Univerzita Karlova
Přírodovědecká fakulta

Algoritmy počítačové kartografie

Úloha 3: Digitální model terénu a jeho analýzy.

Adam Kulich, Markéta Růžičková, Eliška Sieglová
N-GKDPZ
Praha 2023

1 Zadání

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

Výstup: Polyedrický DMT nad množinou P představovaný vrstevnicemi doplněný vizializací sklonů 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 zadanám 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 založen´eho na 2D Delaunay triangulaci. Ve kterých situacích (různé terénní tvary) nebude dávat vhodné výsledky? Tyto situace graficky znázorněte.

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*
Berevná hypsometrie.	*+5b*
Max celkem:	**65b**

Čas zpracování: 4 týdny.

2 Údaje o bonusových úlohách + body

V rámci této úlohy byl řešen bonusový úkol znázornění terénu pomocí barevné hypsometrie.

3 Popis a rozbor problému + vzorce

3.1 Rovinné triangulace

Rovinné triangulace řeší problém dělení roviny na trojúhleníky podle určitých pravidel. V praxi se triangulační algoritmy využívají například v kartografii a GIS k tvorbě digitálních modelů terénu, dále ale také v zpracování obrazu či počítačové grafice, nebo například v biometrii při rozpoznávání otisků prstů.

Definice triangulace: Máme množinu bodů P, nad kterou hledáme triangulaci T. Triangulace T představuje takové planární rozdělení, které vytvoří soubor m trojúhelníků t a hran tak, aby platilo:

- livovolné dva trojúhelníky mají společnou nejvýše hranu,
- sjednocení všech trojúhelníků tvoří $H(P)$,
- uvnitř žádného trojúhelníku neleží žádný další bod z P.

Definice byla převzata z přednášky Rovinné triangulace a jejich využití (Bayer, 2023).

Protože se dnes již pracuje s velkými soubory dat (množiny obsahují až miliardu prvků), máme na triangulační algoritmus nějaké požadavky. Tyto jsou jednoduchost algoritmu a snadná implementace, a dostatečná rychlost pro velká P (tyto požadavky si bohužel protiřečí). Složitost, kterou požadujeme pro velké datasety je $O(n * \log(n)$. Jakákoliv vyšší složitost, například kvadratická či kubická, není použitelná pro datasety o více než 50 000 000 body. Dalším požadavkem je, aby metoda měla co nejméně singulárních případů, a také aby bylo možné ji převézt to vyšších dimenzí. Pro rychlejší běh algoritmu je také důležitá práce ve vláknech, tedy schopnost paralelizace algoritmu, jako je tomu například u strategie *Divide and Conquer*.

Při výběru triangulace se zohledňují další kritéria ohledně tvaru, povinných hran, a triangulace nekonvexních oblastí. Tvar trojúhelníku ovlivňuje jak moc se trojúhelníková síť přimyká terénu a v triangulaci bychom měli chtít dosáhnout trojúhelníků, které se co nejvíce blíží rovnostranným. Pokud by se úhly blížily O či π, znamenalo by to, že vrcholy trojúhelníku jsou velmi daleko od sebe a tudíž neodpovídají terénu. Vkládání povinných hran (constraints) má schopnost ovlivňovat terén v místech s významným tvarem, například v údolích nebo na hřbetech. Schopnost triangulovat nekonvexní oblasti ovlivňuje triangulaci v místech jako jsou například budovy či vodní plochy. Algoritmus by měl být schopný se vypořádat i s triangulací nekonvexních oblastí obsahující díry.

Pro eliminaci trojúhelníku s nevhodnými vlastnostmi dle výše uvedených kritérií se využívají lokální či globální výpočty. Pro lokální kritéria se nejčastěji využívají vlastnosti trojúhelníků jako jsou minimální či maximální úhel (využití např. v Delaunay triangulaci), minimální či maximální výška trojúhelníku, minimální či maximální poloměr vepsané/opsané kružnice, minimální či maximální plocha, a další. Pro globální kritéria se nejčastěji používá suma délek stran

$$\sum_{i=1}^{n_h} h_i = min \tag{1}$$

kde h_i je celková délka hran triangulace. Globálně tento problém ale neumíme exaktně vyřešit, minimu se pouze přiblížíme.

Nejčastěji používaným lokálním kritériem je MIN/MAX strategie pro výpočet úhlu v trojúhelníku, dělící se na Min-max kritérium a Max-min kritérium. Můžeme ho použít na libovolné kritérium, jako je délka stran nebo úhel. V této úloze se pracovalo s kritériem úhlu, proto si to vysvětlíme na příkladu úhlu. Min-max kritérium minimalizuje maximální úhel, což eliminuje trojúhelníky s příliš tupými úhly. Naproti tomu strategiie Max-min maximalizuje minimální úhel, což eliminuje trojúhelníky s příliš ostrými úhly. Největší překážkou tohoto přístupu je, že není rychlý.

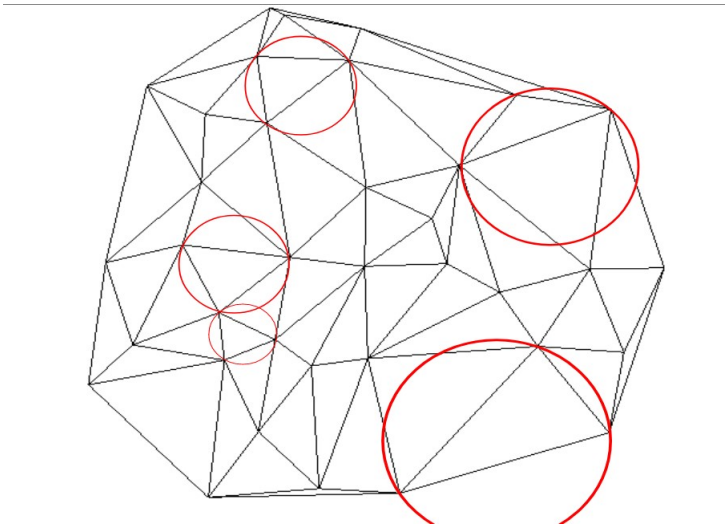
Triangulace se mohou dělit podle geometrické konstrukce (Greedy, Delaunay, Minimum Weight Triangulation, Constrained traingulace atd.), nebo podle využitých kritérií (lokálně optimální, globálně optimální, multikriteriálně

optimalizované).

Existuje mnoho typů triangulace, tento problém patří mezi ty nejvíce rozpracované. Jedna z jednodušších metod triangulace je *Greedy triangulace* patřící do skupiny Greedy (hladových) algoritmů. Pomocí inkrementální konstrukce se v každém kroku do triangulace přidává nejkratší možná hrana. Nevýhodou je její kubická, či po optimalizaci kvadratická složitost, a také to, že v triangulaci vznikají tvarově neoptimální trojúhelníky. Proto v kartografii není moc hojně využívaná. Nejčastěji v praxi bývá používána *Delaunay triangulace*.

3.2 Delaunay triangulace

Delaunay triangulace DT má následující vlastnosti: 1) uvnitř opsané kružnice k libovolnému trojúhelníku neleží žádný jiný bod množiny P, 2) minimalizuje maximální úhel, 3) je lokálně i globálně optimální vůči kritériu minimálního úhlu, 4) je jednoznačná, pokud 4 body z množiny P neleží na kružnici.



Obr. 1: Ukázka opsaných kružnic k vybraným trojúhelníkům Delaunay triangulace.

Delaunay triangulace je taktéž jedinou triangulací, kterou umíme konstruovat rychle, a nejvíce se blíží rovnostranným trojúhelníkům. DT lze také zobecnit do prostoru.

Z DT lze také odvodit další podgrafy a jiné geometrické vlastnosti:

- konvexní obálka: obvod triangulace tvoří konvexní obálku,
- proximity graphs: např. Nearest Neighbourhood Graph, používají se v klasifikacích,
- Minimum Spanning Tree,
- Gabrielle Graph,
- Thiessenovy polygony/Voronoi diagram: jsou přímkovým duálem DT a vznikají spojením středů vepsaných kružnic.

Cílem Delaunay triangulace není přidat takový vrchol, který bude generovat co nejmenší kružnici - to by vyžadovalo kontrolu a zapříčinilo by vyšší výpočetní náročnost. Ve výpočtu se využívá Tháletovy věty, kdy bod ležící uvnitř kružnice opisující dva body přímky (tudíž jedné hrany trojúhelníku) má úhel menší než 90°.

Některými metodami konstrukce Delaunay triangulace jsou například lokální prohazování, inkrementální konstrukce, inkrementální vkládání, divide and conquer nebo sweep line. Z těchto metod je nejvýkonnější metoda divide and conquer, která je také standardně využívána. V tomto cvičení jsme se zabývali inkrementální konstrukcí, která se sice v praxi příliš nepoužívá kvůli časové náročnosti, nicméně patří mezi jednodušší metody.

Metoda inkrementální konstrukce lze použít ve 2D (kružnice) i ve 3D (sféra). Je založena na postupném, ale nikoliv náhodném přidávání bodů do již vytvořené DT. Začíná se určením počátečního bodu, nebo-li pivotu p_1, který může být buď vybrán náhodně, nebo jako bod s minimální x- či y-souřadnicí. Na tom, kde tento bod leží, ale nezáleží. K němu pak nalezneme nejbližší bod, který pojmenujeme p_2, a který spolu s pivotem tvoří hranu $e = (p_1, p_2)$. Dále je hledán Delaunay bod p, který maximalizuje úhel tohoto bodu od hrany e. Základním požadavkem je CCW orientace, tedy, že bod, který přidáváme, musí ležet v levé polorovině. Pokud v levé polorovině žádný bod neleží, je nutné změnit směr. Při konstrukci je používána datová struktura Active Edge List (AEL) obsahující hrany e, které jsou složené z bodů p_i, p_j. Před přidáním do AEL je nutné otestovat, zda se v seznamu již nevyskytuje hrana s opačnou orientací.

Algorithm 1 Delaunay triangulace

```
máme množinu bodů P
inicializace AEL = , DT =
p_1 = rand(P)
naleznutí nejbližší bod p_2
vytvoření hranu e = (p_1, p_2) a opačnou hranu e' = (p_2, p_1)
přidání hrany e a e' do AEL
while AEL not empty do
    e_1 = AEL.pop()                          ▷ vyjmutí první hrany z AEL
    e'_1 = (p2, p1)                          ▷ prohození orientace
    nalezení Delaunay bodu p dle kritéria maximalizující minimální úhel
    if takový bod existuje then
        e_2 = (p_2, p), e_3 = (p, p_1)       ▷ Vytvoření zbývajících hran trojúhelníku
        přidání hran e_2, e_3 do DT
        přidání hran e_2, e_3 do AEL
    end if
end while
```

3.3 Konstrukce vrstevnic

Pro konstrukci vrstevnice na základě triangulační sítě lze využít lineární nebo nelineární interpolační algoritmy. Lineární algoritmy mají za výsledek konstantní rozestup mezi vrstevnicemi a lineární spád terénu, jsou výpočetně jednoduché, ale nevystihují realitu. Nelineárním algoritmům se také přezdívá "geomorfologická interpolace", jelikož předpokládají plynou změnu terénu mezi interpolovanými body. Tyto algoritmy nemají rozestup mezi vrstevnicemi konstantní, protože zohledňují sklon okolních plošek. Oproti lineárním algoritmům jsou ale tyto složitější.

Lineární interpolace vrstevnic je založena na analytické geometrii, jde o hledání průsečnice roviny T (rovina určená trojúhelníkem) a vodorovné roviny s výškou h.

3.4 Analýza sklonu a expozice

Analýza sklonu a expozice nad DTM je jednou z hojně využívaných funkcí v geoinformačních systémech, např. pro analýzu lavinových nebezpečí, hydrologických poměrů, sesuvů, návrhů staveb, a jiné.

Pro analýzu sklonu terénu je nutné vypočítat gradient (vektor maximálního spádu) $\nabla f(x_0, y_0, z_0)$ funkce $f(x, y, z)$ v bodě $p = [x_0, y_0, z_0]$.

$$\nabla f(x_0, y_0, z_0) = (\frac{\partial f}{\partial x}(x_0), \frac{\partial f}{\partial y}(y_0), \frac{\partial f}{\partial z}(z_0)) \tag{2}$$

Do výpočtu gradientu ∇ roviny ρ vstupuje rovnice roviny $\rho : ax + by + vz + d = 0$.

$$\nabla f(x_0, y_0, z_0) = (\frac{\partial f}{\partial x}(x_0), \frac{\partial f}{\partial y}(y_0), \frac{\partial f}{\partial z}(z_0)) = (a, b, c) \tag{3}$$

Pro rovinu ρ procházející třemi body platí:

$$\begin{vmatrix} x - x_1 & y - y_1 & z - z_1 \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{vmatrix} = 0. \tag{4}$$

Odchylka ϕ od roviny ρ se vypočítá následovně:

$$\phi = \arccos \frac{n_1 * n_2}{\| n_1 \| \| n_2 \|} = \arccos \frac{c}{\| n_1 \|}, \tag{5}$$

$$n_1 = (a, b, c), \tag{6}$$

$$n_2 = (0, 0, 1). \tag{7}$$

Tento výpočet se provádí zvlášť nad každou ploškou, trojúhelníky jsou vyplněny barvou na základě hodnoty ϕ.

3.5 Barevná hypsometrie

Bylo provedeno znázornění terénu pomocí barevné hypsometrie. Algoritmus byl postaven na principu vytváření polygonů vzniklých průsečíky trojúhelníků Delauneyho triangulace a vygenerovaných vrstevnic.

Nejprve se pro každý trojúhelník DT zkontroluje, zda jím prochází některá vrstevnice. Pokud ne, vytvoří se polygon odpovídající danému trojúhelníku a jako jeho z-souřadnice se nastaví průměrná hodnota z-souřadnic bodů tvořících vrcholy tohoto trojúhelníku.

Pokud trojúhelníkem prochází jedna nebo více vrstevnic, uloží se všechny tyto linie do listu a seřadí se vzestupně podle jejich z-souřadnice. Poté se vytvářejí polygony mezi okrajem trojúhelníku a jednotlivými vrstevnicemi. Tím dojde k rozdělení trojůhelníku na části s odlišnými z-souřadnicemi. Hodnota z-souřadnice jednotlivých polygonů v trojúhelníku je nastavena na hodnotu z-souřadnice horní (vyšší) vrstevnice. V případě polygonu ohraničeného vrstevnicí zdola a trojúhelníkem shora je z-souřadnice nastavena na hodnotu posledního přidaného bodu tvořícího polygon.

Dále se vykreslí jednotlivé polygony, přičemž jejich barva závisí na hodnotě z-souřadnice.

Algorithm 2 Barevná hypsometrie

```
založ prázdný list pols pro třídu Polygon
for každý Delaunayovský trojúhelník T do
    do p1, p2, p3 ulož vrcholy T
    založ prázdný list edgesinpolygon pro třídu Edge
    clInPol = False
    for každou vrstevnici CL do
        najdi bod MP ve středu CL
        zjisti, zda je MP uvnitř T
        if MP je uvnitř T then
            clInPol = True
            do edgesinpolygon přidej CL
        end if
    end for
    seřaď edgesinpolygon od nejnižší po nejvyšší
    ulož p1, p2, p3 do listu points
    založ prázdný list pointPop
    for každou CL v edgesinpolygon do
        vytvoř list pol a ulož do něj počáteční SP a koncový EP bod CL
        for každý bod P z points do
            if P není v pointPop and z SP ¿ z EP then
                P přidej do pol
                P přidej do pointPop
            end if
        end for
        vytvoř konvexní obálku z pol a přidej ji do pols
        přidej SP do points
        přidej EP do points
        if CL je poslední v edgesinpolygon then
            do pol ulož SP a EP
            for každý bod z points do
                if P není v pointPop and z SP ¡ z EP then
                    P přidej do pol
                    P přidej do pointPop
                end if
            end for
            vytvoř konvexní obálku z pol a přidej ji do pols
        end if
        if clInPol = False then
            vytvoř z p1, p2, p3 polygon a přidej ho do pols
        end if
    end for
end for
```

4 Vstupní data, formát vstupních dat, jejich popis

V rámci této úlohy byl vytvořen algoritmus pro generování bodového mračna. Algoritmus funguje tak, že vytvoří obdélníkové hranice analyzovaného území inicializací bodů o malé nadmořské výšce a následně inicializuje několik náhodných vrcholků. Zbytek bodového mračna generuje algoritmus náhodnou inicializací souřadnic x a y a výpočtem souřadnice z jako průměru souřadnic z tří nejbližších bodů s náhodnou odchylkou.

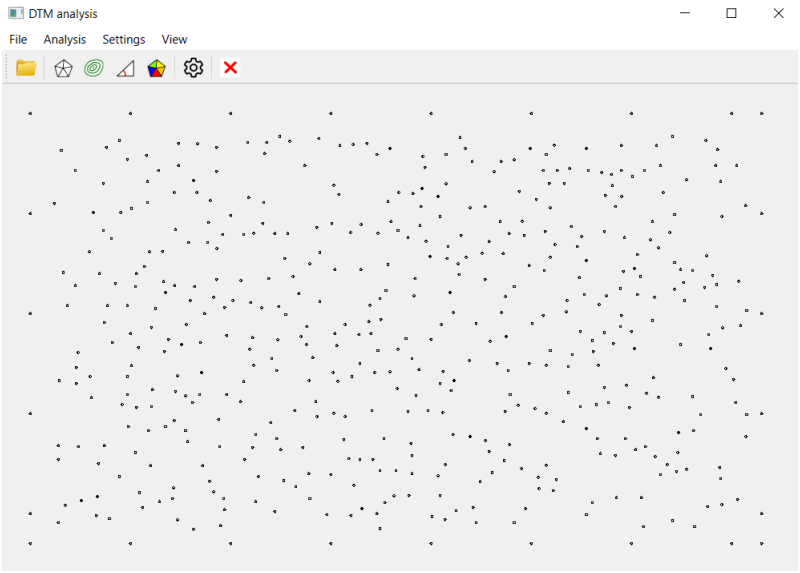
Algorithm 3 Generování syntetického bodového mračna

Inicializuj maximální a minimální souřadnice $x_max = 750, y_max = 450, z_max = 1000, min = x_min = y_min = 50$
Urči celkový počet bodů $max = 500$ a počet vrcholů $s = 4$
Inicializuj rohy bodového mračna jako:
vytvoř bod $roh_a = [x_{min}, min,$ *náhodný float mezi 0 a 100*$]$
vytvoř bod $roh_b = [min, y_{max},$ *náhodný float mezi 0 a 100*$]$
vytvoř bod $roh_c = [x_{max}, min,$ *náhodný float mezi 0 a 100*$]$
vytvoř bod $roh_d = [x_{max}, y_{max},$ *náhodný float mezi 0 a 100*$]$
Inicializuj seznam: $points = [roh_a, roh_b, roh_c, roh_d]$
for i mezi min a y_{max} s krokem 100 **do**
    vytvoř bod $y = [min, i,$*vážený průměr výšek min a* $y_{max}]$
    přidej y do $points$
end for
for i mezi min a x_{max} s krokem 100 **do**
    vytvoř bod $x = [i, min,$*vážený průměr výšek min a* $x_{max}]$
    přidej x do $points$
end for
for i mezi min a y_{max} s krokem 100 **do**
    vytvoř bod $y = [x_{max}, i,$*vážený průměr výšek min a* $y_{max}]$
    přidej y do $points$
end for
for i mezi min a x_{max} s krokem 100 **do**
    vytvoř bod $x = [i, y_{max},$*vážený průměr výšek min a* $x_{max}]$
    přidej x do $points$
end for
for všechny body v s **do**
    vytvoř souřadnici x_i = náhodné číslo mezi $min + 50$ a $x_{max} - 50$
    vytvoř souřadnici y_i = náhodné číslo mezi $min + 50$ a $y_{max} - 50$
    vytvoř souřadnici z_i = náhodné číslo mezi 0 a z_{max}
    vytvoř bod $p_i = [x_i, y_i, z_i]$
    Najdi nejbližší bod k p_i z $points$
    if vzdálenost mezi p_i a nejbližším bodem k p_i je větší než 20 **then**
        přidej p_i do $points$
    end if
end for
počet n = počet prvků v $points$
while n je menší než max **do**
    vytvoř souřadnici x = náhodné číslo mezi $min + 20$ a $x_{max} - 20$
    vytvoř souřadnici y = náhodné číslo mezi $min + 20$ a $y_{max} - 20$
    inicializuj souřadnici $z_0 = 0$
    vytvoř bod $p_0 = [x, y, z_0]$
    zkopíruj $points$ do $templist$
    vytvoř seznam blízkých bodů np a seznam vzdáleností d
    for j od 1 do 3 **do**
        najdi nejbližší bod k p_0 v $templist$ a označ ho p_j
        přidej p_j do np a vzdálenost mezi p_j a p_0 přidej do d
        odstraň p_j z $templist$
    end for
    if vzdálenost od nejbližšího bodu je menší než 10 **then**
        najdi z_1, z_2, z_3 jako souřadnice z 1., 2. a 3. bodu v np
        urči z jako vážený průměr ze z_1, z_2 a z_3 podle převrácené hodnoty vzdálenosti
        přidej $p = [x, y, z]$ do $points$
        zvětši n o 1
    end if
end while

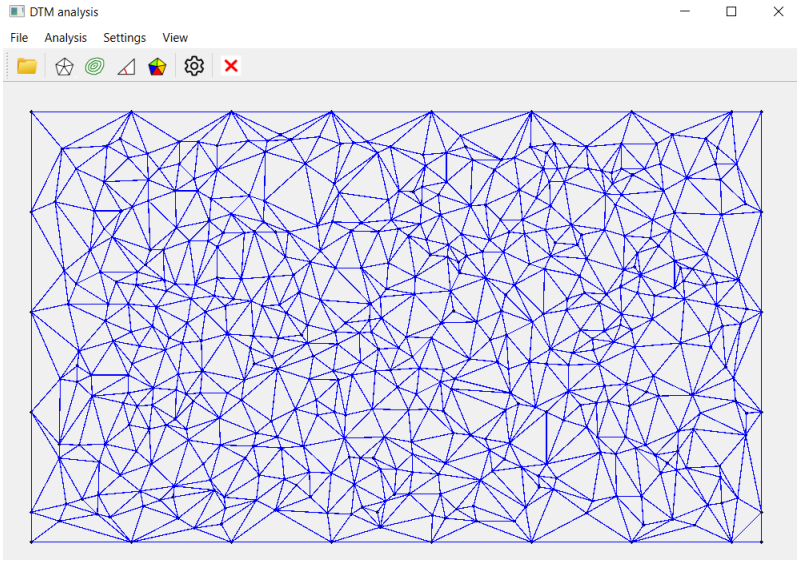
5 Výstupní data, formát výstupních dat, jejich popis

Výstupem je vizualizace vytvořené Delauneyho triangulace, vykreslování vrstevnic, anaýzy sklonu a expozice svahů, a znázornění terénu pomocí barevné hypsometrie.

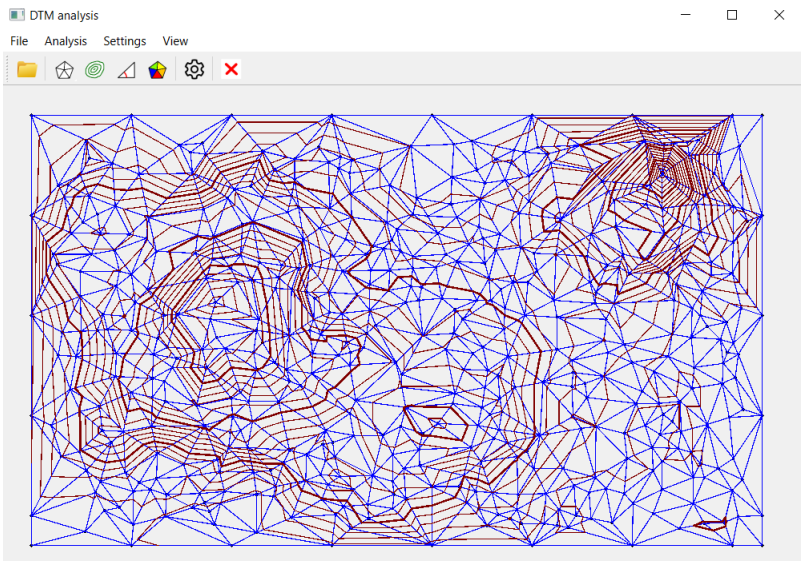
6 Printscreen vytvořené aplikace



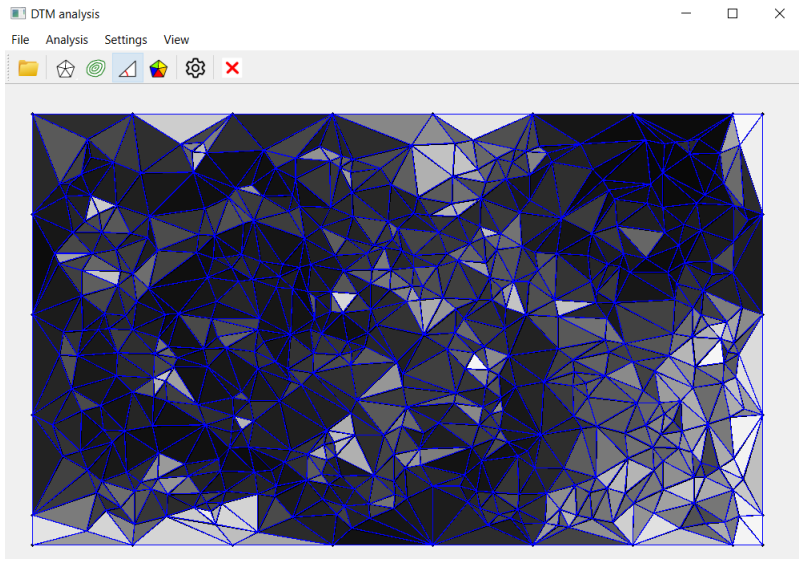
Obr. 2: Vygenerované bodové mračno.



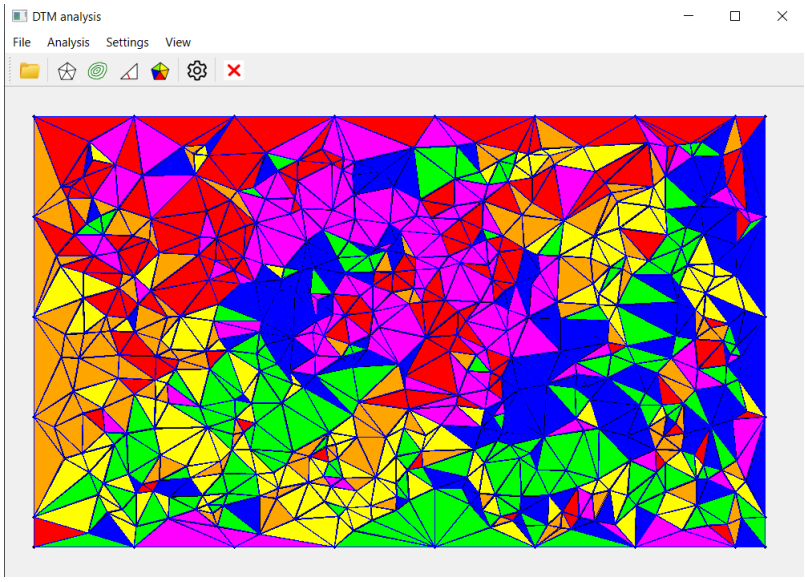
Obr. 3: Delauneyho triangulace nad bodovým mračnem.



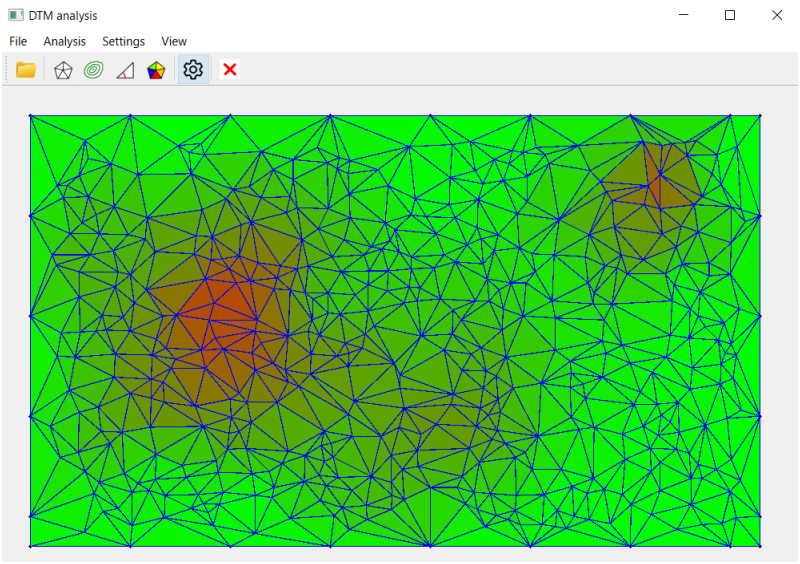
Obr. 4: Vykreslení vrstevnic.



Obr. 5: Analýza sklonu svahů.



Obr. 6: Analýza orientace svahů.



Obr. 7: Znázornění terénu pomocí barevné hypsometrie.

7 Dokumentace: popis tříd, datových položek, jednotlivých metod

7.1 Třída MainForm

Tato třída se nachází ve scriptu mainform.py, který slouží ke spouštění programu a jednotlivých algoritmů. Tato třída byla navržena v rozhraní Qt.

7.2 Třída Draw

Třída sloužící k vykreslování jednotlivých objektů. Obsahuje parametry self.__points, self.__dt, self.__contours, self.__t_contours, self.__triangles, self.__triangles2, self.__pols_hypsometrie. Obsahuje metody mousePressEvent (na místo kliknutí do widgetu přidá bod do seznamu bodů, a vygeneruje mu náhodnou z-souřadnici), paintEvent (vykresluje body, linie a polygony), setDT (list hran typu Edge Delauneyho triangulace přiřazuje do paramentu self.__dt), setContours (list vrstevnic přiřazuje do paramentu self.__contours - základní vrstevnice a self.__t_contours - zvýrazněné vrstevnice), setSlope (list trojúhelníků typu Triangle přiřazuje do parametru self.__triangles), set Aspect (list trojúhelníků typu Triangle přiřazuje do parametru self.__triangles2), setHypsometrie (list polygonů typu Polygon přiřazuje do parametru self.__pols_hypsometrie), getPoints (vrací self.__points), getDT (vrací self.__dt, getContours (vrací self.__contours), input (načte vygenerovaná data bodového mračna), clearLoadedData, clearAnalysis (vyčištění okna).

7.3 Třída Algorithms

Třída obsahující metody potřebné pro výpočetní část algoritmů. Obsahuje funkce getPointAndLinePosition (zjistí, zda je bod v levé či pravé polorovině od dané přímky), getDelaunayPoint (hledá optimální Delaunayovský bod), getNearestPoint (hledá nejbližší bod od daného bodu), updateAEL (testuje, zda hrana s opačnou orientací je v aactive edges list AEL), createDT (vytváření Delauneyho triangulace), getContourLinePoint (hledá průsečík vodorovné roviny a strany trijúhelníka), createContourLines (vytváří vrstevnice v daném intervalu s daným krokem), getSlope (počítá sklon trojúhelníka zadaného jeho vrcholy), analyzeDTMSlope (využívá getSlope a pomocí třídy Triangle z těchto bodů vytváří trojúhelník s daným sklonem), getAspect (zjišťuje orientaci trojúhelníka), analyzeAspect (využívá geAspect a pomocí třídy Triangle z těchto bodů vytváří trojúhelník s danou orientací), hypsometrie (znázorňuje terén pomocí barevné hypsometrie, k vytvoření polygonů vzniklých z průsečíku DT a vrstevnic využívá třídu Polygon).

7.4 Třída Generate

Tato třída obsahuje funkci GeneratePointCloud, která vytvoří syntetické bodové mračno a její pomocné funkce PorM (vrací náhodně 1 nebo -1), number (vrací počet bodů) a getPoi (vrací seznam vygenerovaných bodů).

7.5 Třída Edge

Třída obsahuje funkce getStart (vrací počáteční bod hrany), getEnd (vrací koncový bod hrany), switchOrientation (vytvoří novou hranu s opačnou orientací), definovaný operátor __eq__ (slouží ke zjištění, zda jsou dvě hrany shodné).

7.6 Třída QPoint3DF

Rodičovskou třídou této třídy je třída QPointF. Obsahuje fuknci getZ, která slouží k získání z-souřadnice bodu.

7.7 Třída Polygon

Třída slouží k vytvoření polygonu a získání jeho z-souřadnice.

7.8 Třída Triangle

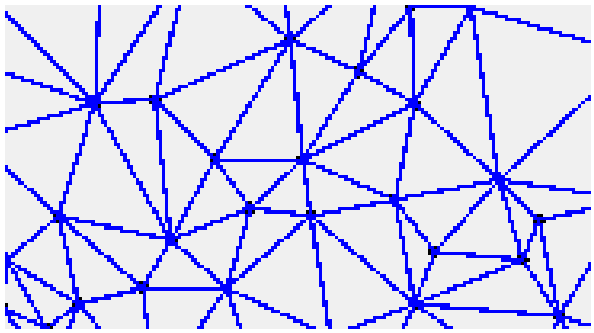
Třída reprezentující trojúhelník DT, vytvořená za účelem následného vybarvování trojúhelníků při analýze sklonu a orientace. Obsahuje funkce getP1, getP2, getP3, getSlope a getAspect.

8 Závěr

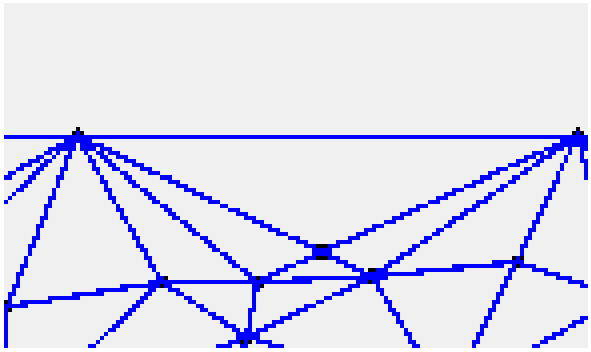
8.1 Zhodnocení činnosti algoritmu

V tomto příkladě je bodové mračno terénu generováno, není tedy založeno na žádném reálném terénu, proto je obtížné hodnotit, jak dobře terén vystihuje. Nicméně dá se zaměřit na vizuální zhodnocení funkcí aplikovaných na DT.

Co se týče samotné Delaunay triangulace nad bodovým mračnem (obr. 3), měla by tvořit ideální trojúhelníky, které co nejvíce vystihují terén. Toto pro naše bodové mračno platí, nicméně tvarově neideální trojúhelníky (trojúhelníky s příliš ostrými a tupými úhly) vznikají na okrajových bodech. Porovnání trojúhelníků prostorově uprostřed bodového mračna, a na okraji, je na následujících obrázcích.

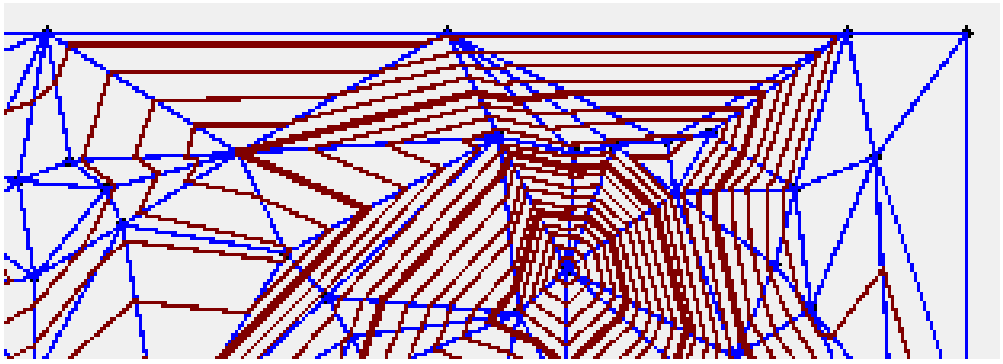


Obr. 8: Ukázka tvorby Delaunay triangulace v hustém bodovém mračnu.

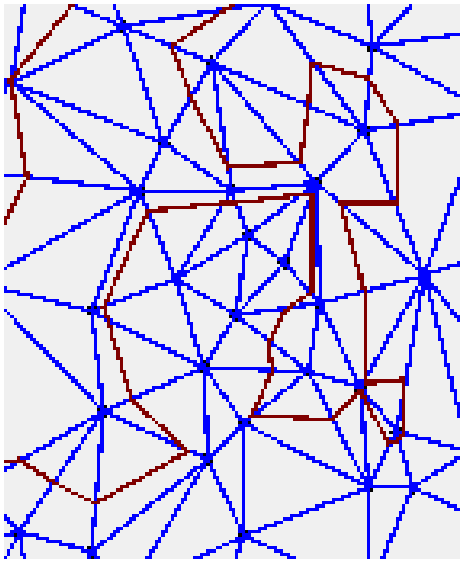


Obr. 9: Ukázka tvorby Delaunay triangulace na okraji bodového mračna.

Dalším krokem byla aplikace funkcí nad vytvořenou triangulací. První z těchto je tvorba vrstevnic pomocí lineární interpolace. Lineární interpolace je asi nejjednodušší možná, nicméně není reprezentativní co se týče reálného terénu. Náš terén je sice vygenerovaný automaticky, nicméně vizualně můžeme říci, že vrstevnice by v reálném případě nevypadaly tak, jak je generuje naše aplikace. Problémem jsou lomené čáry na okrajích trojúhelníků, které dávají vzniknout nepřirozeným tvarům. Některé ukázky jsou uvedené na následujících obrázcích.

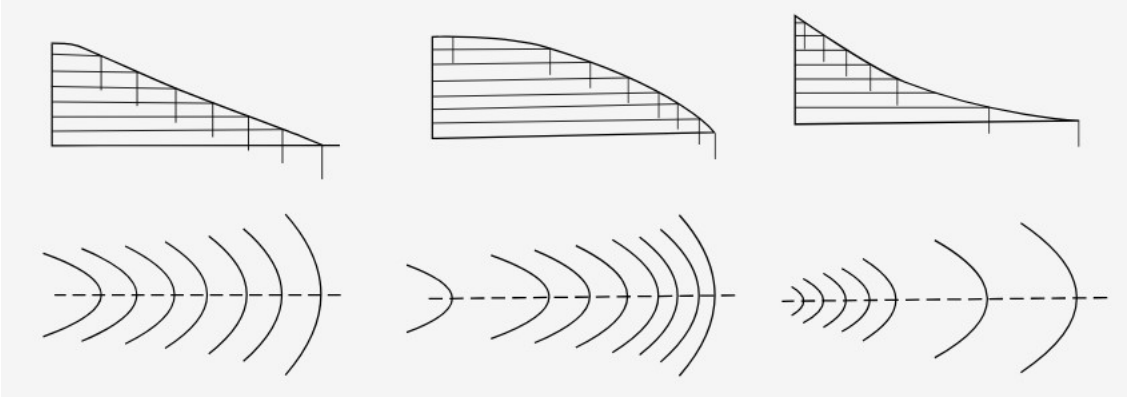


Obr. 10: Nevhodná ukázka vykreslení vrstevnic, vrstevnice paralelní s okrajem datasetu, nepřirozeně lomená čára.



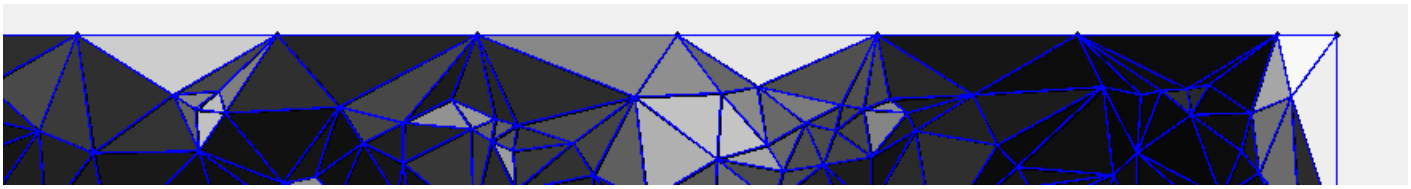
Obr. 11: Další ukázka nepřirozeně vykreslených vrstevnic lineární interpolací. Zde se tvoří nepřirozeně ostré úhly, které bychom v reálném terénu nenašli.

Pro generování vrstevnic je možné využít i jinou interpolaci než lineární. Například lze využít interpolaci morfologickou, která předpokládá plynulý spád terénu mezi interpolovanými body. Nevyužívá exaktní postup, zohledňuje skutečnost, zda se bod nachází blíže vrcholu, kde bývá spád terénu vyšší. Tento postup se ale využívá spíše pro mapy menších měřítek a je znázorněn na obrázku níže. Dále by pak bylo možné využít pro tvorbu vrstevnic beziérovy křivky, nicméně to už je velmi složitý postup. Možná nejlepším řešením pro větší datasety je ponechat pro generaci vrstevnic algoritmus lineární interpolace a tyto linie následně vyhladit, aby lépe připomínaly reálný terén.



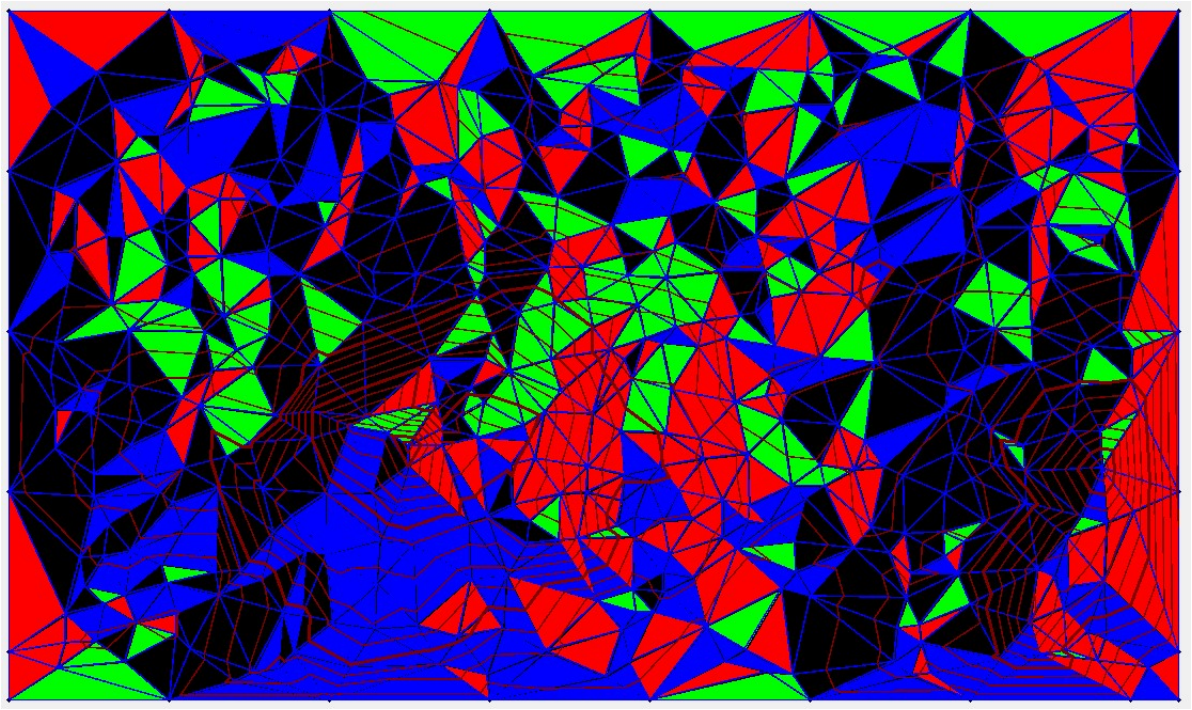
Obr. 12: Rozdíl mezi interpolací lineární (vlevo) a morfologickou (uprostřed, vpravo). Obrázek převzatý z https://pepa.fsv.cvut.cz/ mapovani/web/vyskopis/interpolace.html

Další analýzou nad naší Delaunayho triangulací je analýza sklonu svahů. Tato funkce funguje velice dobře a zabarvuje trojúhelníky s vysokým sklonem do tmavého odstínu, trojúhelníky s nízkým sklonem do světlého odstínu. Je vidět, že náš vygenerovaný dataset (obr. 5) je poměrně sklonitý, nejmenší sklon mají trojúhelníky v "jihovýchodní"části datasetu. Je zde ale vidět, že některé trojúhelníky na okraji datasetu jsou opravdu veliké, a označují poměrně veliké území jednou barvou - to pouze naznačuje to, že DT na okraji datasetu nevystihuje terén úplně ideálně. Celkově ale pro náš dataset platí, že ve výše položených oblastech, tedy okolo vrcholů, je terén sklonitější.



Obr. 13: Ukázka velikých trojúhelníků na okraji datasetu, nevystihuje ideálně terén.

Také jsme se zabývali analýzou sklonitosti terénu. Zde jsou trojúhelníky zabarvené podle jejich orientace na světovou stranu.



Poslední analýzou nad DTM, kterou jsme se zabývali, je barevná hypsometrie, která je znázorněná na obr. 7. Zde je nadmořská výška (nebo-li parametr z u každého bodu) znázorněna barevnou škálou od zelené pro nejmenší

nadmořské výšky po hnědou pro největší nadmořské výšky. Jsou zde vidět 3 vrcholy, nebo alespoň vyvýšené části nad naším datasetem. Oblast s nejvyšší nadmořskou výškou je v západní polovině, oblast s nejnižší nadmořskou výškou se nachází v jihovýchodní části datasetu. Celkově se body na okraji datasetu nachází v nižších nadmořských výškách. Pro barevnou hypsometrii je také možné vytvořit nespojitou stupnici, která by znázornila jednotlivé intervaly (např. po 100 m) v různě odstupňovaných barvách. Dále by také bylo možné vytvořit spojitou stupnici pro více barev (např. od zelené přes hnědou po bílou).

8.2 Náměty na vylepšení

Případné vylepšení aplikace by mohlo spočívat v přidání možnosti uživateli načíst si vlastní bodové mračno. V rámci tohoto cvičení bylo využito vlastní vygenerované bodové mračno, z důvodu obtížné dohledatelnosti reálného bodového mračna využitelného pro tento typ úlohy.

9 Seznam literatury

Bayer, Tomáš. 2023. "Rovinné triangulace a jejich využití." Praha. `http://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk5_new.pdf`.