Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

Programové vybavení pro analýzu geografických dat založenou na shlukování

Plzeň 2015 Ondřej Kaas

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 13. května 2015

Ondřej Kaas

Poděkování

Rád bych poděkoval prof. Dr. Ing. Ivaně Kolingerové za její cenné rady, podporu a velkou trpělivost, kterou mi při konzultacích věnovala.

Rovněž děkuji doc. Ing. Václavu Čadovi, CSc. za poskytnutí odborných rad, ochotu a vstřícný přístup.

Dále bych poděkoval firmě GIS-Stavinvex a.s. za poskytnutá data a jejich kolegiální přístup.

Anotace

Tato diplomová práce se zabývá použitím metody shlukování v reálných geomatematických problémech.

Cílem práce je vytvoření programového vybavení pro analýzu a zpřesňující lokalizaci mračna bodů založenou na shlukování.

V úvodu práce je popsána problematika vizualizace, shlukování a geomatematiky, za ní následuje popis řešení a diskuze výsledků.

Klíčová slova:

Shlukování, lokalizace, mračno bodů, GIS aplikace

Abstract

This Master thesis deals with using clustering methods in real geomathematics problems.

The goal of this thesis is to create a program for analysis and improvement localization of a cloud of points based on the clustering library.

In the beginning the problematics of visualization, clustering and geomathematics are described followed by the description of the solution and discussion of experiments.

Keywords:

Clustering, localization, cloud of points, GIS applications

Tato práce byla podporována z projektu the European Regional Development Fund (ERDF) - projekt NTIS (New technologies for Information Society), European Centre of Excellence CZ.1.05/1.1.00/0.2.0090.

Obsah

1 Úvod

V dnešní době přístrojů, které chrlí enormní množství naměřených dat, je nutné modernizovat dosavadní algoritmy a vymýšlet nové, aby bylo možné zpracovávat takto naměřená data automaticky, a to i v případě, že se nevejdou celá do paměti. Velikost dat obvykle brání výpočtu přesného výsledku, proto je častým požadavkem výsledek, který se alespoň blíží k správnému řešení. Jeden možný přístup spočívá v nalezení menší reprezentativní množiny, která se svými vlastnostmi bude podobat původním datům. Tuto množinu lze nalézt pomocí metody shlukování.

Metodu shlukování lze najít v celé řadě technických oborů, jako je např. zpracování obrazu, data mining a analýza dat. Principem shlukování je sloučení podobných elementů do skupin a reprezentace skupiny jedním prvkem. Druh elementu záleží na aplikaci a může být v podstatě jakýkoli. Shlukování lze realizovat celou řadou algoritmů a mnohé z nich řeší i problém velkých dat.

Hlavní náplní práce je využití metody shlukování na problémech z oblasti zpracování reálných dat pro GIS aplikace. Reálné GIS aplikace poskytují vektorová a rastrová data. Vektorová data uchovávají informace o jednotlivých objektech zájmového území formou bodů, linií a polygonů. Objekty jsou sdružovány do vrstev podle určité tématické souvislosti (např. vodstvo, lesy, budovy, památné stromy). U rastrových formátů dat je nositelem informace pixel, který může reprezentovat jeden celý objekt, jeho část, nebo je v pixelu ukryto více objektů.

Data pro GIS aplikace se získávají např. terénním měřením, leteckým snímkováním nebo z pozemního laserového skenování. Především posledním zmiňovaným způsobem vzniká enormní množství dat. Vlivem pohybu skenovacího systému během skenování však dochází k průběžnému posunu umístění těchto dat a tím vzniká nepřesnost, proto je nutné data před jejich použitím předzpracovat. Data, na kterých bylo v práci experimentováno, poskytla firma GIS-Stavinvex a.s. (se sídlem v Ostravě).

Potřebné shlukovací algoritmy nebylo nutné implementovat od začátku. K práci byla poskytnuta shlukovací knihovna Ing. Jiřího Skály, Ph.D. vytvořená na půdě KIV/ZČU.

2 Teoretické základy

V následujících kapitolách budou tématicky popsány odborné pojmy potřebné v práci.

Kapitola 2.1 se zaměřuje na proces zobrazení bodů na obrazovce počítače. Následuje popis matematického aparátu (kapitola 2.2) používaného v metodách shlukování (kapitola 2.3). Poslední část se věnuje hlavnímu problému v oblasti geomatematiky řešenému v práci (kapitola 2.4).

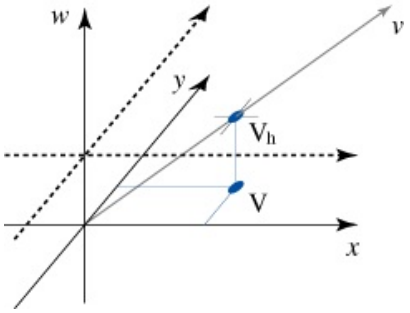
2.1 Grafické základy

Než bude možné definovat samotný proces zobrazení bodů na monitoru, je nutné zavést aparát homogenních souřadnic [zj04].

2.1.1 Homogenní souřadnice

Při práci s objekty jako, jsou přímky, kuželosečky apod., se často dostáváme do problémů. Dvě přímky mohou mít totiž jeden nebo také žádný průsečík. Kdybychom uměli pracovat s body v nekonečnu, mohli bychom například tvrdit, že dvě přímky mají vždy právě jeden průsečík. Z tohoto důvodu vznikla projektivní geometrie, která pro definici bodu umístěného v nekonečnu používá homogenní souřadnice. Zvolený prostor se rozšíří o jednu dimenzi. Pro snadnější představu bylo zvoleno rozšíření z roviny do 3D prostoru.

Nad rovinu xy umístíme rovnoběžnou rovinu ve výšce $w = 1$ (viz Obr. 2.1).



Obr. 2.1: Rozšíření 2D prostoru o homogenní souřadnice [zpg15]

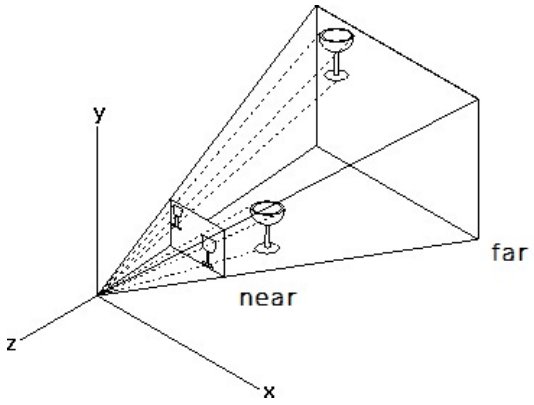
Bod v definován jako $v = [v_x, v_y]^T$, má tedy homogenní protějšek v bodu $v_h = [vh_x, vh_y, w]^T$. Z bodu v_h lze zpětně odvodit původní bod $v = [\frac{vh_x}{w}, \frac{vh_y}{w}]^T$. Bod v_h si můžeme představit i jako vektor a pokud nám nebude záležet na jeho velikosti, pak i jako celou skupinu vektorů $v = [v_x, v_y, w]^T$.

Pokud budeme homogenní souřadnici bodu v_h zvětšovat, jeho protějšek v se bude přibližovat počátku. Naopak zmenšováním se původní bod v bude vzdalovat do nekonečna.

Proto můžeme definovat bod b umístěný v nekonečnu jako $b = [b_x, b_y, 0]^T$. Bod s souřadnicemi $[0, 0, 0]^T$ nemá smysl a v homogenních souřadnicích je zakázán.

2.1.2 Perspektivní projekce

Při zobrazování 3D objektů na 2D zařízení (v našem případě obrazovku monitoru) je třeba stanovit způsob, jakým se toto zobrazení provede. Tímto způsobem je nejčastěji perspektivní projekce [zj04, str. 305, 317]. Největším charakteristickým rysem perspektivní projekce je deformace objektů. Objekty vzdalující se od pozorovatele se na obrazovce zmenšují. K tomu efektu dochází kvůli tvaru zorného pole ve tvaru čtyřbokého komolého jehlanu s úzkým koncem směřující k pozorovateli (viz Obr. 2.2).



Obr. 2.2: Perspektivní projekce [per15]

Čim větší objem v tomto jehlanu objekt vyplňuje, tím větší je zobrazen na obrazovce. Jehlan je omezen dvojicí rovin blízkou (*near*) a vzdálenou (far). V blízké rovině se nachází náš monitor.

K výpočtu pozice bodu v 3D prostoru zobrazeného na 2D obrazovce tzv. projekcí je použito maticového vyjádření. Nejvíce používanými maticemi k popisu projekce jsou :

- Modelová matice **M** - popisuje transformace, které se musí provést na všechny body k jejich převedení do souřadnicového systému zdrojového 3D prostoru. Hodnoty matice dostaneme maticovým násobením matice posunů **T** (*translation matrix*), maticí rotace **R** (*rotation matrix*) a maticí změny měřítka **S** (*scale matrix*). Bod b převedeme do modelového prostoru následující rovnicí (2.1). Předpokládáme sloupcovou notaci vektorů.

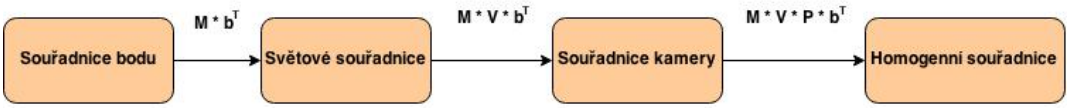
$$b' = \mathbf{M} * b = \mathbf{T} * \mathbf{R} * \mathbf{S} * b \tag{2.1}$$

kde :

`b`	...	bod, na který aplikujeme transformace
`b'`	...	výsledný transformovaný bod
M	...	modelová matice
T	...	matice posunů
R	...	matice rotací
S	...	matice změny měřítka

- Pohledová matice **V** - definuje pozici pozorovatele a jeho natočení
- Projekční matice **P** - definuje samotnou projekci (zorný úhel pozorovatele, poměr stran a hraniční roviny projekce)

Celý proces projekce bodu $b = [b_x, b_y, b_z, 1]^T$ do výsledného bodu $b' = [b'_x, b'_y, b'_z, b'_w]^T$ v 2D prostoru obrazovky lze zobrazit následujícím schématem Obr. 2.3.



Obr. 2.3: Schéma projekce bodu na obrazovku

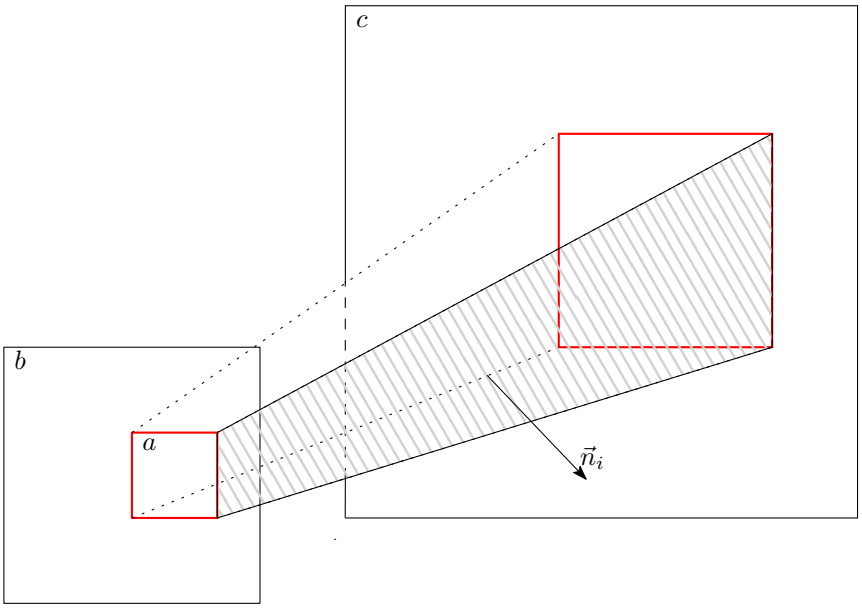
kde :

M	...	modelová matice
V	...	pohledová matice
P	...	perspektivní matice

U výsledného bodu b' poté můžeme zapomenout na homogenní souřadnici b'_w. Pokud by existoval další bod, který by byl promítán na stejné souřadnice jako bod b', porovnáním jejich z-souřadnic by se určilo, který z bodů bude zobrazen na monitoru.

2.1.3 Výběr bodů v prostoru

Na obrazovce definujeme výběrový obdélník a jeho projekcí pak vznikne komolý hranol. Výběrový komolý hranol bude stejně orientovaný jako komolý hranol zorného pole pozorovatele, bude však zabírat jeho menší část. Body výběrového hranol budou definovány pomocí homogenních souřadnic. Body blízko u kamery budou mít souřadnice $[x, y, 0, 1]^T$. Body umístěné v nekonečnu pak $[x, y, 0, 0]^T$. Z těchto bodů se pak vytvoří jednotlivé roviny komolého hranolu (viz Obr. 2.4) s normálovými vektory $\vec{n}_i$.



Obr. 2.4: Výběr bodů v prostoru

kde :

a	...	výběrový obdélník na obrazovce
b	...	obrazovka. Hraniční rovina $near$.
c	...	hraniční rovina far.
$\vec{n}_i$	...	normálové vektory bočních rovin definující výběrový objem.

Souřadnice vyšetřovaných bodů se dosazují do jednotlivých obecných předpisů hraničních rovin. Na základě znaménka výsledku se určí, v jakém poloprostoru se bod nachází. Pokud se bod nachází v druhém poloprostoru než do kterého směřuje vektor $\vec{n}$ dané roviny, je dále vyšetřován vůči dalším rovinám. Pokud splňuje tuto podmínku u všech rovin zároveň, je považován za vnitřní bod výběrového objemu.

2.2 Matematicé základy

V této kapitole bude popsán matematický aparát využitý v metodách shlukování.

2.2.1 Metrický prostor

Metrický prostor [jt13] je dvojice $P = (M, p)$, kde M je libovolná neprázdná množina a p je tzv. metrika, což je zobrazení $p : M \times M \Rightarrow R$, které splňuje následující axiomy (pro libovolná $x, y, z \in M$) :

1. Axiom nezápornosti: $p(x, y) \geq 0$
2. Axiom totožnosti: $p(x, y) = 0 \Leftrightarrow x = y$
3. Axiom symetrie: $p(x, y) = p(y, x)$
4. Trojúhelníková nerovnost: $p(x, z) \leq p(x, y) + p(y, z)$

Metrika poskytuje číselné vyjádření vzdálenosti mezi dvěma prvky. Vzdálenost je pojem relativní a může se měřit různě v závislosti na daném prostoru a konkrétní aplikaci. Výpočet vzdálenosti ale bude klíčový ve výpočtu podobnosti dvou elementů.

Každé množině M lze zadat celou řadu různých metrik. Tím se vytvoří různé metrické prostory, které budou mít stejnou základní množinu, tzv. nosič, ale v každém z nich bude jiným způsobem měřena vzdálenost.

V každé z metrik poté zacházíme s D-dimenzionálními body definovanými jako $x = (x_1, x_2, x_3, ...x_D)^T$ kde $x_1, x_2, ...x_D$ představující souřadnice daného bodu. Vzdálenost dvou bodů x a y je zapsána jako $d(x, y)$.

Obecný předpis pro metriky v D-dimenzionálním prostoru $\mathbb{R}^D$ lze definovat jako :

$x, y \in \mathbb{R}^D, x = (x_1, x_2, ..., x_D)^T,$	$y = (y_1, y_2, ..., y_D)^T$:	
$d(x,y) = \sqrt{\sum_{i=1}^{D}(y_i - x_i)^2}$	...	eukleidovská vzdálenost
$d(x,y) = \sqrt{\sum_{i=1}^{D}\lvert y_i - x_i\rvert}$	pro $D <= 2$,	manhattanská metrika
	pro $D > 2$,	oktaedrická metrika
$d_{max}(x,y) = max\{\overset{D}{\underset{i\in 1}{\forall}}\lvert(y_i - x_i)\rvert\}$	...	maximová metrika

K výpočtu vzdálenosti se nejvíce používá vzorce eukleidovské vzdálenosti, který je jednoduchý a odpovídá základní představě o prostoru.

2.2.2 Váhy

Pomocí vah lze zanést do vztahu pro výpočet podobnosti elementů důležitost některých souřadnic. Toho je docíleno obohacením vzorce metriky o koeficient váhy příslušné souřadnice. Výpočet vzdálenosti d dvou bodů $x = (x_1, x_2, ...x_D)^T$ a $y = (y_1, y_2, ...y_D)^T$ pomocí metriky M, ovlivněný váhami $w = (w_1, w_2, ...w_D)^T$, je zapsán následujícím vzorcem 2.2.

$$d(x,y) = \sqrt{\sum_{i=1}^{D}((x_i - y_i) * w_i)^2} \tag{2.2}$$

kde :

D	...	dimenze daného metrického prostoru
x_i	...	i-souřadnice bodu x
y_i	...	i-souřadnice bodu y
w_i	...	váha souřadnice $i, (w_i \in \mathbb{R})$

Jednotlivé souřadnice mohou mít rozdílné rozsahy. Pro korektní vliv vah u příslušných souřadnic je nutné, aby všechny souřadnice měly stejný rozsah. Toho je docíleno přepočtem hodnoty dané souřadnice na hodnotu souřadnice s největším intervalem ze všech souřadnic. Souřadnice s největším intervalem je vybrána z důvodu zachování přesnosti.

Převod souřadnice A do intervalu souřadnice B s největším intervalem se provede následujícím vzorcem (2.3).

$$k = (A - A_{min}) * \frac{B_{max} - B_{min}}{A_{max} - A_{min}} + B_{min} \tag{2.3}$$

kde:

A	...	hodnota souřadnice
A_{max}, A_{min}	...	maximální a minimální hodnota souřadnice A
B_{max}, B_{min}	...	maximální a minimální hodnota souřadnice B

2.3 Metody shlukování

Principem shlukování je sloučení většího počtu podobných elementů dohromady a jejich reprezentace menším počtem elementů. Podoba elementu poté záleží na aplikaci a může být v podstatě jakákoli - od bodů v 1D prostoru[1] po 3D objekty, celé digitální obrázky, dokumenty nebo databázeové entity. Společným znakem elementů je jejich možné vyjádření specifickým vektorem. Například, body jsou popsány pomocí jejich prostorových souřadnic. Pro některé složitější abstrakce elementů je nutné najít odpovídající vyjádření pro jejich vlastnosti. Shlukování je NP-těžký problém, a proto výsledky algoritmů jsou pouze aproximací správného řešení.

Výsledkem shlukování budou množiny elementů s největší podobností jejich vektorů v rámci jedné množiny. Každou množinu poté reprezentuje jeden element. Rozhodnutí, do jaké míry jsou si dva elementy podobné, a tedy jestli patří do stejného shluku, se provede pomocí tzv. metriky, popsané v kapitole 2.2.1.

V některých případech není žádoucí, aby výsledkem byly shluky elementů podobných v rámci všech vlastností. Některé souřadnice mohou být důleži-

[1] Shlukování hloubkové informace bodu pro renderování.

tější nežli jiné. Například v oblasti digitálního obrazu jsou body (*pixely*) reprezentovány nejen svými polohovými souřadnicemi, ale i souřadnicemi v barevném prostoru *RGB*. Pokud bude cílem shlukovat pouze na základě podobnosti barevných souřadnic bodů, klasická metrika zde neposkytne kýžené řešení. Je nutné do metriky zanést vztah, který bude zvýhodňovat požadované souřadnice. Toho je docíleno pomocí vah, kterým je věnována kapitola 2.2.2.

2.3.1 Shlukovací metody

Existuje mnoho shlukovacích algoritmů, používaných napříč technickými obory. Shlukovací algoritmy mohou být rozděleny podle jejich konkrétních funkcí a principů do dvou protichůdných cest vedoucích k řešení. V následujících odstavcích budou některé z nich představeny [sj13, str. 32,33].

Princip shlukování může být buď hiearchický (*hiearchical*) nebo nehierarchický (*partitional*[2]). Nehierarchický algoritmus rozdělí data mezi přesný počet shluků (segmentů). Hiearchický algoritmus vytvoří hiearchii malých shluků sloučených do shluků větších tvořících stromovou strukturu zvanou dendrogram. Stupněm shlukování pak lze kontrolovat počet vytvořených úrovní samotné hierarchie.

Jiné možné rozdělení algoritmů je aglomerativní (*algomerative*) nebo divizní (*partitional*). Aglomerativní shlukování začíná ve stavu, kdy jsou všechny vstupní elementy považovány za centra shluků. Tato centra jsou postupně spojována na základě jejich podobnosti do té doby, dokud není splněna zastavovací podmínka. Algoritmus je obvykle zastaven ve chvíli, kdy je vytvořeno požadované množství shluků nebo pokud podobnosti elementů klesnou pod krajní mez, kdy již elementy nemají být přiřazeny k sobě. Divizní algoritmus jde k řešení opačnou cestou. Algoritmus začíná s všemi elementy přiřazenými do jednoho velkého shluku. Ten je poté opakovaně rozdělován na základě nepodobnostní podmínky. Algoritmus opět skončí ve chvíli, kdy je vytvořeno požadované množství shluků nebo shluky jsou tak homogenní, že není potřeba dalšího dělení.

Shlukování může být přísné (*hard*) nebo měkké (*fuzzy*). Přísné shlukování přiřazuje každý element právě do jednoho shluku, oproti tomu měkké shlukování určuje počet přiřazení jednoho elementu do více shluků.

[2] v české literatuře se lze setkat s pojmem nehierarchické metody shlukování

Shlukovací algoritmy mohou být deterministické (*deterministic*) nebo stochastické (*stochastic*). Mezi stochastické techniky obvykle patří náhodné algoritmy. Ty jsou většinou používány na velká data pro svoji rychlost.

Shlukovací techniky mohou zpracovávat celá data najednou nebo pracovat postupně (inkrementálně). Pokud bude algoritmus zpracovávat celá vstupní data najednou, lze očekávat přesnější výsledky. Inkrementální algoritmus může být rychlejší a díky menší náročnosti na paměť ho lze použít i na velká data. Málá náročnost na paměť plyne z faktu, že si algoritmus neuchovává všechny informace o vstupních datech, pouze nejdůležitější informace o konkrétních shlucích nutných pro další pokračování.

Z možných shlukovacích algoritmů byl pro tuto diplomovou práci vybrán *facility location*, který bude popsán v následující kapitole 2.3.2. Tento algoritmus byl totiž již dříve implementován v poskytnuté knihovně [sk09] Ing. Jiřího Skály, Ph.D.

2.3.2 Facility location

Algoritmus byl navržen pro velký objem dat, který se zpravidla celý nevejde do paměti počítače a proto se musí načítat po menších blocích (tzv. datastreamové shlukování). Následující odstavec popisuje princip algoritmu (převzato [sj13, str. 35,36]).

V popisu algoritmu jsou používány následující formulace. Nechť písmenem F jsou označeny centra shluků (*facilities*), písmenem C označeny přiřazené body ke shluku (*clients*) a všechny body se mohou stát centry shluků. Problém je v rozhodnutí, který z bodů se má stát centrem shluku a které body mají být k němu přiřazeny. Algoritmus rozhoduje na základě ohodnocení, např. za „otevření shluku“ (prohlášení bodu za centrum shluku) je nutné „zaplatit“ cenu fc (*facility cost*). Další cenou je spojovací cena (*service cost*), většinou závislá na vzájemné vzdálenosti obou elementů. Analogii problému nalezneme v aplikaci z reálného života. Představme si město, kterému musíme dodávat elektřinu. Máme k dispozici několik míst, kde je možné postavit rozvodnu elektřiny. Postavení rozvoden na všechna možná místa je moc drahé, stejně tak jako připojení všech domácností k jedné centrální rozvodně. Je nutné zjistit, na kterých místech postavit rozvodny a která místa budou pouze připojena. Jinými slovy najít optimální řešení, které povede k minimalizaci nákladů na stavby.

Algoritmus se poté snaží minimalizovat celkovou cenu Q definovanou jako

$$Q = \sum_{j \in F} fc + \sum_{i \in C} c_{ij} \tag{2.4}$$

kde:

fc	...	cena za otevření nového centra
F	...	množina center shluků
C	...	množina přiřazených bodů
c_{ij}	...	cena za spojení přiřazeného bodu i k jeho centru shluku j

Vzdálenost je obecně považována za kladnou, symetrickou a splňující trojúhelníkovou nerovnost. Důležité je podotknout, že neexistují žádná omezení mezi množinou center shluků a množinou přiřazených bodů. Množina F může být nezávislá na množině C, podmnožinou C nebo dokonce shodná s C.

Na začátku shlukování není specifikováno, kolik se má vytvořit center shluků, jako tomu může být u jiných algoritmů. Jediným možným ovlivněním je koeficient ceny za otevření shluku. Vysoká hodnota ovlivní výpočet ceny ve prospěch velkých shluků. Otevření nového centra shluku se stane velmi drahé, a tak se body raději přiřadí k centru, než aby se vytvořil nový shluk. Oproti tomu malá cena vytvoří velké množství shluků. Otevření shluku je levné a proto se mnoho bodů stane centrem shluku.

Metod hledajících minimální cenu ohodnocení existuje velké množství, např. lineárním programování se zaokrouhlením na celá čísla (*linear programming rounding*), dualita úloh (*primal-dual algorithm*) nebo níže popsaná metodou lokálního vyhledání (*local search*) [sj13, str. 36-38].

Local search

Metoda local search vytváří graf možných řešení. Jednotlivé uzly v grafu představují dané ohodnocené řešení. Uzly jsou poté spojeny hranami, pokud jedno z řešení lze získat z druhého určitým typem modifikace. Algoritmus poté prochází daný graf a hledá lokální minimum. Jinými slovy, takové řešení, které má menší ohodnocení než všichni jeho sousedi. První řešení je náhodně vygenerované, to je dále iterativně vylepšováno lokálními úpravami. Za cen-

trum shluku je zvolen náhodný bod a poté je zjištěno, zdali tímto otevřením bude vylepšeno dosavadní řešení. Pokud jsou v blízkosti nového centra nějaké body, budou k novému centru přiřazeny. Pokud tímto přeřazením vzniknou shluky s malým počtem bodů, budou jeho body také přeřazeny k novému shluku. Algoritmus poté zavádí funkci vylepšení (*gain*), na základě které lze rozhodnout, zda-li bylo nalezeno lepší řešení.

Nyní si popíšeme algoritmus podrobněji s použitím formulací z předešlých odstavců.

Náhodně se vybere bod a prohlásí se za centrum shluku ($j \in F$), nezáleží, zdali jím již byl nebo ne, a zjistí se možné vylepšení. Pokud jím nebyl, je nutné zaplatit cenu za jeho otevření. Pokud jsou v blízkosti nějaké body, které mají vzdálenost k svému dosavadnímu centru shluku větší než k nově vytvořenému j, přeřadí se k j (tím se zmenšuje spojovací cena c). Takto mohou vzniknout centra shluků s malým počtem přiřazených bodů. Tato centra lze uzavřít a ušetřit tak cenu za jejich otevření (fc). Jejich přiřazené body se přiřadí k nově vytvořenému centru j a zaplatí se nová cena za spojení, která může být menší než cena nového spojení. Po těchto upravách je nutné vypočítat funkci *vylepšení*. Pokud bude *vylepšení(j)* > 0, bod j bude prohlášen za centrum shluku (pokud jím již není) a dané úpravy budou přijaty.

Pro definici funkce *vylepšení* je nutné zavést vzdálenostní rozdíl ds_i (*distance spare*) jako rozdíl vzdáleností bodu k jeho dosavadnímu centru shluku a k novému kandidátovi f. Pokud je rozdíl záporný, tedy dosavadní centrum leží blíže než f, nastaví se $ds_i = 0$. Dále je nutné definovat úsporu ceny za uzavření cs_j (*close spare*) jako cenu, kterou lze ušetřit zavřením daného shluku f_j. Tato cena se rovná ceně za vytvoření shluku mínus cena za všechny přeřazené body z f_i do f. Pokud je cena cs_j záporná, tedy nelze nic ušetřit, nastaví se $cs_j = 0$.

Funkci *vylepšení* lze tedy definovat následujícím vzorcem 2.5 :

$$vylepšeni = -fc + \sum_{c_i \in C} ds_i + \sum_{f_j \in F} cs_j \tag{2.5}$$

kde:

fc	...	cena za otevření nového centra, pokud jím již není
ds_i	...	rozdíl spojovacích cen bodu k jeho dosavadnímu a k novému shluku
cs_j	...	ušetřená cena za uzavření shluku a přeřazení všech jeho bodů k novému shluku
c_i	...	všechny přeřazené body
f_j	...	všechny uzavřené shluky

Teoretické odvození algoritmu ukládá $\mathcal{O}(N \log N)$ opakování, kde N je počet potencionálních center shluků. Experimenty byla zjištěna podobnost řešení už při $\frac{1}{10}N$ opakování.

Algoritmus lze poté popsat následujícím pseudokódem:

```
vygeneruj prvotní řešení;
while opakováno < 1/10 N do
    náhodně zvol bod j a prohlaš za centrum shluku;
    if vylepšení(j) > 0 then
        proveď příslušné změny (přeřazení bodů, uzavření center
        shluků);
    end
end
```

2.3.3 Datastreamové shlukování

Existují tři základní přístupy datastreamového shlukování [sj13, str. 41].

Prvním intuitivním přístupem je rozděl a panuj. Vstupní data jsou rozdělena na několik bloků a každý z nich se poté shlukuje samostatně. Ze zvolených výsledků daných bloků se poté „složí“ výsledné řešení za celá vstupní data. Pokud bude na vstupu velké množství dat, hiearchie může růst do více

úrovní.

Dalším možným přístupem je inkrementální shlukování. Shluk je vytvořen s prvním vstupním elementem. Následující elementy jsou buď'to přiřazeny k již stávajícím shlukům nebo jsou prohlášeny za nová centra shluků. Vytvoření nového shluku nebo přiřazení k některému stávajícímu shluku se děje na základě podobnostního ohodnocení. Hlavní výhodou inkrementálního algoritmu jsou jeho malé nároky na pamět, protože algoritmus nemusí ukládat všechna vstupní data do paměti. Další jeho výhodou je neiterativní přístup. O vstupním elementu je pouze jednou rozhodnuto a již vícekrát se k němu algoritmus nevrací. Hlavní nevýhodou algoritmu jsou data v náhodném pořadí. Jinými slovy, pokud budou tyto podobné elementy v datastreamu umístěny daleko od sebe, pak je bude nucen algoritmus přiřadit nevýhodně k různým dosavadním shlukům nebo vytvořit nové shluky. Tuto chybu již algoritmus nedokáže opravit.

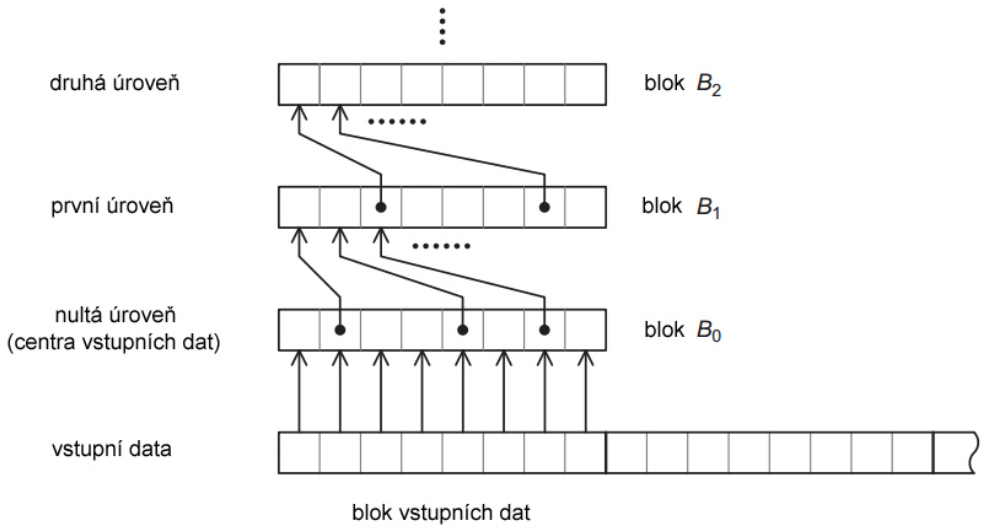
Posledním přístupem, který se v posledních letech stává velmi populární i v jiných odvětvích, je paralelní, distribuované řešení. Algoritmy jsou upraveny tak, aby je bylo možné rozložit na jednotlivé nezávislé etapy. Tyto etapy jsou poté zpracovávány paralelně na více počítačích.

V následující sekci bude popsán hiearchický přístup využívající myšlenku rozděl a panuj [sj13, str. 41-43].

Hiearchické shlukování

Hiearchické datastreamové shlukování rozděluje vstupní data na menší bloky, které poté zpracovává. Po shlukování takového bloku se ohodnocení výsledných center vynásobí počtem přiřazených bodů. Výsledná centra s ohodnocením se poté uloží na externí uložiště (např. na pevný disk). Takto uložená data jsou považována za další datastream a mohou být dále zpracovávána jako původní data. V následujících úrovních bude poté zohledněno jejich dosavadní ohodnocení, kterým bude násobena vzdálenost k jejich novým centrům shluků. Nové ohodnocení center shluků je poté sumou všech přiřazených bodů (původně také center shluků nižší úrovně).

Algoritmus nejlépe popíše následující Obr. 2.5. Načteme úsek dat do úrovně B_0 a shlukujeme. Výsledné shluky uložíme do vyšší úrovně B_1. Zaplňování vyšší úrovně pokračuje do chvíle, než je daná úroveň plná a je nutné je opět shlukovat. Výsledné shluky opět uložíme do vyšší úrovně.



Obr. 2.5: Hiearchická struktura ukládání shluků [cg13]

Algoritmus lze popsat následujícím pseudokódem:

while *jsou vstupní data* **do**
  načti úsek vstupních dat do nejnižšího bloku B_i;
  `zpracuj` (B_i);
end

Funkce `zpracuj` je poté definována:

`shlukuj` B_i;
vypočti nové ohodnocení centrům shluků;
přesuň centra shluků do bloku B_{i+1};
if *blok* B_{i+1} *je plný* **then**
  shlukuj B_{i+1};
end

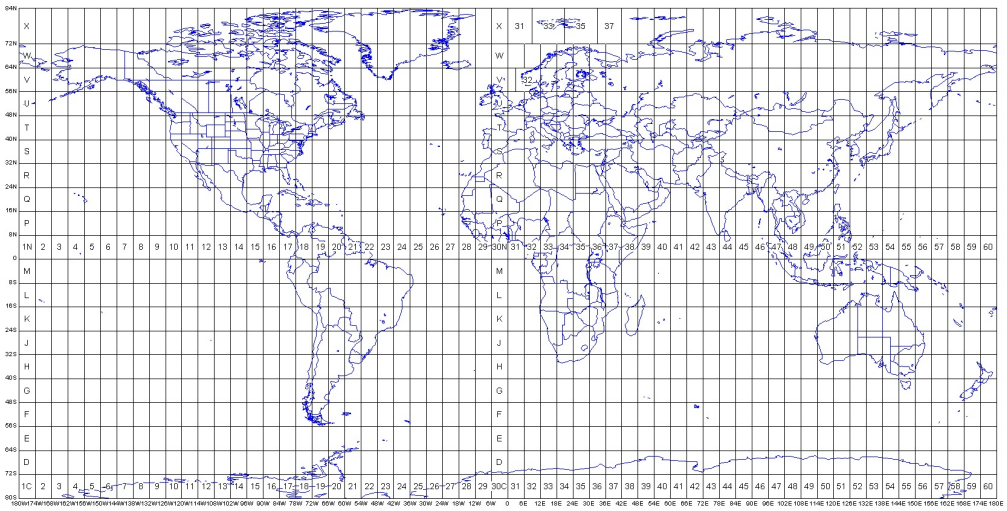
Algoritmus udržuje v každé úrovni maximální počet center shluků. Vyšší úroveň shlukuje až ve chvíli, kdy je potřeba uložit další centra a tím uvolnit místo v paměti.

2.4 Geomatické základy

V následujících kapitolách budou nejdříve popsány souřadnicové systémy používané v průběhu práce, následovány kapitolami popisující konkrétní řešené problémy v oblasti geomatiky.

2.4.1 UTM

Univerzální transverzální Mercatorův systém souřadnic [gen15] je primárně kartografické zobrazení zemského povrchu do roviny. V této rovině jsou definovány souřadnicové systémy po jednotlivých sekcích. Tyto sekce jsou definovány poledníkovými a rovnoběžníkovými pásy (viz Obr. 2.6).



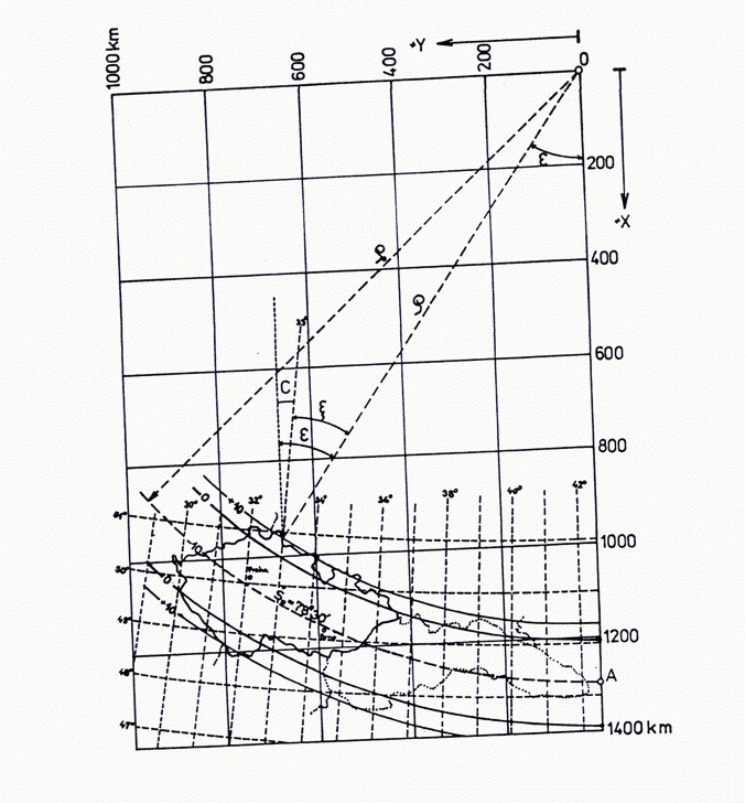
Obr. 2.6: Systém souřadnic UTM [utm15]

Každá tato sekce musí být v tomto systému jednoznačně určena a až následně střed této rovinné sekce tvoří počátek rovinné souřadnicové soustavy. Od tohoto středu se měří vzdálenosti v metrech po ose X rostoucí od středového poledníku směrem na východ (tzv. *eastings*) a po ose Y rostoucí od rovníku směrem na sever (tzv. *northings*).

2.4.2 Systém jednotné trigonometrické sítě katastrální

Systém jednotné trigonometrické sítě katastrální (S-JTSK) je pravoúhlý rovinný souřadnicový systém používaný na území České republiky a Slovenska jako státní referenční souřadnicový systém [gen15].

Vychází z dvojitého konformního kuželového zobrazení v obecné poloze tzv. Křovákova zobrazení (stanoveného Josefem Křovákem v roce 1922). Snahou bylo zavést takový pravoúhlý souřadnicový systém, v němž by se celá tehdy Československá republika nacházela v prvním kvadrantu a měla tedy obě souřadnice kladné (viz Obr. 2.7).



Obr. 2.7: Orientace hlavních os JTSK [kro15]

Kladná část osy X tohoto souřadnicového systému je směřována k jihu, kladná část osy Y k západu. Pro každý libovolný bod na území České i Slovenské republiky navíc platí, že hodnota souřadnice Y je vždy menší než souřadnice X.

2.4.3 Polohopisné body

Polohopis znázorňuje vzájemnou polohu objektů na zemském povrchu bez závislosti na terénním reliéfu. Jeho hlavními složkami jsou pobřežní čáry, vodstvo, vegetační porosty, sídla, hranic pozemků, obvodů budov aj.

Jednotlivé body polohopisu jsou získány přesným terénním geodetickým měřením, které jsou pak zaneseny do podrobné mapy v podobě značek znázorňující daný objekt.

2.4.4 Laserové skenování

Laserové skenování nebo též LIDAR [lid15] využívá principu pulzního bezhranolovýho dálkoměru, který pracuje s vysokou frekvencí, řádově v desítkách tisíc `Hz`. Laserový paprsek je vysílán senzorem k danému objektu a je měřen tranzitní čas od doby vyslání po dobu návratu odraženého parsku. Tímto způsobem laserskenové systémy produkují data, která se dají označit za pseudonáhodná distribuovaná bodová mračna. Tyto body mohou obsahovat více informací než klasický 2.5D model, v kterém je nadmořská hodnota z-hodnotou funkce x a y. To znamená, že vertikální zdi lze v jistých případech opravdu považovat za vertikální, získat povrch pod mostem nebo určit objemy jednotlivých objektů. Toho je mj. docíleno požitím kontiunálního snímání nebo použitím více směrů snímání.

Mračno bodů může být získáno s rozdílnými atributy závisejícími na aplikaci a samotném skanovacím zařízení. Za některé z těchto atributů můžeme považovat:

- hustotu bodů - závisí na nastavení vzorkovací rychlosti systému
- registraci vícenásobných ozvěn - odražený paprsek se může od objektu k čidlu dostat z více směrů
- amplitudu registrace (odrazivost) - napomáhá filtrovacím algoritmům „roztřídit“ objekty dle jejich materiálu, díky tomu lze odlišit např. průčelí staveb od travnatých ploch

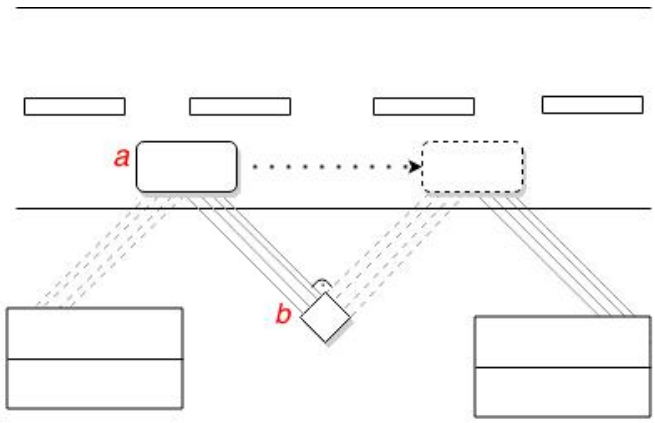
Laserové skenovací systémy lze umístit na statické místo a docílit tak nejpřesnějšího skenování. Této techniky se využívá pro získání přesného 3D

modelu nehybných objektů, jako jsou např. interiéry budov nebo sochy. Další možností je umístění systémů na dopravní prostředky nebo bezpilotní letouny (drony). Ty pak dokáží zdokumentovat jiným způsobem nepřístupné oblasti, jako jsou hustě zabydlené oblasti měst, lesy nebo elektrické rozvodní sítě.

2.4.5 Terénní skenování

Skenovací zařízení se umístí na jedoucí vůz a skenuje se okolí silnice. Otáčením záznamové hlavy laserového systému vznikají jednotlivé řádky bodů tvořící bodové mračno. Ze získaného bodového mračna vzniká digitální obraz lokality. Během terenního laserového skenování jsou získávány a ukládány i další informace jako je GNSS (Globální Navigační Satelitní Systém)[gns15] a rychlost vozidla. Všechny tyto informace se během jízdy vozidla synchronizují a vytvářejí výsledná mračna bodů.

Skenovací systém obsahuje nejméně dva separované laserové skenery umístěné na střeše jedoucího vozidla (viz Obr. 2.8).



Obr. 2.8: Získání bodového mračna

Při jízdě vozidla (a) na Obr. 2.8 je objekt (b) naskenován předním a následně i zadním skenerem. Tyto skenery jsou navzájem kolmo umístěny, aby vzniklo mračno bodů, z kterého lze odvodit střed nebo objem daného objektu.

Před začátkem jízdy se pomocí GNSS získá přesná poloha. V průběhu jízdy ale dochází k výpadkům tohoto GNSS signálu. V těchto „hluchých“ in-

tervalech je pozice vozidla přibližně dopočítána pomocí IMU jednotky (*inertial measuring unit*).

2.4.6 Data z laserového skenování

Celý laserový skenovací systém postupně generuje po dvou binárních souborech formátu `.las` (každý sken zvlášť). Prefixy těchto souborů jsou `left` pro levý, resp. `right` pro pravý sken. Z pohledu směru jízdy vozidla se jedná o první (levý) a druhý (pravý) sken. Za tímto prefixem následuje čas začátku pořizování skenu. Jako sufix, je doplněno inkrementální identifikační číslo začínající vždy od 0. Není vyjímkou, že některý ze skenů při jízdě přestane skenovat. Nové skenování započne s jiným startovacím časem a resetovaným identifikačním číslem. Záléží na konfiguraci systému, jak velké jednotlivé úseky budou zaznamenány.

2.4.7 Formát LAS

Standart formátu `.las` rozděluje soubor do několika úseků viz Tabulka (2.1).

PUBLIC HEADER BLOCK
VARIABLE LENGTH RECORDS
POINT DATA RECORDS

Tabulka 2.1: Definice formátu LAS

- `PUBLIC HEADER BLOCK` - obsahuje mj. následující informace :
  - jméno souboru
  - verzi
  - datum a čas vytvoření souboru
  - identifikaci skenovacího systému
  - formát bodových dat
  - měřítko, offset XYZ aj.

- `VARIABLE LENGTH RECORDS` - obsahuje informace o uživateli, který skenování započal, o jaký v pořadí se jedná sken aj.
- `POINT DATA RECORDS` - informace o bodech získaných během skenování. Existuje 6 formátů bodových záznamů s rozdílnou podrobností popisu bodu. Kromě souřadnic bodů XYZ všechny formáty dále obsahují intenzitu, počet zaznamenaných odrazů daného bodu nebo zdali se jedná o poslední bod v rotaci skenu.

  Vzniká tak popis 4-dimenzionálního vektoru v definovaného jako :

  $$v = (x, y, z, i)$$

  kde :

x, y, z	...	souřadnice bodu v geografickém souřadnicovém systému
i	...	intenzita odraženého paprsku

2.4.8 Zpřesňující lokalizace bodového mračna

Vlivem nepřesného získání pozice pomocí GNSS, samotné jízdy vozidla a s tím spojenými otřesy laserového systému vzniká lokalizované mračno bodů s přesností v desítkách centimetrů až metrů. Před použitím bodové mračna v dalších aplikacích, je nutné tuto přesnost zlepšit na úroveň geodeticky zaměřených polohopisných bodů.

Zpřesňující lokalizace je proces hledání identických bodů to znamená nalezení bodů v mračně korespondující s polohopisnými body. Mračna bodů typicky tyto polohopisné body neobsahují, ať už kvůli malé frekvenci skenování nebo kvůli poloze polohopisného bodu (obvykle střed objektu v místě dotyku se zemským povrchem). Proto je nutné takový bod vytvořit odvozením z okolních bodů mračna.

Proces zpřesňující lokalizace probíhá manuálně. Uživatelé musí postupně identifikovat nebo odvodit všechny identické body v mračnu bodů. Manipulaci s těmito daty ztěžuje jejich enormní velikost pohybující se v řádech 10^8 bodů. Díky této enormní velikosti se mračna rozdělí na menší krychle, které se samostatně lokalizují. Rozdělením celé vstupní množiny vznikají problémy s určením objektů na hranicích krychlí. Nějaká část zaměřeného objektu může být obsažena v sousední krychli nebo daná krychle bude obsahovat malé množství identických bodů. Pro zachování přesnosti se požaduje na 100 m dat z laserového skenování minimálně 1 identický bod.

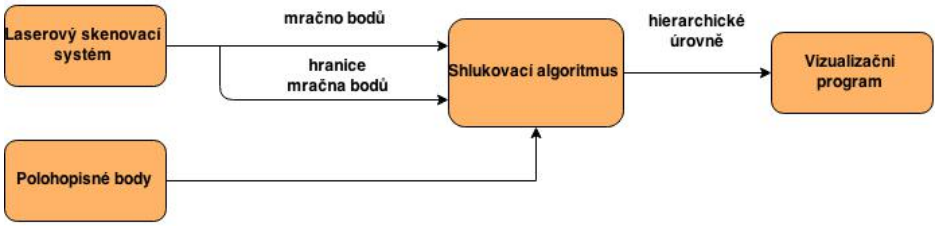
3 Definice úkolu a návrh řešení

Proces lokalizace by zjednodušilo odstranění nutnosti rozdělení vstupních dat na menší krychle. Tím by odpad problém s nalezením identických bodů na hranicích těchto krychlí. Uživatelé by pak mohli pracovat s celým mračnem bodů naskenovaného území a tak mít k dispozici globální náhled na celé území. Další příležitostí k urychlení by bylo v programovém hledání identických bodů. Výsledkem by byla množina identických bodů, které by uživatel měl možnost schválit nebo dle svých zkušeností zvolit jiný bod z konkrétního území.

Hierarchické shlukování vytvoří hierarchickou strukturu bodů rozmístěnou do několika souborů. Jednotlivé soubory obsahují řádově menší počet bodů, a tedy i několik úrovní detailu vstupních dat. Nejvyšší úroveň pak bude možné zobrazit, a získat tak globální náhled na celé naskenované území. Výsledná hierarchie navíc poskytne možnost načtení bodů z libovolné úrovně, a tedy i získání původních naskenovaných dat vybraného objektu či území.

Hledání identických bodů se neobejde bez vložení polohopisných dat do procesu shlukování. Tyto body se předem prohlásí za permanentní centra shluků na celou dobu shlukování. Výsledek bude obsahovat shluky tvořené polohopisnými body, ke kterým budou přiřazeny body z nejbližšího okolí. V těchto přiřazených bodech se budou snadněji hledat identické body.

Zpracování dat bude probíhat v několika etapách navrhnutých v následujícím schématu Obr. 3.1.



Obr. 3.1: Proces hierarchického shlukování

Soubor polohopisných bodů mapuje celé katastrální území. Získaná data z laserového skenování ale pokrývají jen jeho část. Polohopisné body, které budou zahrnuty do shlukování, musí spadat do daného naskenovaného území.

Polohopisné body spolu s mračnem bodů budou tvořit hlavní vstupní data pro shlukovací algoritmus. Výsledkem algoritmu bude hierarchická struktura bodů, se kterou bude dále pracováno ve vizualizačním programu.

Vložení polohopisných bodů jako permanentních center shluků spolu s daty z laserového skenování do procesu shlukování se neobejde bez modifikace metriky a shlukovacího algoritmu. Tyto modifikace budou popsany v následujících kapitolách.

3.1 Násilné určení center shluků

V naskenovaném území mohou existovat polohopisné body, které nemají ve svém okolí žádné jiné body z naskenovaných dat. Z tohoto důvodu je možné, že algoritmus shlukování nikdy polohopisný bod neprohlásí za centrum shluku nebo všechny jeho body a i samotný polohopisný bod v některé fázi přeřadí k jinému bodu. Proto je nutné polohopisné body odlišit od „klasických“ bodů z laserového skenování.

Před zpracováním jsou polohopisné body prohlášeny za centra shluků a všem je nastaven speciální přiznak *dirty*. Tento příznak algoritmus shlukování musí kontrolovat ve fázi:

- přiřazení bodů k jinému centru shluku - nesmí se povolit přiřazení polohopisného bodu k jinému polohopisnému bodu
- uzavření centra shluku - polohopisný bod vždy zůstane jako centrum shluku, i kdyby k němu nebyl přiřazen žádný jiný bod

3.2 Modifikace metriky

Polohopisné body obsahují pouze standardní souřadnice x, y, z. V případě vložení polohopisných bodů do shlukování k datům z laserového skenování nastává problém s nekonzistencí počtu souřadnic mezi body, protože body z laserového skenování obsahují navíc hodnotu intenzity. Proto je reprezentace polohopisných bodů rozšířena o implicitní hodnotu intenzity rovná nule.

Další odlišnost polohopisných bodů je jejich větší rozdíl v z-souřadnici od bodů z laserového skenování. Tento rozdíl se v konkrétních naskenovaných datech pohybuje v intervalu $< 0.3, 1.5 >$. Oproti tomu vzájemný rozdíl mezi body z laserového skenování se pohybuje v intervalu $< 0, 0.3 >$.

V kombinaci s nulovou intenzitou a větším rozdílem v z-souřadnici se přiřazení některého bodu z laserového skenování k centru v podobě polohopisného bodu vyhodnotí jako velmi nevýhodné. Je nutné zvýhodnit přiřazení bodů k polohopisným bodům.

Větší rozdíl v z-souřadnici polohopisných bodů lze odstranit normalizací souřadnic. Všechny souřadnice se převedou na stejný interval $< 0, 1 >$ a s těmito hodnotami se bude dál počítat namísto původních souřadnic. Tyto normalizované hodnoty by se musely ukládat spolu s původními daty nebo je před každým výpočtem znovu vypočítat. Pro snažší budoucí implementaci a úsporu paměti byla navrhnuta následující metrika.

Metrika popisuje váženou vzdálenost d bodů $a = (a_x, a_y, a_z, a_i)$ a $b = (b_x, b_y, b_z, b_i)$ jako:

$$d(a,b) = \begin{cases} \sqrt{(X * w_x)^2 + (Y * w_y)^2 + (Z * w_z)^2 + (I * w_i)^2} & \text{pro } a \vee b \neq dirty \\ \sqrt{(X * w_x)^2 + (Y * w_y)^2} & \text{pro } a \vee b = dirty \end{cases} \tag{3.1}$$

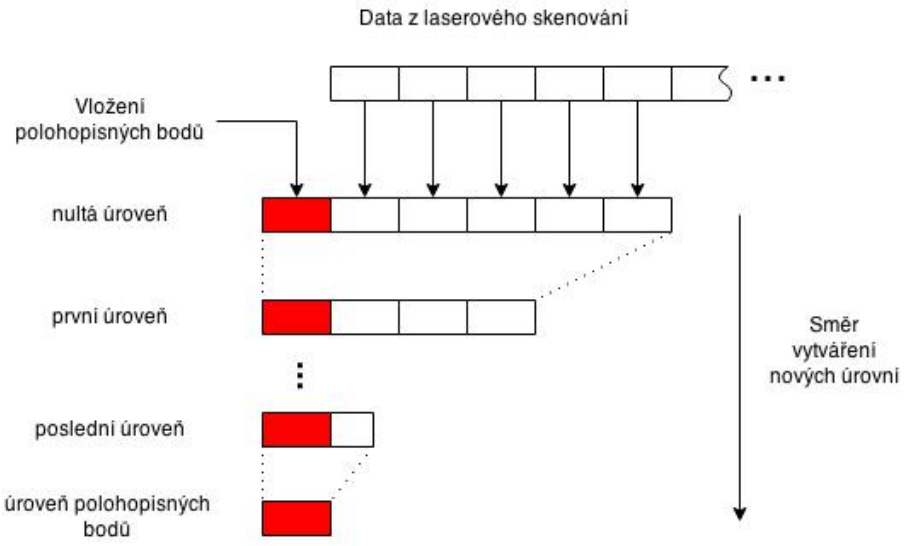
kde:

a_x, a_y, a_z, a_i	...	souřadnice bodu a
b_x, b_y, b_z, b_i	...	souřadnice bodu b
$X = X(a,b) = a_x - b_x$	...	rozdíl x-souřadnic
$Y = Y(a,b) = a_y - b_y$	...	rozdíl y-souřadnic
$Z = Z(a,b) = a_z - b_z$	...	rozdíl z-souřadnic
$I = I(a,b) = a_i - b_i$	...	rozdíl v intenzitě bodů
$w_x \in \mathbb{R}^+$	...	váha x-souřadnice
$w_y \in \mathbb{R}^+$	...	váha y-souřadnice
$w_z \in \mathbb{R}^+$	...	váha z-souřadnice
$w_i \in \mathbb{R}^+$	...	váha intenzity bodu
$a \vee b = dirty, a \vee b \neq dirty$	...	a nebo b obsahuje, resp. neobsahuje příznak *dirty*

V případě, že ani jeden z bodů neobsahuje příznak *dirty*, jsou při výpočtu zahrnuty všechny souřadnice bodů. V opačném případě je vážená vzdálenost vypočtena jako eukleidovská vzdálenost dvou bodů v 2D prostoru. Jinými slovy, polohopisný bod označený příznakem *dirty* má, z hlediska vážené vzdálenosti, stejnou intenzitu a z-souřadnici jako kterýkoli bod z laserového skenování, se kterým je vyšetřován.

3.3 Hierarchie s polohopisnými body

Polohopisné body jsou vloženy jako první blok, aby k nim bylo možné přiřadit jakýkoli bod ze vstupních dat. Hierarchické shlukování započne až po načtení prvního bloku dat z laserového skenování. Celý proces shlukování a výslednou hierarchii lze znázornit následujícím schématem, viz Obr. 3.2.



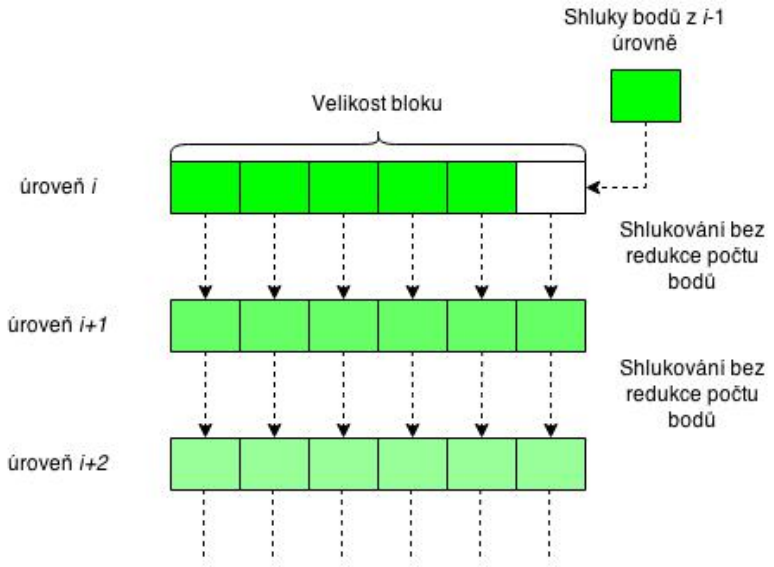
Obr. 3.2: Výsledná hierarchie shlukování s polohopisnými body

Na Obr. 3.2 jsou červenou barvou označeny polohopisné body. Přiřazení bloků klientů z nižších úrovní je naznačeno tečkovanými úsečkami. Polohopisné body zůstanou, díky příznaku *dirty*, po celou dobu shlukování jako centra shluků. Postupným shlukováním úrovní se vždy přesunou do vyšší úrovně, kde získají další přiřazené body. Tento proces pokračuje až do chvíle, kdy se vytvoří poslední úroveň, v které převažují polohopisné body. Dal-

ším uměle způsobeným shlukováním poslední úrovně dojde k vytvoření nové úrovně, která bude obsahovat pouze polohopisné body.

3.4 Modifikace hierarchického shlukování

Algoritmus původního hierarchického shlukování (kapitola 2.3.3) předpokládá snížení počtu bodů (výsledných shluků) oproti vstupním bodům. Zpracování vstupních dat, která obsahují body s velmi rozdílnými souřadnicemi, povede k vytvoření velkého množství osamocených shluků, protože algoritmus shlukování díky velké diverzitě souřadnic vyhodnotí přiřazení bodu k shluku jako velmi nevýhodné. Výsledné shluky poté přecházejí do další úrovně. Problém nastává v případě, kdy tyto výsledné osamocené shluky vyplní celý blok další úrovně i (viz Obr. 3.3).



Obr. 3.3: Opakované vytváření nových úrovní

Algoritmus shlukování úroveň i opět zpracuje, protože počet bodů dosáhl velikosti bloku a je nutné uvolnit místo pro případné další bloky dat. Výsledné shluky se poté přesunou do další úrovně $i+1$. Výsledek shlukování i-té úrovně ale obsahuje stejný počet bodů jako velikost bloku. V další úrovni $i+2$ se proto tento proces opakuje a vzniká tak opakované vytváření nových úrovní a tedy i souborů na disku bez redukce počtu bodů.

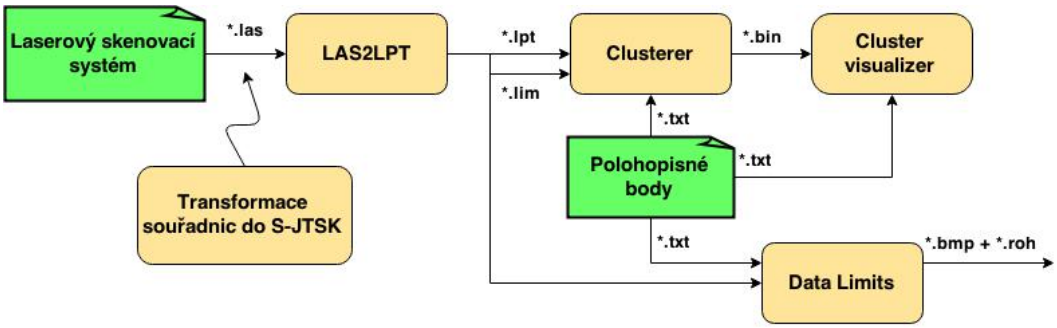
Problém lze zabránit na úrovni $i + 1$. Pokud bezprostředně po vytvoření nové úrovně $i + 1$ je nutné vytvořit novou $i + 2$ úroveň a výsledek shlukování $i + 1$ úrovně je opět rovný velikosti bloku, pak $i + 1$ a každé další $i + 2, i + 3$, ... $i + N$ úrovni je zvětšen velikost bloku na dvojnásobek.

Případ, kdy se takto zvětšené velikosti bloků nevejdou do paměti, nelze nijak jednoduše ošetřit, v zadaných datech nenastává, a proto není brán v úvahu.

4 Implementace řešení

Začátek práce spočíval v získání reálných dat z digitalizace katastrálních map a informací o procesu jejich lokalizace. K tomu posloužila i osobní konzultace s firmou GIS-Stavinvex, a.s., která úzce spolupracuje např. se Seznam.cz v provádění mobilního mapování pro Mapy.cz.

V průběhu implementace řešení bylo nutné vytvořit několik jednoúčelových programů. Každý z těchto programů poté vytváří soubor ve speciálním formátu, který je vstupem dalšího programu v pořadí. Na následujícím schématu je znázorněn výsledný proces zpracování dat z laserového skenování (Obr. 4.1).



Obr. 4.1: Schéma zpracování dat z laserového skenování

Vstupní data jsou na Obr. 4.1 vyznačeny zelenou barvou, oranžovou poté jednotlivé programy. Data tvoří mračno bodů uložené do jednotlivých souborů binárního formátu `las` generovanými laserovým skenovacím systémem a body polohopisu ve formátu `txt`. Mračno bodů je získáno v souřadnicovém systému UTM (kapitola 2.4.1), oproti tomu body polohopisu jsou v souřadnicovém systému S-JTSK (kapitola 2.4.2). K transformaci souřadnicových systémů se používají externí programy [pro15] nebo [gda15].

Další etapou je získání bodových dat z formátu `las`. K tomuto účelu vznikl program `LAS2LPT`, který převádí formát `las` do dále používaného formátu `lpt`. Pro získání vzajemného umístění jednotlivých skenů a celkové hranice naskenovaného území generuje program `LAS2LPT` dále soubory formátu `lim` obsahující maximální a minimální hodnoty v jednotlivých souřadnicích.

O vizuální rozmístění jednotlivých skenů pomocí souborů formátu `lim` se stará program `Data limits`. Pomocí programu lze dále zjistit hustotu rozložení polohopisných bodů v jednotlivých skenech. Pro další možné využití v jiných geomatických programech byl implementován export do rastrového souboru `.bmp`. Pro lokalizované umístění rastrového obrázku vzniká navíc popisný soubor `.roh`, který obsahuje souřadnice umístění rohů obrázku.

Souřadnice jednotlivých bodů mračna (`lpt`), hranice naskenovaného území (`lim`) a polohopisné body (`txt`) tvoří vstupy programu `Clusterer`, který obsahuje samotný algoritmus hierarchického shlukování. Program generuje několik binárních souborů `bin` obsahujících výslednou hierarchickou strukturu.

Pro vizuální kontrolu hierachického shlukování a především jeho jednotlivých modifikací byl vytvořen program `Cluster visualizer`. Program pracuje nad všemi výslednými soubory programu `Clusterer`. Hlavním přínosem programu se pak stala funkce výběru bodů přímo v prostoru a zobrazení přiřazených bodů z nižších úrovní. Důraz během implementace programu byl kladen na možnost vykreslení řádově 10^6 bodů.

Původní shlukovací knihovna byla implementována v jazyce `C#`, z tohoto důvodu byla veškerá řešení realizována v tomto jazyce. Pro základní grafické ovládání byla použita standardní knihovna `WinForms` [wf15].

Pro vykreslení bodů byla zvolena knihovna `OpenTK` (*The Open Toolkit*)[otk15]. Knihovna představuje `C#` wrapper poskytující funkce grafické knihovny `OpenGL, OpenCL` a `OpenAL`. Knihovna `OpenGL` představuje dostatečné grafické zázemí pro vykreslení velkého množství bodů. Další výhodou knihovny je přímý přístup k projekční matici (kapitola 2.1.2), toho je využito během výběru bodů v prostoru.

Důraz byl kladen spíše na využití metody shlukování v oblasti GIS než na samotné uživatelské rozhraní. Z tohoto důvodu bylo ovládání a rozmístění ovládacích komponent voleno autorem této práce na základě jeho uživatelských zkušeností s podobnými programy.

4.1 WinForms a grafický kontext

Základním prvkem knihovny `WinForms` je objekt `Form`, který představuje klasické systémové okno s jménem a základními operacemi (minimalizovat, maximalizovat, zavřít). Do objektu `Form` lze umístit další komponenty tvořící uživatelské rozhraní, např. tlačítka (`Button`), menu (`ContextMenuStrip`) a nebo definovat vlastní komponentu (`UserControl`). Události generované uživatelem předává objekt `Form` svým komponentám. Jedná se především o inicializace (`OnLoad`), překreslení (`OnPaint`), změnu velikosti okna (`OnResize`), pohyb myši (`OnMouseMove`) a další. V případě vlastních komponent je nutné implementovat všechny reakce na tyto události.

Knihovna `OpenTK` poskytuje několik způsobů pro vytvoření grafického kontextu od samostatného okna v podobě `GameWindow` až po integraci do `WinForms` v podobě `GLControl`. V této komponentě již lze přistupovat k knihovním metodám samotného `OpenGL`. Jelikož je `GLControl` potomkem `UserControl`, lze vytvořený objekt vložit přímo do hlavního okna objektu `Form`. Poté stačí inicializovat grafický kontext v samotném objektu `GLControl` a implementovat potřebné reakce na události.

4.2 OpenGL a buffery

Pro maximální využití potenciálu grafické karty se používají jednotlivé vnitřní buffery umístěné přímo v paměti grafické karty. Tyto buffery jsou sekvenčně zpracovány během vykreslování výsledné scény. Knihovna `OpenGL` poskytuje následující buffery:

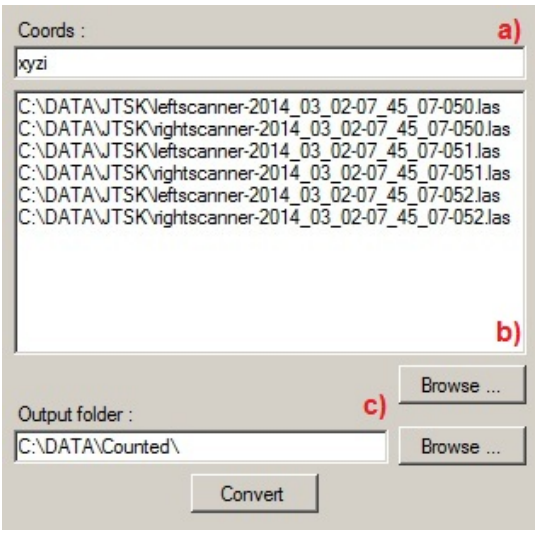
- *vertex buffer* - obsahuje souřadnice jednotlivých bodů, které se budou vykreslovat
- *index buffer* - definuje pořadí bodů, které se mají spojit úsečkou v případě vykreslování primitiv (trojúhélník, čtverec, úsečka)
- *color buffer* - definuje barvu daného bodu
- *texture buffer* - obsahuje souřadnici v textuře, jejichž barva se použije pro obarvení daného bodu

- *normals buffer* - obsahuje předpočítané normálové vektory vykreslovaných primitiv

Do těchto bufferů lze prakticky vložit jakýkoli vytvořený objekt. Objekty jsou v bufferu uloženy za sebou, a proto je nutné bytově definovat velikost jednoho prvku. Logickým požadavkem je ukládání do bufferů pouze těch dat, která budou sloužit k samotnému vykreslení.

4.3 Program LAS2LPT

Pro získání samotných souřadnic bodů z formátu `las` byla použita knihovna `libLAS` [lbs15] implementovaná v jazyce `C++`. Nad jejím zkompilovaným kódem byl vytvořen program `LAS2lpt` s grafickým rozhraním (viz Obr. 4.2) poskytující převedení `las` do plaintextu formátu `lpt`.



Obr. 4.2: Aplikace LAS2lpt

Jednotlivé části rozhraní označené $a), b), c)$ v Obr. 4.2 mají následující význam:

a) Souřadnice které mají být získány ze souboru `las`. V našem případě souřadnice `xyz` a intenzita.
b) Seznam `las` souborů, které se budou převádět.
c) Cílový adresář převedených `lpt` souborů.

Tlačítkem `Convert` se spustí převádění souborů. Výsledné `lpt` soubory nesou stejný název jako zdrojové soubory. Soubory obsahují na každém řádku hodnoty souřadnic jednoho bodu oddělené středníkem. První řádek převedeného `lpt` souboru obsahuje celkový počet bodů v daném souboru. Díky tomu lze alokovat dostatečné množství paměti před samotným zpracováním dat.

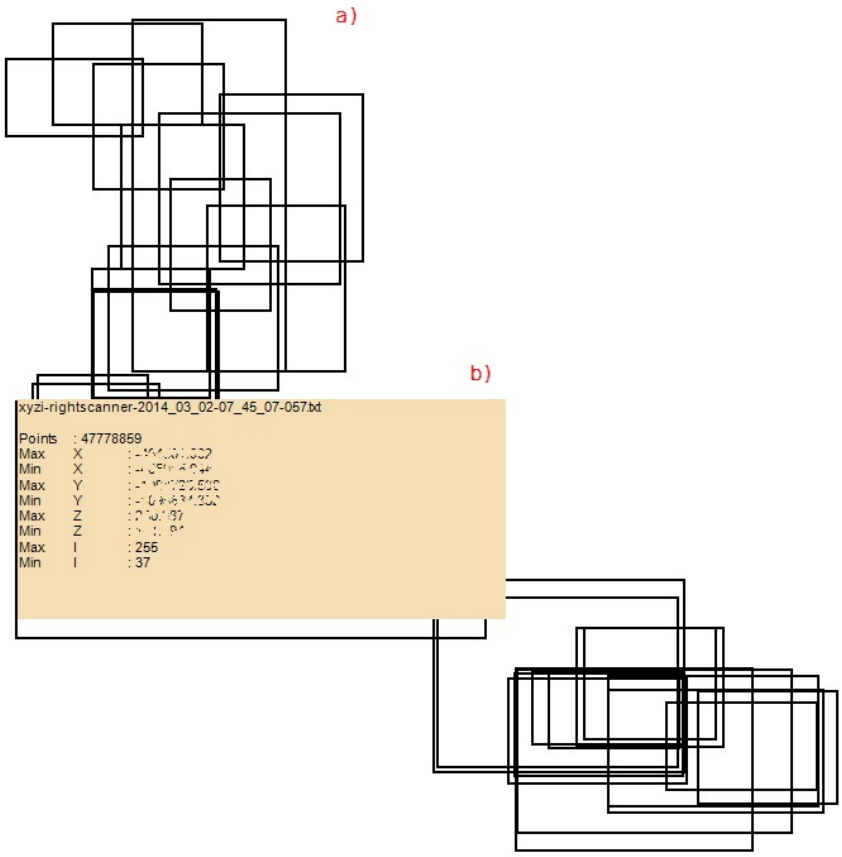
Během převodu jsou vedeny statistiky o minimálních a maximálních hodnotách převáděných bodů. Tyto statistiky mají podobu souborů s koncovkou `.lim` a mají následující formát :

```
Source file : <jméno zdrojového souboru>
Points : <počet bodů>
NOC : <počet souřadnic N >
Max 0 : <maximální hodnota 0 souřadnice>
Min 0: <minimální hodnota 0 souřadnice>
Max 1 : <maximální hodnota 1 souřadnice>
Min 1 : <maximální hodnota 1 souřadnice>
...
Max N : <maximální hodnota N souřadnice>
Min N : <maximální hodnota N souřadnice>
```

4.4 Program Data limits

Statistické soubory `.lim` jsou hlavním vstupem programu `Data limits`. Program zobrazí hranice daného souboru jako obecný čtyřůhelník (viz Obr. 4.3 [1]).

[1] Kvůli anonymizaci dat byly zobrazené souřadnice uměle znehodnoceny.



Obr. 4.3: Přehled rozmístění jednotlivých skanů

Části $a), b)$ v Obr. 4.3 mají tento význam:

a) Obecné čtyřúhelníky představující maximální a minimální hodnoty souřadnic x a y jednotlivých souborů.
b) Najetí myši na některý ze čtyřúhelníků je doprovázen zobrazením informací z statistického souboru.

Program dále umožňuje načíst polohopisné body a určit tak, kolik se jich nachází v jednotlivých skenech. Lze tak snadno nalézt sken s největší hustotou polohopisných bodů.

4.5 Program Clusterer

K hierarchickému shlukování mračen bodů byl vytvořen konzolový program `Clusterer`.

V první fázi program vypočítá globální hranice mračna bodů. Tyto hranice jsou získány z souborů `lim` nebo v případě chybějícího souboru `lim` pro některý sken jsou hranice vypočítány přímo z vstupního souboru `lpt`.

Dále jsou načteny polohopisné body, které spadají do území naskenovaného mračna bodů. Následně se začně po blocích zpracovávat vstupní mračno bodů. Výsledky shlukování jsou ukládány do separovaných souborů (kapitola 4.5.1).

Pomocí parametrů programu lze nastavit následující koeficienty ovlivňující shlukování:

- Cena za vytvoření shluku - jak výhodné resp. nevýhodné bude vytvoření nového shluku. Koeficient je normalizován, tzn. jeho interval se pohybuje v rozmezí $< 0, 1 >$.
- Velikost bloku - počet bodů v bloku. Tato hodnota navíc určuje maximální počet shluků, se kterými je zároveň pracováno v rámci jedné úrovně.
- Váhy jednotlivých souřadnic - koeficienty vah pro jednotlivé souřadnice (kapitola 2.2.2).

Vstup programu pak tvoří následující soubory:

- soubory `lpt` - data z laserového skenování
- soubory `lim` - pro správný vliv vah je nutné získat minimální a maximální hodnoty v jednotlivých souřadnicích
- soubor s polohopisnými body

4.5.1 Uložení výsledků shlukování

Kvůli enormnímu množství dat jsou jednotlivé úrovně shlukování uloženy v podobě binárních souborů. Jména těchto binárních souborů obsahují sufix s číslem představující úroveň v hierachii.

Souřadnice jednotlivých bodů jsou po zpracování kontinuálně zapisovány do souboru své příslušné úrovně. Nejnižší úroveň (nultá) obsahuje pouze souřadnice těchto bodů. Vyšší úrovně jsou obohaceny informacemi o adresu začátku klientů na nižší úrovni, jejich počet a příznak *dirty*. Binární soubor vyšší úrovně má pak následující formát:

$$\texttt{S}\texttt{X}_1\texttt{Y}_1\texttt{Z}_1\texttt{I}_1\texttt{S}_1\texttt{L}_1\texttt{D}_1\texttt{X}_2\texttt{Y}_2\texttt{Z}_2\texttt{I}_2\texttt{S}_2\texttt{L}_2\texttt{D}_2 \ldots \texttt{X}_n\texttt{Y}_n\texttt{Z}_n\texttt{I}_n\texttt{S}_n\texttt{L}_n\texttt{D}_n$$

kde :

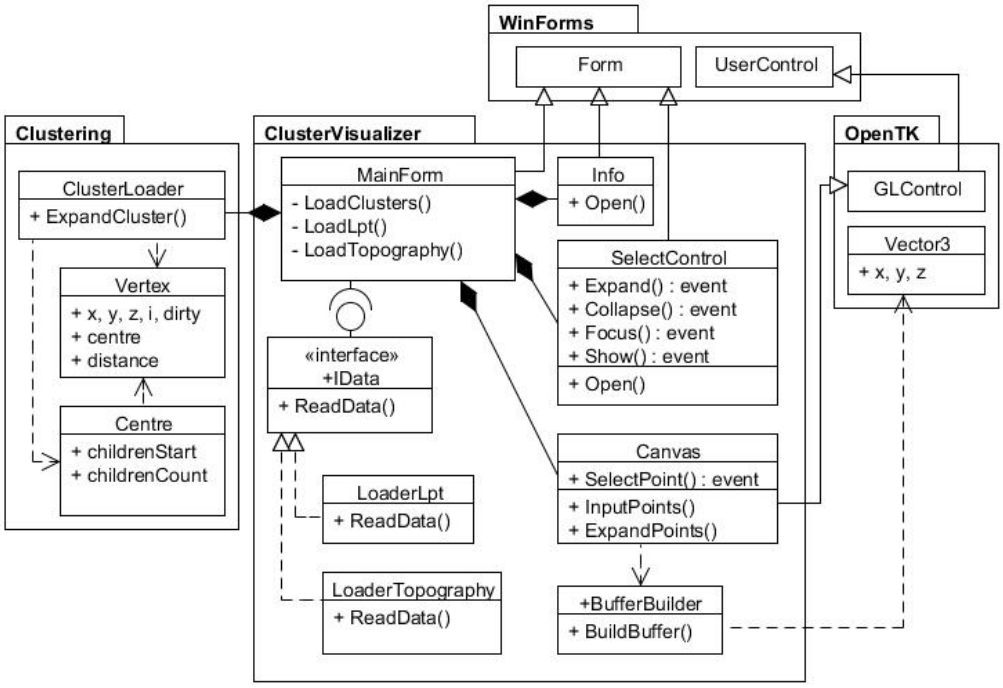
$\texttt{S}$	...	počet souřadnic. Celé číslo bez znaménka (`16bit`).
$\texttt{X}_i\texttt{Y}_i\texttt{Z}_i$	...	souřadnice bodu. Číslo s plovoucí desetinnou čárkou (`64bit`).
$\texttt{I}_i$	...	intenzita daného bodu. Číslo s plovoucí desetinnou čárkou (`64bit`).
$\texttt{S}_i$	...	adresa do souboru nižší úrovně v hierarchii, kde začínají příslušní klienti. Celé číslo se bez znaménka (`64bit`).
$\texttt{L}_i$	...	celkový počet klientů přiřazených k danému centru. Celé číslo se bez znaménka (`32bit`).
$\texttt{D}_i$	...	příznak *dirty* zdali se jedná o polohopisný bod (`1bit`).

Při zpracování výsledných souborů se načte prvních `16 bitů` pro získání počtu souřadnic bodů. Na základě úrovně, získané ze sufixu jména souboru, se vypočte binární velikost jednoho záznamu bodu. Celkový počet bodů v daném souboru se pak vypočte vydělením velikosti souboru velikostí jednoho záznamu.

V budoucí práci se předpokládá nasazení kompresních algoritmů na binární soubory.

4.6 Cluster visualizer

Celý návrh zahrnující integraci potřebných knihoven je znázorněn na následujícím diagramu Obr. 4.4. Pro úsporu místa a lepší názornost jsou zobrazeny pouze nejdůležitější proměnné a metody označené v podobě značek `()`.



Obr. 4.4: Schéma návrhu programu `Cluster visualizer`

Následuje detailnější popis jednotlivých částí.

- ClusterVisualizer
  - `MainForm` - hlavní okno programu. Slouží pouze jako zprostředkovatel mezi ostatními komponentami. Tento `Form` obsahuje další okna v podobě `SelectControl` (viz Obr. 4.10) a `Info`. Tyto okna se zobrazí ve chvíli výběru bodů (kapitola 4.6.3).
  - `Canvas` - grafická komponenta obsahující `OpenGL` kontext. `Canvas` vyvolává jedinou událost `SelectPoint` z které posluchač získá indexi uživatelem vybraných bodů.

  - `BufferBuilder` - vytváří `vertex` a `color` buffer ze vstupních dat. Jelikož jsou vstupní data v souřadnicovém systému `S-JTSK` (kapitola 2.4.2) a `OpenGL` používá pravotočivý souřadnicový systém (viz Obr. 2.2 v kapitole 2.1.2), jsou data před vložením do bufferu ještě transformována tak, aby byly zachovány původní světové strany.
  - `IData` - rozhraní pro načítání budoucích různorodých formátů. Stačí pak implementovat pouze parser konkrétního formátu.

- Clustering
  - `Vertex` - reprezentace bodu. Kromě klasických souřadnic obsahuje odkaz na své centrum shluku a jaká k němu byla vypočítána vážená vzdálenost.
  - `Centre` - základní reprezentace centra shluku. Obsahuje adresu v souboru, kde začínají informace o přiřazených bodech, a jejich počet.
  - `ClusterLoader` - z binárních výsledků hierarchického shlukování vytváří objekty `Centre` a `Vertex`. Třída byla upravena z původní shlukovací knihovny.

- OpenTK
  - `Vector3` - základní reprezentace vektoru v `OpenGL`. Obsahuje pouze souřadnice a základní operace s vektory.
  - `GLControl` - komponenta s přístupem k knihovním funkcím `OpenGL`.

4.6.1 Hlavní funkce programu

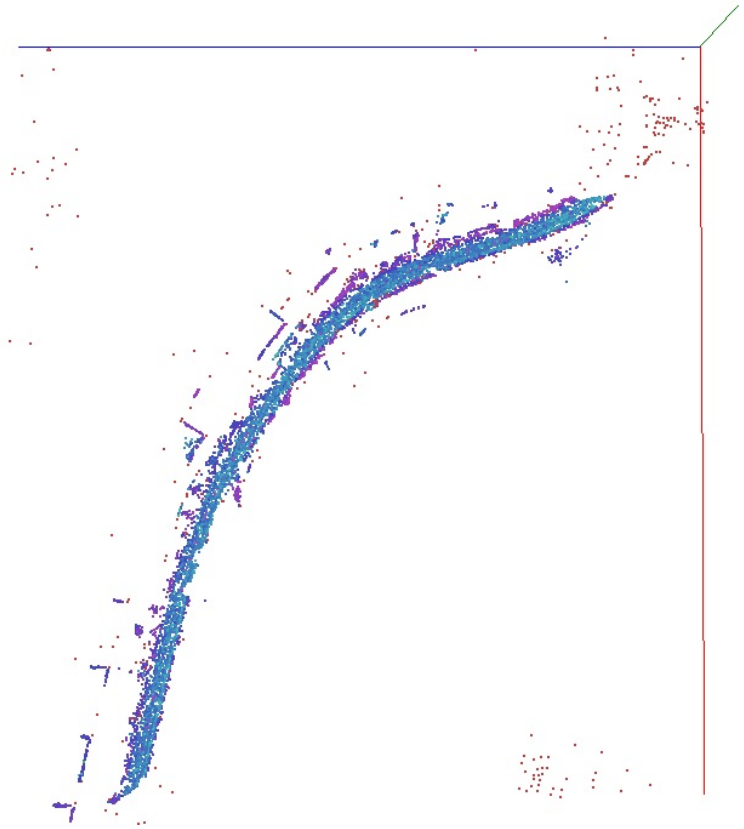
Následuje popis hlavních funkcí programu (viz Obr. 4.5).

Obr. 4.5: Hlavní menu programu `Cluster visualizer`

Části $a), b), c), d)$ v Obr. 4.5 mají tento význam:

a) výběr vstupních souborů
b) výběr, jaká úroveň hierarchie se má zobrazit
c) mód pro výpis souřadnic vybraného bodu
d) mód pro výběr center shluků a jejich následnou expanzi

Hlavním vstupem programu `Cluster visualizer` jsou výsledné binární soubory programu `Clusterer`. Program dále poskytuje náhled na soubory formátu `.lpt` (pouze jejich malou část) a dále pak polohopisné body. Po zvolení daného souboru jsou načteny všechny body a kamera je přesunuta tak, aby poskytovala globální náhled na tato data (viz Obr. 4.6).



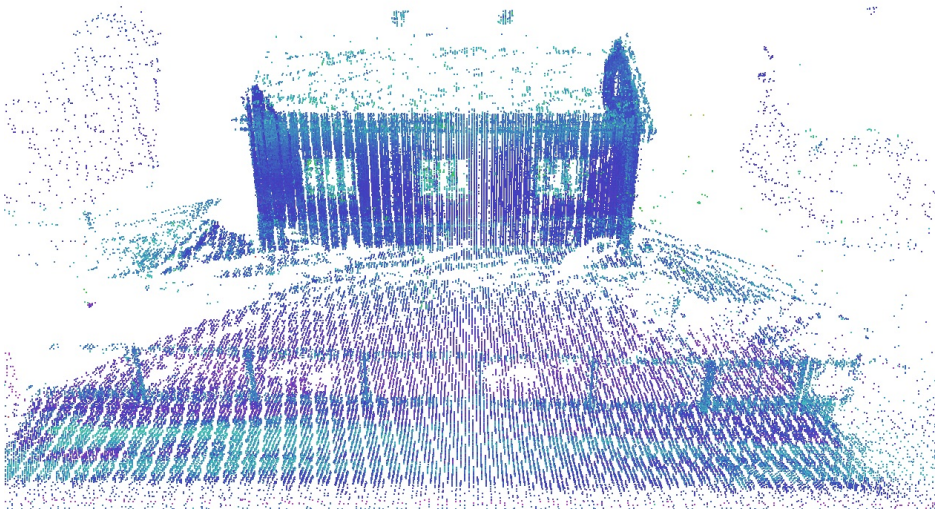
Obr. 4.6: Globální náhled na binární data

Modrou, červenou a zelenou úsečkou jsou zobrazeny hlavní geomatematické osy `XYZ`. Obarvení bodů je odvozeno na základě jejich intenzity. Pro zvýraznění byl zvolen přechod od červené k fialové.

4.6.2 Přesnost vykreslení

Data mají 6 míst před desetinou čárkou a tři místa za desetinou čárkou. Pro uložení souřadnic bodů je použit datový typ `double` s plovoucí desetinnou čárkou. Pohyb kamery je doprovázen přepočítáním souřadnic všech viditelných bodů do souřadnic obrazovky (viz kapitola 2.1.2).

Při pohybu kamery tak vzniká „skákání“ bodů při nepřesném zaokrouhlování. Další nežádoucí efekt byl patrný především v detailním náhledu na data, kdy vznikaly zřetelně viditelné skoky bodů, jak je vidět na detailu rodiného domu s plotem (viz Obr. 4.7).



Obr. 4.7: Přesnost vykreslení

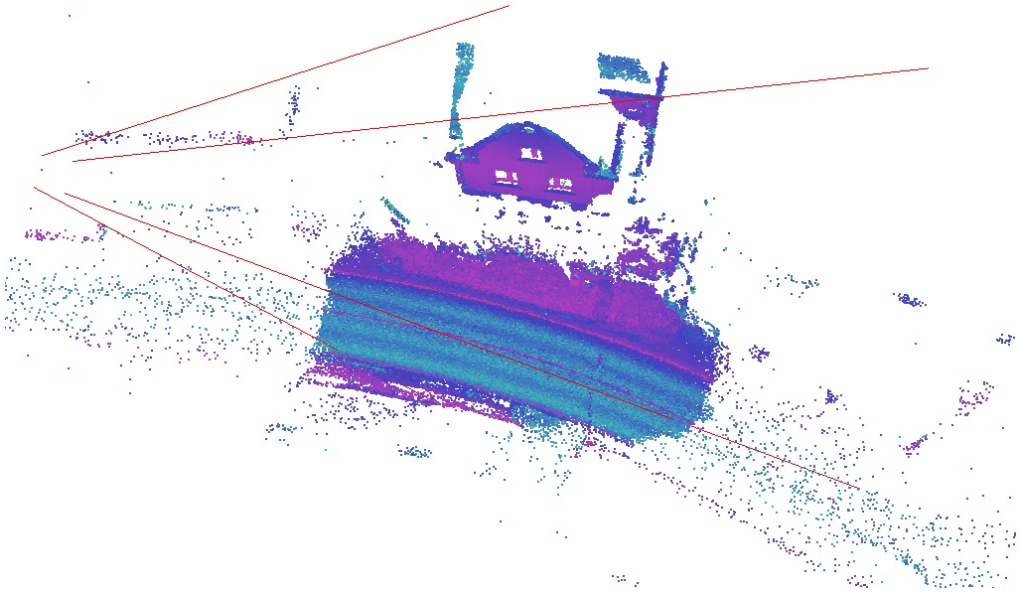
Nepřesnost vykreslení bodů se projevovala při práci v původním rozsahu dat, kde minima a maxima x a y byla velká kladná čísla. Posunutím dat tak, aby souřadnice začínaly v nule se zvýšila přesnost vykreslení.

4.6.3 Výběr bodů v 3D prostoru

Po zvolení příslušného módu probíhá pomocí pohybu myši definování výběrového obdélníku. Body, které spadají do výběru, jsou obarvovany červenou barvou (viz Obr 4.8).

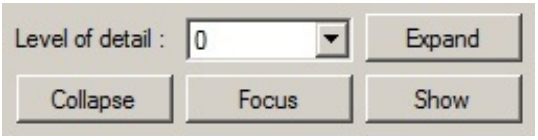
Po výběru bodů, které představují centra shluků dané úrovně, jsou zobrazeny všechny jejich přiřazené body, jak je vidět na Obr. 4.9. Pro ilustraci byly vykresleny i hrany výběrového objemu v podobě červených úseček končících v nekonečnu.

Obr. 4.8: Výběr bodů v prostoru



Obr. 4.9: Expandované vybrané shluky s výběrovým hranolem

Úspěšný výběr bodů je doprovázen zobrazením dalších možností (viz Obr. 4.10).



Obr. 4.10: Možnosti po výběru bodů (`SelectControl`)

Funkce tlačítek v Obr. 4.10 jsou následující:

`Expand`	-	expandování vybraných bodů do zvolené úrovně
`Collapse`	-	zobrazení původně vybraných bodů
`Focus`	-	zobrazení pouze vybraných bodů
`Show`	-	přesunutí a natočení kamery na vybrané body

5 Experimenty a výsledky

První experimenty na malých testovacích datech byly prováděny na sestavě s procesorem Intel® Pentium® Dual CPU T3200 (2,00 GHz, 2 jádra) a 3 GB operační pamětí. V průběhu práce byl zřízen virtuální server, poskytnutý ZČU/KIV, o sestavě Intel® Xeon® CPU E5-4620 v2 (2.60GHz, 2 vlákna) a 6GB operační pamětí.

První část experimentů spočívala v postupném hierarchickém shlukování všech párů poskytnutých skenů. Tedy vždy spolu levý a pravý sken z daného území. Další experimenty probíhaly na všech poskytnutých datech o velikosti 65.8GB.

Druhá část experimentů probíhala v podobě zpracovávala polohopisná data spolu s daty z laserového skenování a ověřovala správnost implementace modifikací shlukovacího algoritmu.

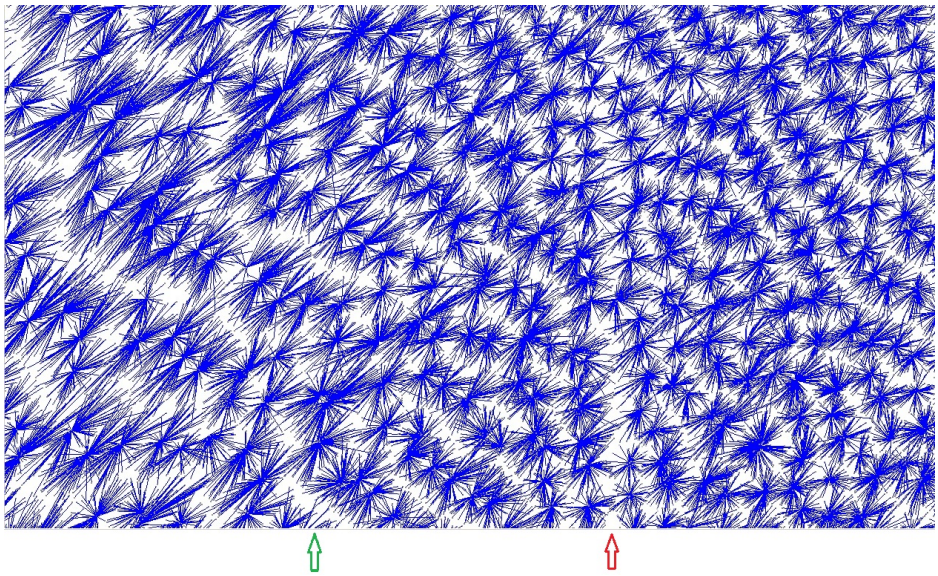
Hlavními sledovanými parametry shlukování byla rychlost shlukování a vypovídací schopnost jednotlivých hiearchickcých úrovní.

5.1 Orientace shluků

S prvními výsledky se objevil nežádoucí efekt v podobě orientace shluků. Orientace shluků byla způsobena řádkovým terénním skenováním (kapitola 2.4.5) a blokovým načítáním hierarchického shlukování.

Z těchto bloků vznikají shluky, které zpravidla nemají optimálně přiřazené body ze svého okolí. Hierarchický algoritmus ve chvíli přiřazení bodů k centru shluku nemá informaci o dalším bloku dat, v kterém zpravidla existuje lépe vyhovující bod. Pokud se tento efekt projeví v místech nějakého objektu, výsledek je shlukově „nařezaný“ objekt.

Na následujícím snímku Obr. 5.1 je vidět detail vozovky směřující na sever při velikosti bloku 10^4 bodů.



Obr. 5.1: Problém se skenově orientovanými shluky (10^4 bodů v bloku)

Pravá část obrázku je samotná vozovka s přerušovanou čárou (červená šipka), která do levé části postupně přechází na travnatou plochu (zelená šipka). Modrou úsečkou je zobrazeno fiktivní spojení přiřazeného bodu k jeho centru shluku.

V kombinaci s větší hustotou zaznamenaných bodů a podobnou intenzitou vznikly v části vozovky menší shluky oproti různorodější travnaté levé části. Dále si můžeme všimnout patrné úhlopříčné orientace shluků ve směru pořizování skenu pod úhlem 45°.

Zvětšením bloku se rozšíří pouze zpracovávaný pruh dat. Algoritmus shlukování pak v tomto pruhu vytvoří shluky, které již nebudou orientovány. Směr pořizování skenu však zůstane zachován v podobě samotných pruhů. Velikost zpracovávaného bloku přímo ovlivňuje celkový čas shlukování (viz tabulka 5.1). Uvedené časy představují celkovou dobu shlukování při zpracování testovacího vzorku o velikosti `3 720 422` bodů.

Jak je vidět na tabulce 5.1, další zvětšení velikosti bloků by se z hlediska časové náročnosti stalo neúnosné. Navíc by se problém rozšířil pouze na „větší pruh“ shluků při zachování orientace.

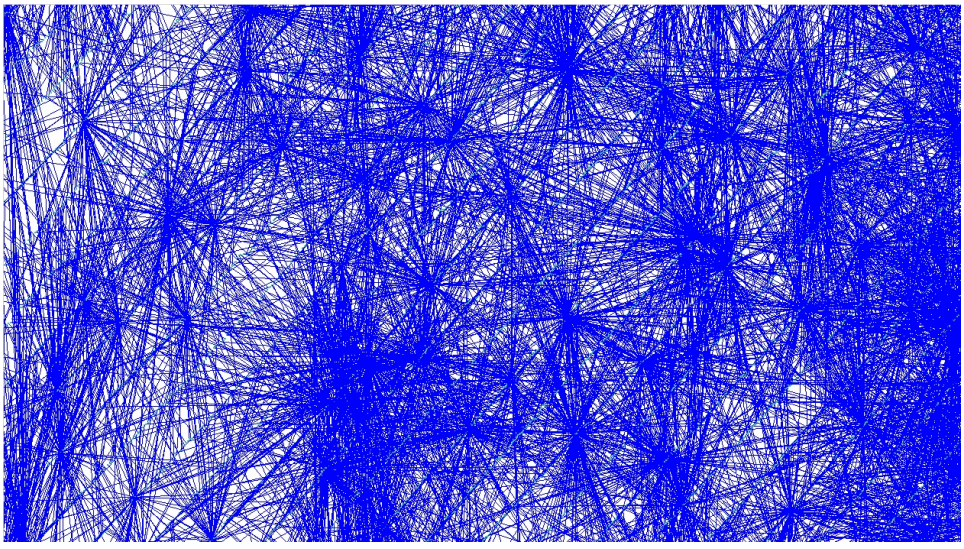
Velikost bloku [počet bodů]	Čas [hh:mm:ss:ms]
1000	00:00:32.546
10 000	00:02:09.265
40 000	00:05:34.906
200 000	00:22:40.750
500 000	00:55:37.282

Tabulka 5.1: Velikost bloku a časová náročnost

5.1.1 Odstranění orientace shluků

Pomocí psedonáhodné změny pořadí jednotlivých bodů v původních datech lze odstranit orientaci shluků. Změna pořadí bodů se provádí po blocích, které budou několikrát větší než velikost budoucího bloku zpracovávaného algoritmem hierarchického shlukování. Algoritmus pak vytváří výhodnější shluky, protože v zpracovávaném bloku budou obsaženy body z několika dalších řádků ve směru jejich pořizování. S změnou pořadí bodů je pak možné použít menší zpracovávaný blok s podobným výsledkem jako u shlukování bez změny pořadí bodů.

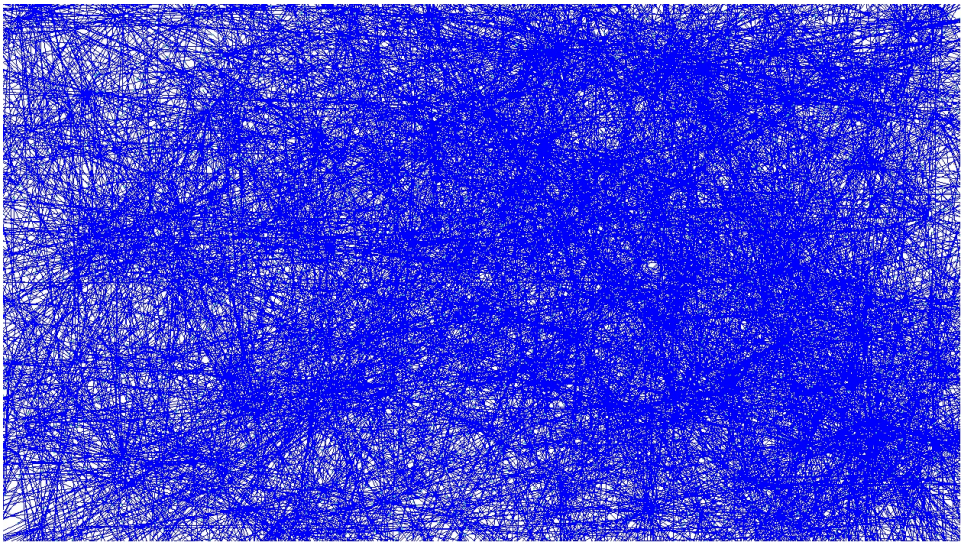
Následuje Obr. 5.2 s výsledkem při zvětšení velikosti bloku, bez změny pořadí bodů s velikostí zpracovávaného bloku 2.10^5 bodů.



Obr. 5.2: Detail vozovky při 2.10^5 bodů v bloku

Na Obr. 5.2 je detail vozovky. Výsledné shluky jsou větší, protože se v jejich okolí nacházely body, které bylo výhodné připojit k shluku. Stále si však můžeme všimnout nepatrné 45° orientace zejména v pravé části obrázku.

Podobného výsledku lze dosáhnout použitím náhodné změny na stejné velikosti bloku jako v předešlém případě (2.10^5 bodů), ale s daleko menší velikostí zpracovávaného bloku 10^4 bodů (viz Obr. 5.3 s stejným detailem vozovky).



Obr. 5.3: Detail vozovky s změnou pořadí bodů

Na Obr. 5.3 si můžeme všimnout úplné ztráty orientace shluků. Díky náhodné změně bodů v vstupním proudu dat, a s tím spojené možnosti přiřadit body i z vzdálenějších řádků skenu, vznikly výhodnější, navzájem se překrývající shluky. Hlavním rozdílem výsledků z posledních dvou obrázků Obr. 5.2 a Obr. 5.3 je čas výpočtu.

Čas předzpracování v podobě náhodné změny pořadí bodů a následného shlukování těchto dat při menší velikosti bloku je vyjádřen v následující tabulce 5.2. V tabulce jsou zaneseny i informace o shlukování, které poskytne podobný výsledek, ale bez změny pořadí bodů (viz Obr. 5.2 a Obr. 5.3).

Činnost	**Velikost bloku**	**Čas** `[hh:mm:ss:ms]`
Náhodná změna pořadí bodů	2.10^5 bodů	`00:00:39:034`
Shlukování s změnou pořadí bodů	10^4 bodů	`00:02:18.381`
Shlukování bez změny pořadí bodů	2.10^5 bodů	`00:22:40.750`

Tabulka 5.2: Časová náročnost

Celkový čas změny pořadí bodů a shlukování (o velikosti bloku 10^4) probíhal necelé tři minuty, což představuje šestinásobnou úsporu času oproti shlukování při velikosti bloku 2.10^5 bodů s podobnými výsledky.

5.2 Výsledky hierarchického shlukování

Algoritmem hierarchického shlukování bylo zpracováno celkových `65,8GB` dat ve formátu `lpt` rozdělených do 42 souborů. Data jednotlivých skenů se rozprostírají na `1522,5 m x 2075 m` fyzicky naskenovaného území v České republice. Hlavními sledovanými parametry byl celkový čas zpracování, počet hierarchických úrovní, počet bodů v jednotlivých úrovní a v neposlední řadě vypovídací schopnost jednotlivých úrovní. Následuje Obr. 5.4 s největším počtem bodů, které již lze zobrazit. Ostatní výsledné úrovně lze nalézt v příloze A.1.

Celkový čas výpočtu hierarchického shlukování trval `5:53:09.157`. Hierarchické shlukování vytvořilo 5 úrovní popsaných v následující tabulce 5.3.

Výsledná úroveň	Velikost souboru [kB]	Počet bodů
`0. úroveň`	61 572 881	1 916 040 187
`1. úroveň`	1 118 522	26 031 054
`2. úroveň`	15 841	368 647
`3. úroveň`	290	6746
`4. úroveň`	5	110

Tabulka 5.3: Velikost úrovní hierarchického shlukování

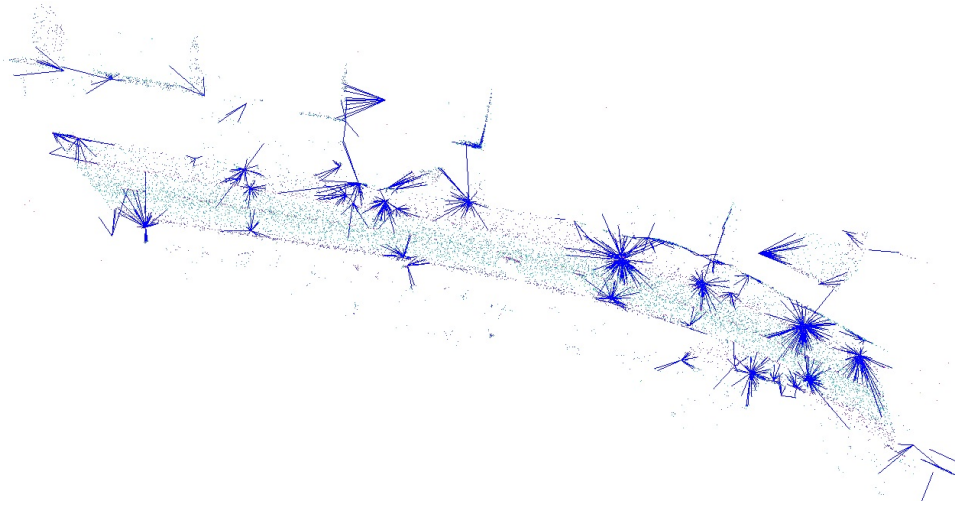
Obr. 5.4: 2. úroveň výsledné hierarchie

Body druhé výsledné úrovně (Obr. 5.4) kopírují silniční komunikaci i okolní zástavbu, lze si tak utvořit představu o naskenovaném území.

5.3 Hierarchické shlukování s polohopisnými body

Ověření správného zpracování polohopisných bodů spolu s daty z laserového skenování probíhalo na testovacím vzorku o velikosti `50MB`. Polohopisný soubor obsahuje `173 311` bodů, díky předpočítaným limitám jich bylo k shlukování přijato `73`.

Na Obr. 5.5 je zobrazena testovaná část silniční komunikace. V obrázku jsou zobrazeny pouze výsledné shluky polohopisných bodů.



Obr. 5.5: Shluky s centrem v polohopisném bodě

Modrou úsečkou je zobrazeno fiktivní spojení bodu k jeho centru shluku v podobě červeně zvýrazněného polohopisného bodu. Modifikace metriky (kapitola 3.2), zvýhodňující polohopisné body z hlediska vážené vzdálenosti, umožnila přiřazení i vzdálenějších bodů k polohopisným bodům.

V následující tabulce 5.4 jsou popsány výsledné soubory hierarchického shlukování na základě výpisu programu `Clusterer`.

Výsledná úroveň	Počet bodů
0. úroveň	2 343 388
1. úroveň	62 520
2. úroveň	1977
3. úroveň	73

Tabulka 5.4: Počet bodů v jednotlivých úrovní

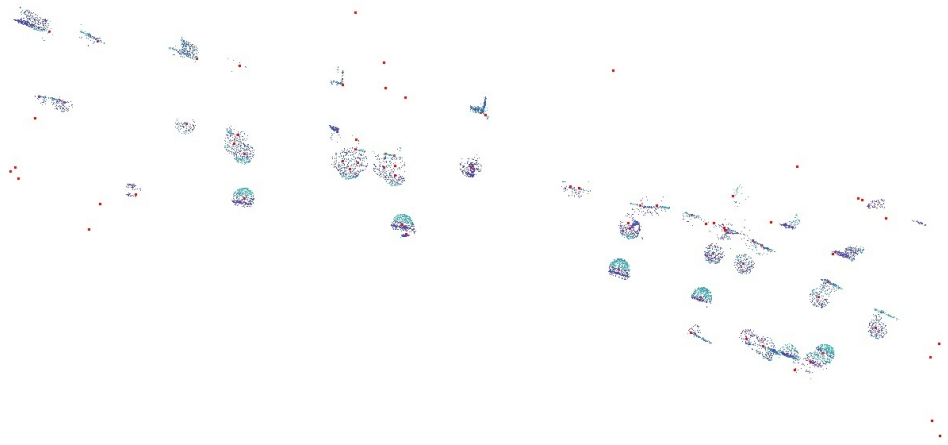
Díky příznaku *dirty* a shlukováním poslední výsledné úrovně (kapitola 3.3) vytvořily vložené polohopisné body další úroveň, viz počet bodů 3. úrovně v tabulce 5.4.

Opakovaným expandováním všech přiřazených bodů k polohopisným shlukům až na úroveň původních dat z laserového skenování získáme jednotlivá

území bodového mračna, v kterých se nachází samotný polohopisný objekt. Zmenšením expandovaného území se usnadní hledání identických bodů, protože se bude vyšetřovat menší počet bodů.

Vytvoření menších území lze částečně změnšením ceny za vytvoření shluku (koeficient fc kapitola 2.3.2), který ovlivňuje velikost shluků. Nelze však předem určit hodnotu této ceny v závislosti na konkrétních datech.

Zmenšení území se proto provede kontrolou vzdálenosti přiřazeného bodu od jeho centra. Tato vzdálenost je ovlivněna konkrétní nepřesností GNSS při určení polohy vozidla a představuje maximální posun mračna bodů od odpovídajících polohopisných bodů.



Obr. 5.6: Množiny s identickými body

Na Obr. 5.6 vidíme expandované území. Všechny body se nacházejí maximálně `1,5m` od svého polohopisného bodu. V tomto území se s vysokou pravděpodobností nacházejí i identické body.

Experimenty prokázaly možnost využití hierarchického shlukování při manipulaci s celým mračnem bodů. Hierarchické shlukování vytváří několik úrovní detailu mračna bodů s možností lokálního detailu až na úroveň původních dat získaných z laserového skenování.

Navržená modifikace metriky pro vložení polohopisných bodů do procesu shlukování spolu s mračnem bodů a následné omezení expandového území zjednodušuje nalezení identických bodů.

6 Závěr

Úkolem práce byla modifikace metody hierarchického shlukování pro manipulaci s celým mračnem bodů získaným z laserového skenování. V průběhu práce bylo vytvořeno několik programů, které zjednodušují práci se zdrojovými daty. Dále byl vytvořen program pro hierarchické shlukování obsahující všechny potřebné modifikace. Poslední program vznikl pro vizualizaci výsledků z hierarchického shlukování.

Dosavadní experimenty prokázaly možné použití hierarchického shlukování pro danou aplikaci. Případné využití vede ke dvěma hlavním přínosům. Prvním přínosem je možnost vytvoření globálního náhledu na celé mračno bodů s možností selektivního zvětšení detailu v vybraném území. Druhý přínos spočívá v předpřipravení území v mračnu bodů, které bude obsahovat identický bod, a tedy zjednodušení procesu lokalizace. Celý proces lokalizace se však stále neobejde bez interakce s uživatelem.

Literatura

[sj13] Skála, Jiří. *Algoritmy pro manipulaci s velkými geometrickými daty*, disertační práce. Západočeská univerzita v Plzni, fakulta aplikovaných věd, Plzeň, 2012. 111 l., str. 28-43, vedoucí práce Ivana Kolingerová.

[cg13] Skála, Jiří - Kolingerová, Ivana. *Dynamic hiearchichal triangulation of a clustered data stream.* Computers & Geosciences, Elsevier, 37(8): 1092-1101, 2011.

[zj04] ŽÁRA, Jiří. *Moderní počítačová grafika. 2.* , přeprac. a rozš. vyd. Praha: Computer Press, 2004, 609 s., 16 s. barev. obr. příl. ISBN 8025104540.

[sk09] Skála, Jiří - Kolingerová, Ivana. *Data Stream Hierarchical Clustering Library* [software]. 2009. Dostupné z: http://www.kiv.zcu.cz/vyzkum/software/2009/hier-clust-datastream.html

[jt13] Tomeček, Jan. *Texty k prednáškám z MMAN3* [online]. [cit. 2013-04-30]. Dostupné z: http://aix-slx.upol.cz/∼tomecek/vyuka/ma1i/metr_prostory.pdf

[igtf15] IGTF. *The Imaging & Geospatial Technology Forum.* [online]. [cit. 2015-04-11]. Dostupné z: http://www.asprs.org/Committee-General/LASer-LAS-File-Format-Exchange-Activities.html

[gns15] GNSS. *Global Navigation Satellite System* [online]. [cit. 2015-04-11]. Dostupné z: http://www.gsa.europa.eu/

[lid15] LIDAR. *Light Detection and Ranging* [online]. [cit. 2015-5-10]. Dostupné z: https://lta.cr.usgs.gov/LIDAR

[lbs15] LibLAS library. *LAS 1.0/1.1/1.2 ASPRS LiDAR data translation toolset* [online]. [cit. 2015-04-11]. Dostupné z: http://www.liblas.org/

[zpg15] Analytická geometrie pro počítačovou grafiku II. *Homogenní souřadnice* [online]. [cit. 2015-05-10]. Dostupné z: http://herakles.zcu.cz/education/zpg/cviceni.php?no=5

[per15] Perspektivní a paralelní projekce. *Perspective and Parallel Projection* [online]. [cit. 2015-5-10]. Dostupné z: http://www.csee.umbc.edu/∼rheingan/435/pages/res/gen-8.Viewing-single-page-0.html

[otk15] OpenTK. *The Open Toolkit* [online]. [cit. 2015-4-11]. Dostupné z: http://www.opentk.com/

[wf15] Microsoft .NET Framework. *Windows Forms* [online]. [cit. 2015-4-11]. Dostupné z: http://msdn.microsoft.com/en-us/library/dd30h2yb.aspx

[kro15] Zobrazení užitá pro ČSR a ČR. *Křovákovo zobrazení* [online]. [cit. 2015-5-10]. Dostupné z: http://gis.zcu.cz/studium/mk2/multimedialni_texty/index_soubory/hlavni_soubory/cechy.html

[gen15] Souřadnicové systémy. *Tvar zemského tělesa a referenční plochy* [online]. [cit. 2015-5-10]. Dostupné z: http://gis.zcu.cz/studium/gen1/html/ch02s03.html

[utm15] UTM. *Grid Zones of the World* [obrázek]. [cit. 2015-5-10]. Dostupné z: http://www.dmap.co.uk/utmworld.htm

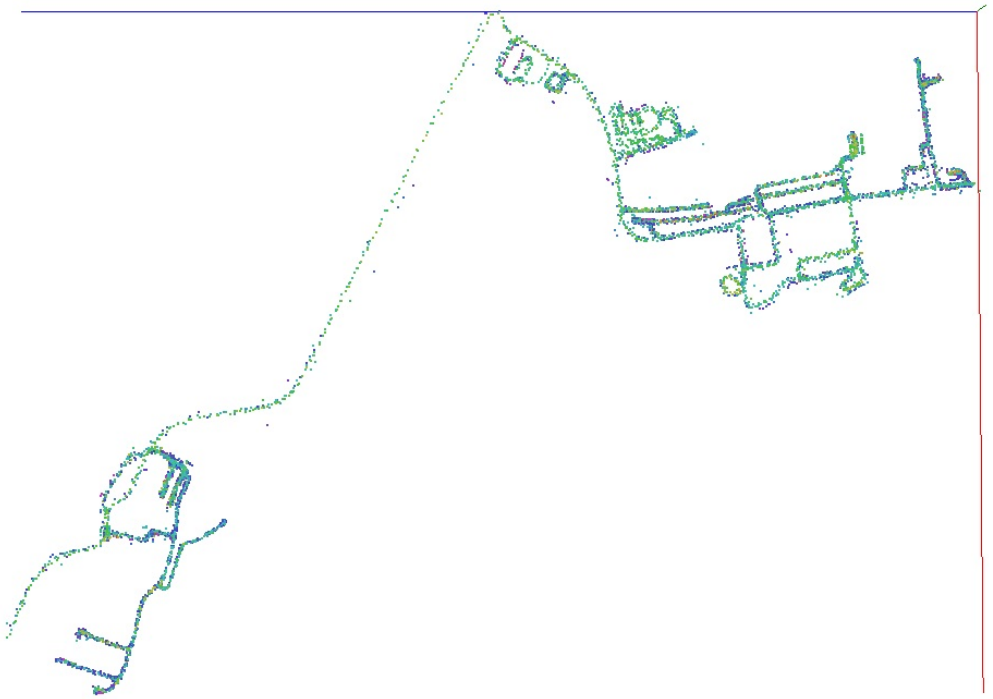
[pro15] PROJ.4. *Cartographic Projections Library* [online]. [cit. 2015-4-11]. Dostupné z: https://trac.osgeo.org/proj/

[gda15] GDAL. *Geospatial Data Abstraction Library* [online]. [cit. 2015-4-11]. Dostupné z: http://www.gdal.org/

A Přílohy

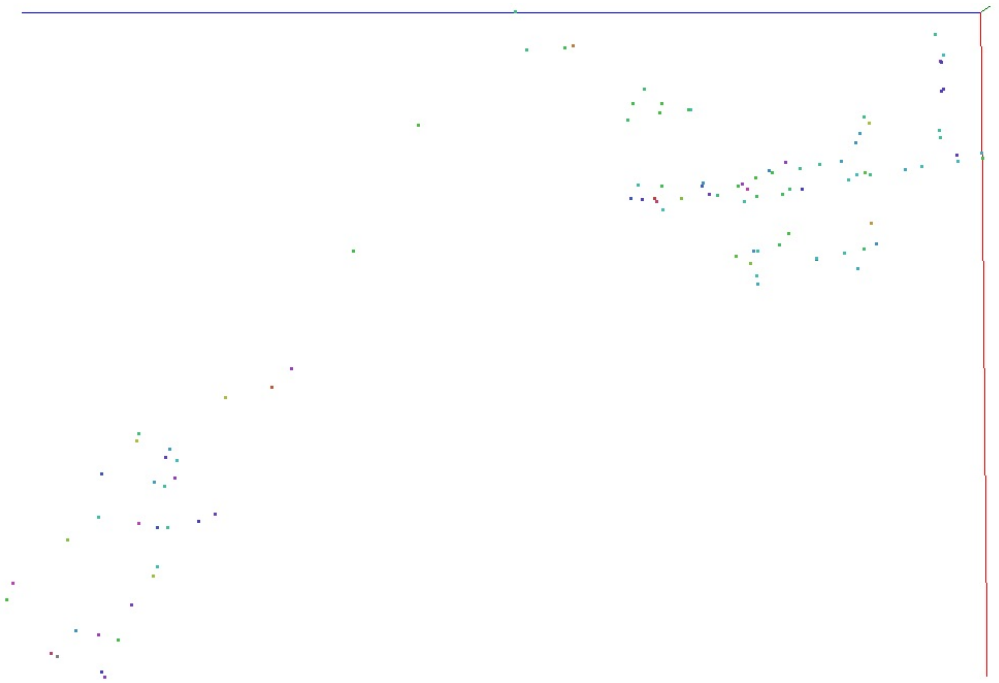
A.1 Hierarchické úrovně

Náhled na 3. výslednou úroveň hierarchického shlukování.



Obr. A.1: 3. úroveň výsledné hierarchie

Náhled na 4. výslednou úroveň hierarchického shlukování.



Obr. A.2: 4. nejvyšší úroveň výsledné hierarchie