tří souřadnic (x, y, z). Skript *edge.py* definuje třídu Edge pro reprezentaci hrany mezi dvěma body v trojrozměrném prostoru pomocí třídy QPoint3DF a implementuje několik metod pro práci s touto třídou, jako jsou získání počátečního a koncového bodu, vytvoření nové hrany s opačnou orientací a porovnání dvou hran.Skript *triangle.py* definuje třídu Triangle, která reprezentuje trojúhelník v digitálním modelu terénu (DMT) a obsahuje metody pro získání jednotlivých vrcholů, sklonu a expozice trojúhelníku.

Obrázek 1: Ukázka aplikace se vstupní množinou bodů, zdroj: autoři.

Metody skriptu *mainform.py* :

- **setupUi**: inicializuje a nastavuje různé prvky uživatelského rozhraní, jako např. velikost okna, hlavní widget, menu, lištu nástrojů a akce (např. otevření souboru, ukončení aplikace, vykreslení polygonu, analýza pozice bodů a polygonů apod.).
- **retranslateUi**: nastavení textů pro různé prvky uživatelského rozhraní.

Metody skriptu *draw.py* :

- **loadData:** Načítá data ze vstupního souboru a vrací seznam bodů s jejich souřadnicemi po přepočtu na zadanou šířku a výšku okna aplikace.
- **getAspectColor:** Vrací barvu na základě směru svahu (aspectu).
- **paintEvent:** Vykresluje body, trojúhelníky a barvy na základě sklonu nebo směru svahu (aspectu) pomocí objektu třídy QPainter.

Metody skriptu *algorithms.py* :

- **get2LinesAngle**: Vypočítá úhel mezi dvěma přímkami zadanými 4 body.
- **getPointLinePosition**: Určí, zda bod leží nalevo nebo napravo od úsečky definované dvěma body.
- **getDelaunayPoint**: Najde optimální Delaunayho bod (tj. bod uvnitř kružnice opsané nad trojúhelníkem).
- **getNearestPoint**: Najde nejbližší bod z množiny bodů k danému bodu p.
- **updateAEL**: Aktualizuje seznam hran AEL (Active Edge List).
- **createDT**: Vytvoří Delaunayho triangulaci dané množiny bodů.

- **getContourLinePoint**: Spočítá průsečík úsečky s horizontální rovinou o zadané výšce z.
- **createContourLines**: Vytvoří konturové linie v zadaném intervalu a kroku. Prochází všechny trojúhelníky a testuje průsečíky rovin s rovinou zadanou z-intervalem. Pokud se z-interval dotýká hrany trojúhelníku, vypočítá se průsečík této hrany s rovinou. Pokud je z průsečíku dělitelný pěti beze zbytku, jedná se o výraznou konturovou linii.
- **getNormVector**: Vypočítá normálový vektor trojúhelníka.
- **getSlope**: Vypočítá sklon trojúhelníku.
- **getAspect**: Vypočítá orientaci trojúhelníku.
- **analyzeDTMSlope**: Prochází všechny trojúhelníky v seznamu a vypočítává sklon. Vytváří nový seznam trojúhelníků, které obsahují informace o sklonu.
- **analyzeDTMAspect**: Prochází všechny trojúhelníky v seznamu a vypočítává orientaci. Vytváří nový seznam trojúhelníků, které obsahují informace o orientaci.

Metody skriptu *edge.py* :

- **getStart**: metoda, která vrací počáteční bod hrany.
- **getEnd**: metoda, která vrací koncový bod hrany.
- **switchOrientation**: metoda, která vytvoří novou hranu s opačnou orientací, tedy vymění počáteční a koncový bod.
- **eq**: metoda, která porovnává dvě hrany a vrací True, pokud jsou stejné (mají stejný počáteční a koncový bod), jinak vrací False

Metody skriptu *QPoint3DF.py* :

- **getZ**: Metoda, která vrací hodnotu souřadnice z objektu QPoint3DF.

Metody skriptu *triangle.py* :

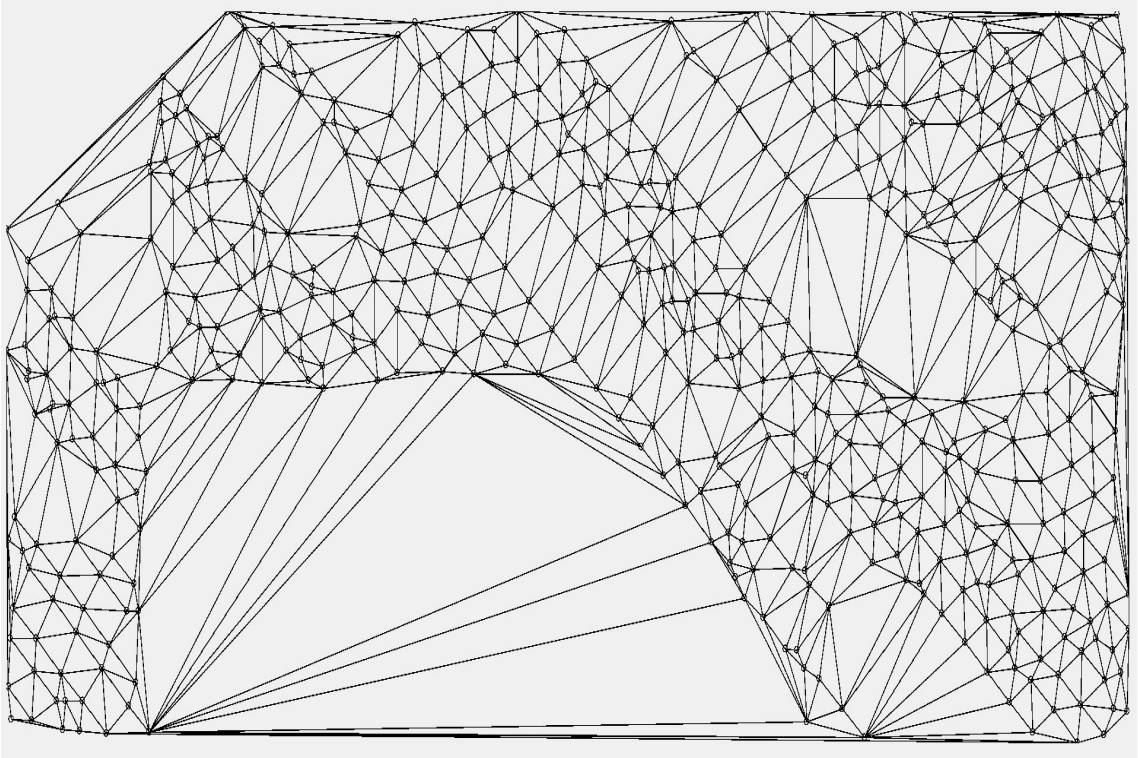
- **getP1**: Vrátí první vrchol trojúhelníku.
- **getP2**: Vrátí druhý vrchol trojúhelníku.
- **getP3**: Vrátí třetí vrchol trojúhelníku.
- **getSlope**: Vrátí sklon trojúhelníku.
- **getAspect**: Vrátí orientaci trojúhelníku.

7 Výsledky

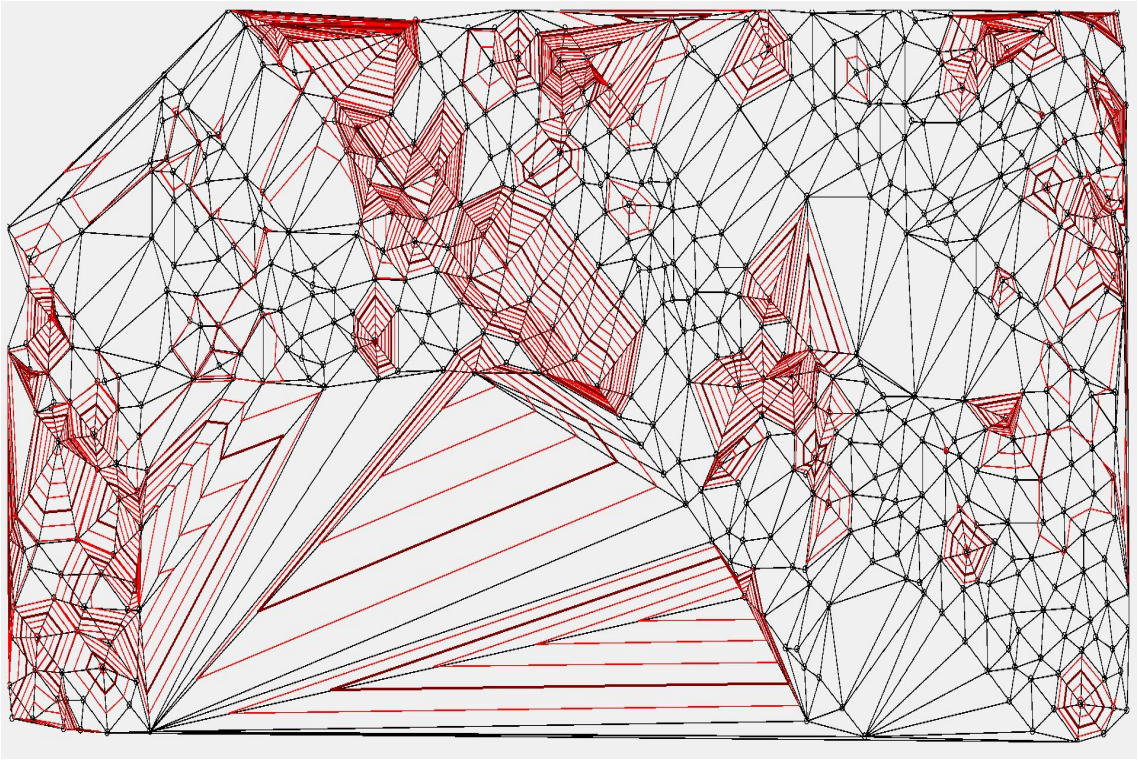
Prvním nezbytným krokem bylo ze vstupní množiny bodů vytvořit Delaunayho triangulaci , neboť se od tohoto procesu odvíjí další postup. Vykreslení triangulace (Obr. 2) lze na první pohled subjektivně hodnotit jako zdařilé. Dalším krokem bylo pro vykreslení vrstevnic (Obr. 3). Před samotným vykreslením byly do aplikace zadány tři nezbytné parametry - minimální a maximální hodnota intervalu nadmořské výšky bodů, která je zde 0 m, resp. 400 m, a velikost rozestupu mezi vrstevnicemi, jež je zde nastavena na 20 m.

Z vykreslených vrstevnic lze usuzovat, že vstupní bodová množina je velice rozmanitá. Lepší představu o členitosti terénu můžeme získat pomocí analýzy sklonu (Obr. 4), resp. jeho velikosti, a analýzou orientace sklonu (Obr. 5). Z analýzy velikosti sklonu lze odvodit, že se střed severní

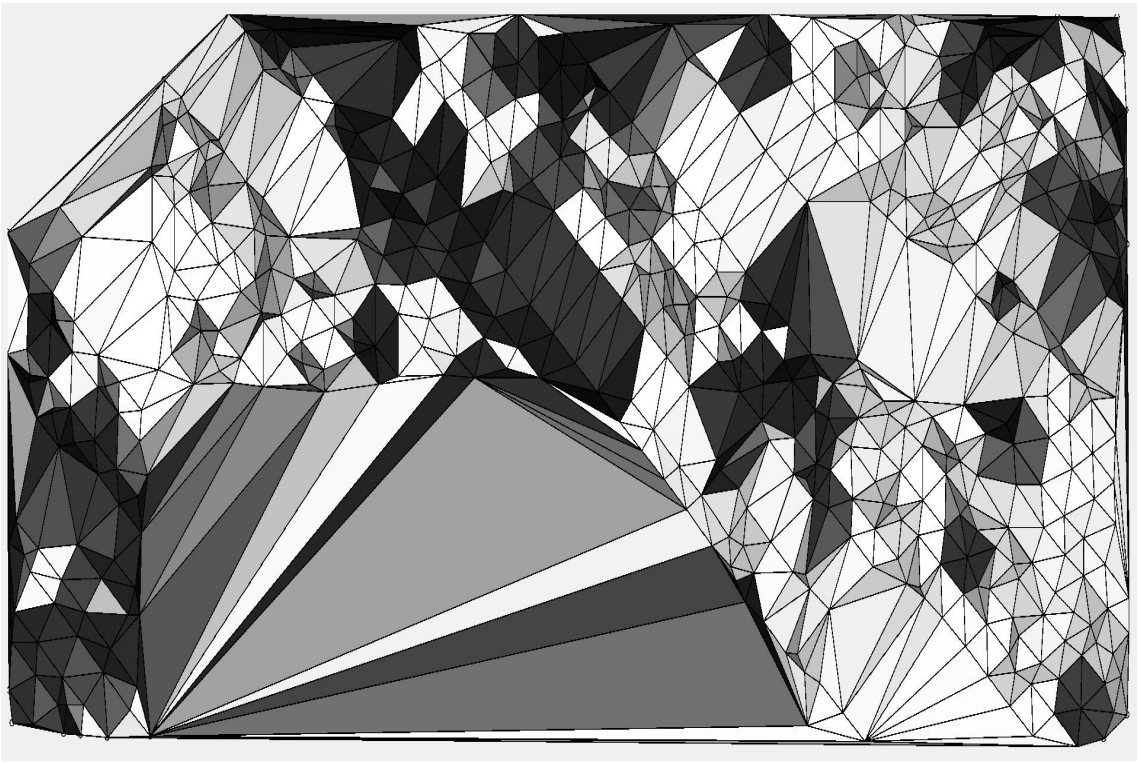
a jihozápadní části obsahuje především skloněnější terén, zatímco skoro celá východní část obsahuje především terén rovinatější.Orientace sklonu bývá často velice náročná na grafické vyjádření. V tomto případě byl azimut rozdělen dle světových stran do osmi intervalů a těm byly přiřazeny náhodné barvy (Obr. 6).



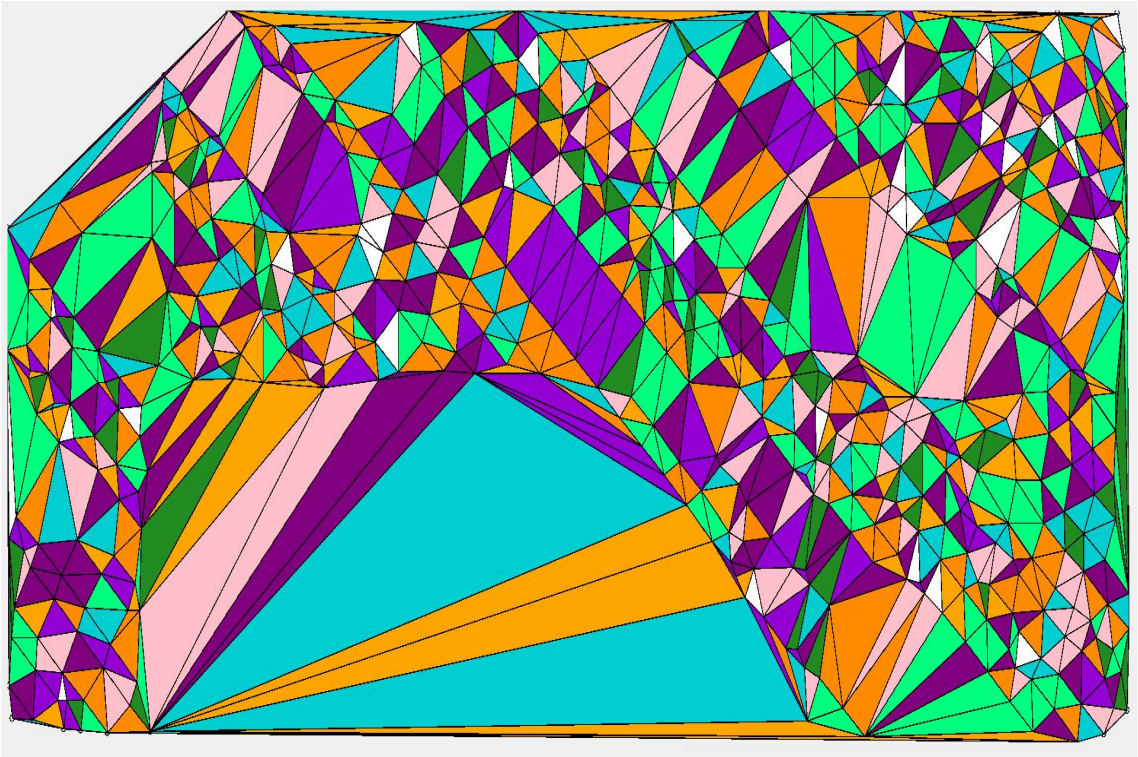
Obrázek 2: Delaunayho triangulace nad vstupní množinou bodů, zdroj: autoři.



Obrázek 3: Vykreslení vrstevnic, zdroj: autoři.



Obrázek 4: Analýza sklonu svahů, zdroj: autoři.



Obrázek 5: Analýza orientace svahů, zdroj: autoři.

```
def getAspectColor(self, aspect):
    ###Get aspect color###
    #Define color ranges for each direction
    color_ranges = [
        (11 / 8 * pi, 13 / 8 * pi, QColor(255, 165, 0)),  # North
        (13 / 8 * pi, 15 / 8 * pi, QColor(148, 0, 211)),  # Northeast
        (0, pi / 8, QColor(34, 139, 34)),  # East
        (pi / 8, 3 / 8 * pi, QColor(255, 192, 203)),  # Southeast
        (3 / 8 * pi, 5 / 8 * pi, QColor(0, 206, 209)),  # South
        (5 / 8 * pi, 7 / 8 * pi, QColor(255, 140, 0)),  # Southwest
        (7 / 8 * pi, 9 / 8 * pi, QColor(0, 255, 127)),  # West
        (9 / 8 * pi, 11 / 8 * pi, QColor(128, 0, 128))  # Northwest
    ]
```

Obrázek 6: Intervaly azimutů a jejich RGB kódy, zdroj: autoři.

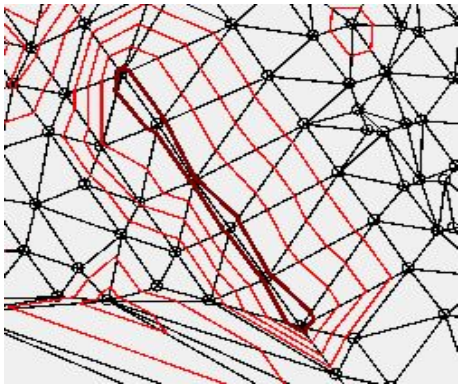
Problematickým místem správného provedení analýz polyedrického modelu může být špatná orientace vrcholů hřebenů nebo údolí. V ukázkových datech je patrná přítomnost hřebene, kde dochází ke snižování nadmořské výšky ve 2 navzájem protilehlých směrech (Obr. 7). Při provedení analýzy orientace (Obr. 8) je patrné, že v průběhu svahu je orientace zachována a že u svahu na druhé straně hřebenu je správně zobrazena opačná orientace svahu.

Taktéž samotné vykreslení vrstevnic se může jevit jako nedostatečné. Z laického pohledu se můžou tyto vrstevnice jevit jako "nevyhlazené", což lze přičítat na vrub problematice a nedostatečnosti vstupních dat a taktéž i samotnému interpolačnímu algoritmu.

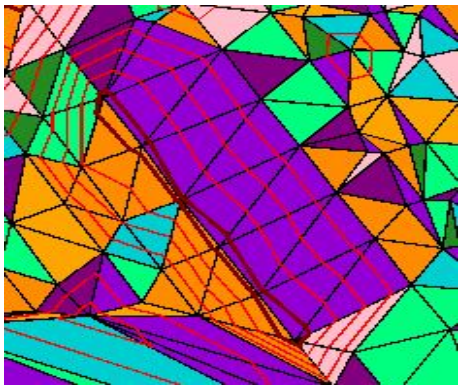
Dalším problémem v případě vrstevnic může být v obecné rovině také vstupní zobrazovaná oblast. Jestliže budeme mít v zobrazovaném území části s nepatrnými změnami nadmořské výšky (roviny, plošiny) a zároveň části s relativně velkými změnami nadmořské výšky (srázy, prudké hřebeny) bude problém tyto rozdílné části území adekvátně zachytit zároveň, resp. nastavit vhodný interval rozchodu vrstevnic. V programu je sice možnost nastavení velikosti rozchodu

vrstevnic, ale není zde možnost využití nadstavbových prvků, jako např. doplňkové či pomocné vrstevnice, jenž by dokázali tato potenciálně problémová místa mnohem lépe adekvátněji znázornit.

Zmiňme také, že program neobsahuje možnost zavedení zdůrazněných, resp. zvýrazněných vrstevnic, jenž by taktéž pomohly k mnohem lepšímu znázornění a popsání terénu.



Obrázek 7: Hřeben, interval vrstevnic: 50 m, zdroj: autoři.



Obrázek 8: Orientace svahů hřebenu, zdroj: autoři.

8 Závěr

Cílem této práce bylo vygenerovat digitální model terénu na základě vstupní množiny bodů spolu s funkcemi, které by za pomocí lineární metody vytvořily vrstevnice a umožnily analyzovat sklon a orientaci trojúhelníků v modelu. Lze konstatovat, že cíl práce byl splněn, neboť zde vytvořený a popsaný program výše uvedené v rámci možností úspěšně a bez závad splňuje.

Program by neměl obsahovat žádné stěžejní nedostatky. Obecně však lze říci, že v případě využívání analýzy polydrických modelů nastává problém špatné orientace vrcholů hřebenů či údolí, což se ale v našich ukázkových datech nepotvrdilo. Možnosti dalšího vylepšení programu lze spatřit v doplnění zbývajících bonusových úloh. Samostatným případem k diskuzi jsou pak alternativní možnosti grafického vyjádření orientace sklonu.

9 Zdroje

Přednášky z předmětu *Algoritmy z počítačové kartografie.*

TRIMBLE (2019). Tutorial: Working With LiDAR Point Cloud Files. eCognition Support Center. Dostupné z:https://support.ecognition.com/hc/en-us/articles/360018797220-Tutorial-Working-With-LiDAR-point-cloud-Files