České vysoké učení technické v Praze
Fakulta stavební
Katedra geomatiky

**Technická zpráva**

# Algoritmy v digitální kartografii

## Úloha č. 3: Digitální model terénu

**Bc. Pane Kuzmanov**
**Bc. František Mužík**

Studijní program: Geodézie a kartografie
Specializace: Geomatika

Praha 2021

## 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.

**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: 4 týdny

# 1 Údaje o bonusových úlohách

## 1.1 Triangulace nekonvexní oblasti zadané polygonem (+5b)

Tato bonusová úloha nebyla řešena.

## 1.2 Výběr barevných stupnic při vizualizaci sklonu a expozice (+3b)

Byla implementována možnost výběru dvou barevných palet pro každou z vizualizací. Pro sklon terénu je možné zvolit jednu z variant "Grayscale" – odstíny šedé nebo "Yellow to Red" – žluto – červená paleta. Ve výběru pro expozici lze najít dvě možnosti: KM – zvolená námi (obr. 3) nebo barevnou paletu, kterou používá ArcGIS (obr. 4). Detailnější popis je v kapitolách 3.2 a 3.3.

## 1.3 Automatický popis vrstevnic (+3b)

Byl implementován automatický popis hlavních vrstevnic. Hlavní vrstevnice je každá pátá vrstevnice, která je zároveň výrazněji vykreslená. Více informací je v kapitole 3.4.1.

## 1.4 Automatický popis vrstevnic respektující kartografické zásady (+10b)

Tato bonusová úloha nebyla řešena.

## 1.5 Algoritmus pro automatické generováních terénních tvarů (+10b)

Bylo implementováno automatické tvoření následujících terénních tvarů: kupa, údolí, spočinek, hřbet. Zároveň je součástí aplikace možnost generování náhodných bodů. Popis řešení je v kapitole 3.5.

## 1.6 3D vizualizace terénu s využitím promítání (+10b)

Tato bonusová úloha nebyla řešena.

## 1.7 Barevná hypsometrie (+5b)

Tato bonusová úloha nebyla vyřešena.

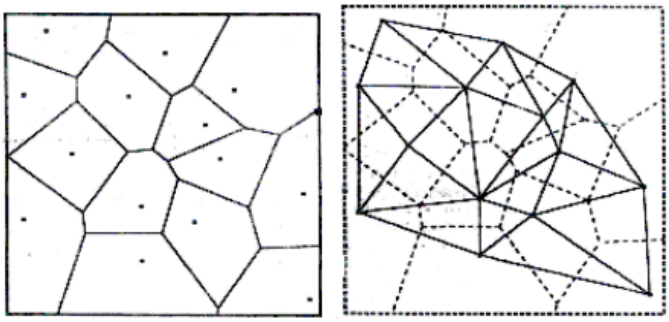
# 2 Popis a rozbor problému

Je zadaná množina bodů s prostorovými souřadnicemi X,Y,Z. Nad těmito body je vytvořen polyedrický digitální model terénu s využitím Delaunayho triangulace. Součástí řešení je taktéž tvorba vrstevnic, včetně jejich popisu. Dále je možné zobrazit sklon terénu (tedy sklon jednotlivých trojúhelníků v triangulaci) a expozici ke světovým stranám (rozděleno na 8 stran).

# 3 Popisy algoritmů formálním jazykem

## 3.1 Delaunay triangulace

Jedná se o nejčastěji využívanou metodu triangulace. Základní funkčnost metody spočívá v rozdělení množiny bodů tzv. Thiessenovými polygony, jejichž vlastností je ohraničení každého z bodů vstupní množiny tak, aby z každého místa v ohraničeném polygonu byl nejbližším bodem právě ten ohraničený (obr. 2). Hrany polygonů tvoří osu spojnice dvou bodů.



Obrázek 1: Ukázka Thiessenových polygonů (vlevo) a výsledné triangulace (vpravo) [4].

Jestliže opsaná kružnice nad spojenou hranou dvou sousedních bodů neprotne jiný bod nebo jestliže se jiný bod nenachází uvnitř kružnice. Průběžným výpočtem je v algoritmu vytvářen seznam hran (AEL – Active Edge List), pro které nebyl nalezen třetí bod pro triangulaci. Při nalezení nové hrany je potřeba zjistit, zda – se již nenachází v AEL a tedy není případně duplicitní. Duplicitní hrany jsou odstraněny. Ve chvíli, kdy je AEL prázdný, je triangulace hotova.

Nutné podmínky pro vytvoření triangulace:

- Žádný bod se nenachází uvnitř kružnice opsané libovolnému trojúhelníku triangulace.
- Libovolné čtyři body by neměly ležet na kružnici – pak je triangulace nejednoznačná.
- Maximalizace nejmenšího úhlu v trojúhelníku.
- Splnění lokálních i globálních kritérií (vyvarování se tupých a velmi malých úhlů v trojúhelníku, minimální suma délek stran...)

**Implementace algoritmu:**

1. Nalezení náhodného a nejbližšího bodu: $p_1 = rand(P), \|p_1 - p_2\| = min.$
2. Vytvoř hranu $e = (p_1 p_2)$
3. $\underline{p} = arg\ min_{\forall p_i \in \sigma_L(e)} r'(k_i), k_i = (a, b, p_i), e = (a, b)$
4. Pokud neexistuje $\underline{p}$, prohoď orientaci $e \leftarrow (p_2 p_1)$. Jdi na 3)
5. Výpočet zbývajících hran trojúhelníka: $e_2 = (p_2, \underline{p}), e_3 = (\underline{p}, p_1)$
6. Přidání 3 hran do AEL: $AEL \leftarrow e, AEL \leftarrow e_2, AEL \leftarrow e_3$

7. Přidání 3 hran do DT: $DT \leftarrow e, DT \leftarrow e_2, DT \leftarrow e_3$
8. Dokud AEL není prázdný:
9. Vybrání první hrany z AEL: $AEL \rightarrow e, e = (p_1 p_2)$
10. Prohození její orientace: $e = (p_2 p_1)$
11. $\underline{p} = arg\ min_{\forall p_i \in \sigma_L(e)} r'(k_i), k_i = (a, b, p_i), e = (a, b)$
12. Pokud existuje $\underline{p}$:
13. Zbývající hrany trojúhelníka: $e_2 = (p_2, p), e_3 = (\underline{p}, p_1)$
14. Přidání hrany do DT, ale nikoliv do AEL: $DT \leftarrow e$
15. Přidání hrany do DT i do AEL: $add(e_2, AEL, DT), add(e_3, AEL, DT)$

## 3.2 Výpočet sklonu

Výpočet sklonu trojúhelníku je učen jako úhel $\varphi$ mezi svislicí a normálou trojúhelníka $n$. Výpočet probíhá vektorově nad každým z trojúhelníků DMT.

1. Určení vektorů $\overrightarrow{u} = (u_x, u_y, u_z) = (x_1 - x_2, y_1 - y_2, z_1 - z_2)$, $\overrightarrow{v} = (v_x, v_y, v_z) = (x_3 - x_2, y_3 - y_2, z_3 - z_2)$
2. Výpočet normálového vektoru trojúhelníka: $\overrightarrow{n} = (n_y, n_y, n_z) = (u_y \cdot v_z - v_y \cdot u_z, -u_x \cdot v_z + v_x \cdot u_z, u_x \cdot v_y - v_x \cdot u_y)$
3. Sklon $\varphi = \dfrac{n_z}{|n|}$

### 3.2.1 Výběr barevné palety

Trojúhelníky jsou vizualizovány na základě hodnoty $\varphi$. Vizualizace je možná s užitím dvou barevných palet. První možností jsou odstíny šedé

Obrázek 2: Barevné palety pro vizualizaci sklonu terénu. Vlevo odstíny šedé a vpravo žluto – červená stupnice.

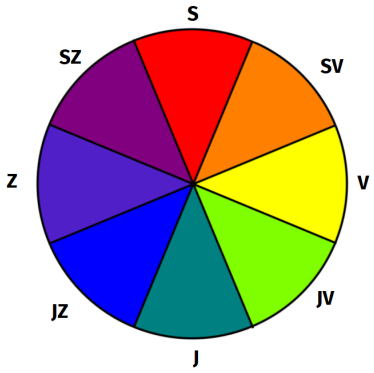
## 3.3 Výpočet expozice

Výpočet expozice probíhá obdobným způsobem jako výpočet sklonu. V podstatě jde o úhel mezi x – ovou a y – ovou složkou normály trojúhelníka. Expozice určuje náklon trojúhelníka k jedné ze světových stran. V aplikaci je uvažováno s 8 možnostmi (jih, jihozápad, západ, severozápad, sever, severovýchod, východ a jihovýchod).

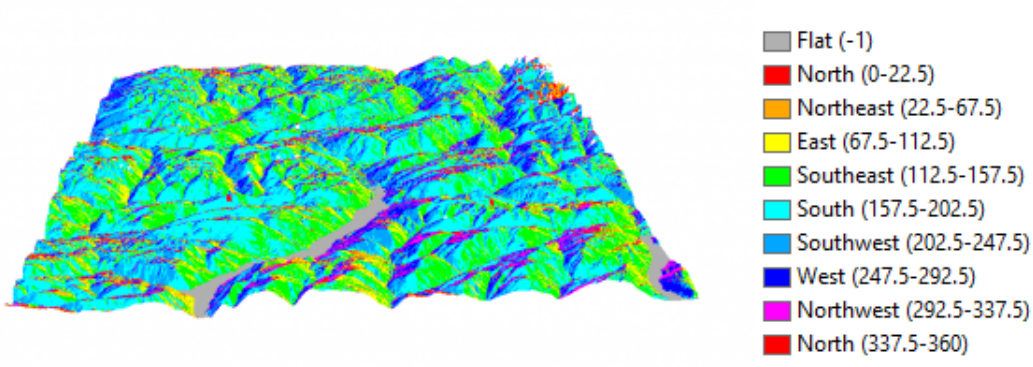
1. Určení vektorů $\overrightarrow{u} = (u_x, u_y, u_z) = (x_1 - x_2, y_1 - y_2, z_1 - z_2)$, $\overrightarrow{v} = (v_x, v_y, v_z) = (x_3 - x_2, y_3 - y_2, z_3 - z_2)$
2. Výpočet normálového vektoru trojúhelníka: $\overrightarrow{n} = (n_y, n_y, n_z) = (u_y \cdot v_z - v_y \cdot u_z, -u_x \cdot v_z + v_x \cdot u_z, u_x \cdot v_y - v_x \cdot u_y)$
3. Určení expozice $exp = arccos\frac{n_z}{|n|}$

### 3.3.1 Výběr barevné palety

Pro expozici jsou na výběr dvě možnosti barevných palet. Námi zvolená KM paleta a standardní paleta, kterou využívá např. ArcGIS.




Obrázek 3: KM barevná paleta pro vizualizaci expozice terénu.




Obrázek 4: Vizualizace expozice terénu dle ArcGIS [1].

## 3.4 Výpočet vrstevnic

Výpočet vrstevnic probíhá na základě vstupní Delaunayho triangulace, minimální a maximální výšky a rozestupu mezi vrstevnicemi. Jedná se o určení vrstevnic lineární interpolací, předpoklad je tedy takový, že spád terénu je mezi podrobnými body konstantní. Jedná se o nejjednodušší a rozšířený způsob tvorby vrstevnic.

Nejprve je nutné zjistit, zda rovina $\rho$ protíná stranu $(p_i, p_{i+1})$ trojúhelníka:

1. Jestliže $(z - z_i)(z - z_{i+1}) < 0$, pak trojúhelník protíná rovinu $\rho$ v obecných bodech.
2. Jestliže $(z - z_i)(z - z_{i+1}) > 0$, pak trojúhelník neprotíná rovinu $\rho$.
3. Jestliže $(z - z_i)(z - z_{i+1}) = 0$, pak jedna ze stran trojúhelníka leží v rovině $\rho$.

Výpočet souřadnic průsečíku A,B vrstevnice o nadmořské výšce $z$ s trojúhelníkem tvořeným body 1,2,3:

$$
\begin{aligned}
x_A &= \frac{x_3 - x_1}{z_3 - z_1}(z - z_1) + x_1 \\
y_A &= \frac{y_3 - y_1}{z_3 - z_1}(z - z_1) + y_1 \\
x_B &= \frac{x_2 - x_1}{z_2 - z_1}(z - z_1) + x_1 \\
y_B &= \frac{y_2 - y_1}{z_2 - z_1}(z - z_1) + y_1
\end{aligned}
$$

### 3.4.1 Popis vrstevnice

Automatický popis vrstevnic byl implementován pro každou hlavní vrstevnici (tedy každou pátou vrstevnici). Výškový údaj je vypsán nad vrstevnicí a umístěn v polovině triangulačního trojúhelníka.

Ve třídě Draw je implementace následovná:

```
//Draw main contours label
 for (Edge mcl : main_contours_label)
 {
    //Start and end point of the contour edge
    QPoint3D sl_point = mcl.getStart();
    QPoint3D el_point = mcl.getEnd();

    //Get label coordintes
    QPoint3D label_point( (sl_point.x()+el_point.x())/2,
                          (sl_point.y()+el_point.y())/2);

    //Draw Z coordinate
    QString z = QString::number(sl_point.getZ());
    qp.drawText(label_point, z);
 }
```

## 3.5 Generování terénních tvarů

Implementace algoritmů pro generování terénních tvarů a náhodných bodů, je popsána níže zvlášť pro jednotlivé případy. Uživatel může manuálně nastavit počet vygenerovaných bodů. Řešené generované terénní tvary jsou následující: náhodné body, kupa, spočinek, hřbet a údolí. Výsledné snímky obrazovky jsou v kapitole 7.

## 3.6 Náhodné body

S využitím funkce *rand*() jsou vygenerovány x – ové, y – ové a z – ové souřadnice podrobných bodů v rámci Canvasu (plátna). Hraničními souřadnicemi jsou rozměry plátna pro X,Y a u souřadnice Z je jakožto horní limit brána daná výška 1000 metrů.

**Implementace algoritmu:**

1. Určení souřadnic $(x_i, y_i, z_i)$ každého bodu $p_i$, přičemž $i \in (1, n)$, kde $n$ je manuálně zvolený počet bodů množiny *points*
2. Výpočet $x_i = rand()\ \%\ width$, kde *width* je šířka plátna
3. Výpočet $y_i = rand()\ \%\ height$, kde *height* je výška plátna
4. Výpočet $z_i = rand()\ \%\ 1000$, přičemž 1000 je horní limit pro výpočet nadmořské výšky
5. $p_i(x_i, y_i, z_i) \rightarrow points$
6. Opakuj pro každé $i < n$

## 3.7 Kupa

S užitím náhodných bodů z předchozího výpočtu, kterým se přiřadí z – ové souřadnice dle vzdálenosti bodů od těžiště množiny bodů *points*, je vypočtena kupa. Body v nejbližším okolí těžiště $t$ mají nejvyšší nadmořské výšky(Z) a s rostoucí vzdáleností od těžiště nadmořské výšky úměrně klesají. Při generování bodů je použita chyba v rámci jednotek metrů, díky které vypadá výsledek reálněji.

**Implementace algoritmu:**

1. Inicializace vstupní množiny náhodných bodů *points* o celkovém počtu $n$
2. Nalezení těžiště $t$ množiny podrobných bodů *points*:
3. $x_t = \dfrac{\Sigma x_i}{n}$
4. $y_t = \dfrac{\Sigma y_i}{n}$
5. $z_t = 1000$, kde 1000 je maximální hodnota pro generování výšky
6. $t \rightarrow points$
7. Výpočet výšky bodů v závislosti na rostoucí vzdálenosti od těžiště. $\forall p_i \in points$:
8. $z_i = 1000 - d + rand()\ \%\ 10$, kde $d$ je vzdálenost bodů $p_i$ a $t$

## 3.8 Spočinek

Podrobné body jsou určeny náhodně, přičemž nadmořská výška je určena postupem popsaným níže. Jedná se o generování mírného svahu. Na základě souřadnicových rozdílu každého bodu s předposledním bodem, je určena Z-ová souřadnice. Tímto je docíleno postupného klesání nadmořské výšky směrem od hřbetnice. Závěrem je do výpočtu vložena náhodná chyba v nadmořské výšce, jejíž velikost je v jednotkách metrů.

**Implementace algoritmu:**

1. Pro množinu bodů *points* platí: $\forall i < n$:
2. $x = x_{points\ i} - x_{points\ n-1}$
3. $y = y_{points\ i} - y_{points\ n-1}$
4. $z = \frac{x + y}{250} + 500$
5. $points\ i \leftarrow z$
6. Opakování pro $i < n$

## 3.9 Hřbet

Výpočet podrobných bodů na hřebu vychází taktéž z náhodných bodů. Nejprve je náhodně určen směr, kterým bude hřbet procházet (směr osy x nebo y). Na základě určení směru proběhne setřídění souřadnic bodů dle souřadnice x nebo y. Dále je určeno těžiště obdobně jako v případě kupy (kapitola 3.7) a výšky podrobných bodů jsou vypočteny v závislosti na vzdálenosti těžiště s přihlédnutím ke směru hřbetu. Při generování bodů je použita chyba v rámci jednotek metrů, díky které vypadá výsledek reálněji.

**Implementace algoritmu:**

1. Náhodné určení směru hřbetu $\rightarrow$ seřazení bodů dle určeného směru, tedy buď podle osy x nebo dle osy y
2. Zadání nadmořských výšek počátečnímu a koncovému bodu (pořadí dle setřídění souřadnic z předchozího kroku)
3. Nalezení těžiště $t$:
4. $x_t = \frac{\Sigma x_i}{n}$
5. $y_t = \frac{\Sigma y_i}{n}$
6. $z_t = 1000$, kde 1000 je maximální hodnota pro generování výšky
7. $t \rightarrow points$
8. Určení výšek podobných bodů $p_i$: $\forall p_i \in points$:
9. Výpočet délek $d_1$(vzdálenost $p_i$ a počátečního bodu z *points*) a $d_2$(vzdálenost $p_i$ a předposledního bodu z *points*)

10. Pokud $d_1 < d_2$:

11. Pak $z_i = 1000 - vzd(p_i, (points_{pocatecni}, points_{posledni})) + rand()\ \%\ 10$

12. Jinak $z_i = 1000 - vzd(p_i, (points_{predposledni}, points_{posledni})) + rand()\ \%\ 10$

13. Opakování pro $i < n$

## 3.10 Údolí

Výpočetní algoritmus pro údolí je v podstatě totožný jako v případě hřbetu (kapitola 3.9), jen s rozdílem ve výpočtu nadmořských výšek podrobných bodů. Zároveň není počítáno s hodnotami menšími než 1000, nýbrž s hodnotami menšími než 150 z důvodu příklonu k realističtějšímu znázornění situace. Při generování bodů je použita chyba v rámci jednotek metrů, díky které vypadá výsledek reálněji.

**Implementace algoritmu:**

1. Náhodné určení směru hřbetu → seřazení bodů dle určeného směru, tedy buď podle osy x nebo dle osy y

2. Zadání nadmořských výšek počátečnímu a koncovému bodu (pořadí dle setřídění souřadnic z předchozího kroku)

3. Nalezení těžiště $t$:

4. $x_t = \frac{\Sigma x_i}{n}$

5. $y_t = \frac{\Sigma y_i}{n}$

6. $z_t = 1000$, kde 1000 je maximální hodnota pro generování výšky

7. $t \rightarrow points$

8. Určení výšek podobných bodů $p_i$: $\forall p_i \in points$:

9. Výpočet délek $d_1$(vzdálenost $p_i$ a počátečního bodu z $points$) a $d_2$(vzdálenost $p_i$ a předposledního bodu z $points$)

10. Pokud $d_1 < d_2$:

11. Pak $z_i = 150 + vzd(p_i, (points_{pocatecni}, points_{posledni})) + rand()\ \%\ 10$

12. Jinak $z_i = 150 + vzd(p_i, (points_{predposledni}, points_{posledni})) + rand()\ \%\ 10$

13. Opakování pro $i < n$

# 4 Problematické situace a jejich rozbor

## 4.1 Automatické nastavení přechodné barevné stupnice mezi žlutou a červenou

Pro zobrazení sklonu terénu byl zvolen přechod žluté do červené jako optimální varianta, nicméně vyladění správně plynule vykreslovaných barev bylo poměrně náročné k představení. Naštěstí existují různé vizualizační online aplikace, které umí velice názorně pomoci – například [2].

```
void Draw::paintEvent(QPaintEvent *event)
{
...
        if (yelred == TRUE)
        {
        //Convert to color
        int col = 255 - k * slope;
        int colg = k * slope;
        QColor color(255, colg, col);

        //Set pen and brush
        qp.setBrush(color);
        qp.setPen(color);
        }
...
}
```

## 4.2 Správné natočení dat vůči světovým stranám

Aplikace umí číst vstupní data v EPSG 5514 (S-JTSK / Krovak East North), s čímž se pojí nutnost přetransformovat souřadnice do místního systému plátna spolu se záměnou souřadnic, jejich natočením a určením měřítka pro rovnoměrné vykreslení dle velikosti plátna. Transformace souřadnice je v kódu sepsána ve třídě Draw takto:

```
void Draw::loadPoints(std::string &path, int height, int width)
{
...
    //Load data from file
    std::ifstream coords(path);

    if(coords.is_open())
    {
        while(coords >> id >> y >> x >> z)
        {
        ...
        }
        coords.close();
    }
```

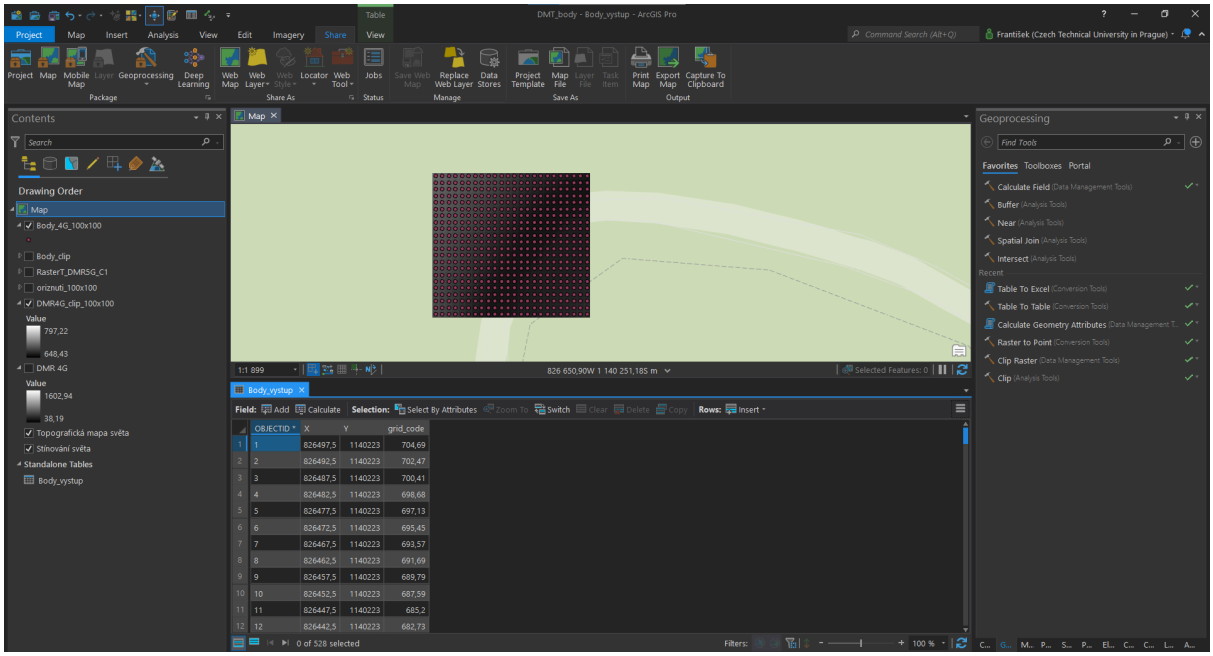
```
    //Transformation coefficient
    double kx = width/fabs(x_max - x_min);
    double ky = height/fabs(y_max - y_min);

    //Scale points to the size of Canvas
    for (int i = 0; i < points.size(); i++)
    {
        double px = points[i].x();
        double py = points[i].y();

        //Rewrite coords
        points[i].setX(-kx*(py - y_max));
        points[i].setY(ky*(px - x_min));
    }
...
}
```

# 5 Vstupní data

Vstupem je textový soubor s prostorovými souřadnicemi X,Y,Z geodetických bodů. Tyto body jsou považovány za vstupní množinu P. Body byly získány exportem z Digitálního modelu reliéfu 4. generace [3] v prosředí ArcGIS Pro [1] (viz. obr. 5). Body byly následně profiltrovány z důvodu jednoduššího a rychlejšího výpočtu aplikace.




Obrázek 5: Export bodů v ArcGIS Pro z DMR4G na vybraném území.

Formát vstupních dat je následující (ID, X, Y, Z):

```
1 846308.0 1129497.0 1141.2
2 846288.0 1129497.0 1131.8
```

```
3 846268.0 1129497.0 1123.2
...
```

# 6 Výstupní data

Za výstup je považována grafický výstup vytvořené aplikace. Grafickým výstupem je polyedrický DMT nad množinou P představovaný vrstevnicemi doplněný vizualizací sklonu trojúhelníků, jejich expozicí a vrstevnicemi s popisem.

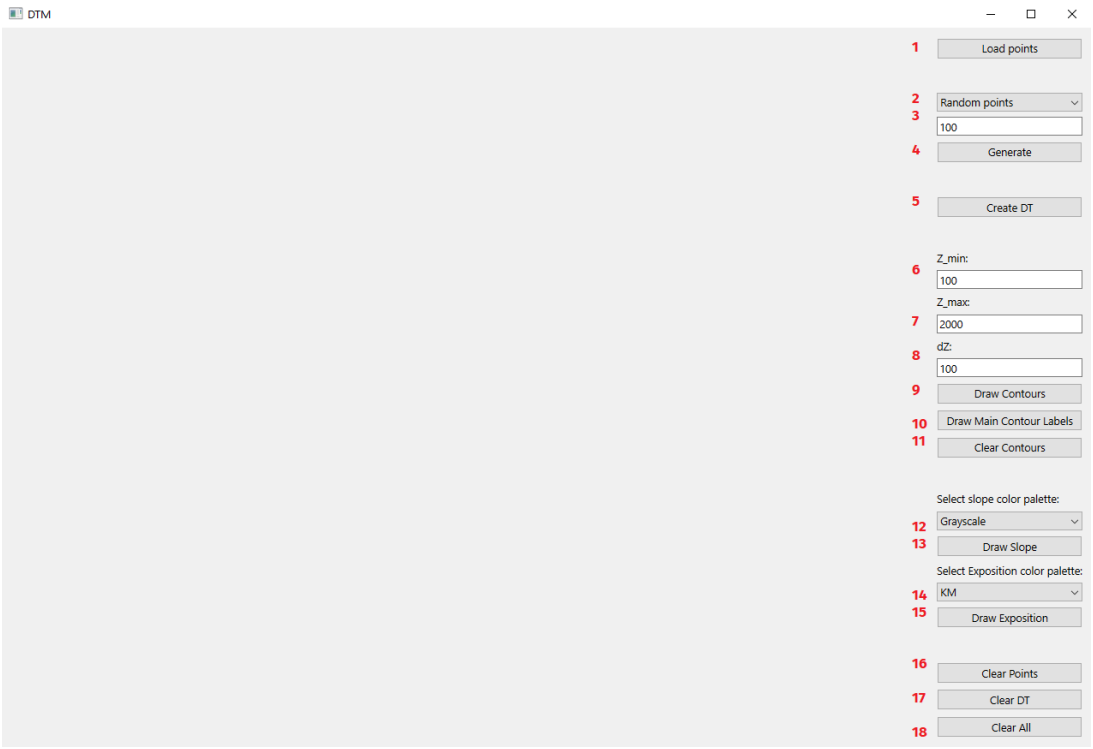
# 7 Snímky obrazovky vytvořené aplikace a její popis

## 7.1 Popis ovládání grafického prostředí aplikace

Grafické prostředí aplikace se skládá z plátna, ve kterém se vykreslují grafické operace a z nabídky funkcí aplikace v pravé části okna. Zde se nacházejí tlačítka, rozevírací nabídky a pole pro zadání čísel. Při nesprávném použití aplikace se objeví chybová hláška s patřičným problémem.

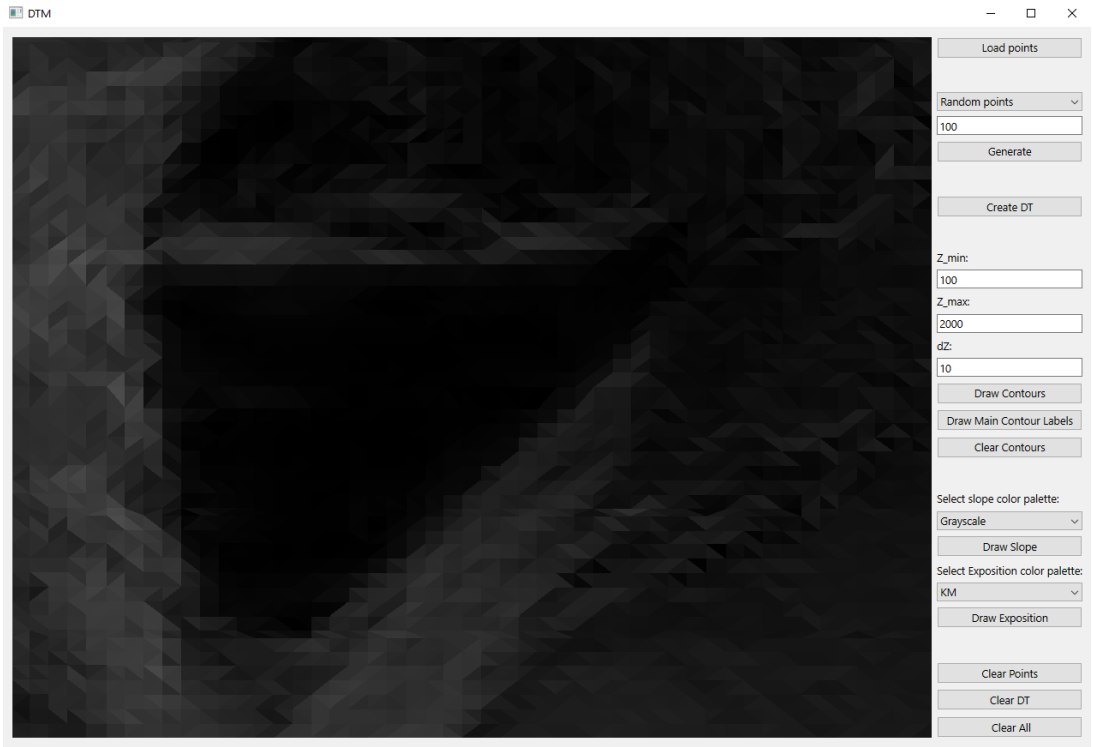
**Popis nabídky funkcí aplikace:** (čísla korespondují s obr. 6)

1. Load Points načte textový soubor s podrobnými body
2. Rozevírací nabídka pro výběr generovaného terénního tvaru
3. Zadání počtu vygenerovaných bodů
4. Spuštění generování bodů
5. Vytvoření Delaunay triangulace
6. Nastavení minimální hodnoty vrstevnic
7. Nastavení maximální hodnoty vrstevnic
8. Nastavení kroku mezi vrstevnicemi
9. Spuštění kresby vrstevnic
10. Spuštění kresby popisku hlavních vrstevnic
11. Vymazání vykreslených vrstevnic
12. Výběr barevné palety pro vybarvení sklonu terénu
13. Vykreslení sklonu terénu
14. Výběr barevné palety pro vybarvení expozice částí terénu
15. Vykreslení expozice
16. Vymazání načtených bodů
17. Vymazání triangulace
18. Vymazání celého okna



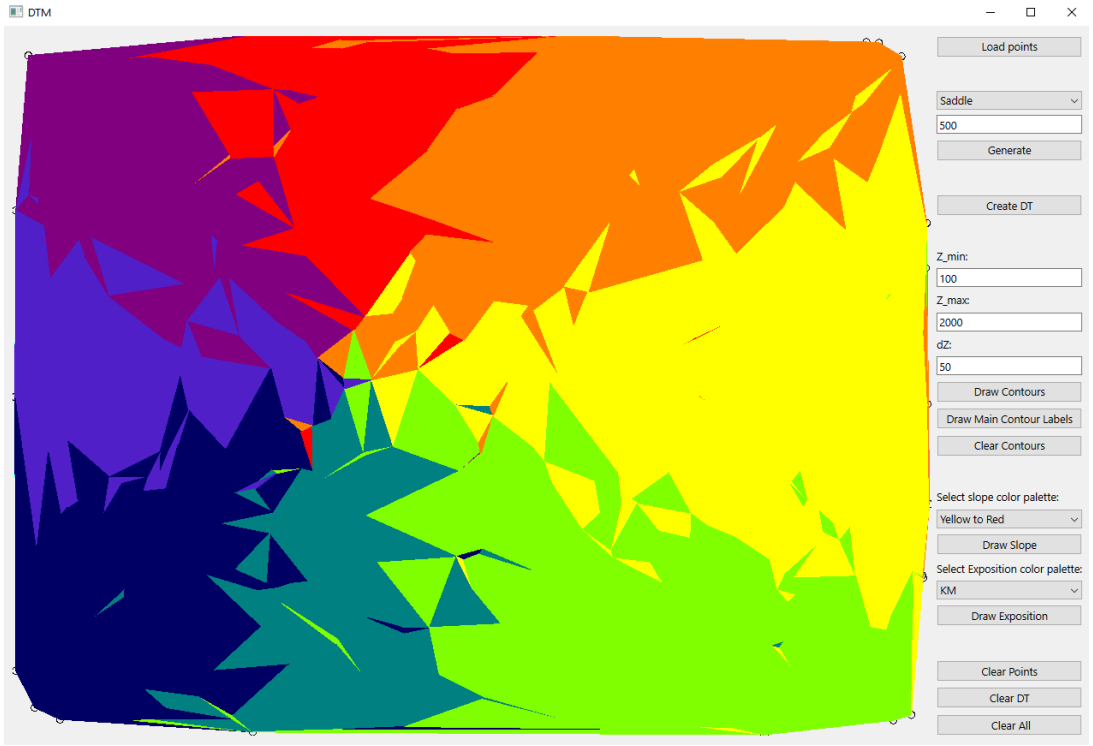

Obrázek 6: Aplikace po spuštění.

## 7.2 Vykreslení sklonu a expozice

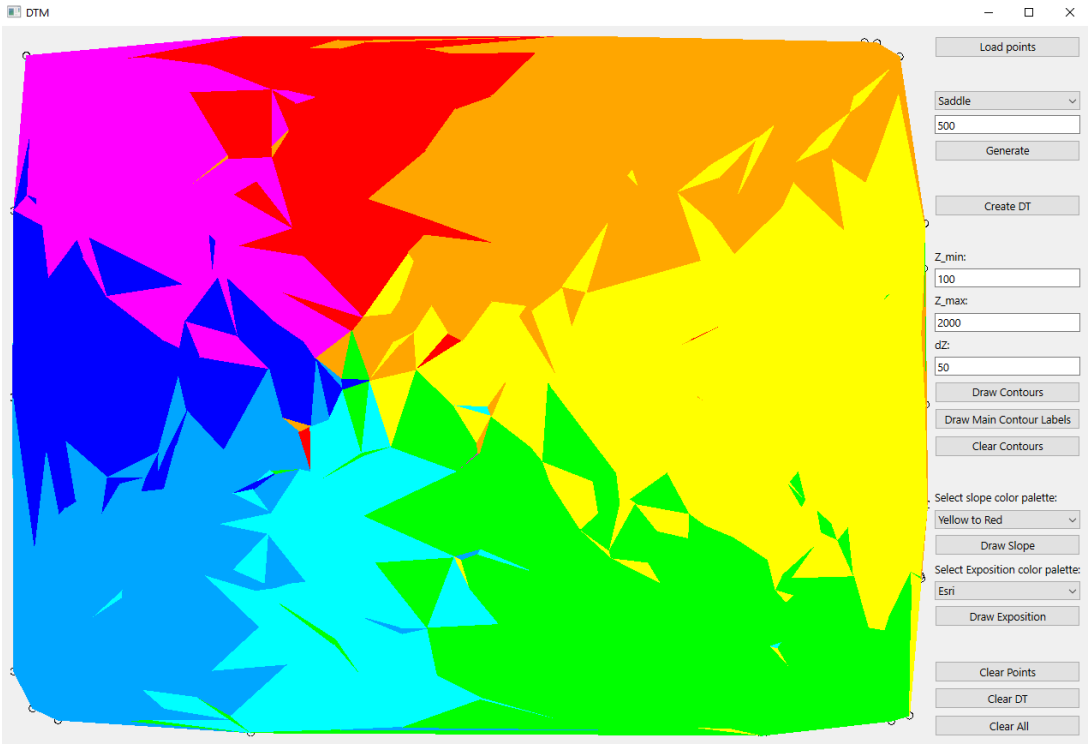



Obrázek 7: Sklon terénu v barevné paletě stupňů šedi.


Obrázek 8: Sklon terénu ve žluto – červené barevné paletě.

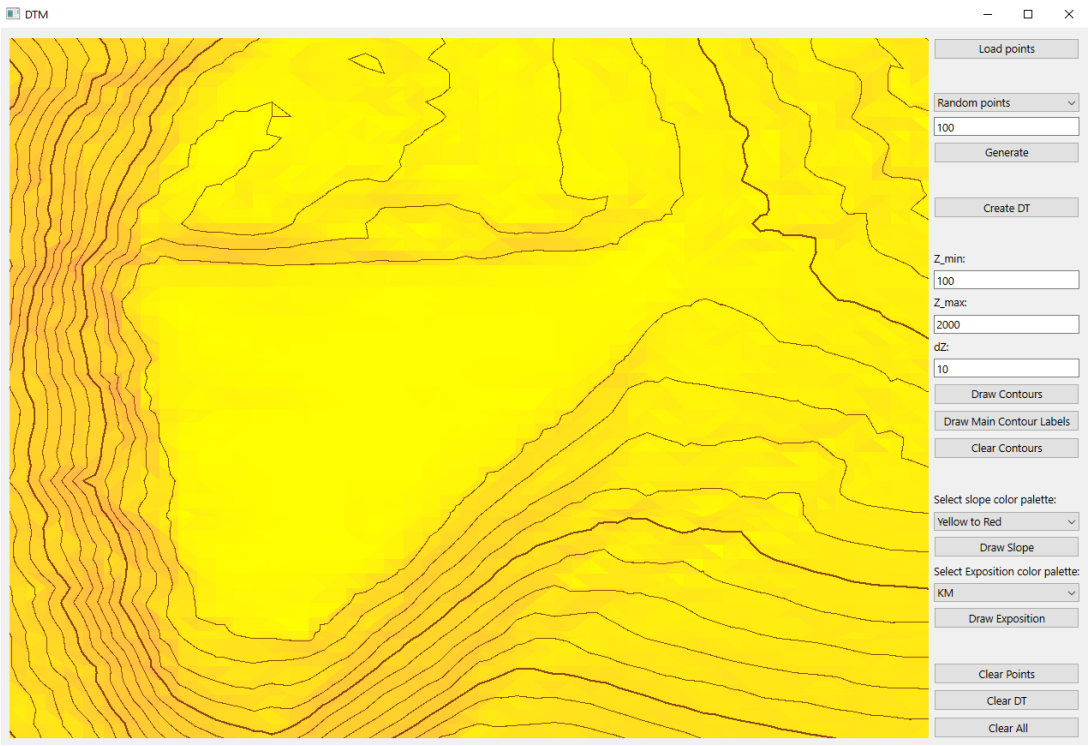



Obrázek 9: Expozice v KM barevné paletě.

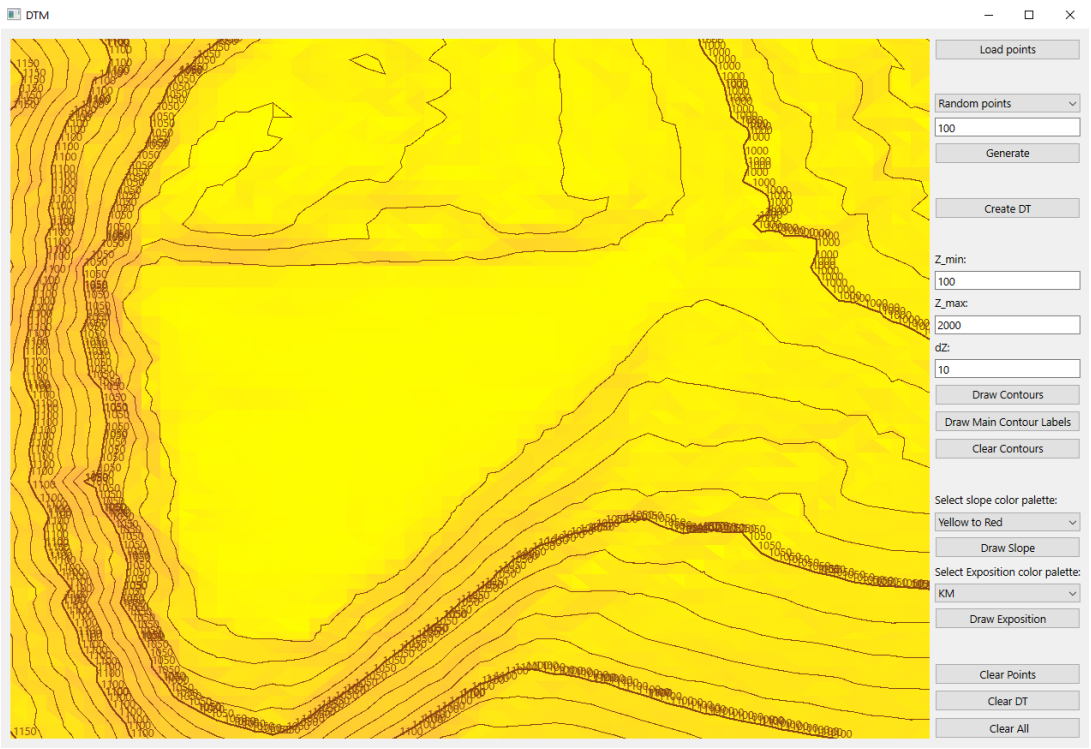



Obrázek 10: Expozice ve standardní barevné paletě, kterou používá např. ArcGIS.

## 7.3 Vykreslení vrstevnic

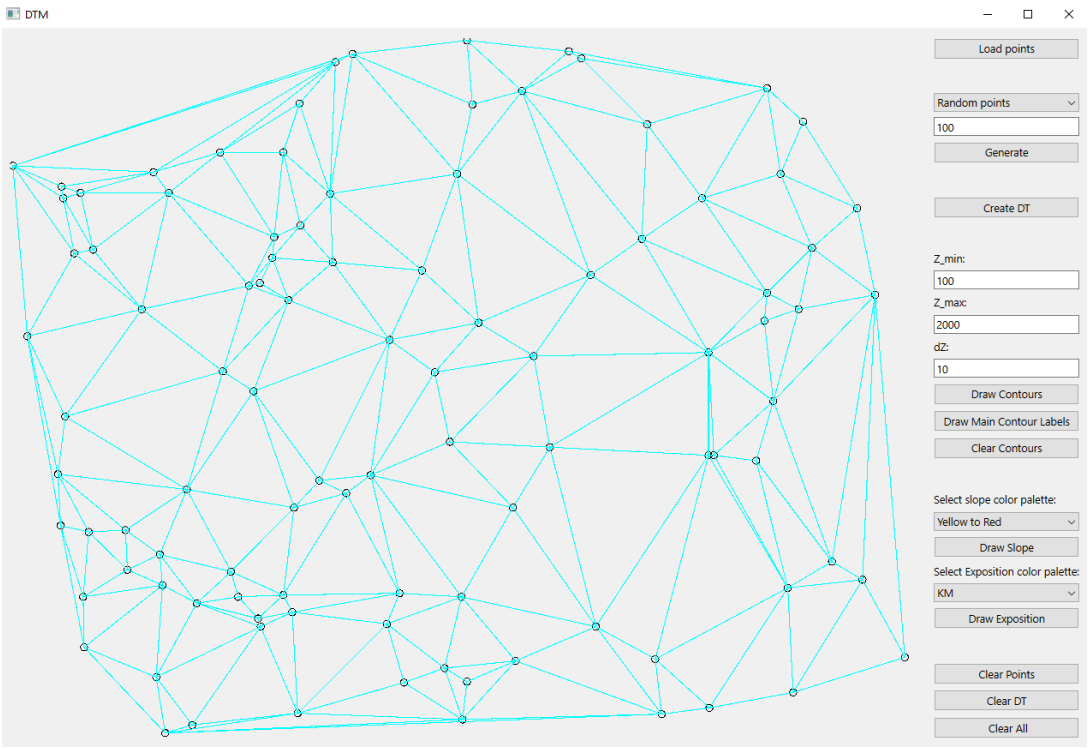



Obrázek 11: Vykreslení vrstevnic bez popisu.

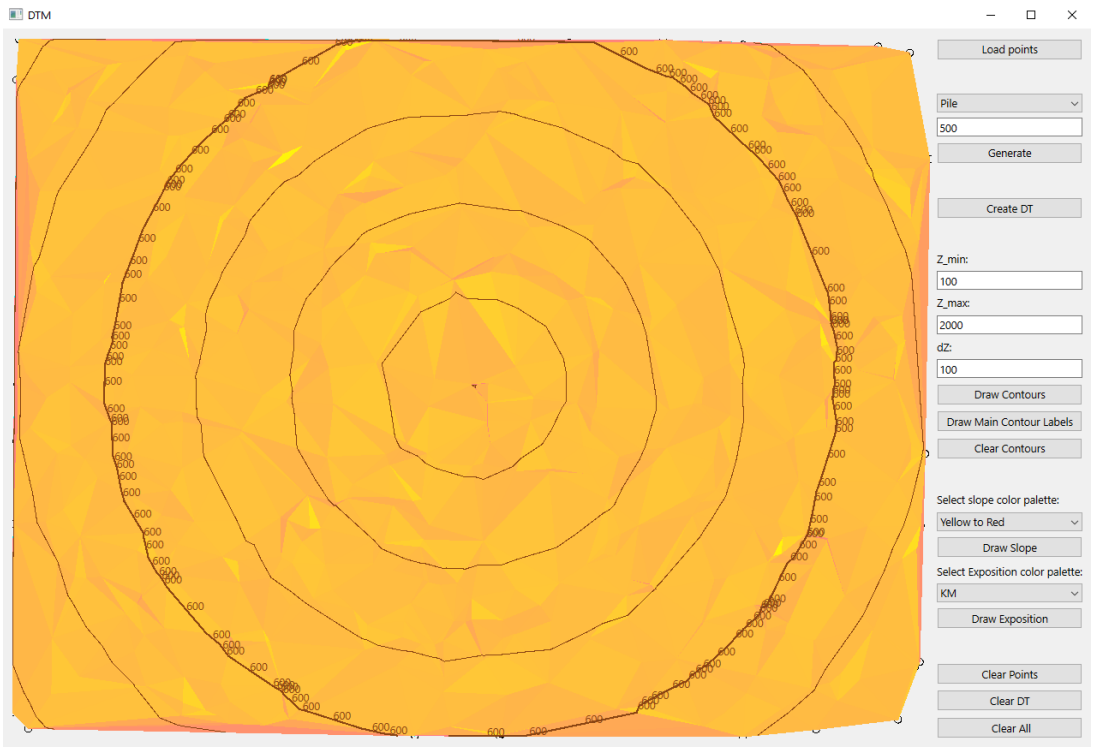



Obrázek 12: Vykreslení vrstevnic s popisem.

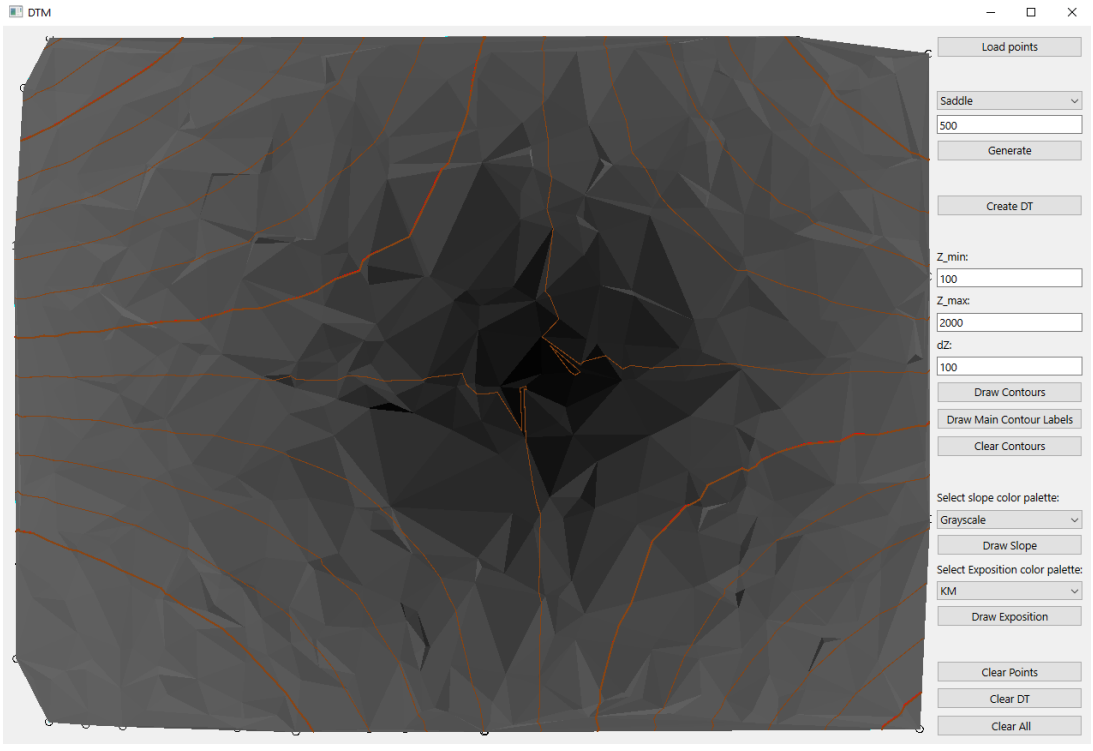
## 7.4 Generování terénních tvarů



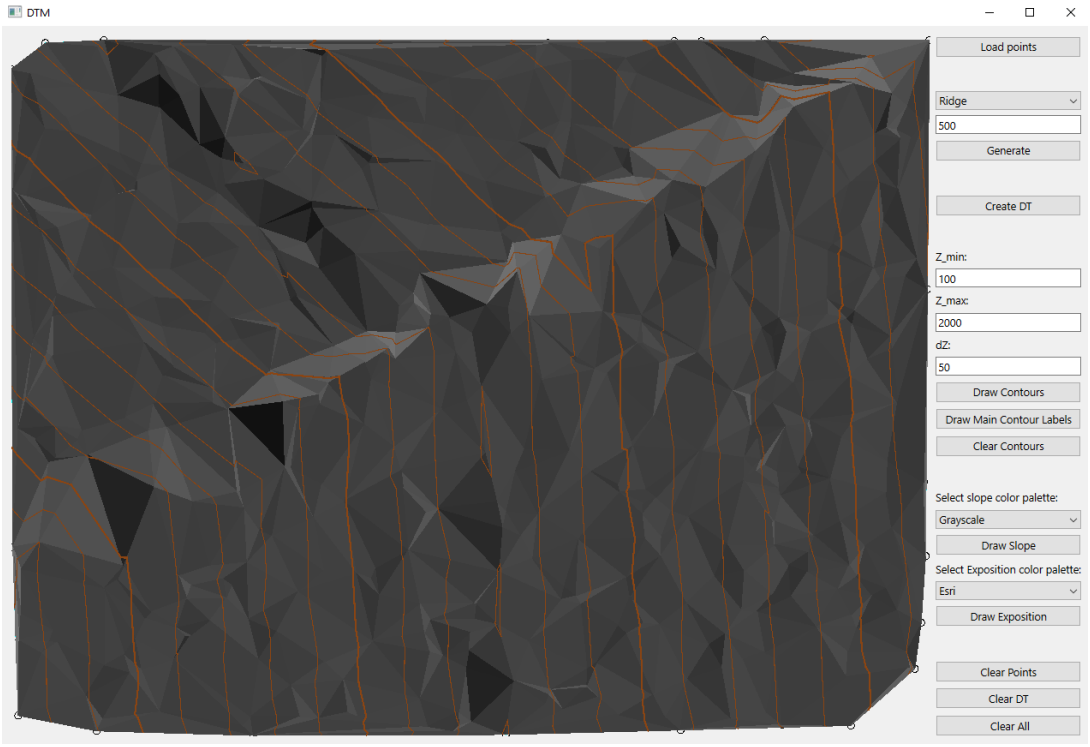

Obrázek 13: Delaunay triangulace nad náhodně generovanými body.



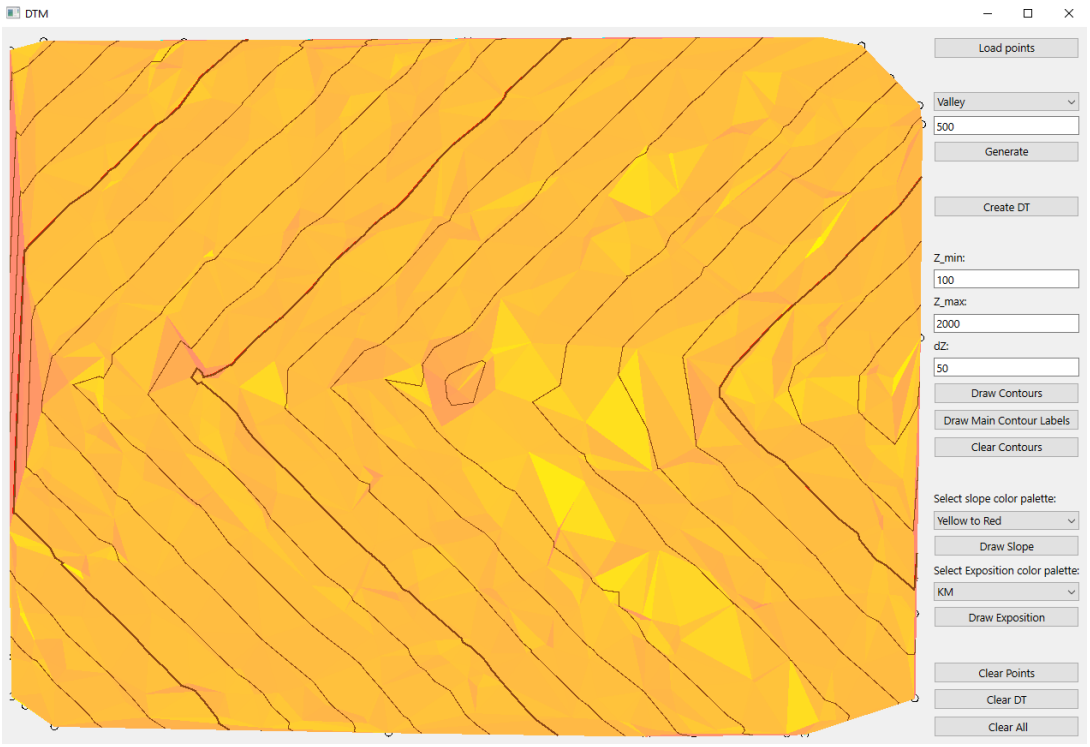

Obrázek 14: Vykreslení kupy.




Obrázek 15: Vykreslení spočinku.

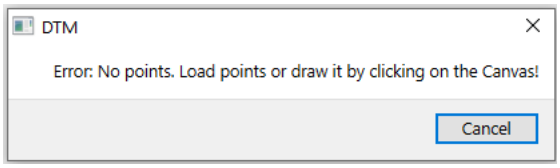



Obrázek 16: Vykreslení hřbetu.

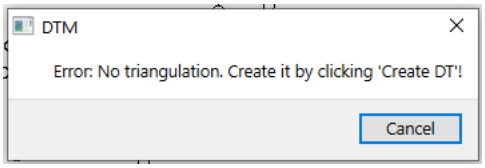



Obrázek 17: Vykreslení údolí.

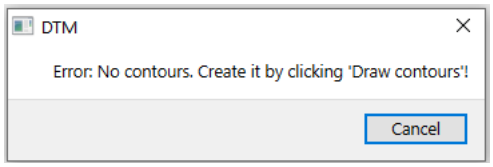
## 7.5 Chybové hlášky




Obrázek 18: Chyb. hl. pro nutnost načtení bodů.




Obrázek 19: Chyb. hl. pro nutnost vytvoření triangulace.




Obrázek 20: Chyb. hl. pro nutnost vytvoření vrstevnic.

# 8 Dokumentace

## 8.1 Třída Algorithms

Tato třída obsahuje výpočetní vzorce pro použité algoritmy.
**Třída algorithms obsahuje následující veřejné metody:**

```
int getPointLinePosition(QPoint &a,QPoint &p1,QPoint &p2)
```

Metoda určuje, zda – li bod leží v levé či v pravé polorovině od přímky (hrany polygonu). Vstupními parametry jsou určovaný bod $a$ a body $p_1, p_2$, které tvoří přímku.

```
std::tuple<QPoint3D,double> getCircleCenterAndRadius(
    QPoint3D &p1,QPoint3D &p2,QPoint3D &p3)
```

Vytvoření opsané kružnice nad třemi body.

```
int getDelaunayPoint(QPoint3D &s,QPoint3D &e,std::vector<QPoint3D> &points)
```

Určení Delaunay bodu pro triangulaci.

```
int getNearestPoint(QPoint3D &p, std::vector<QPoint3D> &points)
```

Nalezení nejbližšího bodu.

```
std::vector<Edge> dT(std::vector<QPoint3D> &points)
```

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

Aktualizace seznamu AEL pro Delaunay triangulaci.

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

Nalezení průsečíku trojúhelníku a horizontální vrstevnice.

```
std::vector<Edge> getContourLines(
    std::vector<Edge> &dt, double zmin, double zmax, double dz)
```

Výpočet vrstevnic na základě zadaných hran triangulace, minimální a maximální hodnoty vrstevnic a rozestupu mezi vrstevnicemi.

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

Výpočet sklonu trojúhelníka tvořeného vstupními body.

```
double getExposition(QPoint3D &p1, QPoint3D &p2, QPoint3D &p3)
```

Výpočet expozice trojúhelníka tvořeného vstupními body.

```
std::vector<Triangle> analyzeDTM(std::vector<Edge> &dt)
```

Výpočet sklonu a expozice pro každý trojúhelník v triangluaci.

```
double distance(QPoint3D &p1, QPoint3D p2)
```

Výpočet vzdálenosti mezi dvěma body.

```
double pointLineDistance(QPoint3D &q, QPoint3D &p1, QPoint3D &p2)
```

Výpočet vzdálenosti bodu od přímky.

```
std::vector<QPoint3D> generateRandomPoints(int n, int height, int width)
```

Generování náhodných bodů na základě jejich počtu a rozměrů plátna.

`std::vector<QPoint3D> generatePile(std::vector<QPoint3D> &points)`

Generování kupy s využitím náhodných bodů.

`std::vector<QPoint3D> generateSaddle(std::vector<QPoint3D> &points)`

Generování spočinku s využitím náhodných bodů.

`std::vector<QPoint3D> generateValley(std::vector<QPoint3D> &points)`

Generování údolí s využitím náhodných bodů.

`std::vector<QPoint3D> generateRidge(std::vector<QPoint3D> &points)`

Generování hřbetu s využitím náhodných bodů.

## 8.2 Třída Draw

Tato třída umožňuje vykreslování bodů, trojúhelníků a vrstevnic.

**Třída draw obsahuje následující privátní proměnné:**

`std::vector<QPoint> points`

Proměnná se souřadnicemi načtených bodů.

`std::vector<Edge> dt`

Proměnná pro ukládání hran tvořených dvěma body.

`std::vector<Edge> contours, main_contours, main_contours_label`

Proměnná pro výpočet vrstevnic a jejich popisů.

`std::vector<Triangle> triangles, triangles_exp`

Proměnná pro ukládání trojúhelníků tvořených třemi body.

`bool grayscale, yelred, kmcol, esricol`

Proměnná pro výběr barevných palet.

**Třída draw obsahuje následující veřejné metody:**

`void clear();`

Vymazání daného objektu.

```
void setPoints(std::vector<QPoint3D> &points_){points = points_;}
```

Nastavení podrobných bodů.

```
std::vector<QPoint3D> getPoints(){return points;}
```

Předání podrobných bodů.

```
void setDT(std::vector<Edge> &dt_){dt = dt_;}
```

Nastavení hran Delaunay triangulace.

```
std::vector<Edge> getDT(){return dt;}
```

Předání hran Delaunay triangulace.

```
void setContours(std::vector<Edge> &contours_){contours = contours_;}
```

Nastavení vrstevnic.

```
std::vector<Edge> getContours(){return contours;}
```

Předání vrstevnic.

```
std::vector<Triangle> getTriangles(){return triangles;}
```

Předání trojúhelníků.

```
void setTriangles(std::vector<Triangle> &triangles_){triangles = triangles_;}
```

Nastavení trojúhelníků.

```
void clearDT();
```

Vymazání Delaunay triangulace z plátna.

```
void clearAll();
```

Vymazání celého plátna.

```
std::vector<Triangle> getTrianglesExp(){return triangles_exp;}
```

Předání trojúhelníků s expozicí.

```
void setTrianglesExp(std::vector<Triangle> &triangles_exp_)
    {triangles_exp = triangles_exp_;}
```

Natavení trojúhelníků s expozicí.

```
void clearSlope(){triangles.clear();}
```

Vymazání sklonu terénu z plátna.

```
void clearExposition(){triangles_exp.clear();}
```

Vymazání expozice z plátna.

```
void clearContours(){
    contours.clear(), main_contours.clear(), main_contours_label.clear();}
```

Vymazání vrstevnic z plátna.

```
void loadPoints(std::string &path, int height, int width)
```

Načtení bodů z textového souboru.

```
void setMainContours(std::vector<Edge> &main_contours_)
    {main_contours = main_contours_;}
```

Nastavení hlavních vrstevnic.

```
std::vector<Edge> getMainContours(){return main_contours;}
```

Předání hlavních vrstevnic.

```
void setMainContoursLabel(std::vector<Edge> &main_contours_label_
    {main_contours_label = main_contours_label_;}
```

Nastavení popisu hlavních vrstevnic.

```
std::vector<Edge> getMainContoursLabel(){return main_contours_label;}
```

Předání popisu hlavních vrstevnic.

`void setGrayscale(bool &grayscale_){grayscale = grayscale_;}`
Natavení šedobílé barevné palety.

`void setYelRed(bool &yelred_){yelred = yelred_;}`
Nastavení žlutočervené barevné palety.

`void setKMCol(bool &kmcol_){kmcol = kmcol_;}`
Nastavení první možnosti pro vykreslení expozice.

`void setEsriCol(bool &esricol_){esricol = esricol_;}`
Nastavení druhé možnosti pro vykreslení expozice.

## 8.3 Edge

Třída uchovávající informace o hranách tvořených dvěma body.
**Třída draw obsahuje následující veřejné metody:**
`void changeOrientation(){QPoint3D temp=s; s=e; e=temp;}`
Změna orientace hrany.

`QPoint3D getStart() const {return s;}`
Vrácení počátečního bodu hrany.

`QPoint3D getEnd() const {return e;}`
Vrácení koncového bodu hrany.

## 8.4 QPoint3D

**Třída draw obsahuje následující privátní proměnnou:**
`double z`
Nadmořská výška bodu.

**Třída draw obsahuje následující veřejné metody:**
`void setZ(double z_){z = z_;}`
Nastavení nadmořské výšky bodu.

`double getZ(){return z;}`
Předání nadmořské výšky bodu.

## 8.5 Třída SortByX

**Třída draw obsahuje následující veřejnou proměnnou:**

```
bool operator() (QPoint &p1, QPoint &p2)
```

Řazení bodů dle jejich x – ové souřadnice.

## 8.6 Třída SortByY

**Třída draw obsahuje následující veřejnou proměnnou:**

```
bool operator() (QPoint &p1, QPoint &p2)
```

Řazení bodů dle jejich y – ové souřadnice.

## 8.7 Triangle

Třída tvořící trojúhelník.
**Třída draw obsahuje následující privátní proměnné:**

```
QPoint3D p1, p2, p3
```

Vrcholy trojúhelníka.

```
double slope, exposition
```

Sklon a expozice trojúhelníka.

**Třída draw obsahuje následující veřejné metody:**

```
QPoint3D getP1(){return p1;}
```

Vrácení souřadnic vrcholu 1.

```
QPoint3D getP2(){return p2;}
```

Vrácení souřadnic vrcholu 2.

```
QPoint3D getP3(){return p3;}
```

Vrácení souřadnic vrcholu 3.

```
double getSlope(){return slope;}
```

Předání sklonu trojúhelníka.

```
double getExposition(){return exposition;}
```

Předání expozice trojúhelníka.

```
void setP1(QPoint3D &p1_){p1 = p1_;}
```

Nastavení vrcholu 1.

```
void setP2(QPoint3D &p2_){p2 = p2_;}
```

Nastavení vrcholu 2.

```
void setP3(QPoint3D &p3_){p3 = p3_;}
```

Nastavení vrcholu 3.

```
void setSlope(double &slope_){slope = slope_;}
```

Natavení sklonu trojúhelníka.

```
void setExposition(double &exposition_){exposition = exposition_;}
```

Nastavení expozice trojúhelníka.

## 8.8 Třída Widget

Tato třída propojuje uživatelské rozhraní aplikace s kódem. Je vytvořena v sekci *Design*.

**Třída widget obsahuje následující privátní metody:**

```
void on_pushButton_clearPoints_clicked()
```

Vymazání podrobných bodů z plátna.

```
void on_pushButton_cleardt_clicked()
```

Vymazání triangulace z plátna.

```
void on_lineEdit_editingFinished()
```

Manuální zápis minimální hodnoty pro vrstevnice.

```
void on_lineEdit_2_editingFinished()
```

Manuální zápis maximální hodnoty pro vrstevnice.

`void on_lineEdit_3_editingFinished()`

Manuální zápis rozestupu mezi vrstevnicemi.

`void on_pushButton_drawContours_clicked()`

Vykreslení vrstevnic.

`void on_pushButton_drawSlope_clicked()`

Vykreslení sklonu terénu.

`void on_pushButton_ClearAll_clicked()`

Vymazání celého plátna.

`void on_pushButton_Exposition_clicked()`

Vykreslení expozice terénu.

`void on_pushButton_LoadPoints_clicked()`

Načtení textového souboru se vstupními body.

`void on_pushButton_mcLabels_clicked()`

Vykreslení popisů vrstevnic.

`void on_pushButton_Generate_clicked()`

Generování náhodných bodů či terénních útvarů.

`void on_pushButton_clearContours_clicked()`

Vymazání vrstevnic z plátna.

`void on_pushButton_createDT_clicked()`

Vykreslení Delaunay triangulace.

# 9 Závěr

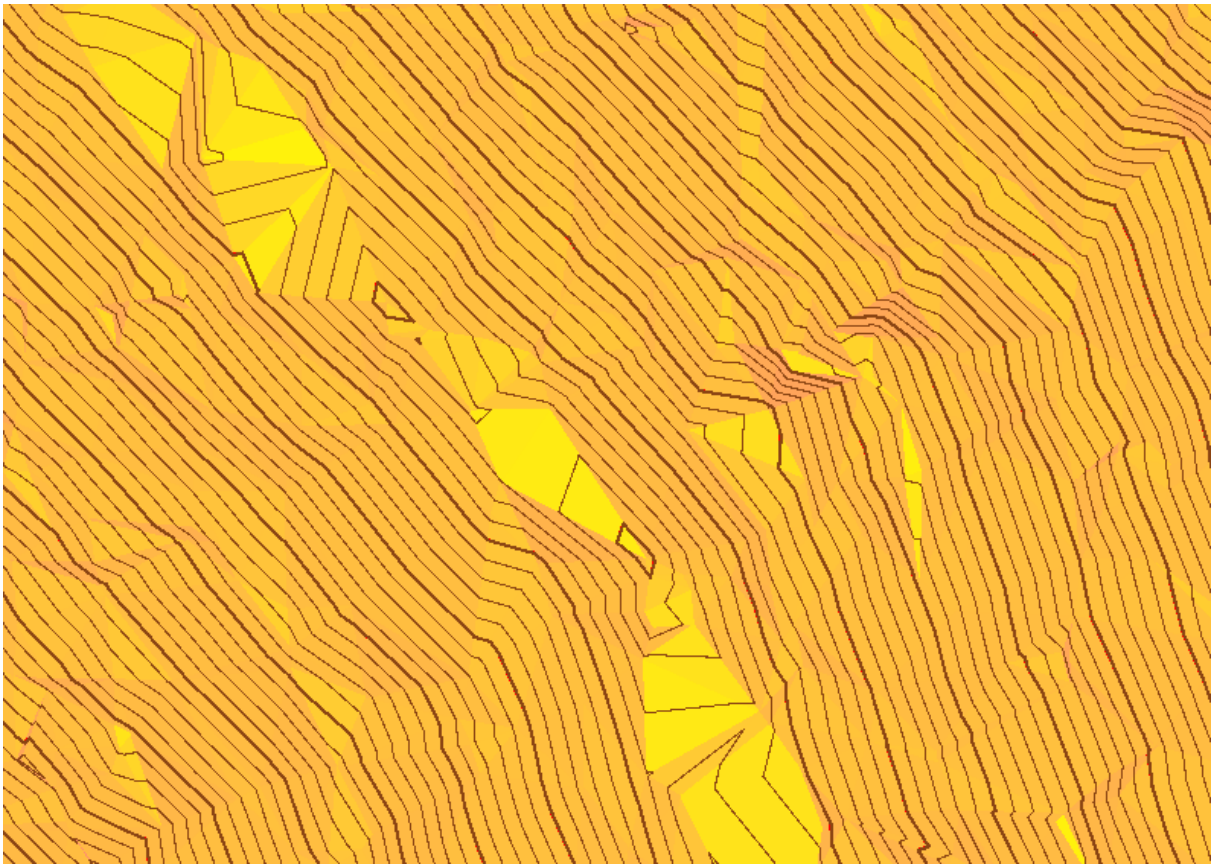
Byla vytvořena desktopová aplikace s interaktivním grafickým prostředí, která uživateli umožňuje vypočítat a vykreslit digitální model terénu nad zadanými či náhodně generovanými bodovými daty. Je na výběr z následujících náhodně generovaných terénních tvarů: kupa, údolí, spočinek, hřbet nebo znázornění bodů čistě náhodných.

V aplikaci je možné vykreslit sklon terénu a expozici částí terénu k jednotlivým světovým stranám, přičemž pro obě z možností jsou na výběr dvě barevné vykreslovací palety. Aplikace dále umožňuje vykreslení vrstevnic, včetně zvýraznění a popisu hlavních vrstevnic. Rozestup vrstevnic, včetně minimální a maximální výšky vrstevnic, lze manuálně měnit v okně aplikace. Není implementován popis vrstevnic respektující kartografické zásady. Vývoj aplikace proběhl v programovacím jazyce C++.

## 9.1 Zhodnocení

Výsledný model terénu slouží jako poměrně kvalitní náhled nad načtenými daty, nicméně bez dalších úprav má stále velké nedostatky. Mezi hlavní problém jistě patří absence povinných hran, díky které se vrstevnice tvoří například po vodní ploše, jak je vidět na ukázce Černého jezera na obr. 11.

Další problematické situace nastávají při generování hřbetu a údolí. Vrstevnice v okolí hřbetnice, respektive údolnice, tvoří jednotlivé "vrcholky"(obr. 21), což by mohlo být pro uživatele matoucí při chápání mapy a zároveň tyto útvary zahlcují mapu. Bylo by tedy vhodné vykreslovat vrstevnice jen nad určitou délku.



Obrázek 21: Chyba při generování hřbetu.

Nicméně přes určité nedostatky se ukazuje, že užití 2D triangulace je vhodné pro prvotní náhled nad daty. Uživatel získá hrubé povědomí o povrchu terénu a jeho členitosti. Pro pokročilejší analýzy nebo přesnější výstupy je žádoucí další úprava algoritmu a odstranění nedostatků.

## 9.2 Možné či neřešené problémy

Nebyly řešeny některé bonusové úlohy. Problém může nastávat při větším objemu vstupních dat, už při 2500 vstupních bodech není triangulace rozhodně "instantní". Při aplikaci nad reálnými daty je objevují nedostatky výpočtu, například kreslení vrstevnic přes vodní plochu.

## 9.3 Náměty na vylepšení

Aplikaci lze jistě vylepšit v mnoha ohledech. Grafické zpracování by mohlo být pokročilejší, bylo by možné přidat více barevných palet nebo při výběru barevné palety zobrazit její barvy, aby je uživatel znal dopředu. Pro zadaná data by mohla aplikace rozpoznat minimální a maximální nadmořskou výšku, a tedy by jí nemusel uživatel manuálně zadávat.

V Praze 5. 12.2021

Bc. Pane Kuzmanov
Bc. František Mužík

# Použitá literatura


[1] ArcGis Pro. Základ systému ArcGIS. https://www.arcdata.cz/produkty/arcgis/desktopovy-gis/arcgis-pro [cit. 2021-12-03].

[2] Computer Science Field Guide. Computer Science Education Research Group at the University of Canterbury, New Zealand. https://www.csfieldguide.org.nz/en/interactives/rgb-mixer/ [cit. 2021-12-04].

[3] Český úřad zeměměřický a katastrální. Základní báze geografických dat České republiky (ZABAGED). https://www.cuzk.cz/ [cit. 2021-12-03].

[4] Západočeská univerzita v Plzni. Katedra geomatiky, Fakulta aplikovaných věd. https://kgm.zcu.cz/ [cit. 2021-12-03].