155ADKG: Digitální model terénu a jeho analýzy

Datum odevzdání: 7.1.2018

Petra Millarová, Oleksiy Maybrodskyy

Contents

1 Zadání

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 tři 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**

Čas zpracování: 3 týdny

2 Popis a rozbor problému

Výstupem je program, který dokáže vytvořit nad prostorovou množinou DMT pomocí triangulace, vrstevnice a vizualizovat sklon terénu a jeho expozici.

3 Popisy algoritmů

3.1 Delaunay triangulace

Delaunay triangulace je nejčastěji používanou triangulací v oblasti GIS, lze jí provádět jak plošně, tak prostorově.

Podmínky DLT

- Uvnitř kružnice v libovolném trojúhelníku triangulace neleží žádný jiný bod množiny.
- Tato triangulace maximalizuje minimální úhel v trojúhelníku, avšak neminimalizuje maximální úhel.
- Je lokálně i globálně optimální vůči kritériu minimálního úhlu.
- Triangulace je jednoznačná, pokud žádné čtyři body neleží na kružnici.

Triangulace byla realizována metodou inkrementální konstrukce. Tato metoda je založena na postupném přidávání bodů do již vytvořené triangulace. Hledá se takový bod, který má spolu s dvěma body triangulace nejmenší opsanou kružnici. Každá hrana trojúhelníku je orientována a body hledáme pouze vlevo od ní.

Pokud je nalezen vyhovující bod, jsou do výsledné triangulace přidány orientované hrany nového trojúhelníku. Pokud neexistuje žádný takový bod, změníme orientaci hrany a hledání opakujeme.

Pro ukládání hran se používá tzv. *Active Edge List*. Tato struktura obsahuje hrany, ke kterým hledáme třetí bod. Když je třetí bod k hraně nalezen, hrana je odebrána ze seznamu. Následně se kontroluje, zda není v AEL tato hrana již přítomna s opačnou orientací. Pokud je, je z tohoto seznamu odstraněna. Pokud není, je hrana do AEL naopak přidána. V každém případě je ale hrana přidána do výsledné triangulace. Takto se v algoritmu postupuje, dokud není AEL prázdný. *Postup při inkrementální konstrukci:*

- náhodný bod p_1 a jemu nejbližší bod p_2
- hrana e_1 z bodů p_1 a p_2
- p_3 je bod s nejmenší Delaunay vzdáleností k hraně e_1 vlevo od hrany.
- pokud se žádný bod nenajde, prohodí se orientace hrany e_1 a hledá se znovu
- po nalezení bodu p_3 se z p_1, p_2, p_3 vytvoří hrany e_2 a e_3 které spolu s hranou e_1 tvoří orientovaný trojúhelník
- dokud není AEL prázdný, přidej e_1, e_2, e_3 do AEL:
  - e_1 první hrana AEL
  - změna orientace e_1
  - hledáme bod p který má nejmenší Delaunay vzdálenost vlevo od otočené e_1
  - pokud takový bod existuje, přidej e_2, e_3 do AEL a hranu e_1 do triangulace
  - pokud takový bod neexistuje, odstraň e_1 z AEL

3.2 Vrstevnice

Vrstevnice byly počítány pomocí lineární interpolace. U lineární interpolace je rozestup mezi dvěma body konstantní, spád terénu taktéž.

Princip spočívá ve vyhledání průsečnice roviny určené postupně každým trojúhelníkem a roviny vodorovné se zadanou výškou h, ve které chceme bod vrstevnice nalézt.

Zda protíná rovina stranu trojúhelníku lze zjistit jednoduchým testem: $(z - z_i)(z - z_{i+1})) < 0$.

Pokud je průsečnicí těchto rovin úsečka, vypočítají se souřadnice jejích bodů následovně:

$$x_a = \frac{(x_3 - x_1)}{z_3 - z_1}(z - z_1) + x_1,$$

$$y_a = \frac{(y_3 - y_1)}{z_3 - z_1}(z - z_1) + y_1,$$

$$x_b = \frac{(x_2 - x_1)}{z_2 - z_1}(z - z_1) + x_1,$$

$$y_b = \frac{(y_2 - y_1)}{z_2 - z_1}(z - z_1) + y_1.$$

3.3 Počítání sklonu

Výpočet sklonu se provádí nad každým trojúhelníkem v DMT. Trojúhelník je jasně daný dvěma vektory, odchylka φ dvou rovin se spočte následovně:

$$\varphi = \arccos(\frac{n_1 n_2}{|n_1 n_2|}).$$

3.4 Orientace svahu

Orientace svahu v bodě je definována jako azimut A průmětu gradientu $\bigtriangledown\rho(x_0, y_0, z_0)$ do roviny xy, který se spočte jako

$$A = \arctan(\frac{n_1}{n_2}), A \in \langle 0, 2\pi),$$

kde n_1, n_2 jsou vektorové součiny vektorů, které tvoří daný trojůhelník.

4 Vstupní data

Do programu lze načíst soubor ve formátu **.txt*, který na každém řádku obsahuje XYZ souřadnice jednoho vkládaného bodu:

```
54625.84 23547.74 250.5
54826.52 23580.78 266.7
54752.66 23612.32 245.8
.
.
.
```

5 Výstupní data

Program načte vstupní data a po stisku tlačítka *Delaunay triangulation* vykreslí body a jejich triangulaci. Při každém vykreslení triangulace se aktualizuje šířka okna a podle té se pak body natransformují a zobrazí.

Poté si uživatel může zvolit, zda chce vizualizaci vrstevnic (tlačítko *Contour lines*) - zde je potřeba si nastavit požadovaný interval a rozmezí, ve kterém se vrstevnice generují (pro orientaci vidí uživatel výšku

nejnižšího a nejvyššího z načtených bodů). Také lze barevně zobrazit sklon terénu (*Slope*) a jeho orientaci (*Orientation*). Pro vyčištění plochy a ponechání pouze načtených bodů slouží tlačítko *Clear*.

6 Ukázky aplikace

Vzhledem k nejasné chybě v algoritmu pro triangulaci je tato kapitola omezená jen na ukázku vzhledu UI a vykreslení vrstevnic.

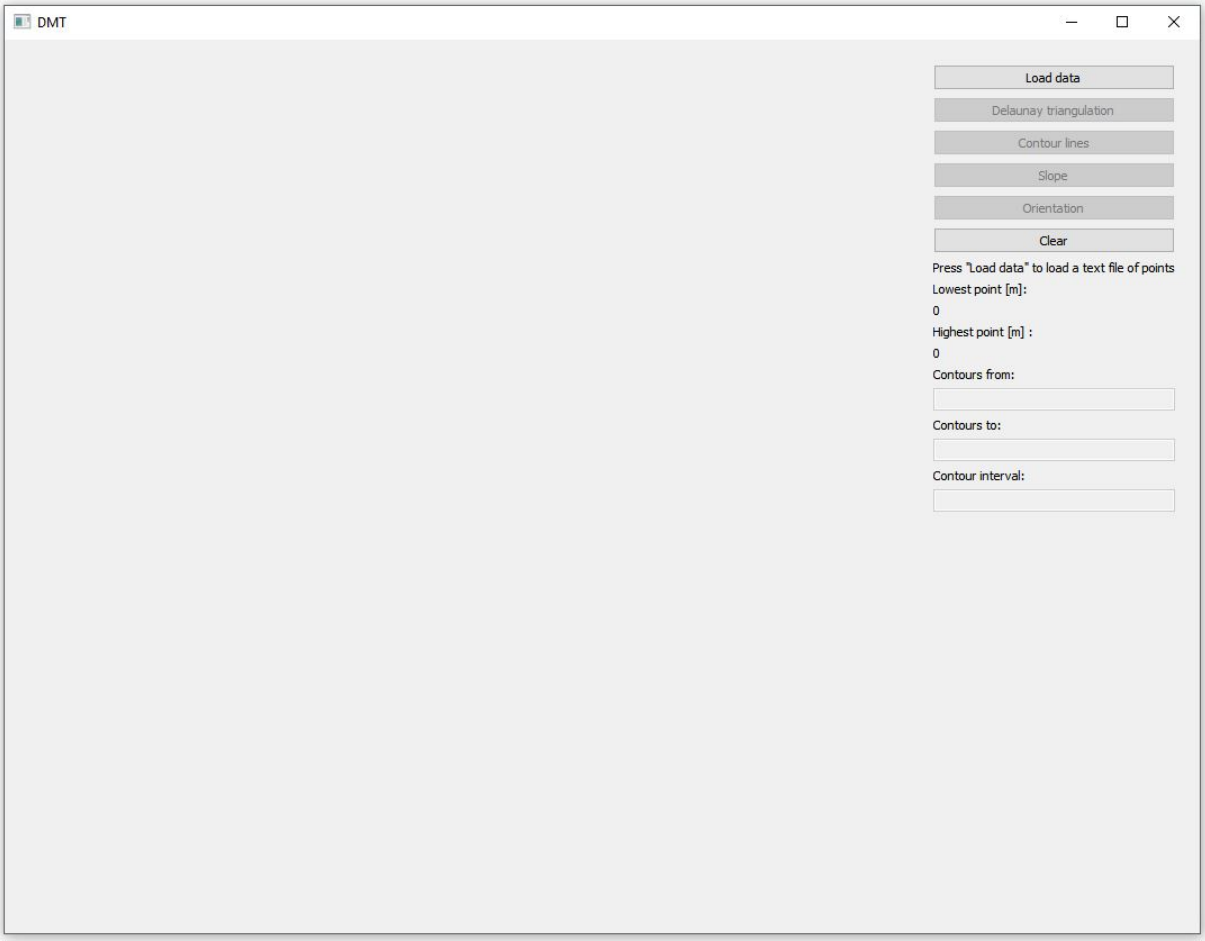

Figure 1: Vzhled aplikace při otevření

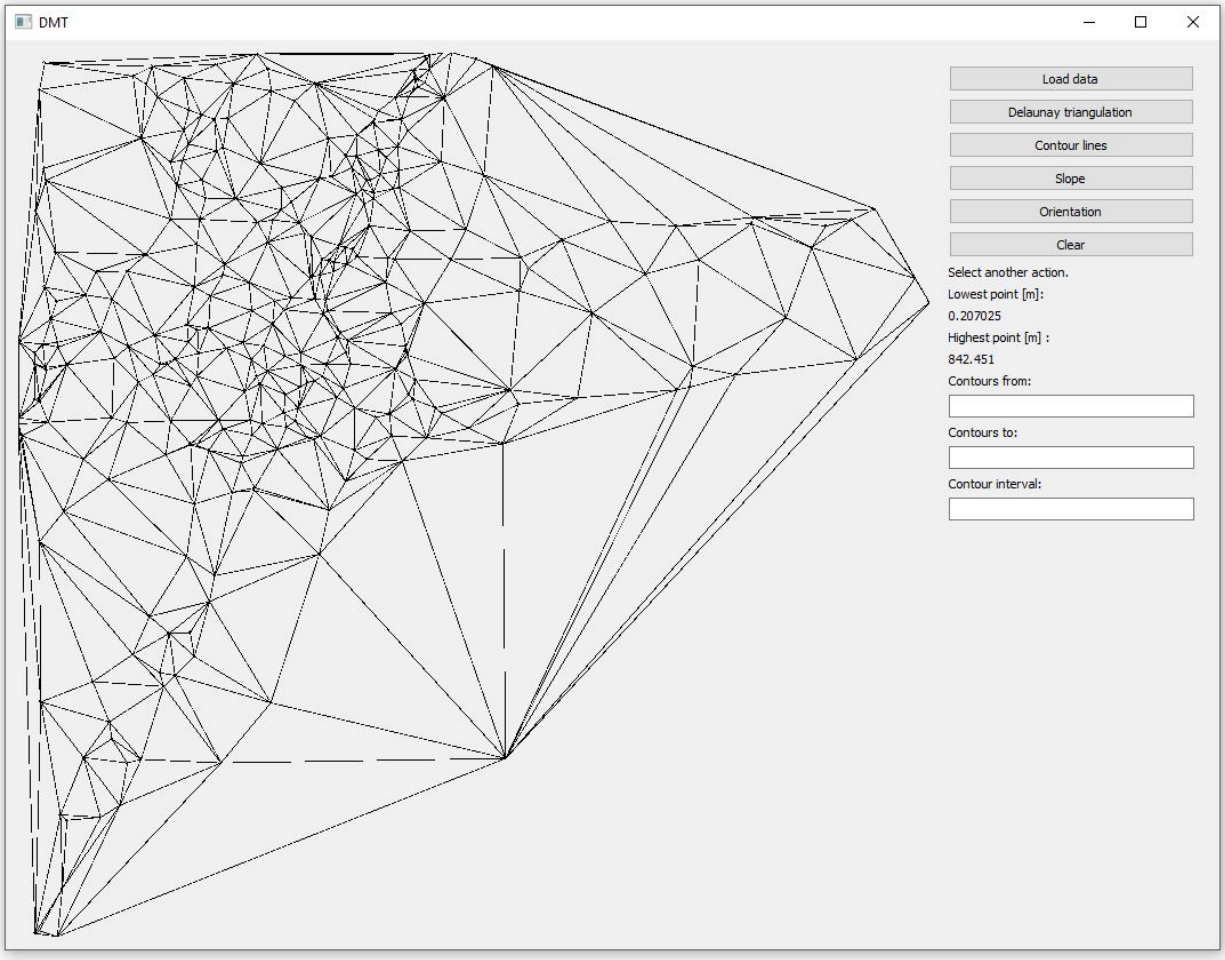

Figure 2: Aplikace po provedení triangulace

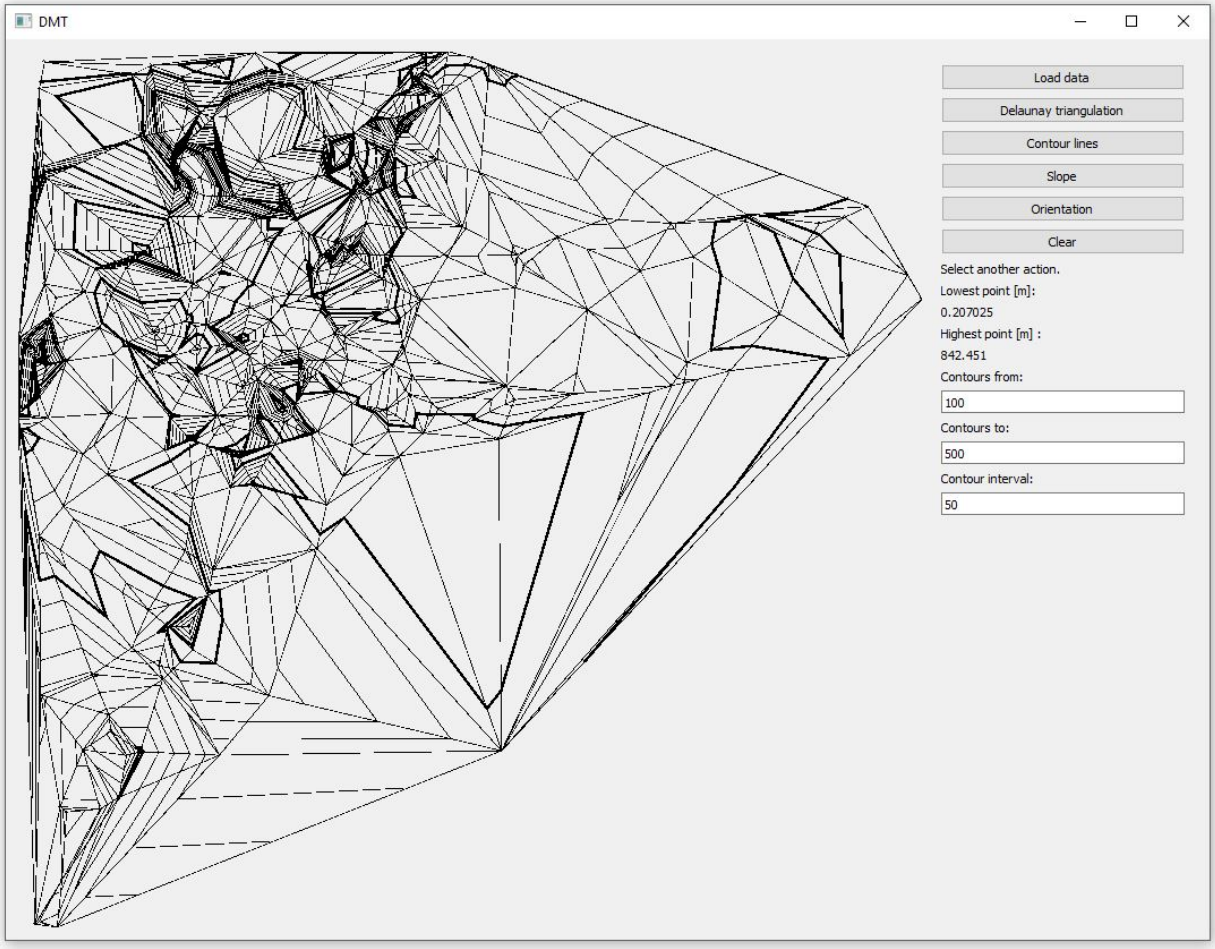

Figure 3: Aplikace po zobrazení vrstevnic

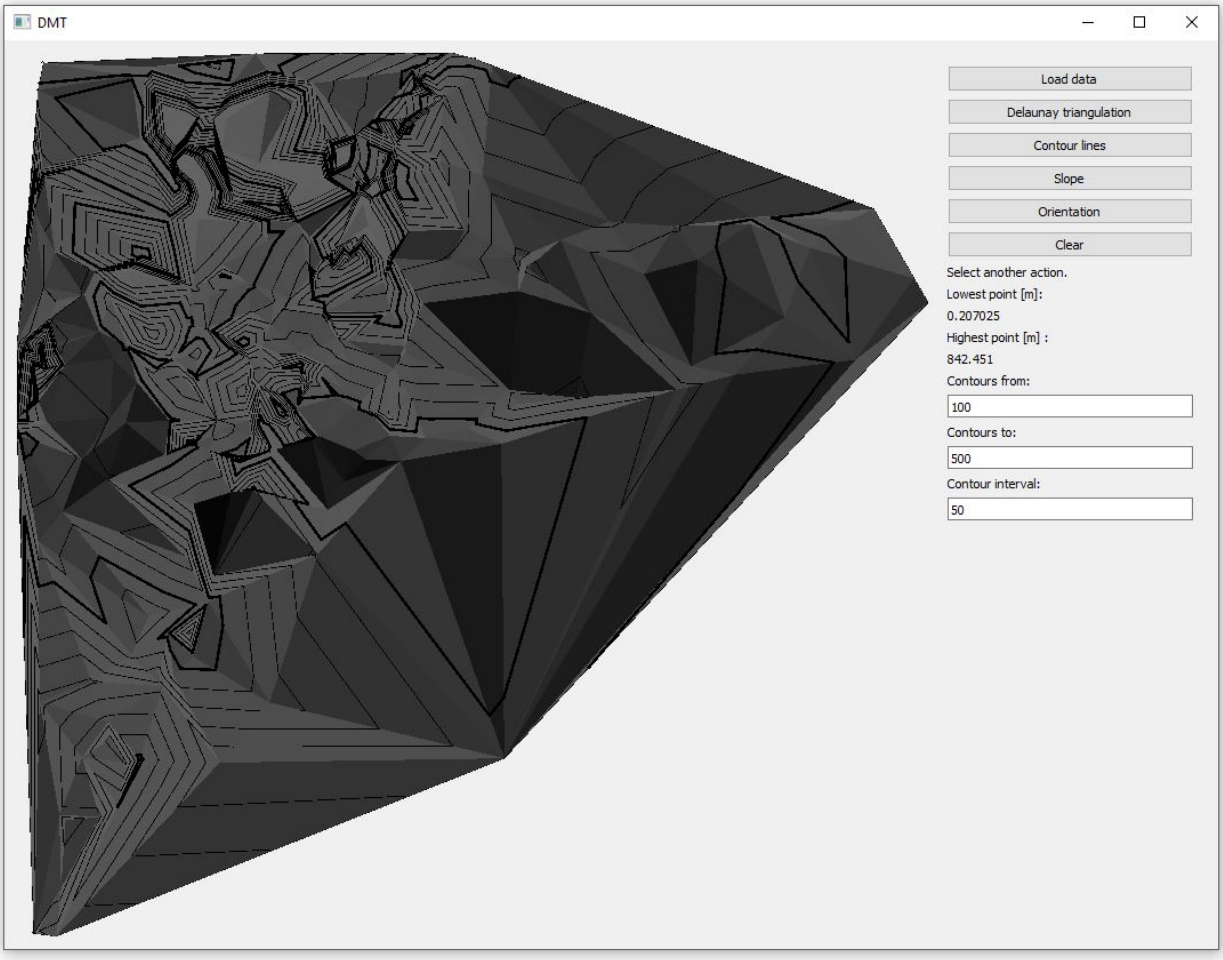

Figure 4: Aplikace po vypočítání sklonu

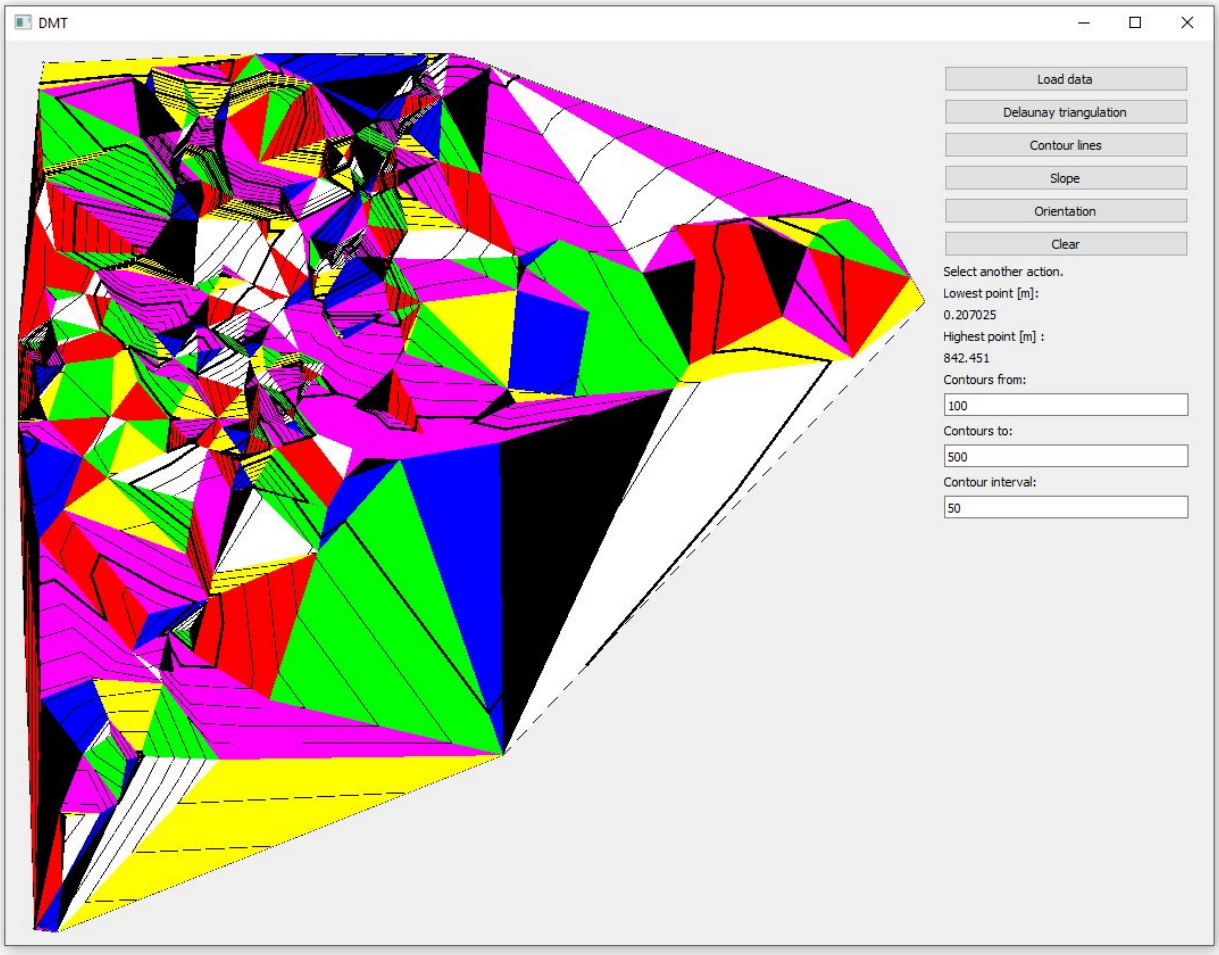

Figure 5: Aplikace po vypočtení orientace svahů

7 Dokumentace

7.0.1 Datové struktury

QPoint3D - třída odvozená od třídy *QPointF*, má navíc i souřadnici *Z* ve formátu double. Nad třídou lze zavolat funkce *getX*, *getY*, *getZ*, které vrací příslušnou souřadnci.
Edge - datový typ obsahující dva body typu *QPoint3D*(počátek a konec hrany). Třída obsahuje přetížení operátoru == pro porovnávání hran, funkce pro vrácení počátečního a koncového bodu hrany a funkci *switchOrientation* která otáčí orientaci hrany.
Triangle je složený ze tří bodů typu *QPoint3D*, svého sklonu a orientace (které jsou typu *double* a implicitně nulové). Tato třída obsahuje funkce pro přístup k těmto hodnotám a funkce pro jejich pozměnění.

7.0.2 Algorithms

Tato třída obsahuje algoritmy, které buď přímo slouží k výpočtům nutným pro zvládnutí této úlohy nebo jsou to algoritmy pomocné (jako např. zjištění vzájemné polohy bodu a přímky) **getPosition** je funkce,

která popisuje polohu bodu vůči linii. Do funkce vstupuji 3 hodnoty typu **QPoint**, které reprezentuji dva body na přímce a bod, jehož poloha se určuje. Návratová hodnota této funkce je celé číslo **intiger**, podle výsledku testování.
Input:

- *QPoint &q* - testovaný bod
- *QPoint &a* - jeden bod linie
- *QPoint &b* - druhý bod linie

Output:

- 0 - bod leží vpravo od přímky
- 1 - bod leží vlevo od přímky

getCircleRadius provádí výpočet poloměru kružnice zadané třemi body a výpočet středu této kružnice, jehož souřadnice se vloží do proměnné předané parametrem.
Input:

- *QPoint3D &p1* - bod kružnice
- *QPoint3D &p2* - bod kružnice
- *QPoint3D &p3* - bod kružnice
- *QPoint3D &cp* - bod do kterého se vloží souřadnice středu kružnice

Output:

- hodnota poloměru kružnice v datovém typu *double*

getDelaunayPoint hledá index takového bodu, který má od zadané hrany minimální poloměr opsané kružnice.
Input:

- *Edge &e* - hrana ke které hledáme bod
- *std::vector <QPoint3D> &points* - vstupní množina bodů

Output:

- index hledaného bodu ve vektoru *points*, při chybě vrací -1

getNearestPoint určuje index nejbližšího bodu k zadanému bodu **Input:**

- *QPoint3D &p* - zadaný bod
- *std::vector <QPoint3D> &points* - vstupní množina bodů

Output:

- index nejbližšího bodu v datovém typu *int*

dt provede vlastní triangulaci nad zadanou množinou bodů a vrací seznam orientovaných hran. **Input:**

- *std::vector <QPoint3D> &points* - vstupní množina bodů

Output:

- výsledný vektor hran v datovém typu *std::vector<Edge>*

getConPoint nalezne bod o zadané výšce mezi dvěma zadanými body **Input:**

- *QPoint3D &p1* - zadaný bod
- *QPoint3D &p2* - zadaný bod
- *double &z* - zadaná výška

Output:

- bod na vrstevnici ve formátu *QPoint3D*

createContours vytvoří vektor hran vrstevnic **Input:**

- *std::vector<Edge>&dt* - triangulace na které hledáme vrstevnice
- *double zmin* - minimální výška vrstevnic
- *double zmax* - maximální výška vrstevnic
- *double h* - vzdálenost mezi vrstevnicemi

Output:

- seznam hran vrstevnic ve formátu *std::vector<Edge>*

convertDTM převede vektor hran výsledné triangulace na vektor trojúhelíků **Input:**

- *std::vector<Edge>* - hrany výsledné triangulace

Output:

- *std::vector<Triangle>* - převedené trojúhelníky

getSlope vypočítá sklon zadaného trojúhelníku **Input:**

- *Triangle* - zadaný trojúhelník

Output:

- velikost sklonu ve formátu *double*

getExposition vypočítá orientaci zadaného trojúhelníku **Input:**

- *Triangle* - vstupní trojúhelník

Output:

- úhel natočení ve formátu *double*

getSlopes nastavují sklony jednotlivých trojúhelníků v zadaném vektoru **Input:**

- *std::vector<Triangle> &dtt* - vstupní vektor trojúhelníků u kterých se nastavuje sklon

Output:

- *void*

7.0.3 Draw

Třída **Draw** slouží k vykreslení vypočtených triangulací, sklonů, orientací a vrstevnic.
loaddata je funkce sloužící k parsování souboru zvoleného uživatelem a vložení dat do proměnné *std::vector<double> &data*.

clear, funkce slouži k vyčištění všech dosavadních výpočtů. Ponechává pouze načtené body.

delaunaydraw vykreslí vypočtenou triangulaci.
slopedraw vykreslí sklon jednotlivých trojúhelníků triangulace.
orientdraw vykreslí orientaci jednotlivých trojúhelníků triangulace.
countoursdrawing vykreslí vrstevnice.
paintEvent funkce volaná při každém překreslení okna, podle nastavených flags volá funkce pro vykreslení různých vlastností triangulace a triangulace samotné.

7.0.4 Widget

Tato třída je vytvořena pro praci s grafickým rozhraním celého programu. Přes ní se provádí načtení souborů a grafické znázornění výsledků aplikace. Po kliknutí na tlačítko v aplikaci se pustí příslušná funkce v této třídě a zavolá funkci na výpočet a vykreslení, případně načtená dat.

7.0.5 sortby-funkce

Tyto třídní funkce slouží k seřazení vektoru souřadnic vzestupně podle velikosti buď souřadnice X nebo Y.

8 Závěr

Byla vytvořena aplikace, která nad Delaunay triangulací počítá a zobrazuje sklon, vrstevnice a orientaci svahu. Aplikace má pár nedostatků, které již bohužel autoři nestihli opravit.

8.1 Náměty na vylepšení

Samozřejmě že aplikace není bez chyb a je zde prostor pro vylepšení. Bylo by vhodné, kdyby si uživatel mohl vybrat, v odstínech jaké barvy se mu zobrazí vypočítaný sklon. Vrstevnice by mohly mít možnost popisu číslem a překreslení bodů podle velikosti okna by se mohlo provádět hned.