Úkol č. 3: Digitálny model terénu a jeho analýzy

Hugo Majer, Júlia Šušková

18. júna 2022

1 Zadanie

Úloha č. 3: Digitální model terénu.

Vstup: množina $P = \{p_1, ..., p_n\}$, $p_i = \{x_i, y_i, z_i\}$.

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.

2 Údaje o bonusových úlohách

Neboli riešené žiadne bonusové úlohy.

3 Polyedrický digitálny model terénu, triangulácia

Digitálny model terénu (DMT) je digitálna reprezentácia zemského povrchu v pamäti počítača, ktorá je zložená z nameraných dát a interpolačného algoritmu, ktorý umožňuje odvodzovať výšky medziľahlých bodov. V ľubovoľnom bode modelu je možné odvodiť nadmorskú výšku.
Jedným z typov DMT je polyedrický DMT nazývaný TIN (Triangulated Irregular Network - Nepravidelná sieť trojuholníkov), ktorý reprezentuje povrch pomocou nepravidelných trojuholníkov, čím poskytuje plne definovaný a spojitý model terénu. Trojuholníková sieť je vytvorená trianguláciou za použitia triangulačného algoritmu. Algoritmus prekladá trojicou vrcholov rovinu v R^3, vzniká tak nepravidelný mnohosten (polyedr), ktorý sa primyká k terénu.

3.1 Triangulácia

Trianguláciou nad zadanou množinou bodov $P = \{P_1, P_2, ..., P_n\}$ rozumieme planárne rozdelenie roviny na m trojuholníkov $T = \{T_1, T_2, ..., T_m\}$ pričom platí:

- Ľubovoľné dva trojuholníky T_i, $T_j \in T$ majú spoločnú nanajvýš hranu alebo vrchol.
- Zjednotenie trojuholníkov je súvislá množina v 2D.
- Vo vnútri žiadneho trojuholníku neleží žiadny ďalší bod z P.

Väčšina metod tvorby triangulácie generuje čo najviac rovnostranné trojuholníky. V takomto prípade každý trojuholník by mal čo najlepšie lokálne reprezentovať povrch, tj. najlepšie sa k nemu primykať.

3.2 Delaunay triangulácia

Bola vyvinutá Borisom Delaunaym v roku 1934 a dnes sa jedná o najpoužívanejšiu metódu triangulácie, obzvlášť v oblasti GIS. Delaunay triangulácia DT spočíva v tom, že vo vnútri opísanej kružnice ku ľubovoľnému trojuholníku $T \in DT$ neleží žiadny iný bod zo vstupnej množiny. Je nutné podotknúť, že DT nemusí byť jednoznačná a môže nastať viacero riešení a to práve vtedy, ak na opísanej kružnici sa nachádza štvrtý bod. Cieľom DT je vytvoriť také trojuholníky, aby najmenší uhol v každom trojuholníku bol maximálny avšak pre túto metódu neplatí, že by sa minimalizoval maximálny uhol v trojuholníku. Výsledné trojuholníky sa svojim tvarom blížia rovnostranným. Jej hranicu predstavuje konvexná obálka.

4 Delaunayho triangulácia inkrementálnou konštrukciou

Myšlienkou je vytvoriť Delaunay trianguláciu DT vkladaním jedného bodu po druhom. Zachováva sa doposiaľ vytvorená DT a po nájdení ďalšieho bodu sa triangulácia jednoducho aktualizuje. Algoritmus začína incializáciou, ktorá môže predstavovať výber náhodného bodu zo vstupnej množiny bodov alebo výberom bodu s minimálnou súradnicou x (varianta použitá v rámci tejto práce), označme ho P_1. K tomuto bodu nájdeme najbližší bod P_2 zo vstupnej množiny bodov, tj. bod s najmenšou Euklidovou vzdialenosťou, čím vytvoríme prvú hranu $e = (P_1, P_2)$, ktorá je orientovaná.

Následne hľadáme Delaunayho bod $\underline{P}$, ktorý leží v ľavej polorovine σ_L voči hrane e a ktorý minimalizuje polomer r kružnice k opísanej hrane e a tomuto bodu:

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

Polohu bodu P_i voči hrane $e = (P_1, P_2)$ určíme za pomoci vektorového súčinu smerového vektoru hrany e (označme $\vec{u}$) a vektoru definovaným bodom P_i a bodom P_1 (označme $\vec{v}$).
Stanovme, že $P_i = [x_p, y_p]$, $P_1 = [x_1, y_1]$ a $P_2 = [x_2, y_2]$, potom:

$$\vec{u} = (x_2 - x_1, y_2 - y_1)$$
$$\vec{v} = (x_p - x_1, y_p - y_1)$$

Vektorový súčin môžeme zapísať maticovo a vypočítať pomocou determinantu, ktorého znamienko určí, v ktorej polrovine voči hrane e sa bod P nachádza:

$$t = \begin{vmatrix} u_x & u_y \\ v_x & v_y \end{vmatrix} = u_x * v_y - v_x * u_y \Rightarrow \begin{cases} t > 0, & P(e) \in \sigma_L \\ t = 0, & P \in p \text{ (kolinearita)} \\ t < 0, & P(e) \in \sigma_R \end{cases}$$

Je nutné dodať, že korektný Delaunayho bod $\underline{P}$ nemusí nutne generovať kružnicu $k = (e, P_i)$ s minimálnym polomerom r, ale uprednostňuje sa vlastnosť, kedy stred kružnice S leží v inej polrovine voči hrane e ako bod P_i, tj. v pravej polrovine, $S(e) \in \sigma_R$. Preferencia sa v takomto prípade zabezpečí negáciou polomeru r:

$$r' = \begin{cases} -r, & S(e) \in \sigma_R \\ r, & S(e) \in \sigma_L \end{cases}$$

Z tohto vyplýva, že je nutné testovať, v ktorej polrovine S voči hrane e leží. Za týmto účelom musíme spočítať, kde leží stred S kružnice $k = (e, P_i)$ a jej polomer r.

Stanovme, že kružnica k so stredom $S = [m, n]$ je daná trojicou bodov $P_1 = [x_1, y_1]$, $P_2 = [x_2, y_2]$, $P_3 = [x_3, y_3]$, potom:

$$m = 0,5 \cdot \frac{k_{12}(-k_4) + k_{11}k_5 - (k_{10} + k_4k_5)k_6}{x_3(-k_4) + x_2k_5 + x_1(-k_6)}$$

$$n = 0,5 \cdot \frac{k_1(-k_9) + k_2k_8 + k_3(-k_7)}{y_1(-k_9) + y_2k_8 + y_3(-k_7)}$$

$$r = \sqrt{(x_1 - m)^2 + (y_1 - n)^2},$$

kde

$$k_1 = x_1^2 + y_1^2 \qquad k_5 = y_1 - y_3 \qquad k_9 = x_2 - x_3$$

$$k_2 = x_2^2 + y_2^2 \qquad k_6 = y_2 - y_3 \qquad k_{10} = x_1^2$$

$$k_3 = x_3^2 + y_3^2 \qquad k_7 = x_1 - x_2 \qquad k_{11} = x_2^2$$

$$k_4 = y_1 - y_2 \qquad k_8 = x_1 - x_3 \qquad k_{12} = x_3^2$$

Polohu stredu kružnice S voči hrane e určíme už vyššie uvedeným spôsobom.

Ak vyhovujúci Delaunayho bod $\underline{P}$ nebol nájdený, tak sa otočí orientácia hrany e a vyhľadávanie bodu sa opakuje, opäť v ľavej polrovine ak uvažujeme už novú orientáciu hrany. Po nájdení bodu $\underline{P}$ dostávame prvý trojuholník tvorený troma hranami:

$$e = (P_1, P_2), e_2 = (P_2, \underline{P}), e_3 = (\underline{P}, P_1)$$

$$\triangle(P_1, P_2, \underline{P})$$

Tieto tri hrany pridáme do pomocnej dátovej štruktúry *Active Edge List* (AEL) a zároveň tieto hrany budú tvoriť výslednú trianguláciu DT. Následne sa z AEL vezme (a zmaže) posledná pridaná hrana, prehodí sa jej orientácia a vyhľadá sa pre ňu Delaunayho bod $\underline{P}$. Ak je nájdený, vytvoria sa ďalšie dve nové hrany. Tieto dve hrany, spoločne s tou, ktorá bola vzatá z AEL a preorientovaná, budú tvoriť trianguláciu DT.

Následne je ale nutné rozhodnúť, či tieto dve nové hrany budú pridané aj do AEL. Pri pridávaní hrany $e = (P_i, P_j)$ do AEL dochádza ku kontrole, či AEL už neobsahuje túto hranu ale s opačnou orientáciou, tj. $e' = (P_j, P_i)$. Ak áno, hrana e' je z AEL odstránená. Ak nie, hrana e je pridána do AEL.

Ak pre aktuálnu hranu nebol nájdený Delaunayho bod $\underline{P}$, tak to znamená, že hrana je súčasťou konvexného obalu a je pridaná do výslednej triangulácie DT.

Opísaný postup končí vo chvíli, kedy AEL je prázdne. Vďaka udržiavaniu orientácie hrán v jednom smere na hranici aktuálnej triangulácie uloženej v AEL hľadáme Delaunayho bod $\underline{P}$ vždy naľavo od orientovaných hrán.

Inkrementálna konštrukcia Delaunayho triangulácie sa dá formálne zapísať nasledovne:

Algorithm 1 *Inkrementálna konštrukcia Delaunayho triangulácie*

1: Inicializuj DT a AEL ako prázdne listy: DT, $AEL = [\,]$.
2: Nájdi bod P_1 s minimálnou x súradnicou: $P_1 = \min(X_i)$.
3: Nájdi bod P_2 s minimálnou Euklidovou vzdialenosťou ku P_1: $\|P_2 - P_1\|_2 = \min$.
4: Vytvor prvú hranu $e = (P_1, P_2)$.
5: Nájdi Delaunayho bod $\underline{P}$: $\underline{P} = \arg\min_{\forall P_i(e) \in \sigma_L} \ r'(k_i), \quad k_i = (e, P_i), \ \ P_i(e) \in \sigma_L$
6: Ak $\nexists \underline{P}$, prehoď orientáciu hrany e: $e = (P_2, P_1)$ a zopakuj bod 5.
7: Vytvor zvyšné dve hrany prvého trojuholníku: $e_2 = (P_2, \underline{P})$, $e_3 = (\underline{P}, P_1)$.
8: Pridaj hrany e, e_2, e_3 do AEL: $AEL \leftarrow e, e_2, e_3$.
9: Pridaj hrany e, e_2, e_3 do DT: $DT \leftarrow e, e_2, e_3$.
10: Pokým $AEL \neq \varnothing$:
11: Vezmi a zmaž poslednú hranu z AEL: $e(P_a, P_b) = \text{AEL.pop()}$
12: Prehoď jej orientáciu: $e = (P_b, P_a)$.
13: Nájdi Delaunayho bod $\underline{P}$: $\underline{P} = \arg\min_{\forall P_i(e) \in \sigma_L} \ r'(k_i), \quad k_i = (e, P_i), \ \ P_i(e) \in \sigma_L$
14: Ak $\exists \underline{P}$:
15: Vytvor ďalšie dve hrany nového trojuholníku: $e_2 = (P_b, \underline{P}), e_3 = (\underline{P}, P_a)$
16: Pridaj hrany e, e_2, e_3 do DT: $DT \leftarrow e, e_2, e_3$.
17: Prehoď orientáciu hrán e_2, e_3: $e_2 = (\underline{P}, P_b)$, $e_3 = (P_a, \underline{P})$.
18: Ak e_2 alebo $e_3 \in AEL$, zmaž e_2 alebo e_3 z AEL.
19: V inom prípade: Zmeň orientáciu e_2 alebo e_3 a pridaj e_2 alebo e_3 do AEL.

5 Konštrukcia vrstevníc

Vrstevnice boli skonštruované lineárnou interpoláciou, ktorá predpokladá lineárny spád terénu medzi bodmi a vo výsledky aj rozostup vrstevníc je rovnaký, čo len ťažko môže vystihovať reálny povrch. Riešenie je založené na analytickej geometrií, kedy hľadáme priesečnicu roviny trojuholníku T tvoriaceho trianguláciu DT a vodorovnej roviny ρ s výškou h.

Nech máme trojuholník T s hranami e_i, e_{i+1}, e_{i+2} a rovinu vrstevnice ρ o výške z. Medzi hranou trojuholníka e_i a rovinou vrstevnice ρ nás zaujímajú dva vzťahy:

- Hrana e_i náleží rovine ρ, vtedy platí

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

  Ak hrana trojuholníka T náleží ρ, táto hrana je vrstevnicou.

  Ak všetky hrany trojuholníka T náležia ρ, tak pre takýto T nie je nutné riešiť vrstevnice.

- Hrana e_i pretína rovinu ρ, vtedy platí

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

V tomto prípade rovina ρ musí pretnúť dve hrany, vzniknú dva priesečníky. Spojením vypočítaných priesečníkov vznikne vrstevnica. Je teda nutné vypočítať polohu priesečníkov hrany $e_i = (P_1, P_2)$ a roviny vrstevnice ρ o výške z. Ak $P_1 = [x_1, y_1, z_1]$, $P_2 = [x_2, y_2, z_2]$, potom súradnice x, y priesečníku spočítame ako:

$$x = \frac{x_2 - x_1}{z_2 - z_1}(z - z_1) + x_1,$$
$$y = \frac{y_2 - y_1}{z_2 - z_1}(z - z_1) + y_1$$

6 Analýza sklonu

Sklon je uhol φ medzi zvislicou (normálový vektor (0,0,1)) a normálovým vektorom roviny trojuholníku z DT. Rovina trojuholníku $T_i \in DT$ je určená trojzložkovými vektormi $\vec{u} = (u_1, u_2, u_3)$ a $\vec{v} = (v_1, v_2, v_3)$. Nech T_i má vrcholy $P_1 = [x_1, y_1, z_1]$, $P_2 = [x_2, y_2, z_2]$, $P_3 = [x_3, y_3, z_3]$, potom:

$$\vec{u}: \quad u_1 = x_2 - x_1, \qquad u_2 = y_2 - y_1, \qquad u_3 = z_2 - z_1;$$
$$\vec{v}: \quad v_1 = x_3 - x_1, \qquad v_2 = y_3 - y_1, \qquad v_3 = z_3 - z_1.$$

Normálový vektor roviny trojuholníku $n_T = (a, b, c)$ následne spočítame ako $\vec{u} \times \vec{v}$:

$$a = u_2 v_3, \qquad b = -(u_1 v_3 - u_3 * v_1), \qquad c = u_1 v_2 - u_2 v_1.$$

Po spočítaní normy $\|\vec{n_T}\|$ známym vzorcom nasleduje samotný výpočet sklonu

$$\cos\varphi = \left| \frac{c}{\|\vec{n_T}\|} \right|.$$

7 Analýza orientácie

Orientácia je azimut priemetu normálového vektoru roviny trojuholníku n_T do roviny x, y

$$n_T = (a, b, 0),$$

pričom zložky a, b určíme spôsobom uvedením pri výpočte sklonu (kapitola 6). Azimut A vektoru a tým pádom orientáciu trojuholníku napokon určíme zo vzťahu

$$\tan A = \frac{b}{a}.$$

8 Aplikácia

Aplikácia bola vytvorená vo vývojovom prostredí QT 6. Slúži na trianguláciu množiny bodov a vytvorenie polyedrického digitálneho modelu terénu. Implementovaná je aj konštrukcia vrstevníc, analýza sklonu a orientácie.

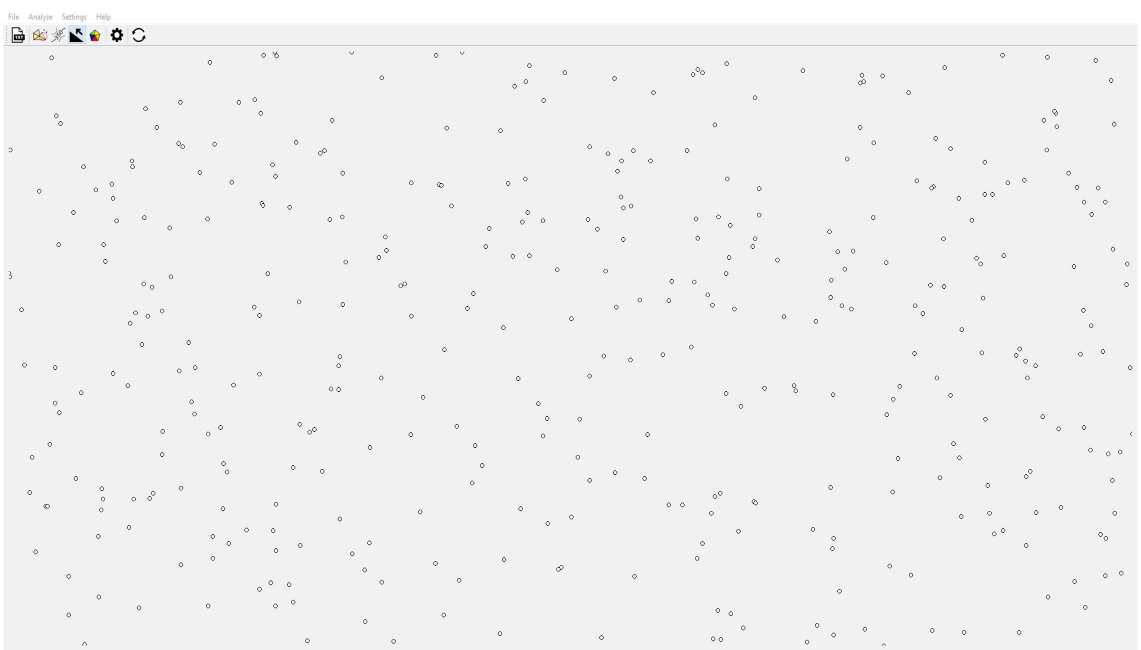
Užívateľské rozhranie je veľmi jednoduché a intuitívne. Prevažnú časť okna aplikácie tvorí zobrazovacia plocha, na ktorej sa vizualizujú do aplikácie nahrané dáta a jednotlivé výsledky. Aplikácia v hornej lište obsahuje menu, pod menu tlačidla rýchlej voľby.

Vstupné dáta užívateľ do aplikácie nahrá cez *File – Open .TXT* alebo tlačidlom *Open .TXT*. Po kliknutí naň vyskočí pop-up okno s možnosťou prehliadať súbory v PC. Je nutné zvoliť TXT súbor súradníc X, Y, Z bodov. Stĺpce súradníc musia byť oddelené TAB alebo stredníkom.

Po nahratí dát užívateľ môže pristúpiť k triangulácií a to cez *Analyze – Create DT* alebo opäť aj tlačidlom *Create DT*. Rovnakou cestou je možné vyvolať aj zvyšné analýzy. Tieto analýzy je možné vykonať aj bez nutnosti (manuálne) spustiť trianguláciu, pretože pri dotaze na analýzu sa triangulácia vytvorí automaticky.

Ďalšou položkou sú *Settings*, v ktorých užívateľ môže nastaviť základný interval generovaných vrstevníc a interval v ktorom sa majú vrstevnice generovať.

Posledné tlačidlo *Refresh* resetuje zobrazovaciu plochu.



Obr. 1: Rozhranie aplikácie s načítanými vstupnými dátami.

9 Zhodnotenie

V prvom rade na testovacích dátach bola porovnaná výsledná triangulácia, sklon a orientácia vykonaná našou aplikáciou s trianguláciou a analýzami vykonanými v ArcGIS. Ak porovnávame samotnú trianguláciu (pozri Obr. 2, Obr. 3), nedá sa povedať, že by boli úplne totožné, ale v niektorých miestach nachádzame totožné výsledky (trojuholníky), čo síce zo samotnej vizualizácie triangulácie nemusí byť zrejmé. Ak porovnávame analýzy sklonu medzi aplikácou a ArcGIS, kde samotné trojuholníky sú vplyvom zafarbenia lepšie rozlíšiteľné, totožné trojuholníky medzi týmito dvoma výstupmi môžeme ľahko pozorovať. Čo sa týka samotnej analýzy sklonu, výsledky medzi aplikáciou a ArcGIS sú až na niektoré výnimky rovnaké (pozri Obr. 4, Obr. 5). Takéto rozdiely môžu byť spôsobené tým, že legenda (intervaly a farby) v aplikácií nie je úplne totožná s tou z ArcGIS. Takisto vrstevnice vygenerované aplikáciou odpovedajú tým z ArcGIS (pozri Obr. 6, Obr. 7). Takisto ak porovnávame orientáciu medzi výstupom z aplikácie a ArcGIS, na mnohých miestach sa výsledky zhodujú, avšak nie na všetkých. Nájdené rozdiely sa nepodarilo nijak zovšeobecniť, na každom mieste je rozdiel v orientácií individuálny (pozri Obr. 8, Obr. 9).

Samotná kvalita triangulácie a analýz bola zhodnotená pre tri terénne tvary a to údolie, hrebeň a kopa.

9.1 Údolie

Údolie bolo nasimulované pomocou vytvorených dát, a to tak, že v strede je rovinná oblasť a smerom ku obidvom krajom nadmorská výška bodov stúpa (rovnomerne na oboch stranách). Pri tomto terénnom tvare vrstevnice sú zobrazené korektne (pozri Obr. 10), takisto sklon (pozri Obr. 11). Čo sa týka samotnej triangulácie, tá je v rovinnej časti údolia nejednoznačná, čo je spôsobené aj tým, že sa jedná o grid. Orientácia v stredovej rovinnej časti je problémová, čo je spôsobené tým, že pri výpočte (a stanovení) orientácie neuvažujeme rovinu (pozri Obr. 12.

9.2 Hrebeň

Hrebeň je tvar podobný údoliu s tým rozdielom, že výšky od stredovej rovinnej oblasti ku okrajom nestúpajú ale klesajú. Tým pádom platia rovnaké závery ako pri údoliu, tj. triangulácia v stredovej rovinnej časti nejednoznačná, vrstevnice a sklon správne, orientácia je problémová. Závery nie je nutné demonštrovať na obrázkoch.

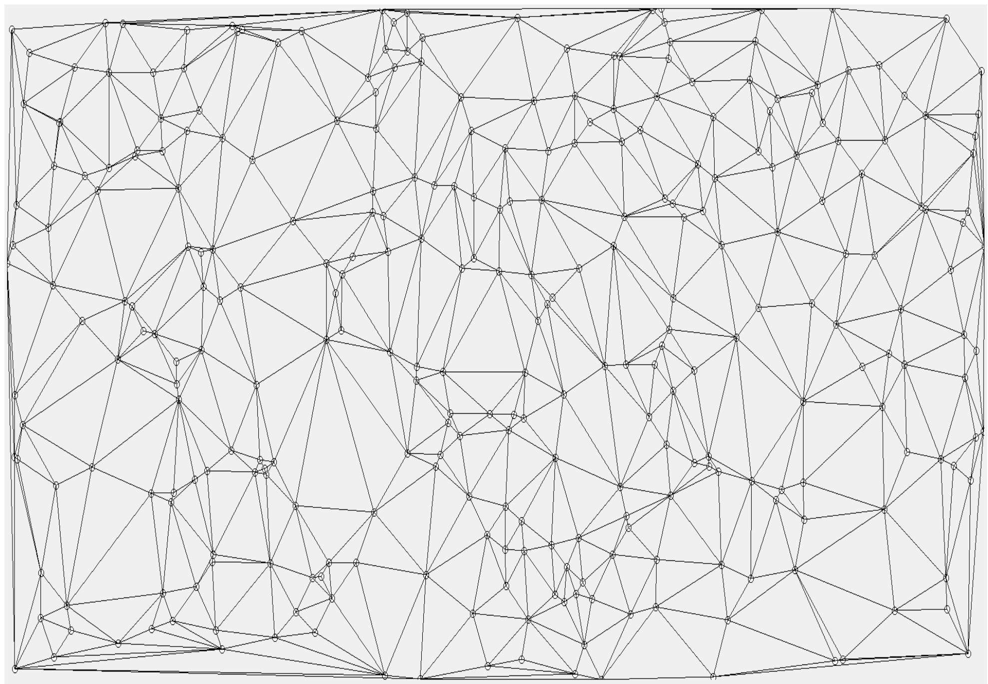
9.3 Kopa

Na demonštráciu výsledkov pre kopu boli zvolené geografické dáta z Lužických hôr. Keďže sa jedná o reálne dáta z prírody, kopa prirodzene nie je úplne tvarovo dokonalá. Triangulácia, vrstevnice, sklon a orientácia vychádza pri tomto terénnom tvare veľmi dobre a nepozorujeme nijaké vážnejšie nedostatky (pozri Obr. 13, Obr. 14, Obr. 15).

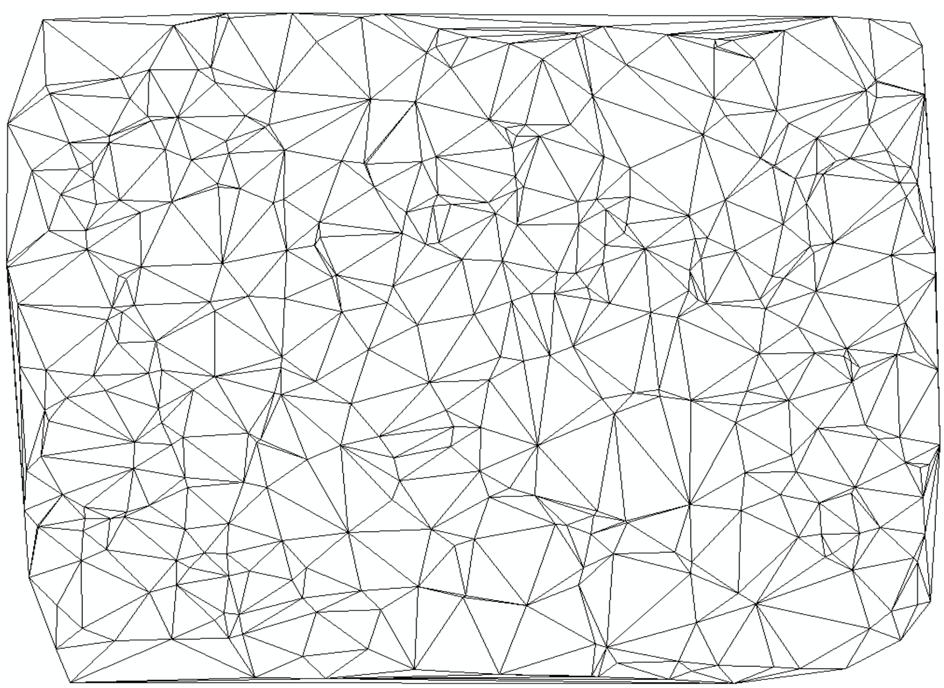
9.4 Terénne hrany a okraj triangulácie

Terénnou hranou rozumieme styk dvoch rôzne sklonených častí terénnej plochy. Hrana spája miesta, kde sa zlomovo mení sklon týchto terénnych plôch. Nakoľko využitá Delaunayho triangulácia zohľadňuje len vzdialenosť bodov, ich výška (súradnica Z) nie je braná do úvahy a tým pádom vo všeobecnosti nevhodne modeluje terénne hrany a ich priebeh. Tento problém rieši *Data Dependent Triangulation*.

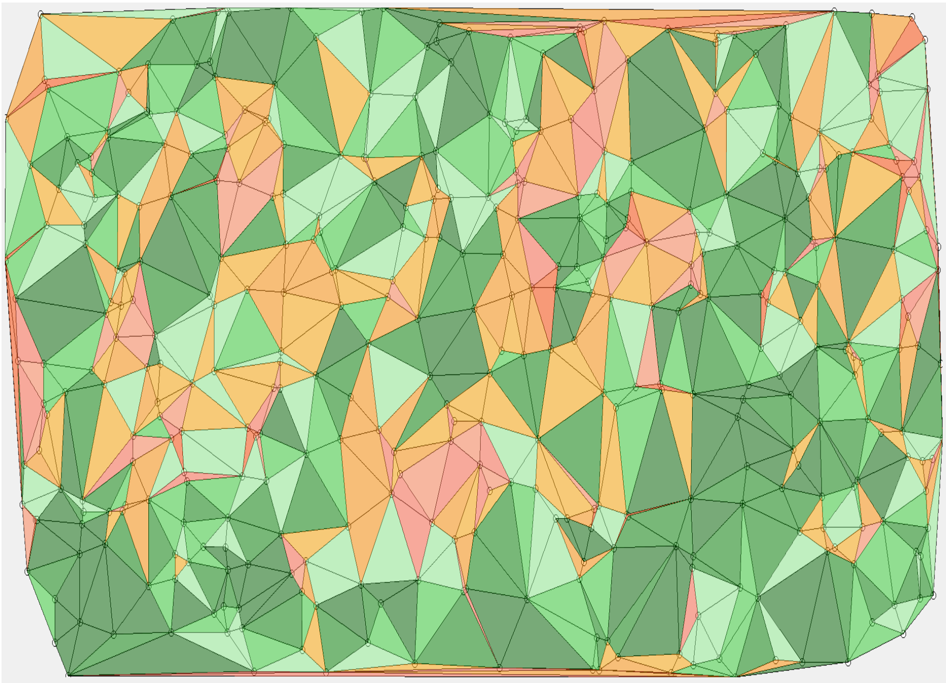
Problémovým miestom z hľadiska aproximácie terénu sú aj okraje triangulácie, ktoré neodzrkadľujú priebeh terénu. Na okrajoch pozorujeme trojuholníky, ktoré sa svojim tvarom neblížia ku rovnostranným trojuholníkom, ale sú "štíhle". Tento problém sa môže riešiť pokročilejším algoritmom, ktorý v týchto okrajových častiach bude vkladať nové body, čím zlepší tvar okrajových trojuholníkov.



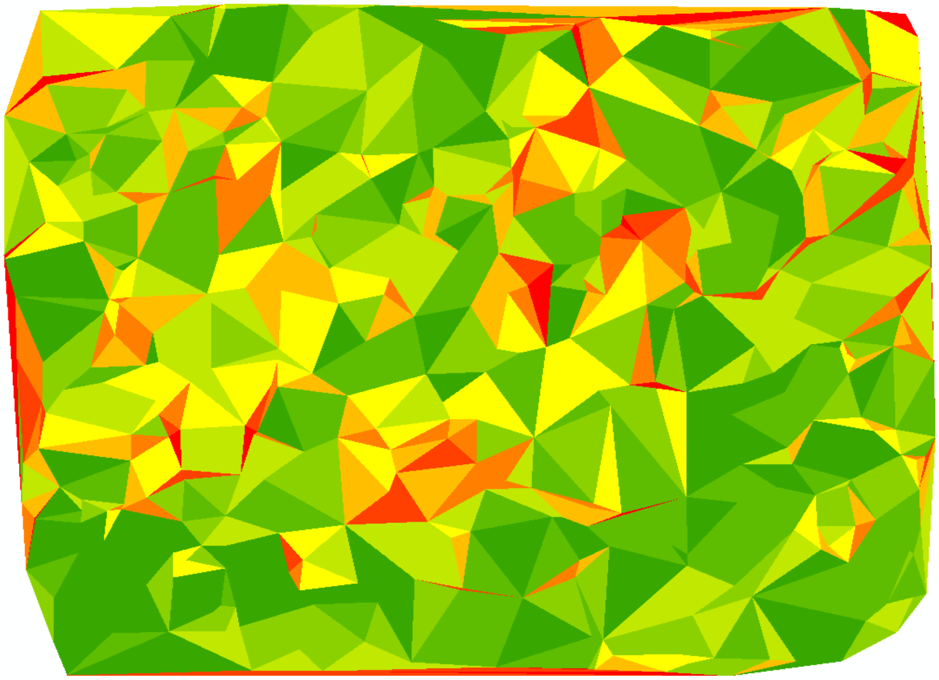
Obr. 2: Triangulácia vytvorenou aplikáciou.



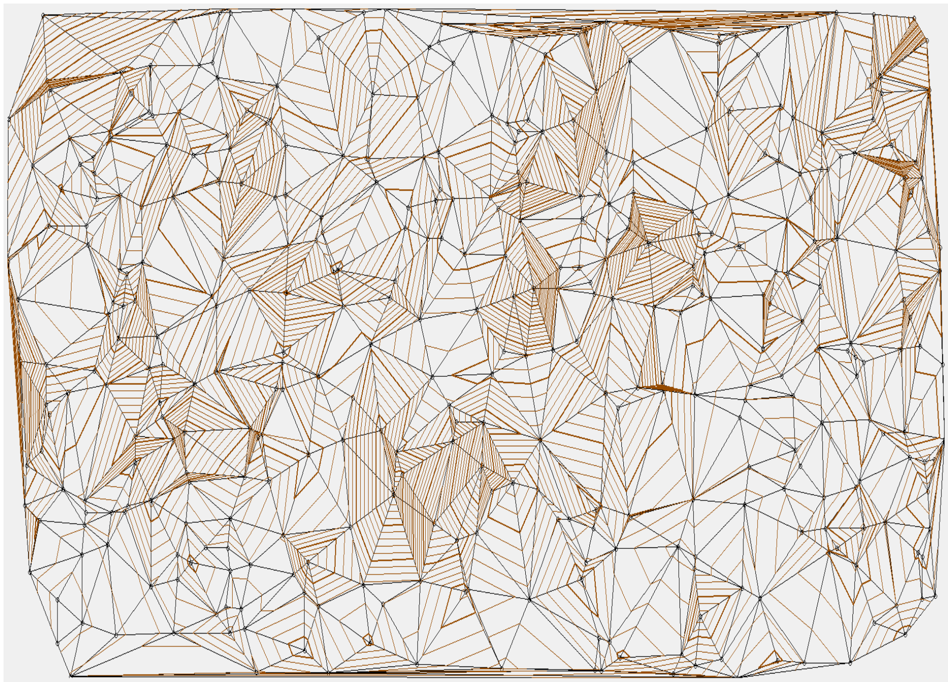
Obr. 3: Triangulácia vykonaná softvérom ArcGIS.



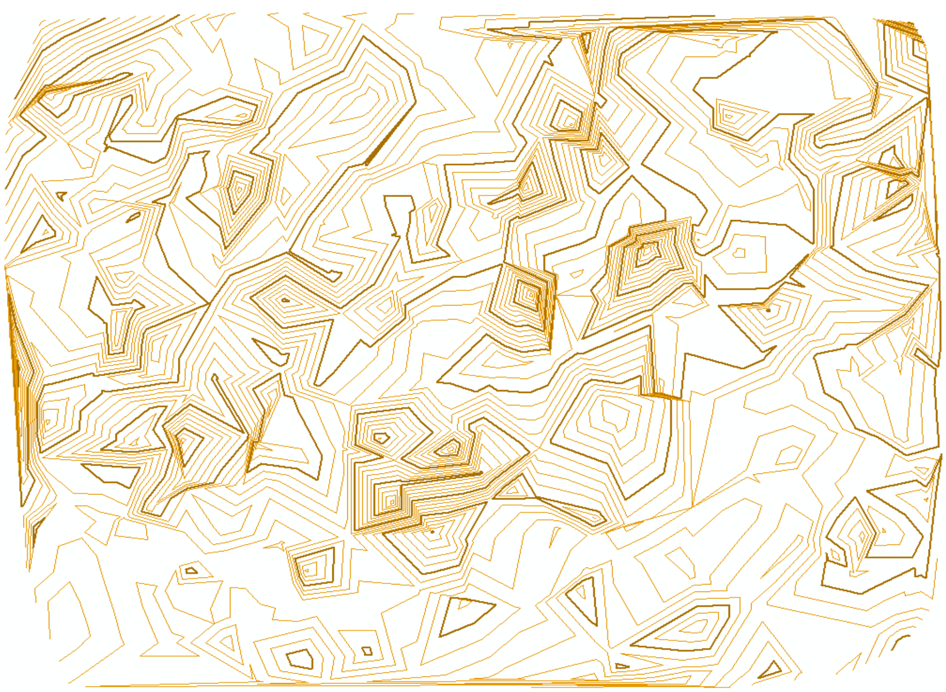
Obr. 4: Analýza sklonu vykonaná aplikáciou.



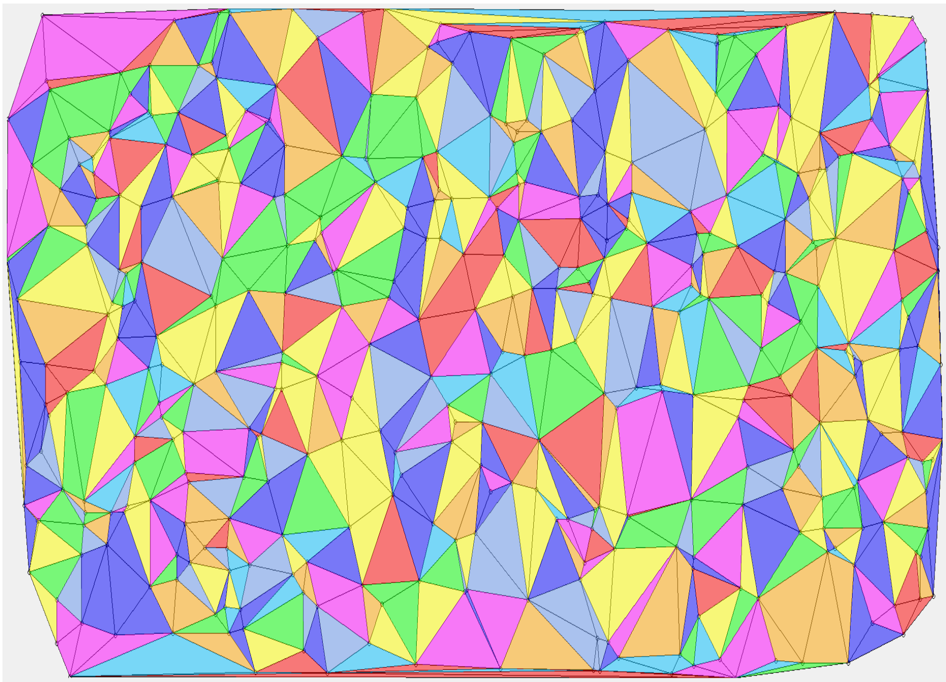
Obr. 5: Analýza sklonu vykonaná softvérom ArcGIS.



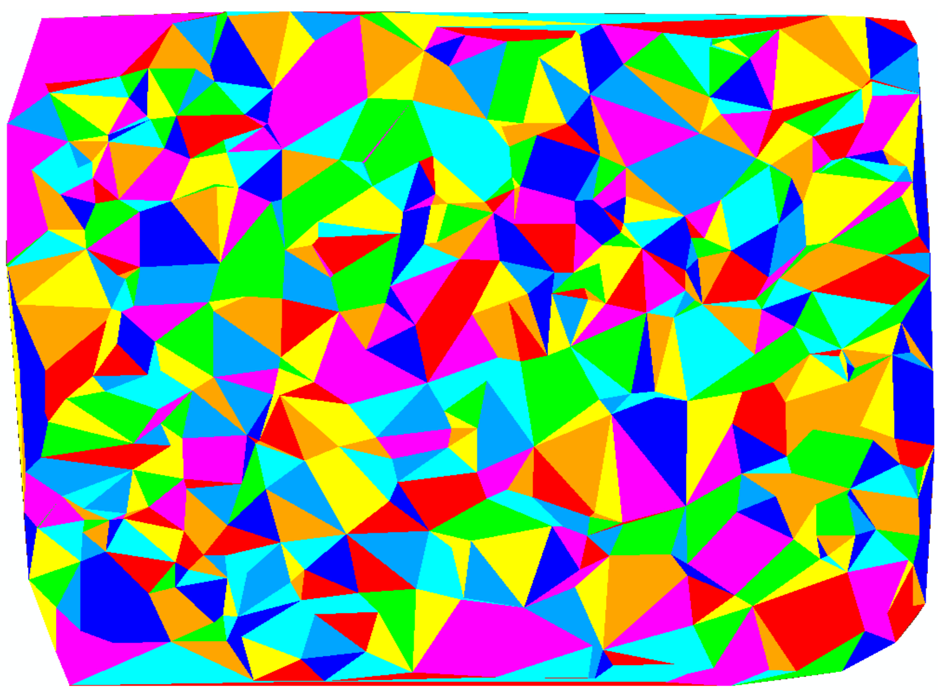
Obr. 6: Vygenerované vrstevnice aplikáciou.



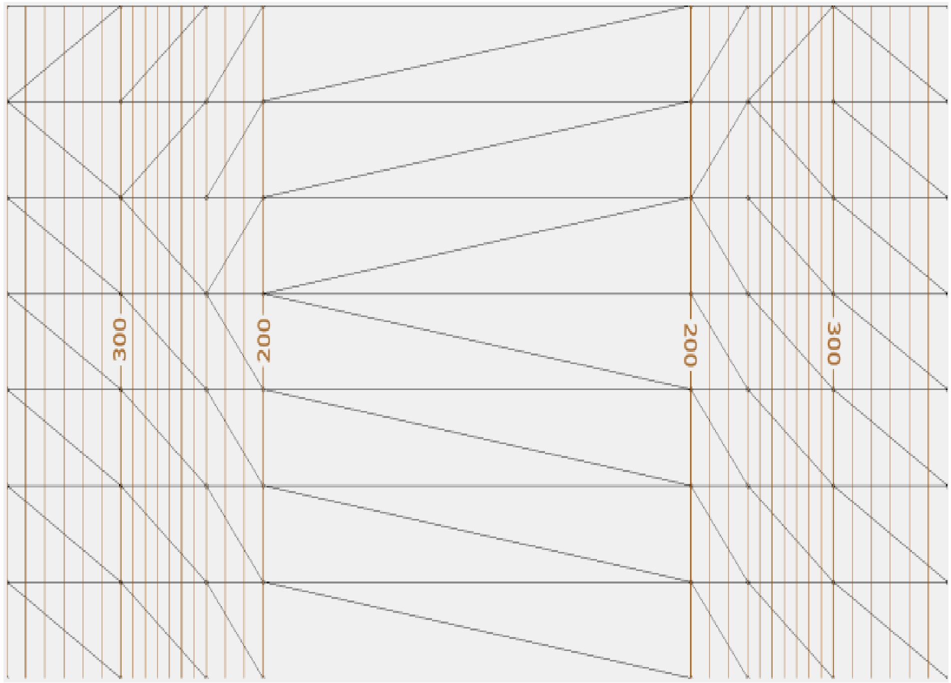
Obr. 7: Vygenerované vrstevnice softvérom ArcGIS.



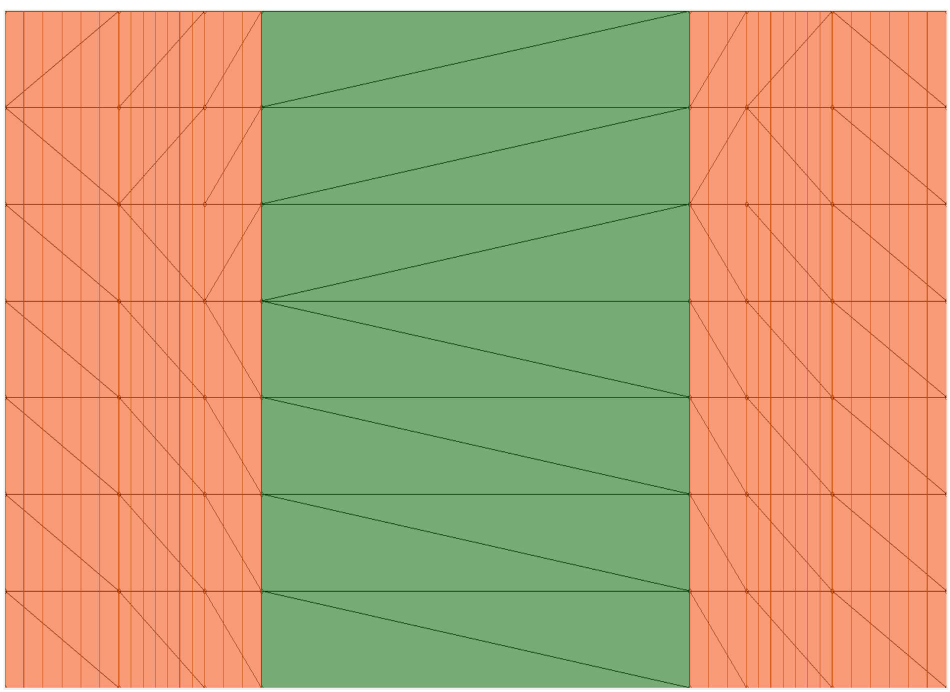
Obr. 8: Analýza orientácie vykonaná aplikáciou.



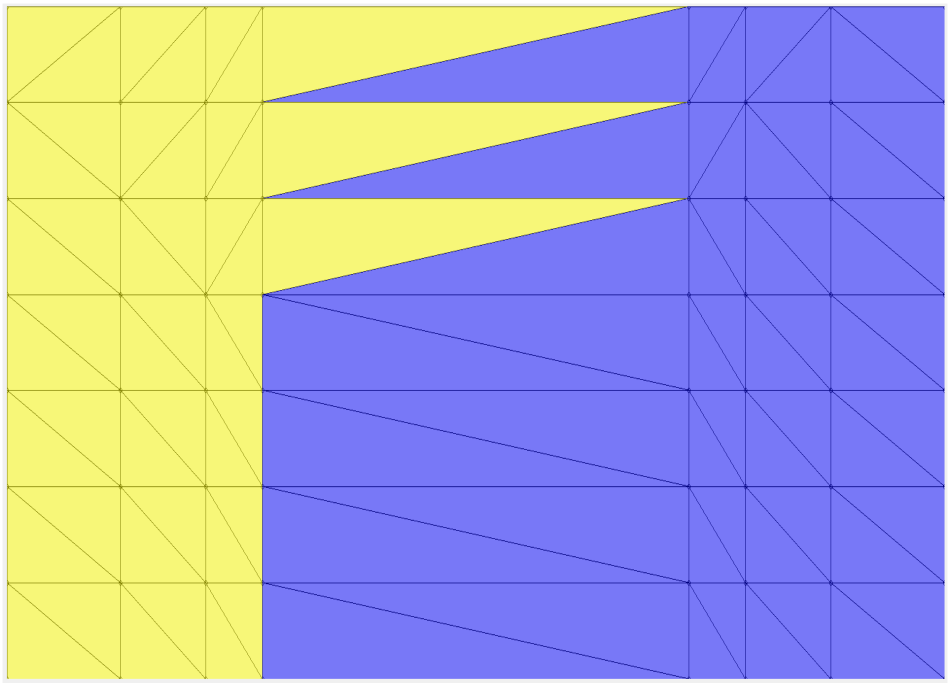
Obr. 9: Analýza orientácie vykonaná softvérom ArcGIS.



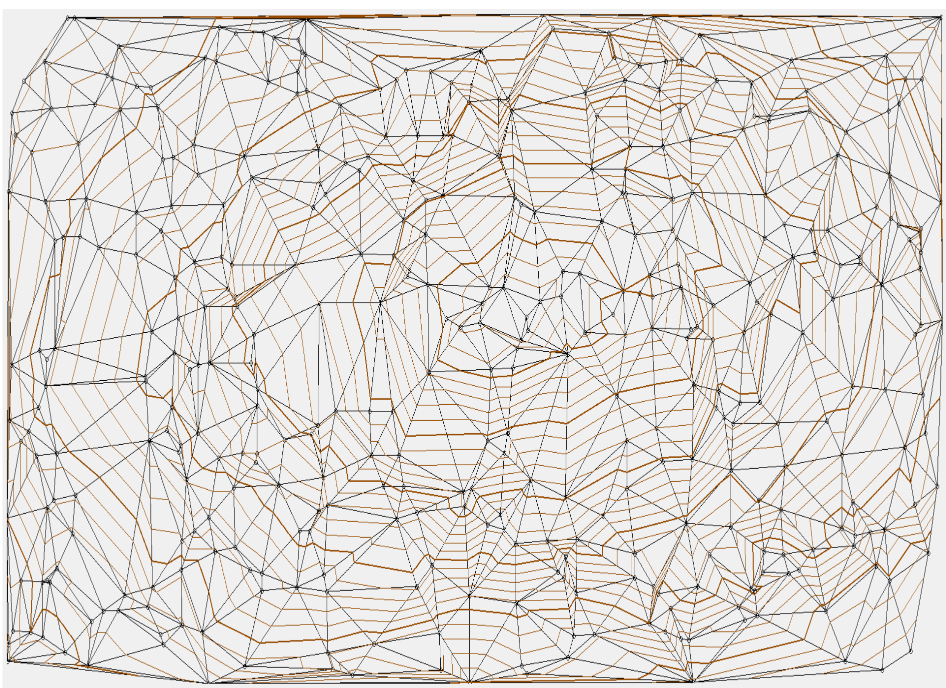
Obr. 10: Vytvorené údolie s vygenerovanými vrstevnicami. Pre lepšie znázornenie výšok v grafickom softvéry doplnené popisky vrstevníc.



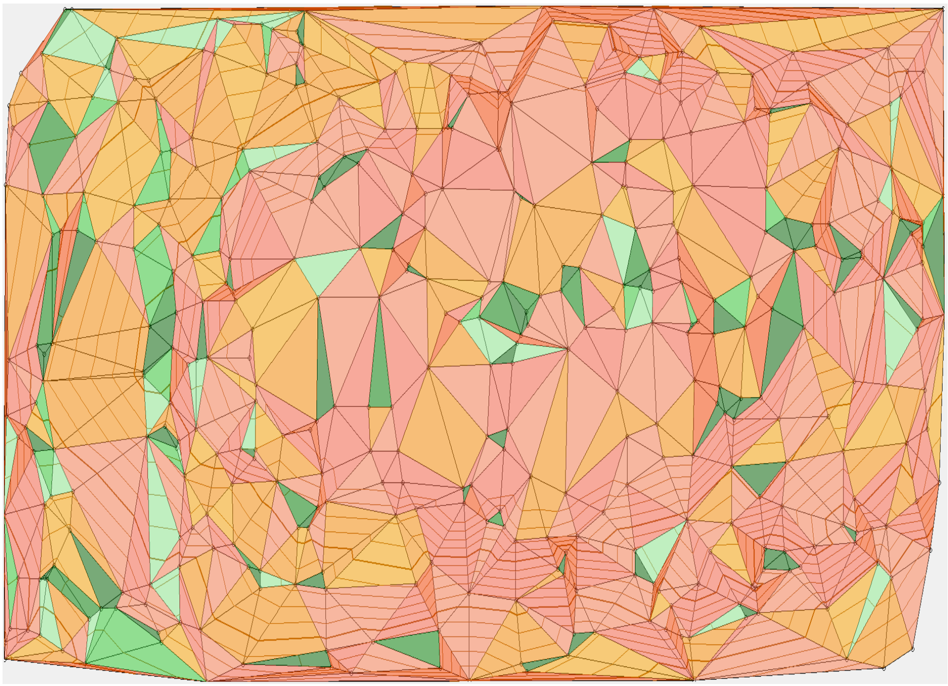
Obr. 11: Analýza sklonu pre vytvorené údolie.



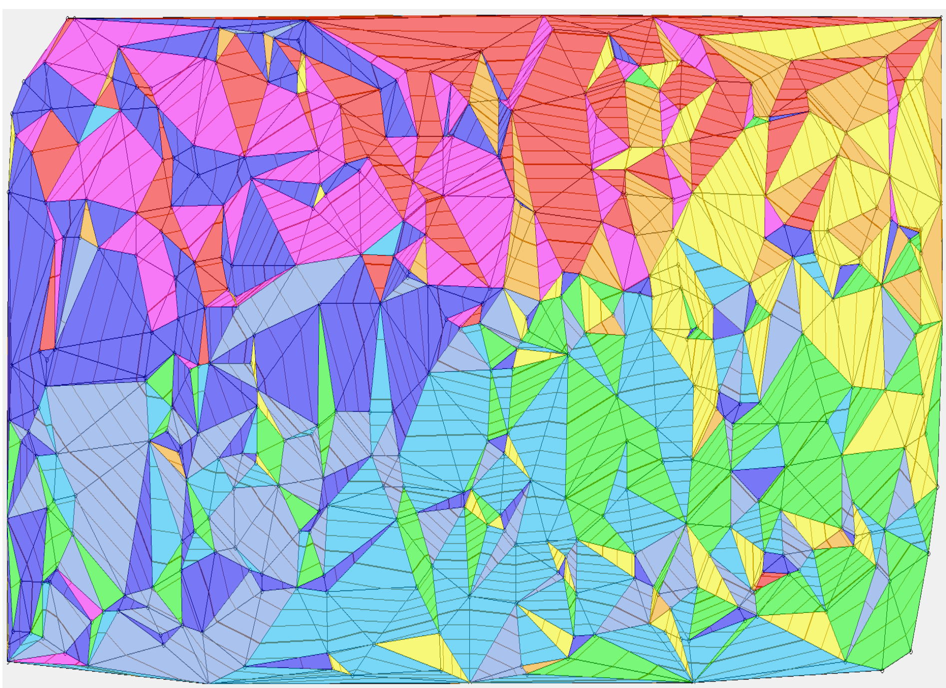
Obr. 12: Analýza orientácie pre vytvorené údolie. (modrá - západ, žltá - východ)



Obr. 13: Triangulácia a vrstevnice u kopy.



Obr. 14: Analýza sklonu pre kopu.



Obr. 15: Analýza orientácie pre kopu.

10 Dokumentácia

Samotný program bol napísaný v jazyku Python 3.9 v SW PyCharm. Okrem externých modulov QT bol využitý aj modul `csv`, ktorý slúži na prečítanie CSV súboru. Program pozostáva z piatich tried:

1. Trieda `Mainform`:

   Metóda `openFile` inicializuje vloženie vstupných dát do programu. Zisťuje výšku a šírku zobrazovacej plochy aplikácie z dôvodu nutného preškalovania súradníc vstupných dát do súradníc zobrazovacej plochy aplikácie. Hodnoty výšky a šírky predáva metóde `setPath`.

   Metóda `refreshCanvas` slúži na zmazanie všetkých zvizualizovaných objektov na zobrazovacej ploche aplikácie.

   Metóda `runDT` inicializuje výpočet a vizualizáciu Delaunayho triangulácie.

   Metóda `runContourLines` inicializuje výpočet a vizualizáciu vrstevníc. Metóda takisto získava užívateľské parametre, ktoré predáva funkcii na výpočet vrstevníc. V prípade, ak v predošlom kroku nebola vygenerovaná Delaunayho triangulácia, metóda takisto volá funkciu `runDT`.

   Metóda `runCalculateSlope` inicializuje výpočet a vizualizáciu sklonu. V prípade, ak v predošlom kroku nebola vygenerovaná Delaunayho triangulácia, metóda takisto volá funkciu `runDT`.

   Metóda `runCalculateExposition` inicializuje výpočet a vizualizáciu orientácie roviny. V prípade, ak v predošlom kroku nebola vygenerovaná Delaunayho triangulácia, metóda takisto volá funkciu `runDT`.

   Metóda `showSettings` slúži pre zobrazenie dialogóveho okna, ktoré užívateľovi umožňuje zadávanie parametrov pre výpočet vrstevníc uplatnených v metóde `runContourLines`.

2. Trieda `Draw`: Je zodpovedná za grafickú stránku aplikácie a zobrazovacej plochy. Trieda obsahuje inicializátor a 9 metód:

   `Inicializátor` má dva pozičné argumenty a inicializuje premenné pre túto triedu. Inicializované sú prázdne listy pre vstupné body, Delaunayho trianguláciu, vrstevnice, sklony a orientáciu rovín ako aj potrebné parametre pre tranformáciu súradníc zobrazovacej plochy aplikácie do reálneho intervalu.

   Metódy `getPoints`, `getTransformationParameters` a `getDT` vracajú konkrétne premenné treidy `Draw`.

   Metódy `setDT`, `setCL`, `setSlope`, `setExposition` vyžadujú jeden argument, ktorý je následne metódou priradený konkrétnym premenným triedy `Draw`.

Metóda `setPath` má dva argumenty a to výšku a šírku zobrazovacej plochy aplikácie. Do premennej `points` ukladá vstupné body načítané z `CSV` alebo `TXT` súboru. Na prečitanie súboru sa používa externý modul `csv`.

Z dôvodu rozličného súradnicového systému vstupných dát a zobrazovacej plochy aplikácie vstupné dáta navyše preškálováva. V prvom rade je bod najbližší počiatku súradnicového systému zobrazovacej plochy presunutý do jeho počiatku. Následne sa vykoná normalizácia do intervalu $< 0, 1 >$ a hodnoty sú vynásobené šírkou a výškou zobrazovacej plochy. Navyše Y súradnice preklápa cez osu X.
Metóda vracia list bodov.

Metóda `paintEvent` má argument `QPaintEvent` z modulu `QtGui`. Dochádza tu ku začiatku vizualizácie na zobrazovacej ploche. Vykreslené sú body, línie ako aj polygony tvorené vstupnými bodmi, ktoré sú sfarbené podľa špecifickej hodnoty extrahovanej z listov `slopes` a `exposition`.

3. Trieda `QPoint3D`: Táto trieda je odvodená z rodičovskej triedy `QPointF`. Obsahuje `Inicializátor` a 1 metódu:

   `Inicializátor` má 3 argumenty dátového typu `float`, ktoré predstavujú priestorové súradnice bodu - `x`, `y`, `z`. Hodnota `z` je defaultne nulová.

   Metóda `getZ` vracia hodnotu premennej `z`.

4. Trieda `Edge`: Trieda definuje líniu tvorenú počiatočným a koncovým bodom typu `QPoint3D`. Obsahuje `Inicializátor`, 3 metódy a metódu $_eq$:

   `Inicializátor` má 2 argumenty typu `QPoint3D`, ktoré predstavujú počiatočný a koncový bod línie.

   Metódy `getStart` a `getEnd` vracajú počiatočný alebo koncový bod línie.

   Metóda `switch` vracia objekt typu `Edge` s opačnou orientáciou pôvodnej línie.

   Metóda $\_\_eq\_\_$ má jeden argument objektového typu `Edge`. Metóda slúži na porovnanie totožnosti dvoch hrán. Metóda vracia hodnotu $True/False$.

5. Trieda `Algorithms`:

   Metóda `getPointAndLinePosititon` má na vstupe tri argumenty, všetky predstavujú body v dátovom type `QPoint3D`. Slúži na určenie polohy analyzovaného bodu a priamky.
   V prvom kroku je definovaná prijateľná odchýlka `epsilon`, nasleduje výpočet zložiek dvoch vektorov a výpočet vektorového súčinu pomoocu determinantu a testovanie podmienok, do

ktorých vstupuje hodnota determinantu a podľa ktorej sa určí pozícia bodu voči priamke. Metóda vracia hodnotu 1 ak je bod v ľavej polrovine, hodnotu 0 ak je bod v pravej polrovine, v prípade kolinearity vráti hodnotu -1.

Metóda `get2LinesAngle` má na vstupe štyri body, opäť všetky v dátovom type `QPoint`. Slúži na výpočet uhlu dvoch priamok. Opäť sú spočítané dva vektory, ich skalárny súčin a ich norma.
Figuruje v nej podmienka, ktorá rieši singularitu analyzovaného bodu na vrchole polygónu, a to na základe nulovej hodnoty jednej z noriem vektorov. Metóda v tomto prípade vracia hodnotu 0.
Druhá podmienka slúži na vyhnutie sa prípadu, kedy arccos uhlu dvoch priamok sa počíta z hodnoty, ktorá je väčšia ako 1. V takomto prípade metóda vracia výslednú hodnotu z $|\arccos(1)|$.
Mimo spomenutých prípadov funkcia vracia hodnotu uhlu dvoch priamok v kladných hodnotách.

Metóda `getCircleCenterAndRadius` má na vstupe tri argumenty typu `QPoint3D`, z krotých je vo funkcii definovaná kružnica. Metóda vracia stred kružnice ako dátový typ `QPoint3D` a hodnotu polomeru tejto kružnice vo forme `float`.

Metóda `getNearestPointIdx` má na vstupe argument `p` typu `QPoint3D` a list bodov typu `QPoint3D`. Metóda vracia `integer` index najbližšieho bodu k vstupnému bodu `p`.

Metóda `getDelaunayPointIdx` má na vstupe argument `e` objektového typu `Edge` a list bodov typu `QPoint3D`. Metóda vracia `integer` index vhodného bodu pre proces generovania Delaunayho triangulácie.

Metóda `DT` má na vstupe list bodov typu `QPoint3D`. Výstupom metódy je zoznam hrán typu `Edge`, ktoré tvoria Delaunayho trianguláciu.

Metóda `updateAEL` má na vstupe argument `e` objektového typu `Edge` a zoznam hrán typu `Edge`. Metóda nevracia nijakú premennú, je potrebná pre spustenie procesu aktualizácie vstupného zoznamu hrán.

Metóda `getCLpoint` má na vstupe 3 argumenty a to p_1, p_2 typu `QPoint3D` a argument `z` typu `float`. Metóda vracia priesečník hrany Delaunayho trojuholníku a vrstevnice o výške `z` vo forme `QPoint3D`.

Metóda `createCL` má na vstupe 4 argumenty - `dt` zoznam hrán `Edge`, ktoré tvoria Delaunayho trianguláciu a argumenty `zmin`, `zmax` a `dz` dátového typu `float`, ktoré určujú interval a inkrement generovaných vrstevníc. Výstupom metódy je zoznam `cl` obsahujúci vrstevnice objektového typu `Edge`.

Metóda `calculateSlope` má na vstupe 4 argumenty a to zoznam `dt` obsahujúci hrany Delaunayho triangulácie, `parameters` dátového typu `tuple`, výšku a šírku zobrazovacej plochy aplikácie `width` a `height` dátového typu `int`. Metóda vracia zoznam `slopes`, ktorý obsahuje hodnotu sklonu dátového typu `float` pre každú plochu Delaunayho triangulácie.

Metóda `calculateExposition` má na vstupe 4 argumenty a to zoznam `dt` obsahujúci hrany Delaunayho triangulácie, `parameters` dátového typu `tuple`, výšku a šírku zobrazovacej plochy aplikácie `width` a `height` dátového typu `int`. Metóda vracia zoznam `exposition`, ktorý obsahuje hodnotu orientácie dátového typu `float` pre každú plochu Delaunayho triangulácie.

11 Záver

Vytvorená aplikácia úspešne implementuje tvorbu Delaunayho triangulácie inkrementálnou konštrukciou čim vytvára polyedrický digitálny model nad vstupnou množinou bodov, načítanou z TXT súboru. Nad takto vytvoreným modelom je schopná generovať vrstevnice lineárnou interpoláciou, analyzovať sklon a orientáciou jednotlivých trojuholníkov.

Najviac problémov bolo s analýzou orientácie. Jej výsledky pravdepodobne nie sú úplne správne. Chyba je zrejme pri prevode súradníc zo zobrazovacej plochy naspäť do pôvodných (vstupných). Takisto zostáva aj mierny nedostatok už z predošlých úloh, kedy vstupné dáta sú mierne deformované z dôvodu roztiahnutia vstupných dát v oboch smeroch do maximálneho možného intervalu.

12 Použité zdroje

Prednášky z predmetu *Algoritmy počítačové kartografie.*

BRŮHA, L. 2016: Digitální modely terénu. Přírodovědecká fakulta Univerzity Karlovy v Praze.

LEIFER, V. 2006: Delaunayho triangulace a její aplikace. Diplomová práca. VUT v Brne, FSI ústav automatizace a informatiky.