ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ, OBOR GEODÉZIE A KARTOGRAFIE
KATEDRA GEOMATIKY

název předmětu

ALGORITMY V DIGITÁLNÍ KARTOGRAFII

číslo úlohy	název úlohy			
3	Digitální model terénu a jeho analýzy			
školní rok	studijní skupina	zpracovali:	datum	klasifikace
2020/21	60	Michal Zíma, Tomáš Lauwereys	19. 12. 2020	

Zadání

Anotace:

V rámci úlohy byla vytvořena aplikace v prostředí QT, která umí vygenerovat trojúhelníkovou síť pomocí Delaunay triangulace. Aplikace dále umí vizualizovat sklon a expozici sítě, popsat hlavní vrstevnice.

Přesné zadání:

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.

Bonusové úlohy

1. Triangulace nekonvexní oblasti zadaná polygonem.
2. **Výběr barevných stupnic při vizualizaci sklonu a expozice.**
3. **Automatický popis vrstevnic.**
4. Automatický popis vrstevnic respektující kartografické zásady (orientace, vhodné rozložení).
5. Algoritmus pro automatické generování terénních tvarů (kupa, údolí, spočinek, hřeb, …).
6. 3D vizualizace terénu s využitím promítání.
7. Barevná hypsometrie.

V rámci úlohy byly řešeny dvě bonusové úlohy (2. a 3.)

Zvolené algoritmy

Delaunay triangulace

K vytvoření Delaunay triangulace byla použita metoda inkrementální konstrukce. Metoda funguje na principu hledání minimální opsané kružnice vedoucí bodem, který je nejvhodnější k již vytvořené orientované

hraně a to v její levé polorovině. Jestliže nastane případ, kdy takový bod existuje, tak dojde ke změně orientace hrany a pokračuje se dále v hledání dalších bodů. Algoritmus má vlastní seznam hran, ke kterým nebyl nalezen třetí bod. Jakmile algoritmus nalezne novou hranu, tak je zapotřebí ověřit, zda hrana s opačnou orientací již v seznamu hran není. V případě, kdy tomu tak není, tak dojde k vložení nově nalezené hrany do seznamu a algoritmus hledá další hrany. Pokud by nastala situace, kdy k nějaké hraně v seznamu dojde k nalezení třetího bodu, tak algoritmus hranu odstraní ze seznamu. Algoritmus běží do situace, dokud nebude seznam hran prázdný.

Nalezení bodu p probíhá následovně. Nejprve se určí poloha bodu od dané hrany:

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

$$t = \begin{vmatrix} u_x & u_y \\ v_x & v_y \end{vmatrix} .$$

Pokud t < 0, tak se bod p nachází v levé polorovině. Určení poloměru se provede následovně:

$$m = 0{,}5 \cdot \frac{k_{12} \cdot (-k_4) + k_{11} \cdot k_5 - (k_{10} + k_4 \cdot k_5) \cdot k_6}{x_3 \cdot (-k_4) + x_2 \cdot k_5 + x_1(-k_6)}$$

$$n = 0{,}5 \cdot \frac{k_1 \cdot (-k_9) + k_2 \cdot k_8 + k_3 \cdot (-k_7)}{y_1 \cdot (-k_9) + y_2 \cdot k_8 + y_3 \cdot (-k_7)}$$

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

kde

$k_1 = x_1^2 + y_1^2$ $k_2 = x_2^2 + y_2^2$ $k_3 = x_3^2 + y_3^2$ $k_4 = y_1 - y_2$

$k_5 = y_1 - y_3$ $k_6 = y_2 - y_3$ $k_7 = x_1 - x_2$ $k_8 = x_1 - x_3$

$k_9 = x_2 - x_3$ $k_{10} = x_1^2$ $k_{11} = x_2^2$ $k_{12} = x_3^2$

Pro Delaunay triangulaci platí následující vlastnosti:

- Triangulace je jednoznačná, pokud čtyři body neleží na kružnici.
- Maximalizuje minimální úhel v trojúhelníku.
- Vůči kritériu min. úhlu je lokálně i globálně optimální.
- Uvnitř opsané kružnice libovolného trojúhelníku triangulace neleží žádný jiný bod.

Zápis Delaunay triangulace

Vstupem množiny AEL (Active Edge List) jsou hrany, ke kterým jsou hledány body p.

1. *Nalezení pivota q s min. souřadnicí X: $q = min(x_i)$*
2. *Hledání nejbližšího bodu k pivotovi: $||p_1 - q_2|| = min$*
3. *Vytvoření první hrany: $e = (q, p_1)$*
4. *Hledání Delaunay bodu: $p=argmin_{\forall pi \in \sigma L}(e)r'(k_i);k_i =(a,b,p_i);e=(a,b)$*
    a. *Při nenalezení Delaunay bodu změna orientace: $\nexists p : e \leftarrow (p_1, q)$; go to 4.*
5. *Vytvoření zbývajících hran trojúhelníka: $e_2 = (p_1, p)$; $e_3 = (p, q)$*
6. *Přidání hran trojúhelníka do DT: $DT \leftarrow e$; $DT \leftarrow e_2$; $DT \leftarrow e_3$*
7. *Přidání hran trojúhelníka do AEL (Active Edge List): $AEL \leftarrow e$; $AEL \leftarrow e_2$; $AEL \leftarrow e_3$*
8. *Dokud není AEL prázdný, tak proveď:*
    a. *Hledání Delaunay bodu k hraně z AEL (viz 4).*
    b. *Pokud Delaunay bod existuje if $\exists p$*
        i. *Vytvoření zbývajících hran trojúhelníku: $e_2 = (p_1, p)$; $e_3 = (p, q)$*
        ii. *Pokud nová hrana není v AEL, přidej ji.*

Výstupem je množina DT, která obsahuje 3 hrany tvořící trojúhelník.

Výběr barevných stupnic

U této bonusové úlohy byla definována barevná stupnice pro zobrazení sklonu a expozice. V rámci ComboBoxu je možné si zvolit stupně šedi, modré, zelené a všechny barvy. Aby byly dobře čitelné a viditelné vrstevnice, byla aplikována průhlednost barev. Samotné barvy byly implementovány ve třídě Draw, kde jsou procházeny jednotlivé trojúhelníky Delaunay triangulace a ke každému trojúhelníku podle spočtené analýzy a podle vybraného typu škály je přiřazena konkrétní barva.

Analýza sklonu DMT

Sklon udává, zda terén stoupá, klesá, nebo zůstává neměnný. Pokud máme dva normálové vektory (0, 0 , 1) a normálový vektor roviny trojúhelníku z trojúhelníkové sítě, pak nám úhel mezi těmito normálovými vektory definuje sklon terénu.

1. *Pro všechny trojúhelníky triangulace:* $\forall t_i \in DT$ *:*
2. *Výpočet normálového vektoru roviny trojúhelníku:*

$$u_x = x_2 - x_1;\ \ u_y = y_2 - y_1;\ \ u_z = z_2 - z_1;$$
$$v_x = x_2 - x_3;\ \ u_y = y_2 - y_3;\ \ u_z = z_2 - z_3;$$
$$n_t = \sqrt{(uy \cdot vz - uz \cdot vy)^2 - (ux \cdot vz - uz \cdot vx)^2 + (ux \cdot vy - uy \cdot vx)^2}$$

3. *Výpočet sklonu:* $\varphi = \arccos\left(\frac{n_z}{|n_t|}\right)$

Analýza expozice DMT

Expozice označuje orientaci terénu vůči světlu (tj. na jakou část terénu dopadá kolik světla (stín vs. dostatek světla)). Pokud definujeme rovinu určenou souřadnicovými osami x a y a promítneme do ní norm. vektor roviny trojúhelníku, pak azimut tohoto průmětu definuje expozici terénu.

1. *Pro všechny trojúhelníky triangulace:* $\forall t_i \in DT$ *:*
2. *Výpočet normálového vektoru roviny trojúhelníku:*

$$u_x = x_2 - x_1;\ \ u_y = y_2 - y_1;\ \ u_z = z_2 - z_1;$$
$$v_x = x_2 - x_3;\ \ u_y = y_2 - y_3;\ \ u_z = z_2 - z_3;$$
$$n_x = (u_y \cdot v_z - u_z \cdot v_y)$$
$$n_y = -(u_x \cdot v_z - u_z \cdot v_x)$$

3. *Výpočet expozice:* $A = atan2\left(\frac{n_x}{n_y}\right)$

Tvorba vrstevnic

Vrstevnice jsou tvořeny pomocí lineární interpolace, která je založená na analytické geometrii, kde jejím úkol je najít průsečnici roviny tvořenou trojúhelníkem a vodorovné roviny s výškou z. Koncové body A, B této průsečnice se určí z podobnosti trojúhelníků.

$$x_A = \frac{x_3 - x_1}{z_3 - z_1} \cdot (z - z_1) + x_1$$

$$y_A = \frac{y_3 - y_1}{z_3 - z_1} \cdot (z - z_1) + y_1$$

$$x_B = \frac{x_2 - x_1}{z_2 - z_1} \cdot (z - z_1) + x_1$$

$$y_B = \frac{y_2 - y_1}{z_2 - z_1} \cdot (z - z_1) + y_1$$

Výsledkem je tedy vektor bodů které jsou lomovými body vrstevnice. Po spojení těchto bodů vznikne hrana tvořící vrstevnici v daném trojúhelníku o dané výšce.

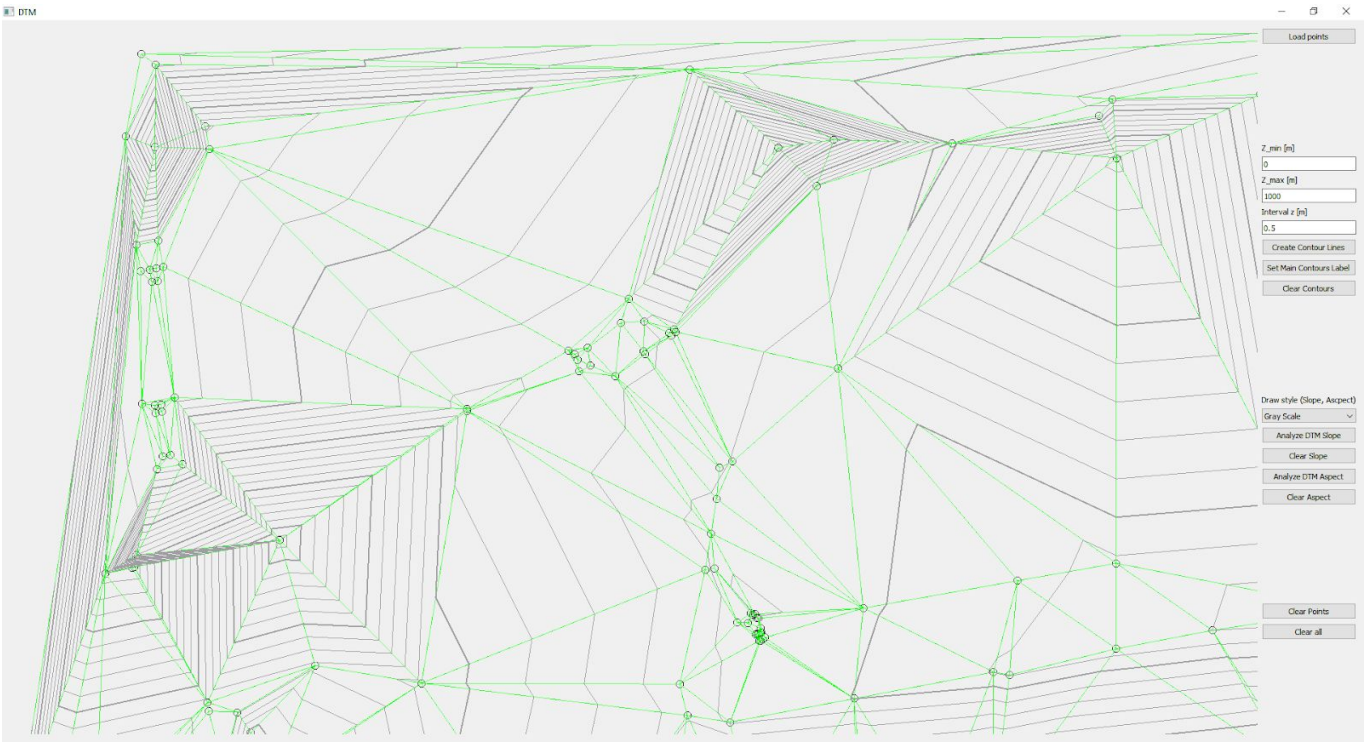
Automatický popis vrstevnic

Popis byl vytvořen pro hlavní vrstevnice (každou pátou). Ve funkci contourLine bylo zapotřebí zajistit, aby nevracela pouze vektor hran vrstevnic, ale i vektor příslušných výšek. Výšky byly následně pomocí funkce setMainContours převedeny do třídy Draw a zde byly využity pro vykreslování hlavních vrstevnic. Samotný text byl umístěn do průměrných souřadnic mezi počátkem a koncem každé hrany hl. vrstevnice bez natočení textu do kopce.

Problematické situace a jejich rozbor

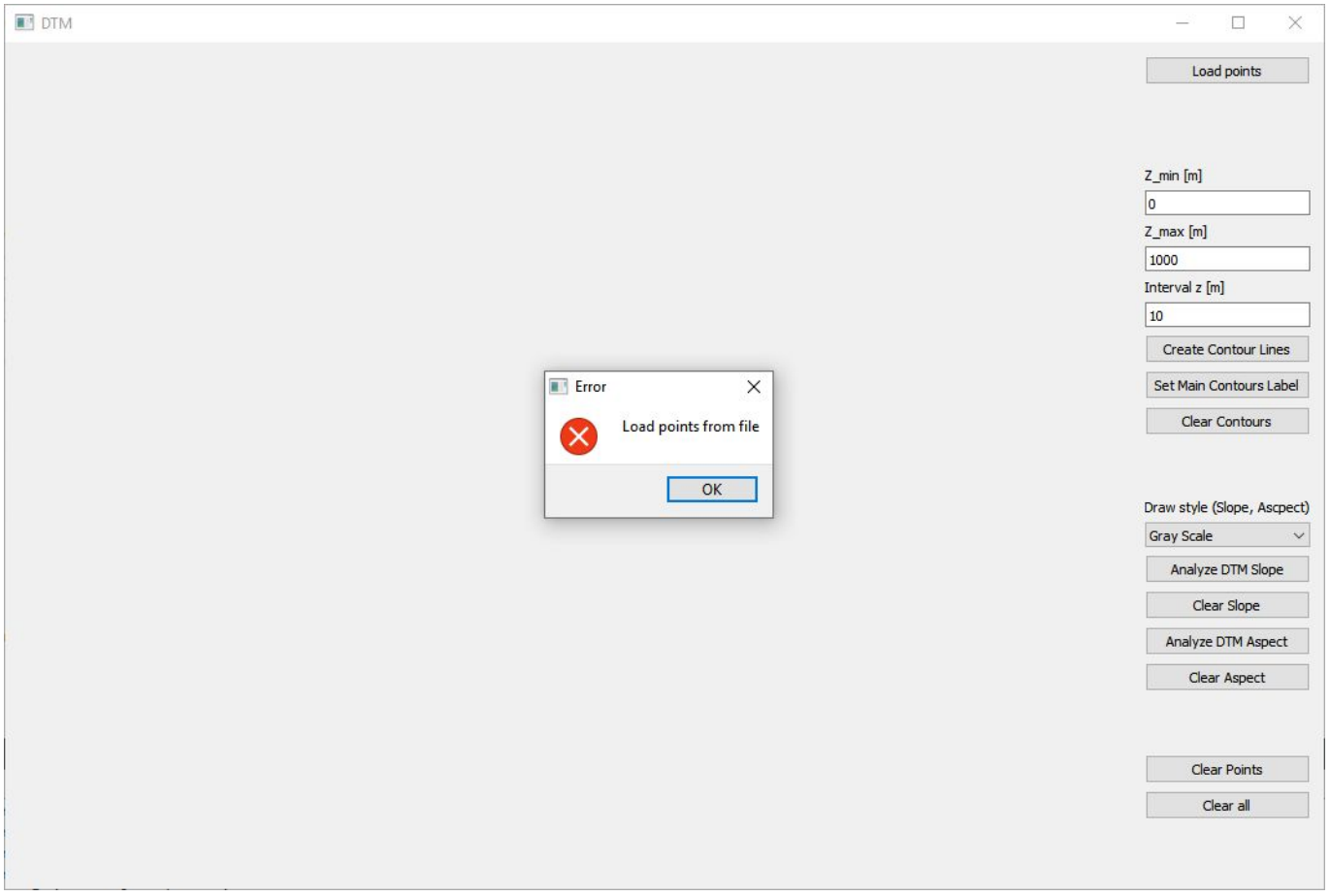
Při testování na reálných datech je program omezen velikostí vykreslovacího okna. Uživatel musí kontrolovat min a mix Z souřadnice a interval vygenerovaných vrstevnic. Chybí také definice povinné a ostrovní hrany.

Ukázka vygenerování vrstevnic (interval 0.5 m) nad reálnými daty



Při spuštění analýzy sklonu svahu a orientace a při vygenerování vrstevnic v situaci, kdy nebyly načteny vstupní body, aplikace spadla. Aplikace byla doplněna o chybovou hlášku, která se zobrazí, když tato situace nastane.

Chybová hláška



Vstupní data

Uživatel může vstupní body definovat kliknutím myší v kreslícím okně nebo může načíst textový soubor souřadnic vstupních bodů ve formátu[x y z].

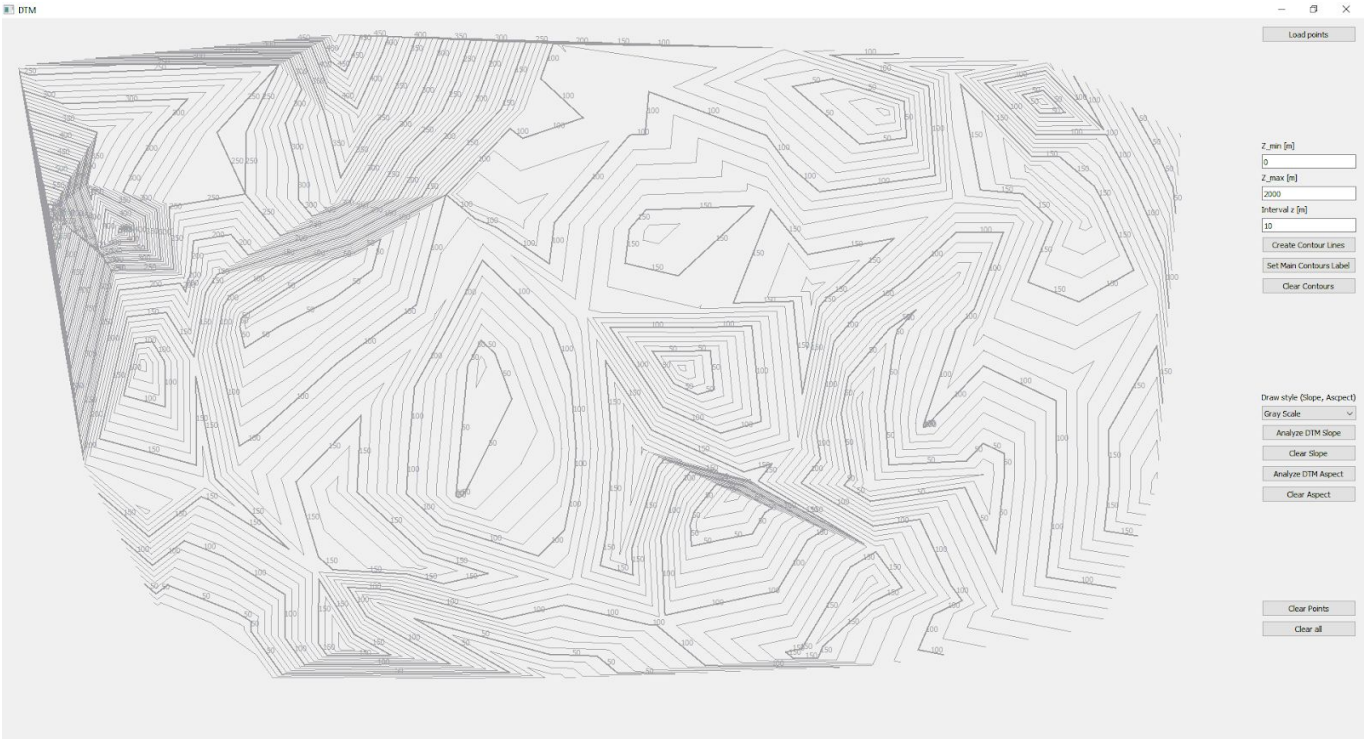
Výstupní data

Výstupem je grafická aplikace, která generuje nad vstupními body vrstevnice, digitální model terénu podle Delaunay triangulace. Dále analyzuje sklon a orientaci DMT.

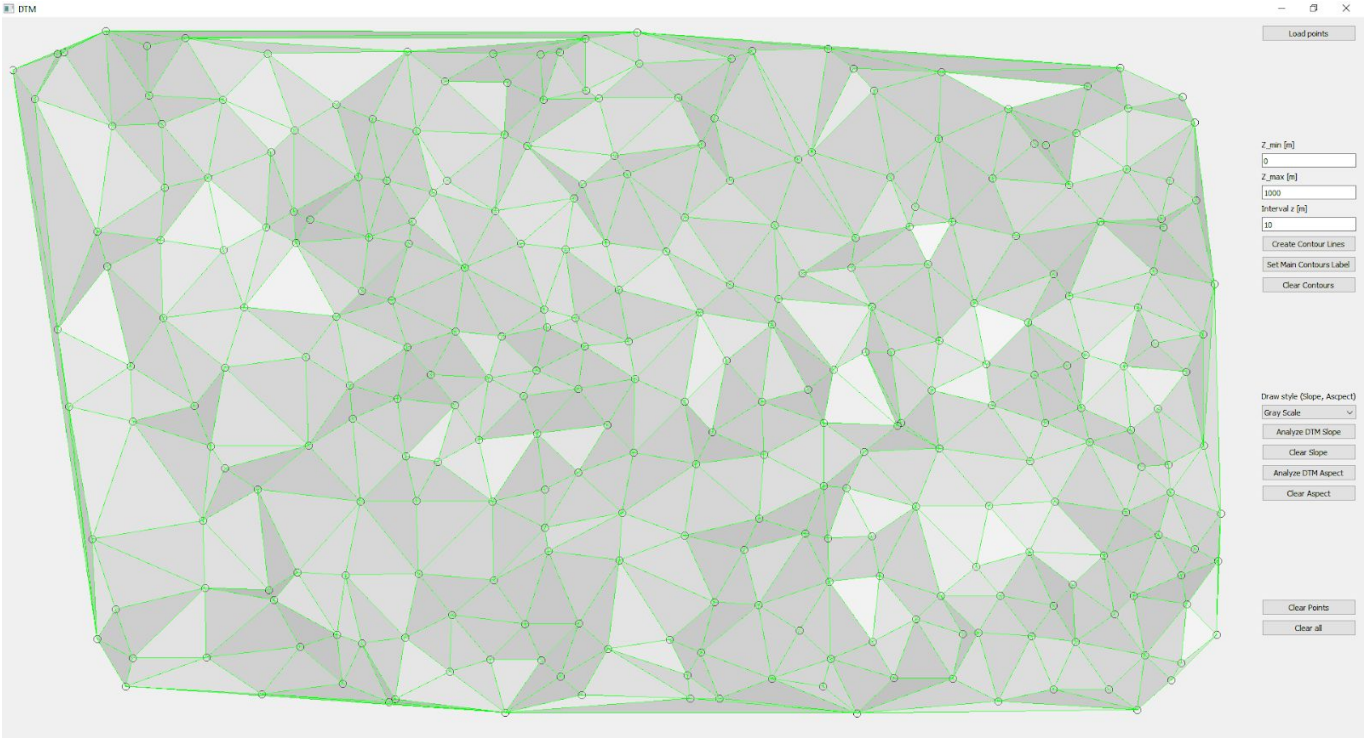
Vytvořená aplikace a zhodnocení činnosti algoritmu

Následující snímky vytvořené aplikace zobrazují řešení daných situací:

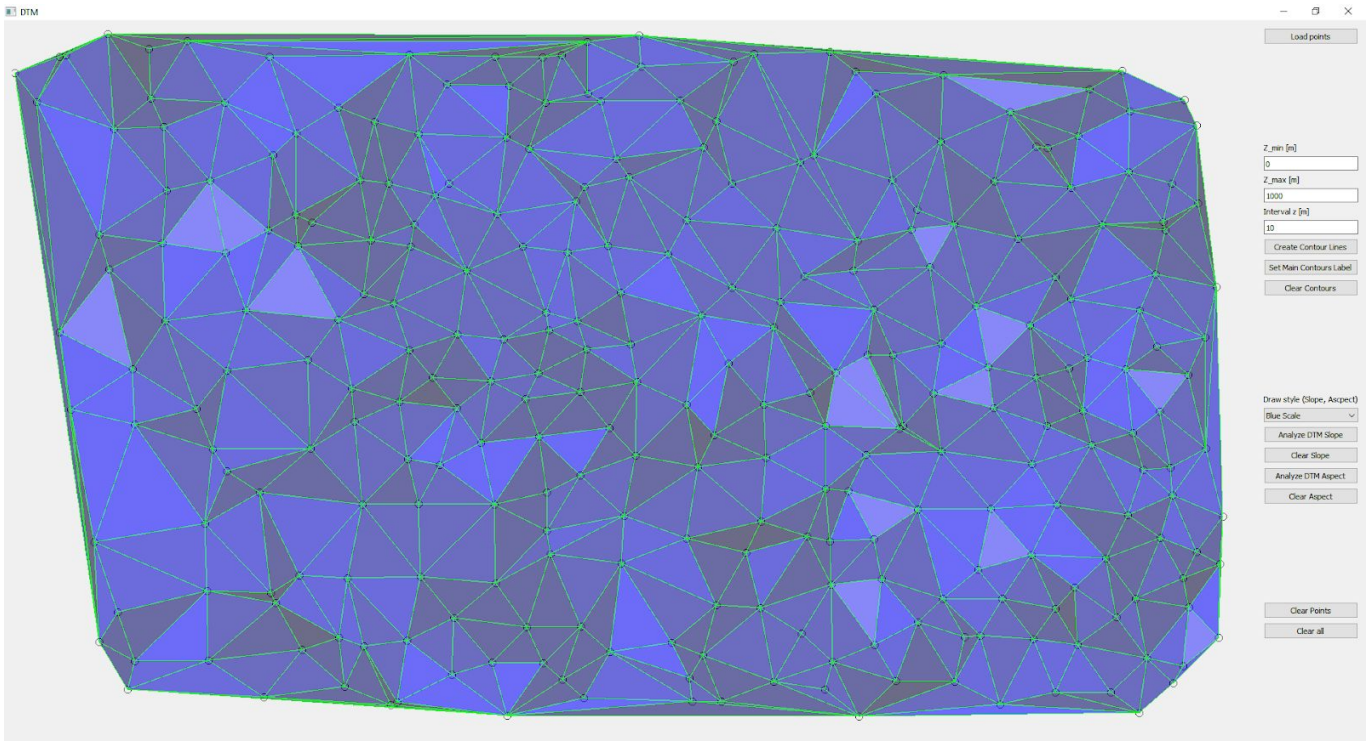
Vygenerované vrstevnice a popis hlavních vrstevnic



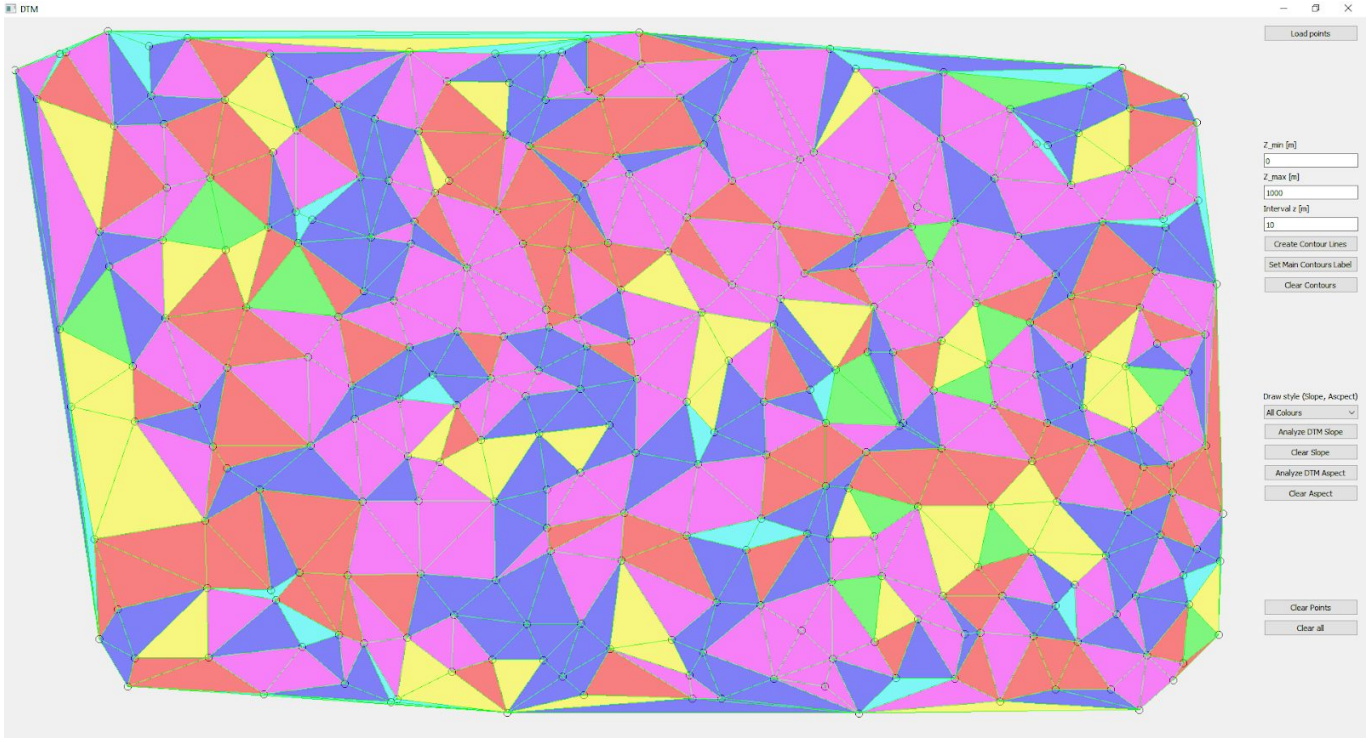
Analýza svahu (stupnice šedé barvy)



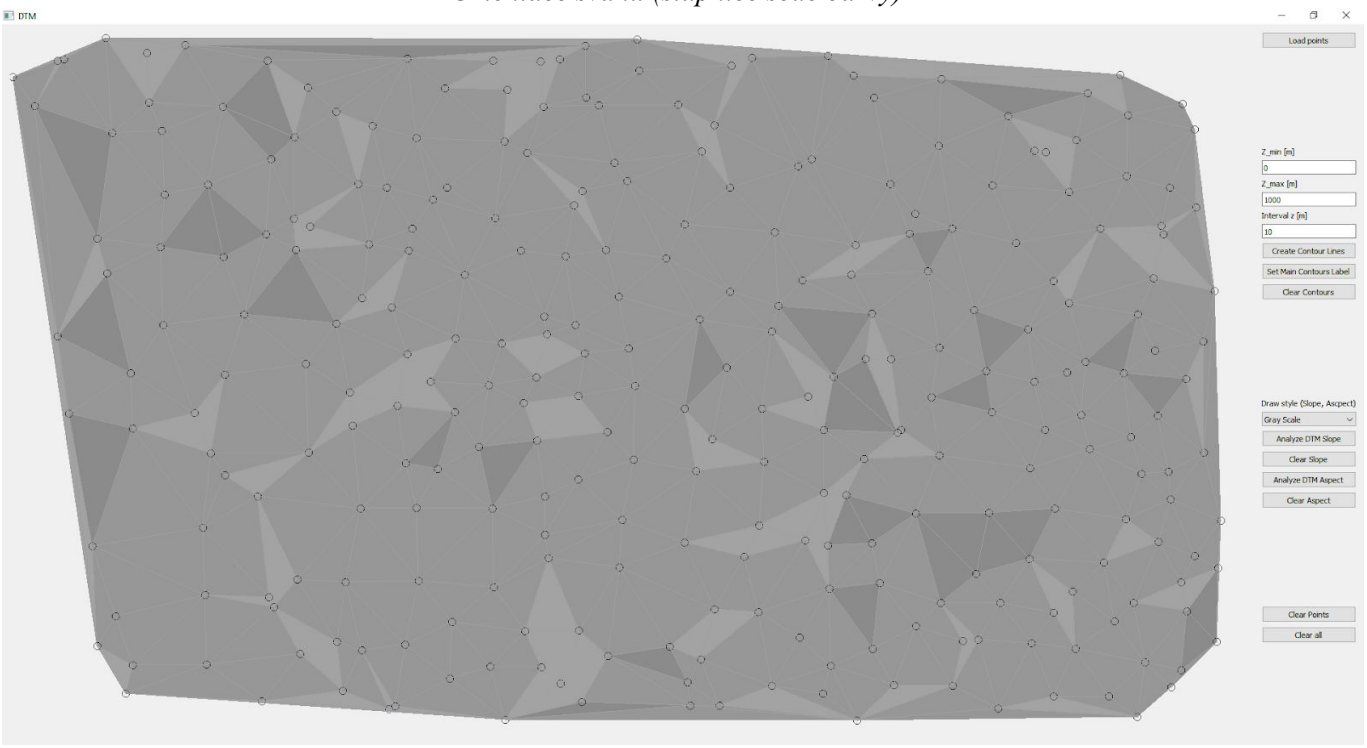
Analýza svahu (stupnice modré barvy)



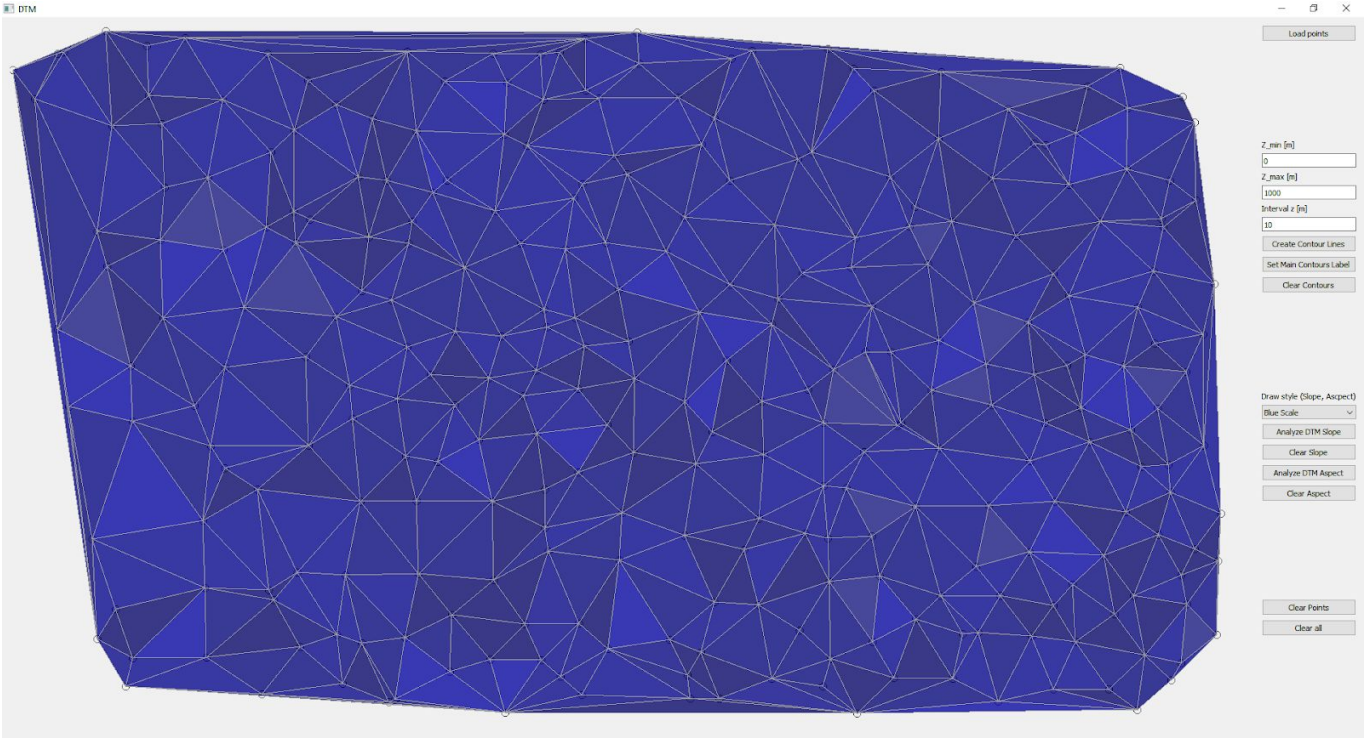
Analýza svahu (Různé barvy)



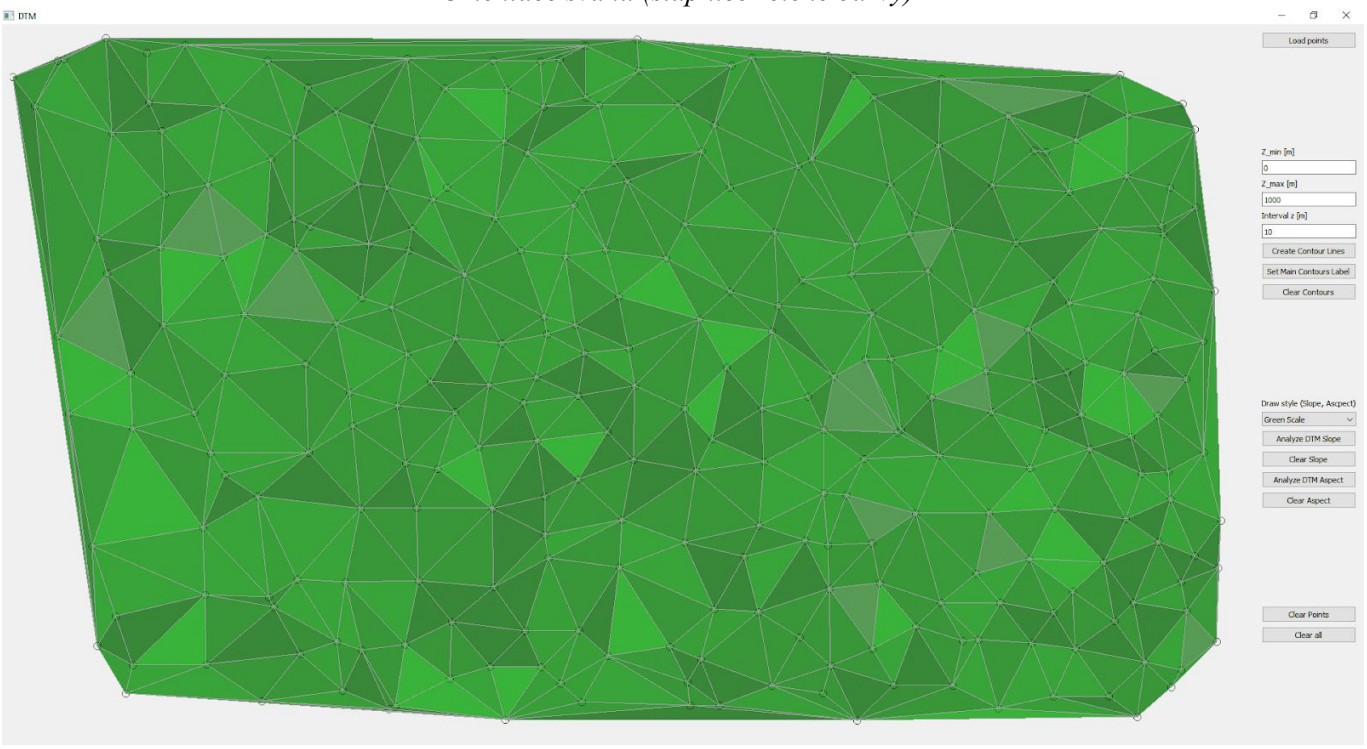
Orientace svahu (stupnice šedé barvy)



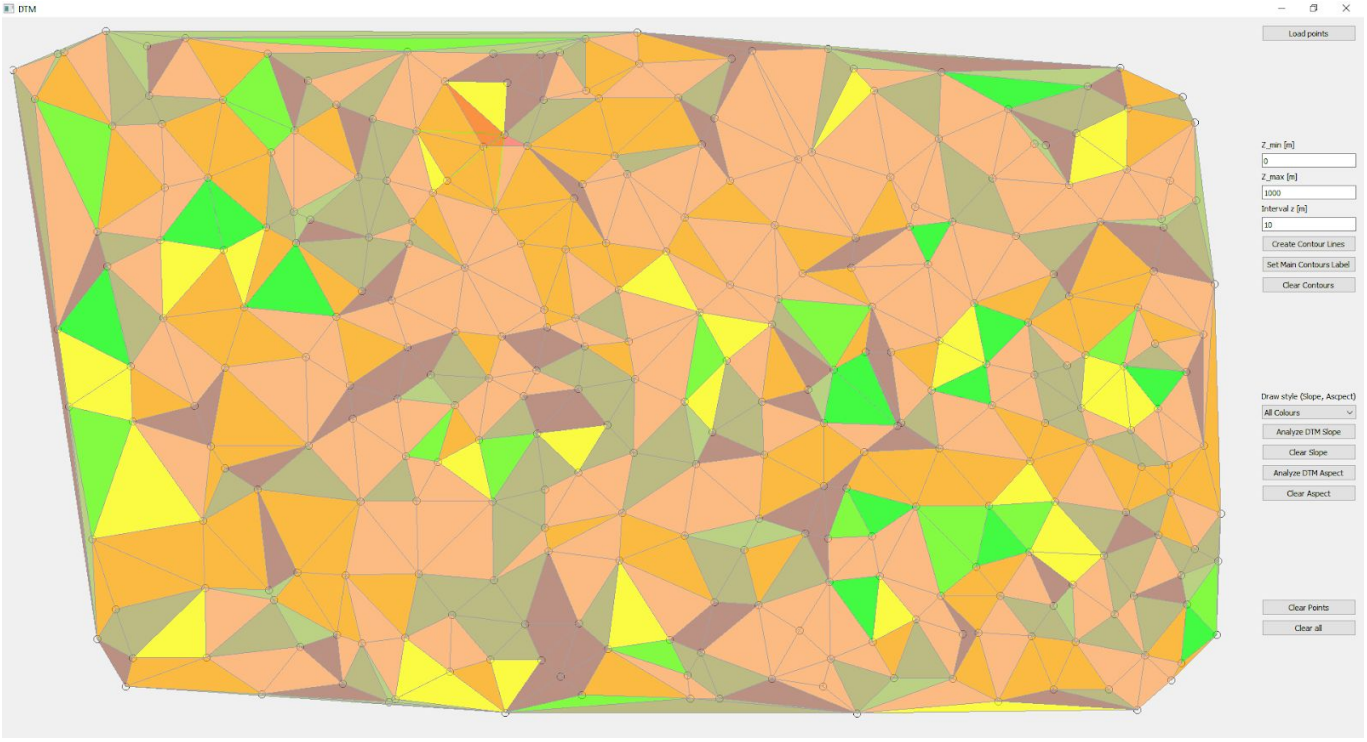
Orientace svahu (stupnice modré barvy)



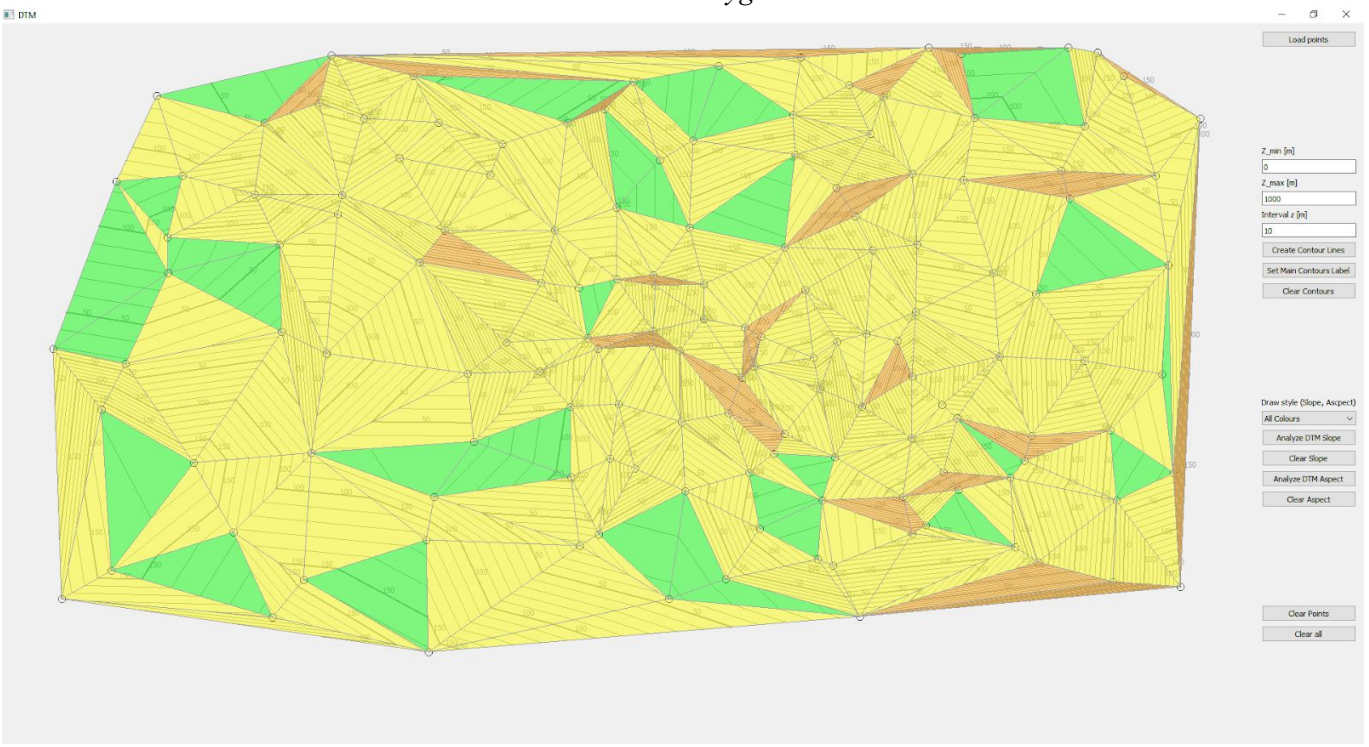
Orientace svahu (stupnice zelené barvy)



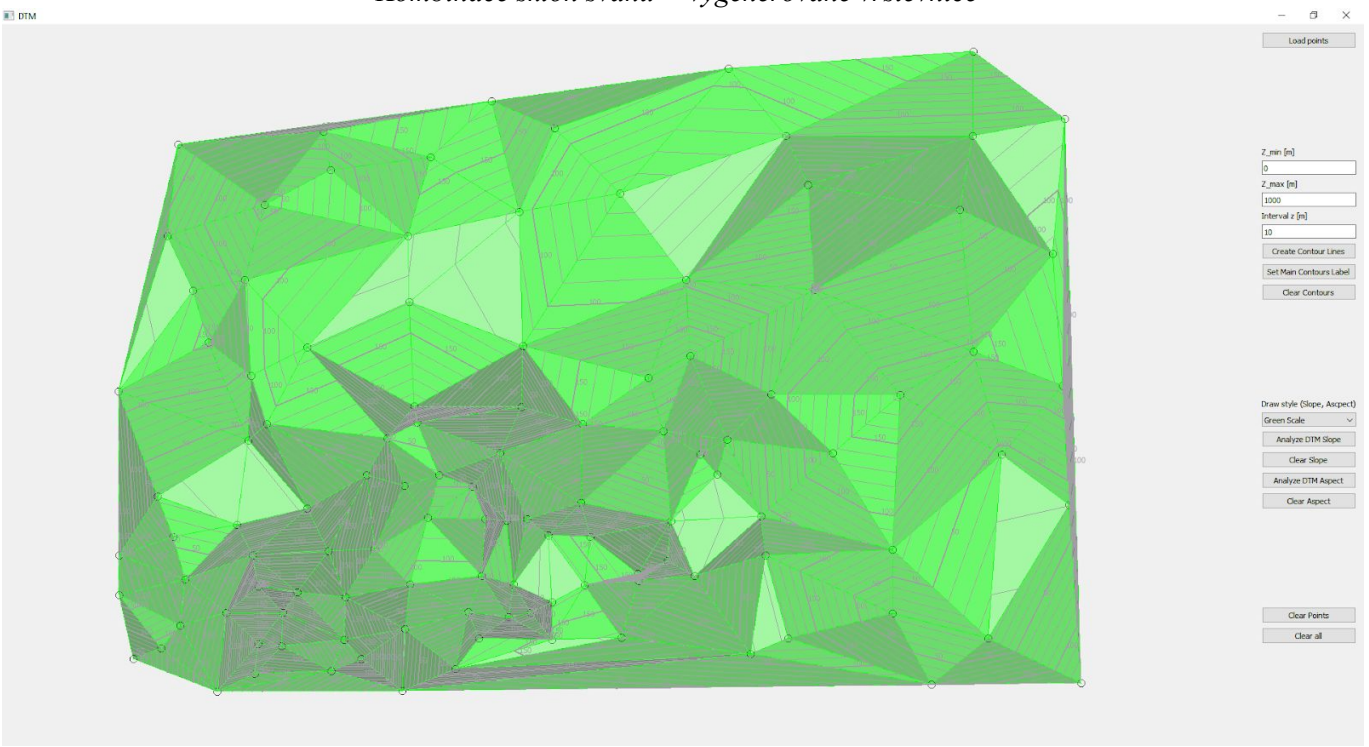
Orientace svahu (různé barvy)



Kombinace orientace svahu + vygenerované vrstevnice



Kombinace sklon svahu + vygenerované vrstevnice



Z předešlých obrázku je patrné, že algoritmu dělají problémy velmi ostré trojúhelníky. Tyto trojúhelníky vznikají většinou na okrajích oblasti vstupních bodů a při nevhodném rozložení vstupních bodů.

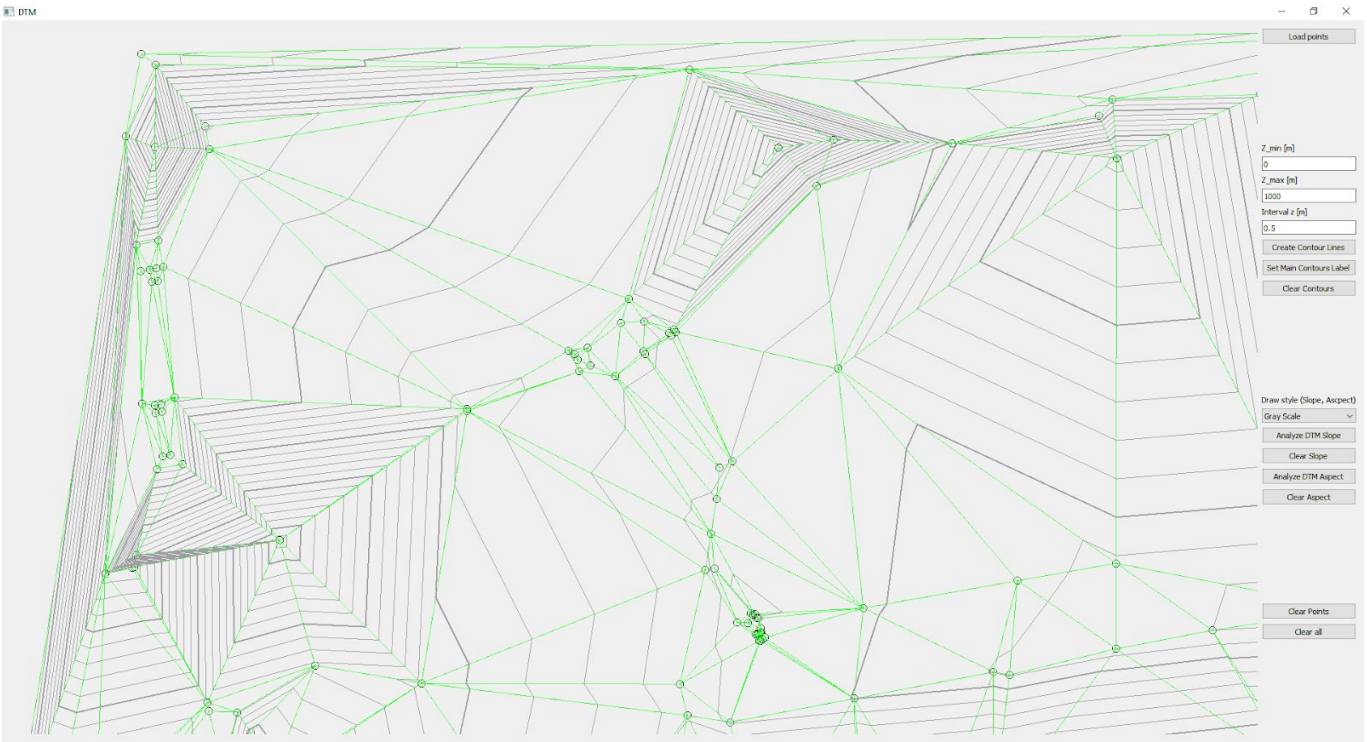
Pro snadnější orientaci ve výsledcích analýzy svahu a orientace svahu je pro uživatele vhodné, aby vygeneroval zároveň vrstevnice a popisy hlavních vrstevnic. Díky průhlednosti barvy se zobrazí barevné výsledky analýzy společně s vrstevnicemi. Dále by uživateli pomohlo generování legendy na základě zvoleném způsobu vykreslení analýzy.

Vrstevnice díky hranám obsahují lomené body. Pro lepší grafické výsledky by bylo vhodné zahrnout do algoritmu další postup na vyhlazení vrstevnic.

Při testování nad vstupními daty, které jsou přílohou, nedošlo k žádným zásadním chybám při sestrojení Delaunay triangulace a následném vygenerování vrstevnic.

Při testování na reálných datech je program omezen velikostí vykreslovacího okna. Uživatel musí kontrolovat min a mix Z souřadnice a interval vygenerovaných vrstevnic. Chybí také definice povinné a ostrovní hrany.

Ukázka vygenerování vrstevnic (interval 0.5 m) nad reálnými daty



Dokumentace

Třídy, datové položky a metody

Aplikace obsahuje sedm tříd - Algorithms, Draw, sortByX, Edge, QPoint3D, Triangle a Widget. Každá třída je zastoupena hlavičkovým souborem a zdrojovým souborem. V hlavičkových souborech jsou definovány společně se třídou její proměnné a metody.

- **Třída Algorithms**

  Třída Algorithms obsahuje celkem čtyři metody, které jsou použity pro vyřešení zadaného problému. Datovými typy metod byly QPointF a QPolygonF, oba s plovoucí desetinnou čárkou.

  int getPointLinePosition(QPointF &q, QPointF &p1, QPointF &p2);

  Tato metoda určuje pozici bodu *q* vůči zadané hraně polygonu *P*. Vrací hodnoty 1 (bod leží v levé polorovině), 0 (bod leží v pravé polorovině) a -1 (bod leží na hraně). Ošetření případu, že bod leží na hraně, bylo vytvořeno na základě podmínky sestrojení trojúhelníku. Trojúhelník lze sestrojit tehdy pokud součet délek dvou stran je větší než délka třetí strany. Pokud podmínka neplatí, body leží v rovině. Jelikož uživatel kliká myší a né vždy klikne na hranu byla zvolena tolerance 0,2. Pokud podmínku, že leží na hraně nesplňuje, je následně vypočítán determinant vektorů p1p2 a qp1. Pokud je determinant větší než 0 funkce vrací hodnotu 1, pokud je záporný, vrací hodnotu 0.

  int getPointLinePosition(QPoint3D &q, QPoint3D &p1, QPoint3D &p2);

  Metoda určuje pozici bodu vůči přímce (v levé polorovině, v právě, na přímce).

  void circleCenterAndRadius(QPoint3D &p1, QPoint3D &p2, QPoint3D &p3, double &r, QPoint3D &s);

  Funkce určující střed kružnice.

  int findDelaunayPoint(QPoint3D &pi, QPoint3D &pj, std::vector<QPoint3D> &points);

  Metoda, která určuje vhodný bod pro Delaunay triangulaci.

  double dist(QPoint3D &p1, QPoint3D &p2);

  Metoda, která počítá vzdálenost 2 bodů.

  int getNearestpoint(QPoint3D &p, std::vector<QPoint3D> &points);

  metoda k nalezení nejbližšího bodu.

  std::vector<Edge> DT(std::vector<QPoint3D> &points);

  Metoda generující Delaunay triangulaci na základě vstupní množiny bodů.

  void updateAEL(Edge &e, std::list<Edge> &ael);

  Metoda, která přidá hranu do seznamu hran.

  QPoint3D getContourPoint(QPoint3D &p1, QPoint3D &p2, double z);

  Metoda, která hledá body na vrstevnici.

  std::vector<Edge> contourLines(std::vector<Edge> &dt, double z_min, double z_max, double dz);

  Metoda, která generuje vrstevnice.

  double getSlope(QPoint3D &p1, QPoint3D &p2, QPoint3D &p3);

  Metod, která vrací hodnotu sklonu.

  double getAspect(QPoint3D &p1, QPoint3D &p2, QPoint3D &p3);

Metoda, která vrací hodnotu orientace svahu.

`std::vector<Triangle> analyzeDTM(std::vector<Edge> &dt);`
metoda, která analyzuje DTM podle sklonu a orientace.

- **Třída Draw**

`void mousePressEvent(QMouseEvent *event);`
Tato metoda ukládá souřadnice bodu *p*, které uživatel zadá kliknutím myši do kreslícího pole.

`void paintEvent(QPaintEvent *e);`
Touto metodou se vykreslují body, hrany DTM, vygenerované vrstevnice, sklon a orientace DTM.

`void setPoints(std::vector<QPoint3D> &points_){points=points_;}`
Metoda, která načítá vstupní body.

`std::vector<QPoint3D> & getPoints(){return points;}`
Metoda, která vrací souřadnice načtených bodů.

`void setDT(std::vector<Edge> &dt_){dt = dt_;}`
Metoda, která načítá hrany DTM.

`std::vector<Edge> & getDT(){return dt;}`
Metoda, která vrací hrany DTM.

`void setContours(std::vector<Edge> &contours_){contours = contours_;}`
Metoda, která načítá vygenerované vrstevnice.

`std::vector<Edge> & getContours(){return contours;}`
Metoda, která vrací vygenerované vrstevnice.

`void setDTM(std::vector<Triangle> &dtm_){dtm = dtm_;}`
Metoda, která načítá vygenerovaný DTM.

`std::vector<Triangle> & getDTM(){return dtm;}`
Metoda, která vrací vygenerovaný DTM.

`void loadFile(std::string &path);`
Metoda, která načítá cestu vstupní souboru.

`void setMainContours(std::vector<Edge> &main_contours_){main_contours = main_contours_;}`
Metoda, která načítá hlavní vrstevnice.

`void setLabelContours(std::vector<Edge> &label_contours_){label_contours = label_contours_;}`
Metoda, která načítá název hlavní vrstevnice.

`std::vector<Edge>& getMainContours() {return main_contours;}`
Metoda, která vrací hlavní vrstevnice.

`std::vector<Edge>& getContoursLabel() {return label_contours;}`
Metoda, která vrací název hlavní vrstevnice.

`void setAspectDTM(std::vector<Triangle> &aspect_dtm_){aspect_dtm = aspect_dtm_;}`
Metoda, která načítá hodnotu sklonu DTM.

std::vector<Triangle>&getAspectDTM() {return aspect_dtm;}
Metoda, která vrací hodnotu sklonu DTM.

void setStyle (int style_){style=style_;}
Metoda, která načítá způsob vykreslení sklonu a orientace.

- **Třída sortByX**
  Tato třída třídí vstupní body podle souřadnice X

- **Třída Edge**
  Třída k uložení hrany triangulace a vrstevnice.

- **Třída QPoint3D**
  Třída definuje souřadnice vstupního bodu (X, Y, Z)

- **Třída Triangle**
  Tato třída ukládá trojúhelníky definované vrcholovými body a ukládá informace o sklonu a orientaci trojúhelníku.

- **Třída Widget**
  void on_pushButton_7_clicked();
  Po kliknutí na tlačítko *Create Contour Line* jsou vygenerovány vrstevnice.

  void on_pushButton_11_clicked();
  Po kliknutí smaže vstupní souřadnice.

  void on_pushButton_12_clicked();
  Po kliknutí smaže vygenerované vrstevnice a názvy hl. vrstevnic

  void on_lineEdit_editingFinished();
  Prostor, pro definování minimální hodnoty vrstevnic.

  void on_lineEdit_2_editingFinished();
  Prostor, pro definování maximální hodnoty vrstevnic.

  void on_lineEdit_3_editingFinished();
  Prostor, pro definování intervalu vrstevnic.

  void on_pushButton_2_clicked();
  Po kliknutí je provedena analýza skolnu DMT.
  void on_pushButton_3_clicked();
  Po kliknutí je smazán výsledek analýzy sklonu DMT

  void on_pushButton_4_clicked();
  Po kliknutí je smazán obsah grafického okna.

  void on_pushButton_clicked**();**
  Umožňuje načíst uživateli textový soubor souřadnic bodů.

  void on_pushButton_5_clicked**();**
  Po kliknutí jsou vygenerovány názvy hlavních vrstevnic.

  void on_pushButton_6_clicked**();**
  Po kliknutí je provedena analýza orientace svahů DMT.

  void on_pushButton_8_clicked**();**
  Po kliknutí je smazán výsledek analýzy orientace svahů DMT.

Závěr

V rámci této úlohy byla vytvořena aplikace, která je schopna na základě vygenerovaných bodů zkonstruovat trojúhelníkovou síť pomocí Delaunay triangulace a dále vizualizovat sklon i expozici sítě a popsat hlavní vrstevnice.

V Praze dne 17. 1. 2021

Bc. Michal Zíma,
Bc. Tomáš Lauwereys

Přílohy:

[1] data_real.txt (skutečná data z terénu)
[2] data.txt (data vytvořená k testování)