Univerzita Karlova
Přírodovědecká fakulta

ALGORITMY POČÍTAČOVÉ KARTOGRAFIE

Digitální model terénu

Martina Pavlová, Martin Šíma, Ludmila Vítková
1 N-GKDPZ
Praha 2024

Zadání

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 trojúhelníků 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, proveďte 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 slabinami 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.

Hodnocení

Krok	**hodnocení**
Delaunay triangulace, polyedrický model terénu.	10b
Konstrukce vrstevnic, analýza sklonu a expozice	10b
Triangulace nekonvexní oblasti zadané polygonem	*+5b*
Výběr barevných stupnic při vizualizaci sklonu a expozice	*+3b*
Automatický popis vrstevnic	*+3b*
Automatický popis vrstevnic respektující kartografické zásady (orientace, vhodné rozložení)	*+10b*
Algoritmus pro automatické generování terénních tvarů (kupa, údolí, spočinek, hřbet, ...)	*+10b*
3D vizualizace terénu s využitím promítání	*+10b*
Barevná hypsometrie	*+5b*
Max celkem:	**65b**

V rámci této úlohy byla zpracována bonusová úloha triangulace nekonvexní oblasti.

1 Popis a rozbor problému

Digitální model terénu je matematické zobrazení povrchu země, kde je pro libovolný bod modelu možné určit jeho nadmořskou výšku. Jsou v něm zahrnuty i terénní rysy, kterými jsou například kopce, údolí a další topografické prvky. Jako podkladová data mohou být použita data získaná pomocí LIDARu nebo fotogrammetrie. Ke zpřesnění modelů lze použít satelitní snímky nebo letecké fotografie. Digitální model terénu lze využít pro vizualizaci a analýzu zemského povrchu ve 3D například pro plánování staveb silnic a mostů, k ochraně přírody nebo pro modelování povodní a půdní eroze.

Často používanou metodou tvorby digitálního modelu terénu je TIN neboli *Triangular Irregular Network*. Tyto modely jsou vytvářeny za pomoci sítě trojúhelníků, které vzniknou na základě vstupní bodové vrstvy. Pro každý bod počítá TIN nadmořskou výšku za použití interpolace výšek okolních bodů. Výška bodu je tedy odhadnuta přesněji na rozdíl od modelů, které například využívají pravidelné sítě nebo vrstevnice. V oblastech s dynamičtějšími změnami lze vytvořit více trojúhelníků různých velikostí a tím pádem dojde k lepšímu přizpůsobení nepravidelnému tvaru terénu. Mimo informace o nadmořské výšce zachycuje tento model i informace o sklonu terénu.

1.1 Delaunay triangulace

Jednou z nejčastěji používaných metod triangulace je Delaunay triangulace. Tuto triangulace lze tvořit nejen ve 2D, ale i ve 3D prostoru. V rámci této metody jsou vytvořeny trojúhelníky tak, aby byly co nejvíce rovnostranné. Tím dochází k minimalizaci případné deformaci trojúhelníků. Snaží se tedy maximalizovat minimální vnitřní úhel, čímž vzniká pravidelnější síť. Jednou z jejích vlastní je, že uvnitř kružnice opsané libovolnému trojúhelníku neleží žádný jiný bod za zadané vstupní množiny P.

Do vytvořené Dalaunay triangulace se postupně ukládají jednotlivé body, přičemž nejprve je ze vstupní množiny vybrán náhodný bod P_1. Na základě Eukleidovské vzdálenosti dojde k určení bodu P_2 jemu nejbližšímu. Z těchto dvou bodů vznikne hrana $e = (P_1, P_2)$. Dalším krokem je hledání bodu $\underline{P}$, který se vůči vzniklé hraně e nachází v levé polorovině. Tento bod zároveň minimalizuje poloměr kružnice opsané hraně e a tomuto bodu. Po nalezení tohoto bodu dochází ke vzniku dvou nových hran $e_2 = (\underline{P}, P_1)$ a $e_3 = (P_2, \underline{P})$. Vniklé hrany tvoří trojúhelník. V případě, že bod $\underline{P}$ nebyl nalezen, dojde k otočení orientace hrany e a následuje opět hledání bodu v levé polorovině.

Všechny nově vytvořené hrany jsou následně přidány do *Active Edges List (AEL)* a tvoří výslednou triangulaci. AEL obsahuje hrany e pro které jsou hledány body $\underline{P}$. Po vyprázdnění tohoto seznamu dojde k vytvoření Delaunay triangulace.

Implementace Delaunay triangulace

Algorithm 1 *Delaunay triangulace*

Inicializuj prázdný seznam dt
Inicializuj prázdný seznamael
Najdi bod P_1 s nejmenší x-ovou souřadnicí
Najdi bod P_2, který je nejblíže bodu P_1
Vytvoř z bodů P_1 a P_2 hranu e
Vytvoř opačnou hranu e_op z bodů P_2 a P_1
Přidej hranu e do ael
Přidej hranu e_op do ael
Dokud není ael prázdná
    Vezmi první hranu $e1$ a otoč její orientaci
    Najdi k této hraně Delaunayovský bod p_dt
    Pokud existuje pq_dt
        Vytvoř hranu $e2$ z bodů e2_op a p_dt
        Vytvoř hranu $e3$ z bodů pq_dt a $e1$
        Vzniklé hrany přidej do dt
        Aktualizuj ael
Vrať dt

1.2 Konstrukce vrstevnic

Pomocí lineární interpolace lze konstruovat vrstevnice tak, že jsou lineárními funkcemi prokládány křivky. Pokud máme zadané souřadnice dvou bodů x_a a y_a, pak je lineární interpolací přímka mezi těmito dvěma body. Rovnici vzájemných vztahů můžeme odvodit z podobnosti trojúhelníků.

$$x_a = \frac{x_3 - x_1}{z_3 - z_1}(z - z_1) + x_1 \qquad x_b = \frac{x_2 - x_1}{z_2 - z_1}(z - z_1) + x_1$$

$$y_a = \frac{y_3 - y_1}{z_3 - z_1}(z - z_1) + y_1 \qquad y_b = \frac{y_2 - y_1}{z_2 - z_1}(z - z_1) + y_1$$

Principem této metody je hledání průsečnic roviny určené trojúhelníkem z Delaunay triangulace a vodorovné roviny ρ, která má výšku h. Pomocí následující nerovnice lze určit, zda rovina ρ prochází hranou tvořenou zadanými body.

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

Implementace

Algorithm 2 *Vrstevnice*

```
Inicializuj prázdný seznam contours
Projdi všechny trojúhelníky v dt
    Získej vrcholy trojúhelníku P1, P2, P3
    Získej z-ovou souřadnici vrcholů P1, P2, P3
    Projdi všechny hodnoty od zmin do zmax s krokem dz
        Vypočítej rozdíl mezi aktuální hodnotou z a z-ovými souřadnicemi vrcholů trojúhelníku
        Pokud jsou všechny vrcholy na stejné z-ové úrovni
            Pokračuj
        Pokud jsou dvě hrany kolineární
            Přidej trojúhelník do seznamu vrstevnic
        Pokud jsou hrany trojúhleníku protínány rovinou dané z-ové souřadnice
            Vypočítej průsečíky
            Vytvoř nové hrany
            Přidej hranu do contours
Vrať contours
```

1.3 Analýza sklonu terénu

Výpočet sklonu je proveden pro každý trojúhelník Delaunay triangulace. Pro rovinu ρ, jejíž obecná rovina vypadá následovně:

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

je vypočítán gradient $\nabla\rho$, neboli maximální vektor spádu. Tento gradient má v daném bodě směr normály k vrstevnici a zároveň je orientován ve směru funkce p.

$$\nabla\rho(x_0, y_0, z_0) = \left(\frac{\partial p}{\partial x}(x_0), \frac{\partial p}{\partial y}(y_0), \frac{\partial p}{\partial z}(z_0)\right) = (a, b, c)$$

V případě, že máme vodorovnou rovinu π, pak mají roviny ρ a π normálové vektory n_1, n_2. Pro rovinu π uvažujeme s jednotkovým vektorem

$$n_1 = (a, b, c), \quad n_2 = (0, 0, 1)$$

Odchylku φ od rovin ρ a π lze vypočítat ze vztahu:

$$\varphi = arccos\frac{n_1 \cdot n_2}{\|n_1\|\|n_2\|} = arccos\frac{c}{\|n_1\|}$$

Implementace

Algorithm 3 *Sklon terénu*

1: Vypočítej rozdíl x-ových souřadnic bodů P_1 a P_2
2: Vypočítej rozdíl y-ových souřadnic bodů P_1 a P_2
3: Vypočítej rozdíl z-ových souřadnic bodů P_1 a P_2
4: Vypočítej rozdíl x-ových souřadnic bodů P_1 a P_2
5: Vypočítej rozdíl y-ových souřadnic bodů P_1 a P_2
6: Vypočítej rozdíl z-ových souřadnic bodů P_1 a P_2
7: Vypočítej x-ovou souřadnici normálového vektoru trojúhelníku
8: Vypočítej y-ovou souřadnici normálového vektoru trojúhelníku
9: Vypočítej z-ovou souřadnici normálového vektoru trojúhelníku
10: Vypočítej normu normálového vektoru
11: Vrať hodnotu úhlu mezi normálovým vektorem a osou z

1.4 Analýza orientace terénu

Expozici neboli orientaci terénu lze definovat jako azimut průmětu gradientu $\nabla\rho$ do roviny x, y. Vznikne tedy vektor $\vec{v}$ s nulovou složkou z.

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

Azimut tohoto vektoru je měřen od osy y a lze vypočítat pomocí vztahu:

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

Implementace

Algorithm 4 *Orientace terénu*

1: Vypočítej rozdíl x-ových souřadnic bodů P_1 a P_2
2: Vypočítej rozdíl y-ových souřadnic bodů P_1 a P_2
3: Vypočítej rozdíl z-ových souřadnic bodů P_1 a P_2
4: Vypočítej rozdíl x-ových souřadnic bodů P_1 a P_2
5: Vypočítej rozdíl y-ových souřadnic bodů P_1 a P_2
6: Vypočítej rozdíl z-ových souřadnic bodů P_1 a P_2
7: Vypočítej x-ovou souřadnici normálového vektoru trojúhelníku
8: Vypočítej y-ovou souřadnici normálového vektoru trojúhelníku
9: Vypočítej normu normálového vektoru
10: Vrať hodnotu úhlu mezi normálovým vektorem a osou x

1.5 Triangulace nekonvexní oblasti

Při triangulace nekonvexní oblasti je nejdřive provedena obyčejná triangulace, která vytvoří triangulaci uvnitř konvexní obálky všech bodů. V dalším kroku je tato triangulace upravena pomocí funkce *clipDT*, kde je pro každý trojúhelník spočítán centroid. Pokud tento centroid leží uvnitř nekonvexní oblasti, příslušný trojúhelník je vyřazen z triangulace. Poloha bodu vůči polygonu je určována pomocí Ray Crossing algoritmu.

2 Struktura programu

Program se skládá z 10 souborů, kterými jsou *Data*, *Images*, *Algorithms*, *Draw*, *Edge*, *load*, *MainForm*, *QPoint3DF*, *Settings* a *Triangle*.

Ve složce *Data* se nacházejí vstupná data, která se skládají z 5 různých souborů. Dva testovací soubory (test_2.txt, text_3.txt) jsou uměle vytvořené. Zbylé soubory pocházejí z reálného bodového mračna (zámková dlažba v Hřensku). Testovací soubor č. 8 (Test_8.json) zobrazuje oblast Krušných hor a Českého středohoří (ČUZK, 2024). Všechny tyto datasety jsou ve formátu JSON. Ve vstupních datech je klíč *POINTS*, který obsahuje seznam bodů, které mají souřadnice *[x, y, z]*. Druhým klíčem je pak *BORDER*, ve kterém jsou body tvořící hranici. Soubor také může obsahovat klíč *HOLE*, který obsahuje body díry. Body triangulace uvnitř díry budou vyjmuty z výsledné triangulace. Pokud soubor neobsahuje klíč BORDER, program ho při výpočtech nahradí konvexní obálkou všech bodů (pomocí algoritmu Jarvis Scan). V případě absence klíče HOLE, je jako díra použito prázdné pole. Hranici a díru obsahuje testovací soubor test_9.json. Další možností, jak přidávat body do již načteného bodového mračna je klikáním. Takto vložené body však mají náhodnou nadmořskou výšku. body však mají náhodnou nadmořskou výšku. Vkládání hranice triangulace a díry lze provést také manuálně pomocí tlačítek Create border a Create hole. Body jsou vkládány do výsledných polygonů (hranice, díra) postupně, tak jak jsou vkládány na vstupu. Lze vložit pouze jednu díru, která by měla být topologicky korektní.

Ve složce *images* se nachází celkem 12 ikon, které slouží k vytvoření aplikace. Těmito ikonami jsou *clear.png*, *clear_all.png*, *contours2.png*, *exit.png*, *open_file.png*, *orientation2.png*, *save.png*, *settings.png*, *slope2.png* a *triangles2.png*, *Hole.png* a *Border.png*.

V *algorithms.py* je definována třída *Algorithms*, která obsahuje 15 metod. Pomocí *getPointLinePosition* dochází k určení polohy bodu p vzhledem k přímce určené body p_1 a p_2. Pokud se bod nachází vlevo, je vrácena hodnota 1, pokud se bod nachází vpravo, je vrácena hodnota 0 a pokud se bod nachází na přímce, je vrácena hodnota -1. Metoda *getNearestPoint* vypočítává vzdálenost každého bodu k bodu q a následně vrací bod s nejmenší vzdáleností, tedy ten, který je k bodu q nejblíže. Další metodou je *getTwoLineAngle*, která vypočítá pomocí skalárního součinu úhel mezi dvěma liniemi definovanými čtyřmi body. Metoda *getDelaunayPoin*t vyhledává bod, který maximalizuje úhel mezi dvěma přímkami a zároveň leží vlevo od zadané hrany. Metoda *updateAE*L aktualizuje seznam aktivních hran (Active Edge List). Pomocí metody *createDT* dochází k vytvoření Delaunay triangulace pro danou množinu bodů. Další metodou je *getContourPoint*, která bere dva body definující hranu trojúhelníku a z-ovou souřadnici roviny, následně vypočítá průsečík této hrany s rovinou na základě lineární interpolace, a nakonec vrací tento průsečík. Metoda *createContourLines* generuje vrstevnicové čáry pro zadanou množinu bodů pomocí Delaunayovy triangulace a zadaných intervalů vrstevnic na základě výšek. Pro každý trojúhelník v triangulaci vypočítává průsečíky s rovinami vrstevnic a přidává výsledné hrany do seznamu vrstevnicových čar. Pomocí metody *computeSlope* dochází k výpočtu sklonu trojúhelníku a pomocí metody *computeAspect* dochází k výpočtu orientace. Metoda *analyzeDTMSlope* vytváří seznam trojúhelníků s informacemi o sklonu a vrací ho a stejně tak metoda *analyzeCTMAspec*t vytváří seznam trojúhelníků s informacemi o aspektu a vrací ho. Další metodou je *rayCrossingAlgorithm*, která implementuje Ray

Crossing algoritmus pro učení toho, zda bod q leží uvnitř nebo vně polygonu. Tato metoda přijímá bod q a polygon *pol* a vrací 1, pokud se bod nachází uvnitř polygonu. Pokud se bod nenachází uvnitř polygonu, vrací hodnotu 0. Předposlední metodou je *clipDT*, která provádí oříznutí Delaunayho triangulace na základě polygonu. Pro každý trojúhelník v triangulaci se vypočítá jeho centroid a zkontroluje se, zda leží uvnitř nebo vně polygonu pomocí Ray Crossing Algorithm. Trojúhelníky, jejichž centroidy jsou uvnitř polygonu, jsou přidány do seznamu oříznuté triangulace, který je nakonec vrácen. Pokud se ve vstupním souboru nenachází body hranice polygonu (klíč "BORDER"), je jako hranice určena konvexní obálka bodů. Pro vytvoření konvexní obálky lze použít metodu *jarvisScan*.

V *draw.py* se nachází třída *Draw*, ve které je definovaných celkem skoro 20 různých metod. Metoda *clearAll* slouží k vymazání všech dat a metoda *clearResults* slouží k vymazání výsledků analýzy DMT. Metoda *getPoint*s vrátí seznam bodů. Metoda *getBorder* vrátí hranici. Metoda *setBorder* nastaví hranici na zadaný seznam bodů. Metoda *getDT* vrátí Delaunay triangulaci. Metoda *setDT* nastaví Delaunay triangulaci na zadaný seznam hran. Metoda *getDTMSlope* vrátí seznam trojúhelníků s informacemi o sklonu. Metoda *setDTMSlope* nastaví seznam trojúhelníků s informacemi o sklonu. Metoda *getDTMAspect* vrátí seznam trojúhelníků s informacemi o aspektu. Metoda *setDTMAspect* nastaví seznam trojúhelníků s informacemi o aspektu. Metoda *setContours* nastaví seznam hran vrstevnic. K nastavení parametrů pro zobrazení slouží metody *setViewDT*, *setViewContourLine*, *setViewSlope* a *setViewAspect*. Metoda *mousePressEvent* je zavolána při stisknutí tlačítka myši, přičemž získává souřadnice $[x, y]$ a generuje náhodnou výšku bodu v rozmezí *zmin* a *zmax*. Následně dojde k vytvoření nového bodu p s těmito souřadnicemi, který je přidán do seznamu bodů points. Poslední metodou je *paintEvent*, která je volána v případě, kdy je potřeba překreslit plátno.

Soubor *Edge.py* obsahuje implemntaci třídy *Edge*, která reprezentuje hranu Delaunay triangulace. Třída má atributy *start* a *end,* v kterých jsou uloženy objekty třídy *QPoint3DF*, reprezentující koncové body hrany. Dále jsou ve třídě implementovány gettery pro přístup k jednotlivým položkám (*getStart*, *getEnd*) a metoda pro změnu orientace hrany (*switchOrientation*). Tato třída má také přetížený operátor == pro porovnávání (*__eq__*).

Základem skriptu *MainForm.py* je automaticky vygenerovaný kód reprezentující uživatelské rozhraní. Dále je skript doplněn o metody obsluhující základní funkcionality programu. Metoda *openClick* otevře File Dialog pro výběr vstupního souboru. Dále se zde nachází metody vytvářející samotné vrstvy, které vzešly z jednotlivých analýz (*createContourClick*, *analyzeSlopeClick*, *analyzeAspectClick*). Mazání bodů, popř. analýz je prováděno pomocí metod *clearClick* a *clearAllClick*. U jednotlivých vrstev lze měnit viditelnost pomocí metod *viewContourClick*, *viewSlopeClick* a *viewExpositionClick*. Poslední metodou je *setParameter*, který otevře dialogové okno, ve kterém je možné zvolit základní interval vrstevnic a hodnoty minimální a maximální vykreslené vrstevnice.

Soubor *Settings.py* obsahuje automaticky vygenerovaný kód popisující uživatelské rozhraní dialogového okna pro volbu parametrů vykreslovaných vrstevnic.

Soubor *Triangle.py* inplementuje třídu *Triangle*. Objekt této třídy reprezentuje jeden trojúhelník Delaunay triangulace. Třída má atributy *vertices*, *slope* a *exposition*. Atribut *vertices* je objekt třídy *QPolygonF* a uchovává vrcholy trojúhelníku (objekt třídy *QPoint3DF*). Atributy *slope* a *exposition*

uchovávají vlastnosti trojúhelníku (sklon a expozice). Třída dále obsahuje gettery k jednotlivým atributům (*getVertices*, *getAspect*, *getSlope*).

Soubor *QPoint3DF.py* obsahuje implementaci třídy QPoint3DF reprezentující bod v trojrozměrném prostoru. Tato třída dědí z objektu *QPointF*. V této třídě je také implementován atribut z uchovávající nadmořskou výšku a příslušný getter (*getZ*). Dále je v této třídě přetížen operátor ==.

Soubor *Load.py* obsahuje tři metody obsluhující načítání dat. Samotné načítání bodů je umožněno pomocí metody *loadPoints*, v prvním kroku je ze všech bodů vypočten minmax box, z kterého je vypočten posun na jednotlivých osách a škálovací faktor. Body jsou následně transformovány pomocí funkce *transformPoints*. Kontrola vstupních dat probíhá pomocí funkce *isNumber*.

Výstupy jednotlivých analýz jsou vykreslovány ve vrstvách. Od nejsvrchnějších vrstev to jsou digitální model terénu, vrstevnice, sklon terénu a orientace. Digitální model terénu a vrstevnice jsou vykresleny pouze jako liniové prvky. Sklon a orientace jsou polygonové vrstvy, tudíž po provedení všech analýz a zobrazení všech vrstev není vrstva orientace vidět.

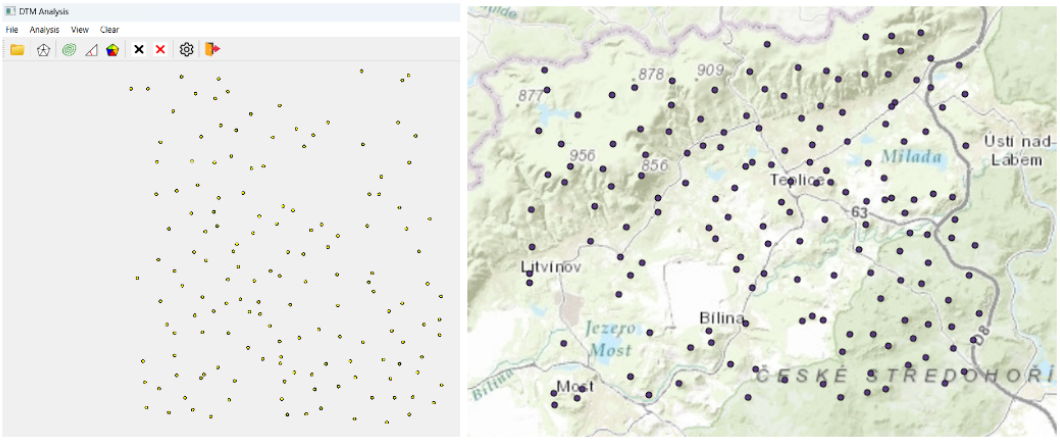
3 Triangulace nekonvexní oblasti

Při triangulace nekonvexní oblasti je nejdříve provedena obyčejná triangulace, která vytvoří triangulaci uvnitř konvexní obálky všech bodů. V dalším kroku je tato triangulace upravena pomocí funkce clipDT. Postupně jsou procházeny všechny trojúhelníky triangulace. U každého trojúhelníku bylo zkontrolováno, zda je celý uvnitř hranic, popřípadě vně díry. Pokud tyto podmínky nebyly splněny, trojúhelník byl vyřazen z triangulace. U každého trojúhelníku musí být všechny body uvnitř hranice (border) a vně díry (hole), toto je určeno pomocí Ray Crossing algoritmu. Dále se všechny hrany trojúhelníku nesmí jakkoliv protínat hranici, popřípadě obsahovat díru (metoda intersectionTwoLines). Díky metodě triangulace nekonvexní oblasti může za určitých podmínek nastat situace, kdy je bod uvnitř hranice a vně díry, ale i tak nejsou zařazeny do triangulace. Toto je způsobeno tím, že všechny trojúhelníky, které obsahují tento bod nebyly zařazeny do triangulace.

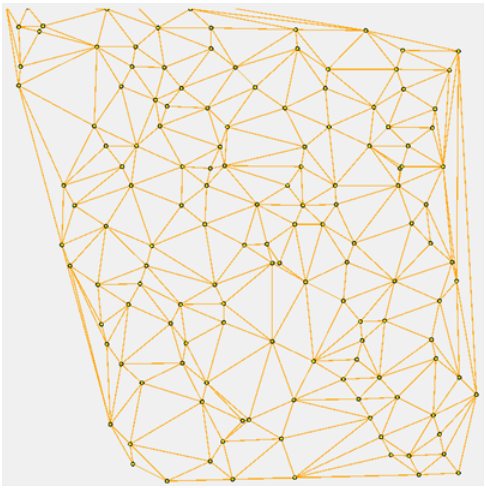
4 Výsledky

V rámci této úlohy došlo k vytvoření uživatelského rozhraní s využitím frameworku QT, ve kterém lze vytvořit digitální model terénu s využitím algoritmů popsaných v teoretické části této zprávy. Toto grafické rozhraní aplikace vytvořené v prostředí Qt Designer lze vidět na obrázku 1 a dále bylo upravováno v prostředí programovacího jazyka Python. Po spuštění aplikace může uživatel otevřít soubor obsahující bodové mračno.

Prvním krokem při tvorbě digitálního modelu terénu (DMT) a jeho analýz je vytvoření Delaunay triangulace, jejíž výsledek lze vidět na obrázku 2.

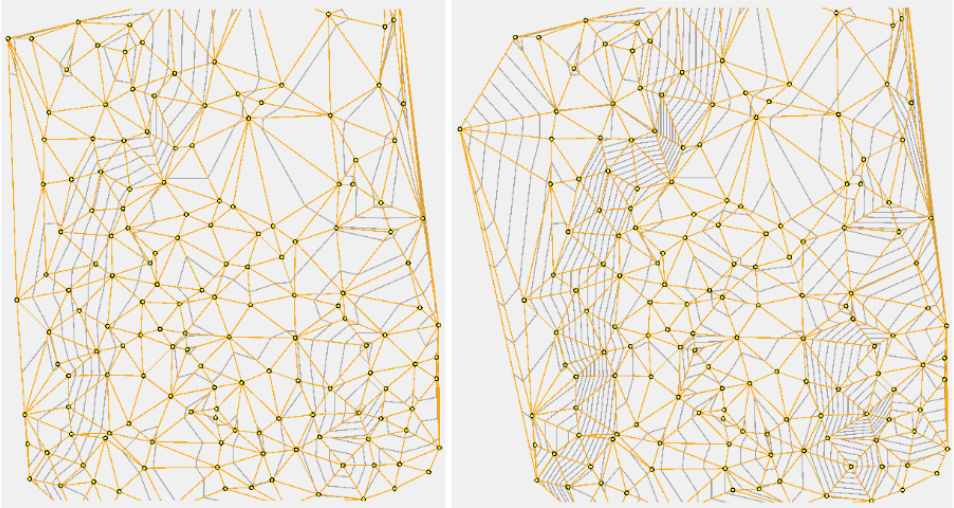


Obrázek 1: Grafické rozhraní aplikace po načtení bodového mračna (vlevo), stejné body na mapě (vpravo)



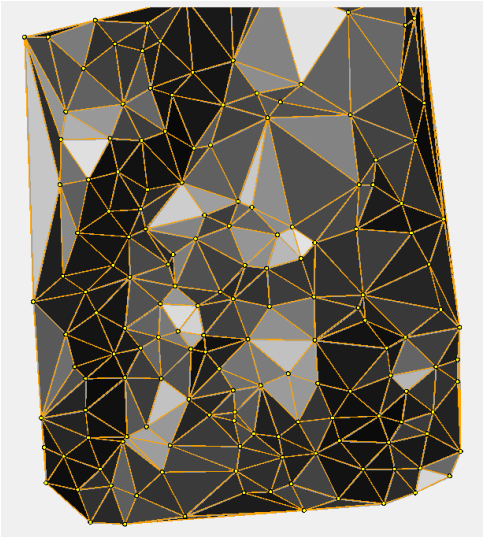
Obrázek 2: Výsledek Delaunay triangulace

Z důvodu různé výškové členitosti bodového mračna je potřeba nastavit parametry vykreslování vrstevnic. Defaultní základní interval vrstevnic (ZIV) je 100 m a vrstevnice jsou vykreslovány v rozsahu 0 – 1 610 m. Toto nastavení závisí na výškových poměrech ve zkoumané oblasti. V případě datasetu č. 8 (*test_8.json*) se jedná o členitou oblast Krušných hor (vlevo), Teplic a Českého Středohoří (vpravo) (obr. 1 vpravo), tudíž správná volba ZIV závisí na účelu (obr. 3). Pokud by se jednalo a oblast s nízkými výškovými rozdíly, ZIV by se muselo správně zvolit. Příliš velký velký interval (např. defaultních 100 m) by v tomto případě nevykreslil žádné vrstevnice. Naopak příliš malý ZIV by zvýraznil malé výškové rozdíly způsobené šumem, v takovém případě je potřeba vrstevnice dostatečně generalizovat. U těchto datasetů s malými výškovými rozdíly není není možné znázornit sklon dostatečně detailně, jelikož je zde malý sklon povrchu (*test_2.json*, *test_3.json*) Výsledné vrstevnice při použití dvou zmíněných intervalů lze vidět na obrázku 3.



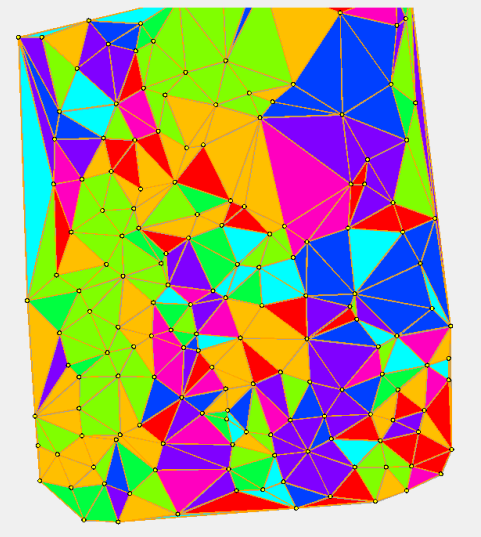
Obrázek 3: Různá nastavení základního intervalu vrstevnic, vlevo 100 m, vpravo 50 m

Další analyzovanou vlastností námi vytvořeného digitálního modelu terénu byl sklon jednotlivých trojúhelníků, který je znázorněn odstíny šedi. Světlé plochy značí rovinaté oblasti, zatímco tmavé plochy reprezentují strmé oblasti. Výsledek sklonu lze vidět na obrázku 4.

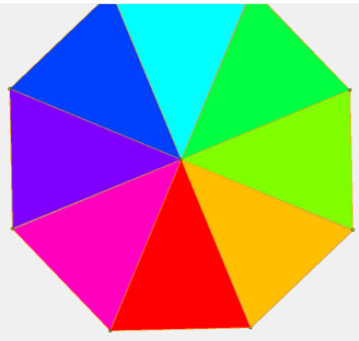


Obrázek 4: Výsledek analýzy sklonu

Poslední analyzovanou vlastností digitálního modelu terénu byla orientace vůči světovým stranám, kterou lze vidět na obrázku 5. Pro vizualizaci orientace vůči světovým stranám byla využita barevná stupnice, kterou lze vidět na obrázku 6, kde tyrkysová značí sever, limetková východ, červená jih a fialová západ.

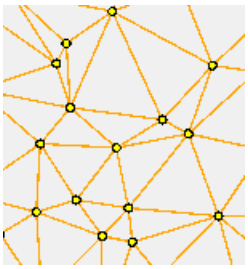


Obrázek 5: Výsledek analýzy orientace vůči světovým stranám

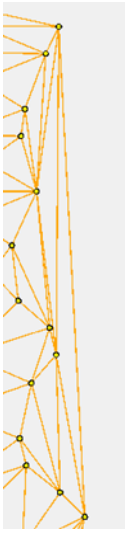


Obrázek 6: Barevná stupnice použitá pro vizualizaci orientace

Posledním krokem v rámci této úlohy bylo zhodnocení úspěšnosti aplikace. Výsledkem Delaunay triangulace by měly být pokud možno tvarově ideální trojúhelníky. Pokud se podíváme na trojúhelníky vytvořené uprostřed bodového mračna, jako například na obrázku 7, lze vidět, že vytvořené trojúhelníky mají víceméně ideální tvar. Naopak u výsledných krajních trojúhelníků je na první pohled vidět, že mají neideální tvar. Tyto trojúhelníky lze vidět na obrázku 8, kde můžeme pozorovat jejich příliš ostré či naopak příliš tupé úhly.

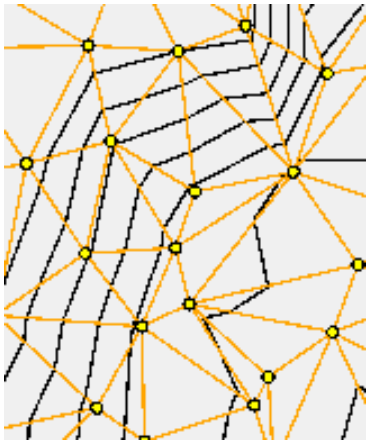


Obrázek 7: Trojúhelníky uprostřed bodového mračna



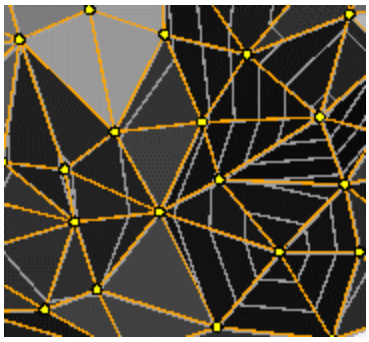
Obrázek 8: Trojúhelníky na okraji bodového mračna

Vrstevnice byly vytvořeny pomocí lineární interpolace a proto úplně stoprocentně neodpovídají skutečnosti. Jak lze vidět na obrázku 9, na některých míst vznikají lomené čáry, které ve skutečnosti mají být spojité křivky jdoucí do oblouků. V námi vytvořené aplikaci také nejspíše nedojde k vykreslení velmi jemných detailů v terénu.

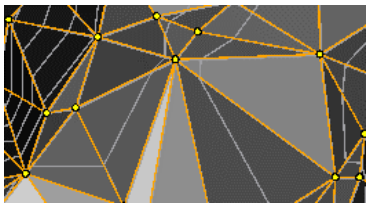


Obrázek 9: Detail výsledných vrstevnic

Při detailnější analýze výsledného sklonu lze konstatovat, že výsledek odpovídá skutečnosti. V oblastech většího množství vrstevnic (jako tomu je například na obrázku 10) je barva sklonu tmavší, zatímco v oblastech, kde se nachází velmi málo vrstevnic nebo dokonce žádné (jako na obrázku 11) je sklon znázorněn světlou barvou.

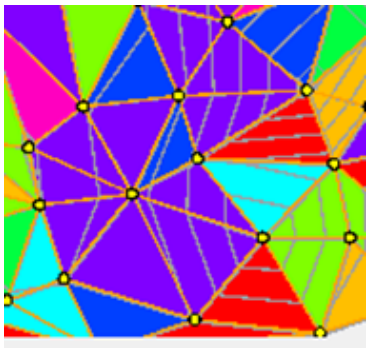


Obrázek 10: Výsledný sklon ve strmější oblasti



Obrázek 11: Výsledný sklon v rovinatější oblasti

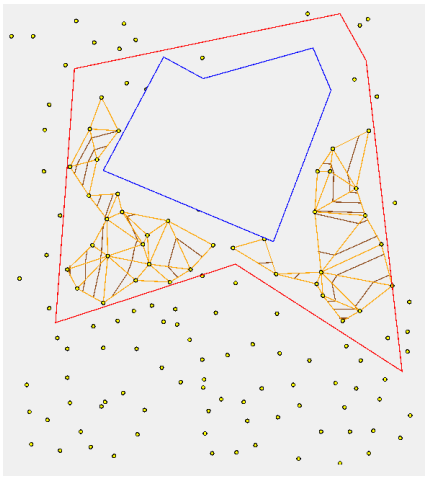
Výsledná vizualizace orientace terénu vůči světovým stranám je detailní vzhledem k tomu, že došlo k použití osmi světových stran místo čtyř. Základní směry (sever, východ, jih, západ) jsou tedy doplněny o severovýchod, jihovýchod, jihozápad a severozápad. Na obrázku 12 lze vidět detailní výsledek analýzy orientace.



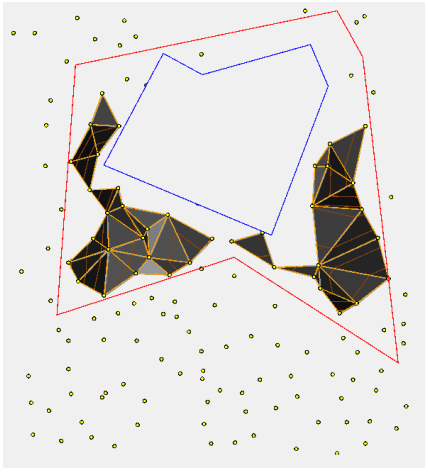
Obrázek 12: Detail výsledné orientace vůči světovým stranám

Při použití algoritmu 2D Delaunay triangulace mohou nastat problémy ve znázorňování určitých částí terénu. Obecně se hůře bude znázorňovat velmi členitý terénu s jemnými detaily, jelikož by ke správné reprezentaci terénu bylo potřeba velké množství bodů. Při nedostatečném množství bodů dochází ke vzniku příliš velkých trojúhelníků a tedy k zanedbání malých terénních útvarů. Z důvodu malého množství bodů se například u testovacího souboru č. 8 nejsou znázorněné detailní terénní tvary. Například povrchový důl Bílina není vůbec zobrazen nebo Krušných hor vypadají jako stolová hora. Vrcholy Českého středohoří vypadají jako pyramidy, také z důvodu lineární interpolace vrstevnic (obr. 10). Další sproblém může nastat při znázorňování hřbetů, údolí, říčních sítí či obecně všech liniových prvků. Pokud jsou body rozmístěny po zájmové oblasti nehomogenně, může interpolace vyústit v nepřesnou reprezentaci strmých svahů.

Na obrázku 13 lze v levé části vidět výsledek triangulace nekonvexní oblasti. Napravo je pak vidět výsledek analýzy sklonu v území. Nekonvexní oblast je ohraničena červenou čarou, přičemž vnitřní ohraničení oblasti je znázorněno modrou čarou.



Obrázek 13: Triangulace nekonvexní oblasti



Obrázek 14: Analýza sklonu u nekonvexní oblasti

5 Závěr

V rámci této úlohy došlo k vytvoření aplikace, ve které lze z bodového mračna vytvořit digitální model terénu (DMT). Nad vytvořeným modelem lze vytvořit vrstevnice nebo provádět analýzy jako je sklon a orientace terénu vůči světovým stranám. V této úloze bylo dále popsáno zhodnocení úspěšnosti aplikace, přičemž bylo konstatováno, že aplikace funguje správně. Výpočet vrstevnic byl proveden pomocí lineární interpolace, což by znamenalo, že by se hodnoty nadmořské výšky v celém území měnily lineárně. To se však v přírodě neděje, avšak tento výpočet je nejjednodušší. Detailnější analýzu by bylo možné provést v případě, že by bylo k dispozici mnohem větší množství bodů. Došlo by tak ke znázornění i menších terénním útvarů.

6 Zdroje

CZUK (2024): Stahovací služba WFS - Bodová pole.
https://ags.cuzk.cz/arcgis/services/BodovaPole/MapServer/WFSServer? (30. 5. 2024).