České vysoké učení technické v Praze

Fakulta elektrotechnická

Katedra řídicí techniky

DIPLOMOVÁ PRÁCE

Identifikace dopravní infrastruktury na základě dat z GPS měření

Vypracoval: Bc. Milan Janeček
Vedoucí práce. Ing. Michal Kutil

Abstrakt

Cílem práce bylo analyzovat naměřená data, získaná z měření pomocí plovoucích vozidel vybavených GPS (Global Positioning System) měřícím zařízením. Tyto vozidla jezdí v městském provozu a v pravidelných intervalech zaznamenávají svoji GPS pozici.

Analýza naměřených dat se dělí na několik fází. První fáze je výběr analyzovaných dat. Druhá filtrace, po ní následuje generování shluků. Závěrečná fáze spojuje shluky a výsledkem je virtuální mapa.

Druhým úkolem bylo verifikovat získanou infrastrukturu pomocí mapových podkladů. Mapové podklady se stahují z Google Maps a lze je zobrazit během analýzy.

V praxi by měla tato práce posloužit k účelům monitorování hustoty dopravy a k následnému jejímu efektivnímu řízení.

Abstract

The aim of the project was to analyse measuring data which are obtained from measure using vehicles which have GPS receiver. This vehicles go in the city traffic and in the regular intervals record actual GPS position.

Analysis of measuring data is divided on a few parts. The first stage is choose of analysed data. The second is filtering. After the second stage follows the generating of clusters. The final stage links the clusters and the virtual map originate.

The second task was to verificate the infrastructure using the map. The map are downloaded from Google Maps and it can be show during the analysis.

In practice this work should answer for purposes of monitoring traffic density and for the efective traffic control.

Poděkování:

Chtěl bych poděkovat všem, kteří se podíleli na této diplomové práci, neboť bez jejich pomoci a rad by tato práce nemohla vzniknout. Především vedoucímu diplomové práce Ing. Michalu Kutilovi za jeho vedení, připomínky a poznámky k práci.

Dále bych poděkoval všem (především rodině), kteří mě po celou dobu studia podporují.

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, SW atd.) uvedené v přiloženém seznamu.

V Praze dne 21. 5. 2009 Milan Janeček

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Milan Janeček**

Studijní program: Elektrotechnika a informatika (magisterský), strukturovaný
Obor: Kybernetika a měření, blok KM1 - Řídicí technika

Název tématu: **Identifikace dopravní infrastruktury na základě dat z GPS měření**

Pokyny pro vypracování:

1. Seznamte se s principy měřením dat pomocí GPS a jejich následné analýzy. Prostudujte možnosti GPS při využití monitorování dopravy.
2. Navrhněte a implementujte algoritmus, jenž z naměřených dat, získaných z měření pomocí plovoucího vozidla vybaveného GPS měřícím zařízením, provede identifikaci dopravní infrastruktury.
3. Verifikujte získanou infrastrukturu pomocí mapových podkladů.

Seznam odborné literatury:

Dodá vedoucí práce

Vedoucí: Ing. Michal Kutil

Platnost zadání: do konce zimního semestru 2009/10

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 27. 2. 2009

Obsah

I. Úvod 1
II. Shluková analýza – teoretický rozbor 2
1. Úvod 2
1.1. Matematická formulace 2
1.2. Dělení shlukových metod 3
2. Matice dat – znaky a objekty 4
2.1. Standardizace dat 4
2.2. Normalizace objektů 4
3. Podobnost objektů 5
3.1. Metriky 5
4. Metody hierarchického shlukování 6
4.1. Aglomerativní hierarchické shlukování 6
4.1.1. Algoritmus aglomerativního shlukování 6
4.2. Koeficienty nepodobnosti shluků 7
4.3. Divizivní hierarchické shlukování 8
5. Nehierarchické shlukování 8
5.1. Problém počátečního počtu shluků 8
5.2. Problém počátečního rozkladu 9
5.3. Metoda K-means (MacQueenova metoda) 9
5.4. Metoda K-medoid 10
5.5. Fuzzy C-means algoritmus 11
5.6. The Gustafson–Kessel algoritmus 12
6. Porovnání metod 13
III. Získání virtuální mapy z množiny GPS bodů 14
1. Úvod 14
2. Získání vstupních dat pomocí kamery 14
3. Algoritmy pro zpracování GPS data 15
3.1. Algoritmus COP–k-means 15
3.2. Algoritmus DJ–Cluster 16
3.3. Algoritmus CLATIN 18
4. Schroedelova metoda 18

5. Zjišťování hustoty provozu 20
6. Spojování shluků podle směru vektoru 20
7. Zpřesňování virtuálních map 22
8. Prediktivní určování pohybu 22
IV. Specifikace pro mapu 24
1. Úvod 24
2. Filtrace dat 24
3. Generování shluků 26
3.1. Optimální počet shluků 26
3.2. Výpočetní náročnost algoritmů 35
4. Spojovací (linkage) metoda 36
4.1. Příklady výsledné virtuální mapy 41
V. Matlab GUI (Graphical User Interface) 43
1. Systém Handle Graphics 43
2. Vlastní realizace GUI, verifikace – Google mapy 44
VI. Závěr 52
VII. Literatura 53
VIII. Příloha CD 56

Seznam obrázků

II. Shluková analýza

Obr. 1 Učení bez učitele 2

Obr. 2 Ukázka dendrogramu 3

Obr. 3 Znázornění metody k-means pomocí diagramu 10

Obr. 4 Výsledek GK shlukovacího algoritmu (pro 3 shluky) a „contour map“ 13

III. Získání virtuální mapy z množiny GPS bodů

Obr. 1 Kamerový systém 14

Obr. 2 GPS přijímač 14

Obr.3 Zachycení tras (černé čáry) pomocí kamery 15

Obr. 4 K-means data pro k = 4, jednotlivé shluky jsou označeny rozdílnými symboly 15

Obr. 5 Generování shluků pomocí Cop–k–means algoritmu 16

Obr. 6 Density-based 17

Obr. 7 Změna směru při chůzi 18

Obr. 8 Spojení center (označeno modrým kruhem) shluků (cluster seeds) 19

Obr. 9 Spojení shluků do křižovatky 19

Obr. 10 Vygenerované směrové shluky 21

Obr. 11 Vstupní data 21

Obr. 12 Ilustrace algoritmu (x znamená střed shluku, který je tvořen bílými body, body uvnitř čárkovaného shluku tvoří předešlý shluk) 23

IV. Specifikace pro mapu

Obr. 1 Originální a vyfiltrovaná data 24

Obr. 2 Data na mapovém podkladu 24

Obr. 3 Špatně a dobře vyfiltrovaná data 25

Obr. 4 Ukázka obdélníkové sítě 26

Obr. 5 Data 1 27

Obr. 6 Data 3 27

Obr. 7 Data 2 27

Obr. 8 Data 4 27

Obr. 9 Data 5 27

Obr. 10 Hodnoty indexů PC a CE 27

Obr. 11 Hodnoty indexů SC, S 27

Obr. 12 Hodnoty indexů XB a DI 28
Obr. 13 Index ADI a sedm vygenerovaných shluků 28
Obr. 14 Hodnota indexu ADI pro Data 2 28
Obr. 15 Šest shluků pro Data 2 28
Obr. 16 Hodnota indexu ADI pro Data 3 29
Obr. 17 Sedm shluků pro Data 3 29
Obr. 18 Hodnota indexu ADI pro Data 4 29
Obr. 19 Sedm shluků pro Data 4 29
Obr. 20 Hodnota indexu ADI pro Data 5 29
Obr. 21 11 shluků pro Data 5 29
Obr. 22 Shluky pro Data 1 29
Obr. 23 Shluky pro Data 2 29
Obr. 24 Shluky pro Data 2 29
Obr. 25 Shluky pro Data 3 30
Obr. 26 Shluky pro Data 4 30
Obr. 27 Shluky pro Data 5 30
Obr. 28 Typy shluků: (1) Kulovitý, (2) Podlouhlý a (3) Složený („špatně oddělený“) 30
Obr. 29 Shluky pro Data 1 31
Obr. 30 Shluky pro Data 3 31
Obr. 31 Shluky pro Data 2 31
Obr. 32 Shluky pro Data 4 31
Obr. 33 Shluky pro Data 5 31
Obr. 34 Vygenerované shluky při použití algoritmu Fuzzy C-means 32
Obr. 35 Vygenerované shluky při použití algoritmu K-means 32
Obr. 36 Vygenerované shluky při použití aglomerativního hierarchického shlukování 33
Obr. 37 Vygenerované shluky při použití algoritmu Fuzzy C-means 34
Obr. 38 Vygenerované shluky při použití algoritmu K-means 34
Obr. 39 Doba výpočtu 35
Obr. 40 Vstupní data – neorientovaný graf 36
Obr. 41 Vývojový diagram 37
Obr. 42 Ukázka dílčích směrových vektorů 37
Obr. 43 Spojení shluků po druhé fázi „Hledání vrcholů stupně 2“ 38
Obr. 44 Hledání třetí hrany trojúhelníka 38
Obr. 45 Ukázka fáze „Nevykreslení kombinace“ 39

Obr. 46 Ukázka fáze „Vykreslení kombinace“ 39
Obr. 47 Ukázka fáze „Nevykreslení kombinace“ – druhá v pořadí 40
Obr. 48 Ukázková křižovatka na mapovém podkladu 40
Obr. 49 Virtuální mapa na mapovém podkladu 41
Obr. 50 Virtuální mapa na mapovém podkladu 42
V. Matlab GUI (Graphical User Interface)
Obr. 1 Hierarchie základních grafických objektů systému Matlab 43
Obr. 2 Wireframe (drátěný model) aplikace 44
Obr. 3 Zobrazení průběhu výpočtu 44
Obr. 4 Hledání zoomu 45
Obr. 5 Výsledná mapa s patřičným zoomem 46
Obr. 6 Chybové okno 46
Obr. 7 Panel s parametry 47
Obr. 8 Panel s funkcemi 47
Obr. 9 Menu Bar 48
Obr. 10 Hlavní část okna figury Google Map 49
Obr. 11 O programu 49
Obr. 12 GUI s výslednými daty 50
Obr. 13 Graf závislosti jednotlivých funkcí 51

Seznam tabulek

Tabulka 1 Časová náročnost shlukových metod 35
Tabulka 2 Časová náročnost shlukových metod pro složitější infrastrukturu 35

Seznam kódu

Kód Matlabu 1 Struktura uložených dat 48

I. Úvod

Hlavním úkolem této práce je analýza naměřených dat, získaných z měření pomocí plovoucích vozidel vybavených GPS (Global Positioning System) měřícím zařízením. Tato vozidla jezdí v městském provozu a v pravidelných intervalech zaznamenávají svoji GPS pozici.

Diplomová práce navazuje na projekt Lomon (Monitorování městské dopravy pomocí plovoucích vozidel), ve kterém jsme s Markem Riedelem řešili sběr GPS dat. Jednotlivá data jsme prostřednictvím komunikátoru GC 072 240F od firmy Levelna přenášeli na server, kde se pomocí šifrovacího klíče, který je uložen v databázi, přeloží a následně se ověří jejich správnost pomocí kontrolního součtu CRC.

Zpracování naměřených dat lze rozdělit na několik hlavních fází. Prvotní fází je výběr analyzovaných dat. Poté následuje filtrace vybraných dat. Vyfiltrovaná data se používají ve třetí fázi pro vygenerování shluků a v poslední fázi dochází ke spojení vygenerovaných shluků a vytvoření virtuální mapy. Podrobnosti těchto fází lze nalézt v *kapitole IV*. *Kapitola V* popisuje grafické řešení (GUI). V *kapitole II* se nachází teoretický rozbor shlukové analýzy a v *kapitole III* rešeršní zpráva zabývající se získáním virtuální mapy z GPS bodů.

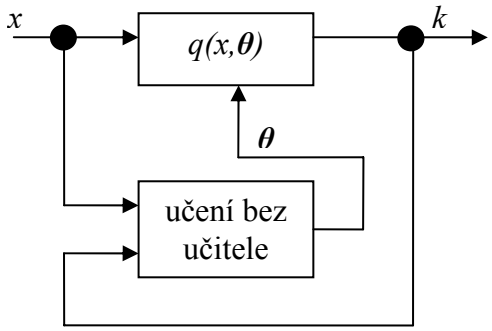
Dalším úkolem je verifikace získané infrastruktury pomocí mapových podkladů. Tento problém je také popsán v *kapitole V*.

V praxi by měly tato práce a předešlý projekt být využity za účelem monitorování hustoty dopravy a k jejímu následnému efektivnímu řízení. Pro splnění těchto cílů bude ještě nutné vyřešit vhodné uložení výsledných dat a případnou aktualizaci.

II. Shluková analýza – teoretický rozbor

1. Úvod

Shluková analýza (cluster analysis) je statistická metoda používaná ke klasifikaci objektů. Klasifikace je činnost vytvářející rozklad nějaké množiny objektů na menší množiny. Je zahrnovaná mezi metody učení bez učitele. Učení bez učitele Obr. 1 se používá pro analýzu pozorování (nebo dat), když není k dispozici informace od učitele, tj. trénovací množina. Jejím cílem je nalézt v dané množině objektů podmnožiny (shluky objektů) tak, aby si byli navzájem podobné, ale ne mimo tento shluk. Učení bez učitele znamená, že parametr $\boldsymbol{\theta}$ neurčuje učitel, ale závisí jak na posloupnosti pozorování tak i na jednotlivých výsledcích k.



trénovací množina = výstup z klasifikátoru
$\boldsymbol{\theta}$ = učení (x, k)

kde
$x = (x_1, \ldots, x_n)$ je posloupnost pozorování
$k = (k_1, \ldots, k_n)$ je posloupnost výsledků
$\boldsymbol{\theta}$ je parametr, na němž závisí rozhodovací strategie q

Obr. 1 Učení bez učitele

1.1. Matematická formulace

Základní množina $\boldsymbol{\Psi}$ (n objektů) se skládá z disjunktních neprázdných podmnožin C_i, které dohromady tvoří množinu $\boldsymbol{\Psi}$. To znamená, že pro $i \neq j$

$$C_i \cap C_j = \emptyset, C_1 \cup C_2 \cup \cdots \cup C_m = \boldsymbol{\Psi}, \qquad \textbf{(1)}$$

kde m udává počet rozkladů. Každá množina C_i se nazývá komponentou rozkladu. Shlukování náleží k třídě úloh tohoto tvaru:

Nechť x je náhodný objekt z množiny objektů $\boldsymbol{\Psi}$ s rozdělením pravděpodobnosti p_x: $\boldsymbol{\Psi} \rightarrow \boldsymbol{R}$. Nechť $\boldsymbol{D}$ je množina rozhodnutí taková, že pro každý objekt $x \in \boldsymbol{\Psi}$ se určí jisté rozhodnutí $d \in \boldsymbol{D}$. Nechť $\boldsymbol{W} : \boldsymbol{\Psi} \times \boldsymbol{D} \rightarrow \boldsymbol{R}$ je pokutová funkce, jejíž hodnota $\boldsymbol{W}\ (x,d)$ představuje ztráty v případě volby rozhodnutí d pro objekt x. Cílem úlohy je zkonstruovat strategii $\boldsymbol{Q} : \boldsymbol{\Psi} \rightarrow \boldsymbol{D}$, která minimalizuje hodnotu:

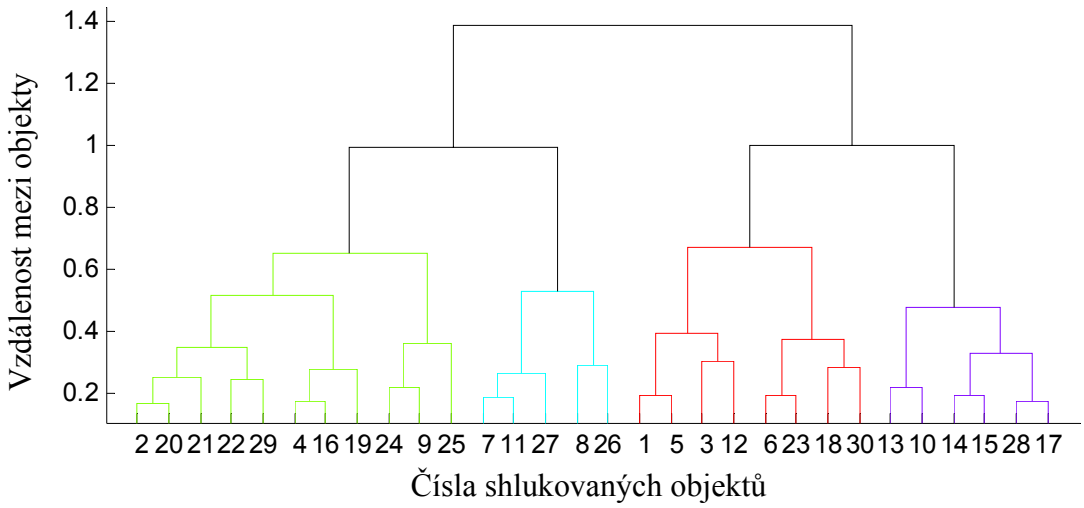
$$\sum_{x \in \Psi} p(x) \cdot W(x, Q(x)) \,. \qquad \textbf{(2)}$$

Výsledkem strategie ***Q*** je vytvořit jednotlivé shluky tak, aby objekty splňovaly dané rozhodnutí ***D***.

1.2. Dělení shlukových metod

Základní dělení shlukových metod je podle cíle – ke kterému směřují – a to na hierarchické a nehierarchické.

Hierarchické shlukování je systém podmnožin, kde průnikem dvou podmnožin shluků je buď prázdná množina nebo jedna z nich. Pokud alespoň jednou nastane druhý případ hovoříme o hierarchickém shlukování. Podle přístupu k základní množině dat dělíme hierarchické shlukování na *divizivní* (vycházíme z celku, jednoho shluku a ten následně dělíme) a *aglomerativní* (vycházíme z jednotlivých objektů, shluků o jednom prvku a ty spojujeme). Ve skutečnosti se jedná o větvení. Graficky lze hierarchické shlukování vyjádřit pomocí dendrogramu (Obr. 2). Jedná se o podobnostní strom, kde se na jednu osu vynáší pořadová čísla shlukovaných objektů a druhou osu tvoří stupnice, z níž je možné číst velikost shlukovací hladiny neboli vzdálenost mezi objekty. Pro velké množství dat se dendrogram vykresluje od určité hladiny, aby nedocházelo k přeplnění obrázku. Dále je možné na dendrogramu barevně znázornit shluky, kde při spojování jednotlivých objektů nebyl překročen tzv. *threshold* (prahová vzdálenost mezi objekty).



Obr. 2 Ukázka dendrogramu

Nehierarchické shlukování je systém navzájem různých neprázdných podmnožin vstupní množiny dat, v němž průnik libovolných dvou podmnožin (shluků) je prázdný. Tento typ shlukování vede k disjunktnímu rozkladu množiny.

2. Matice dat – znaky a objekty

Každý objekt musí být popsán vektorem v o p znacích (např. délka, šířka, …), kde v_j označuje j-tý znak vektoru v. Matici dat pro shlukovou analýzu obvykle sestavujeme tak, že objekty tvoří její řádky, znaky její sloupce.

2.1. Standardizace dat

Velmi často jsou jednotlivé znaky objektů v různých jednotkách. Tato vlastnost může ovlivňovat průběh shlukování, neboť se pak některé znaky mohou jevit jako dominantní. Abychom dosáhli správných výsledků je nutné provést standardizaci dat a to tak, že v každém sloupci matice $\boldsymbol{Z} = (z_{ij})$ pro každý znak v_j $(j = 1, 2, \ldots, p)$ provedeme tyto úpravy dat:

1. Vypočteme střední hodnotu $\overline{v_j}$ j-tého znaku v_j a směrodatnou odchylku s_j pro $j = 1, 2, \ldots, p$ podle vztahů:

$$\bar{v}_j = \frac{1}{n}\sum_{i=1}^{n} z_{ij}, \qquad s_j = \left[\frac{1}{n}\sum_{i=1}^{n}\left(z_{ij} - \bar{v}_j\right)^2\right]^{1/2}. \qquad \textbf{(3)}$$

2. původní hodnoty z_{ij} j-tého znaku i-tého objektu přepočteme na tzv. standardizované hodnoty:

$$x_{ij} = \frac{z_{ij} - \bar{v}_j}{s_j}. \qquad \textbf{(4)}$$

Tyto nové hodnoty jsou již nezávislé na jednotkách měření a stávají se veličinami bezrozměrnými (střední hodnotu mají rovnou nule a rozptyl roven jedné).

2.2. Normalizace objektů

Objekty pro shlukovou analýzu jsou určeny vektory o p znacích. Normy těchto vektorů mohou někdy nežádoucím způsobem ovlivňovat výsledky kvantitativního hodnocení podobností objektů. V takových případech je vhodné normalizovat tyto vektory, aby měli

stejnou normu (nejlépe jednotkovou). Normalizace docílíme tak, že budeme normalizovat všechny vektory (řádky) matice dat na jednotkovou normu tak, že všechny složky každého z vektorů dělíme normou tohoto vektoru.

3. Podobnost objektů

Jedním ze základních problémů shlukové analýzy je vzájemná podobnost objektů a kvantitativní vyjádření této podobnosti. Někdy se způsob hodnocení podobnostních vztahů váže přímo k určité shlukovací metodě. V některých případech jde o dvě na sobě nezávislé části problematiky shlukové analýzy.

3.1. Metriky

Metriky patří mezi nejběžnější způsob vyjádření podobnostních vztahů mezi objekty vycházející z geometrického modelu matice dat. Přiřadíme-li objektům charakterizovaným p znaky jako modely body p-rozměrného euklidovského prostoru $\boldsymbol{E}_p$, pak lze pro libovolné dva body $R = (r_1, r_2, \ldots, r_p)$ a $S = (s_1, s_2, \ldots, r_p)$ psát euklidovskou vzdálenost:

$$\delta(R,S) = \left[\sum_{i=1}^{p} (r_i - s_i)^2 \right]^{1/2}. \qquad \textbf{(5)}$$

Obecně je metrika δ funkce definovaná na $\boldsymbol{E}_p \times \boldsymbol{E}_p$, přiřazující každé dvojici bodů (R, S) číslo $\delta(R, S)$ splňující tyto čtyři podmínky:

- $\delta(R, S) \geq 0$
- $\delta(R, S) = 0 \Leftrightarrow R = S$ (identita)
- $\delta(R, S) = \delta(S, R)$ (symetrie)
- $\delta(R, T) \leq \delta(R, S) + \delta(S, T)$ (trojúhelníková nerovnost)

Další metrika vhodná pro posuzování vzájemné podobnosti objektů je Minkowského metrika, která je definována vztahem:

$$\delta_k(R,S) = \left(\sum_{i=1}^{p} |r_i - s_i|^k \right)^{1/k}. \qquad \textbf{(6)}$$

Pro různá k dostáváme např. tyto metriky [2]:

- vzdálenost v „městských blocích“ (city block) pro $k = 1$

- euklidovskou metriku pro $k = 2$
- sup-metriku $\delta_\infty(R,S) = \max_{i=1,\dots p}\{|r_i - s_i|\}$ pro $k \to \infty$

Podobnost mezi objekty se nechá popsat řadou dalších podobnostních koeficientů [2, 3], např. pomocí koeficientu korelace, koeficientu nepodobnosti objektů, …

4. Metody hierarchického shlukování

Hierarchické shlukování je sekvence vnořených rozkladů, která na jedné straně začíná triviálním rozkladem, kdy každý objekt dané množiny objektů tvoří jednoprvkový shluk, a na druhé straně končí triviálním rozkladem s jedním shlukem obsahujícím všechny objekty. Podle směru postupu při shlukování dělíme metody hierarchického shlukování na aglomerativní a divizivní.

4.1. Aglomerativní hierarchické shlukování

Vycházíme z množiny objektů tvořených n jednoprvkovými shluky – často označováno jako nultý rozklad. Pro další rozklady množiny objektů je nutné definovat způsob hodnocení podobnostních vztahů mezi shluky. V každém následujícím kroku vybíráme ty dva shluky, které jsou si ve smyslu definice nejpodobnější. Z těchto dvou shluků vytvoříme nový shluk. Takto postupujeme až do té doby než vznikne pouze jediný shluk. Tento algoritmus snadno zapíšeme pomocí rekurze.

4.1.1. Algoritmus aglomerativního shlukování

- Nultý zápis počáteční množiny objektů
- V i-tém kroku ($0 < i < n - 1$) najdeme takovou dvojici shluků, pro kterou je hodnota koeficientu nepodobnosti shluků minimální - sjednocení těchto dvou shluků nahradí v rozkladu samostatné prvky shluků
- V posledním n-1 kroku jsou všechny shluky sjednoceny v jediný shluk

4.2. Koeficienty nepodobnosti shluků

Koeficienty nepodobnosti shluků jsou považovány za kvantitativní hodnocení podobnostních vztahů mezi shluky. Existuje celá řada těchto koeficientů. Mezi nejznámější patří:

Metoda nejbližšího souseda (nearest neighbour) byla poprvé uvedena panem Sneath pod názvem „single link“ [8] a definuje koeficient nepodobnosti shluků vztahem

$$\boldsymbol{D}(C_i, C_j) = \min_{\substack{O_i \in C_i \\ O_j \in C_j}} \{\boldsymbol{d}(O_i, O_j)\}. \qquad \textbf{(7)}$$

Metoda nejvzdálenějšího souseda – jejímž autorem je Sörensen, je známá též pod názvem „complete linkage“ [9] a je definována vztahem

$$\begin{aligned} &\boldsymbol{D}(C_i, C_j) = \max_{\substack{O_i \in C_i \\ O_j \in C_j}} \{\boldsymbol{d}(O_i, O_j)\} \qquad pro\ C_i \neq C_j, \\ &\boldsymbol{D}(C_i, C_i) = 0, \end{aligned} \qquad \textbf{(8)}$$

kde

$\boldsymbol{d}$	je libovolný koeficient nepodobnosti objektů,
C_i, C_j	jsou shluky rozkladu,
$\boldsymbol{D}(C_i, C_j)$	koeficient nepodobnosti shluků danou metodou.

Centroidní metoda – nepodobnost mezi dvěma shluky je vyjádřena pomocí euklidovské vzdálenosti jejich těžišť [10]. Pro zavedení koeficientu nepodobnosti shluků centroidní metodou je výhodnější použít čtverec euklidovské vzdálenosti $\boldsymbol{d}_E^2$ než samotné vzdálenosti $\boldsymbol{d}_E$, a to pro zjednodušení výpočtu hodnot. Potom pro koeficient platí:

$$\boldsymbol{D}(C_i, C_j) = \boldsymbol{d}_E^2(\overline{T_1}, \overline{T_2}), \qquad \textbf{(9)}$$

kde

C_i, C_j	jsou shluky rozkladu,
$\boldsymbol{D}(C_i, C_j)$	koeficient nepodobnosti shluků centroidní metodou,
$\overline{T_1}, \overline{T_2}$	jsou těžiště shluků.

Dalšími metodami pro určení nepodobnosti shluků jsou např. Mediánová metoda, Wardova - Wishartova metoda a mnoho jiných [2].

4.3. Divizivní hierarchické shlukování

Divizivní metody vytvářejí hierarchický systém rozkladů množiny objektů postupným rozdělováním existujících shluků. Při aplikaci divizivního algoritmu postupujeme tak, že za počáteční shluk pokládáme celou množinu objektů a postupně rozdělujeme existující shluky tak dlouho, až jsou všechny shluky jednoprvkové. Postup divizivního shlukování spočívá v postupném rozdělování každého z existujících shluků na dva nové tak, aby výsledný rozklad tohoto shluku byl optimální vzhledem k nějakému kritériu. Nalezení absolutně optimálního rozkladu množiny n objektů na dvě podmnožiny však vyžaduje prozkoumání $(2^{n-1} - 1)$ možností.

Za nejvhodnější divizivní metodu je považována MacNaughton–Smithova metoda [11]. Ta v prvním kroku algoritmu vypočte pro danou množinu objektů matici vzdáleností s použitím euklidovské metriky $\boldsymbol{d}_E$. Jednotlivé prvky matice se používají k výpočtům průměrných vzdáleností jednotlivých objektů se skupinami objektů. V rozdělovaném shluku ζ najdeme objekt, jehož průměrná vzdálenost od ostatních objektů je maximální. Tento objekt tvoří základ nově vznikajícího shluku χ. Průměrná vzdálenost tohoto objektu od nově vznikajícího shluku χ obsahujícího pouze tento objekt je nulová. Rozdíl průměrných vzdáleností od objektů zbývajícího shluku ζ a nově vznikajícího shluku χ je kladný. Najdeme další objekt, jehož rozdíl průměrných vzdáleností je maximální. Je-li tento rozdíl kladný, přemístíme nalezený objekt k nově vznikajícímu shluku χ. Přemisťování se provádí tak dlouho, dokud je rozdíl kladný, v opačném případě ukončíme rozdělování. Po ukončení rozkladu jednoho shluku vybereme z existujících shluků další shluk a opět na něm provádíme rozklad na dva shluky. Rozklady provádíme tak dlouho, až jsou shluky jednoprvkové. Podrobněji viz. [2].

5. Nehierarchické shlukování

5.1. Problém počátečního počtu shluků

Při použití nehierarchického shlukování nastává problém, jak stanovit optimální počet shluků. Mnoho předních teoretiků věnovalo značnou pozornost této problematice, ale nebylo nalezeno optimální uspokojivé řešení. Bylo vytvořeno několik indexů pro nalezení

optimálního počtu shluků např.: Partition index, Separation index – více o indexech lze nalézt v [5].

5.2. Problém počátečního rozkladu

Jedná se o problém stanovení typických vzorových objektů (v Euklidovském prostoru typických bodů), kolem nich se začnou vytvářet jednotlivé shluky. Pro výběr počátečních typických bodů se využívají tyto jednoduché postupy:

- výběr prvních *k*-bodů z libovolně uspořádané množiny bodů
- generovat *k* umělých bodů, jejichž každá souřadnice je náhodné číslo z příslušného variačního intervalu
- generovat (2^p + 1) počátečních typických bodů (*p* je počet rozměrů prostoru), z nichž první je těžištěm celé množiny bodů, a další pak jsou vrcholy *p*-rozměrného kvádru se středem v těžišti a hranami délky $2s_i$ (s_i je směrodatná odchylka hodnot *i*-té souřadnice) rovnoběžnými s osami soustavy souřadnic. Je důležité, aby platilo $2^p > n$, kde *n* je počet bodů. Tento postup popisují páni Fromm a Northouse ve svém algoritmu CLASS [12].

5.3. Metoda K-means (MacQueenova metoda)

Algoritmus *K-means* [13] byl publikován již v roce 1967 a doposud je hojně využíván k vytváření shluků. Algoritmus má dva vstupní parametry, prvním je množina dat $X = \{x_1,...,x_n\}$ a druhým je číslo *k* udávající počáteční počet typických bodů – vektorů μ_j. Na počátku se vektory $\mu_j, j = 1,...,k$ inicializují na náhodně zvolenou hodnotu nebo se inicializují pomocí vhodně zvolené heuristiky (využívající apriorní znalosti o úloze). Následně po inicializaci se začnou iterativně provádět následující dva kroky:

1. **Klasifikace:** Všechna data $x_i, i = 1,...,n$ se klasifikují do tříd určených vektory $\mu_j, j = 1,...,k$ podle minima euklidovské vzdálenosti (dle nejbližšího souseda, *nearest-neighbour classification*). Vzor x_i se tedy přiřadí do třídy

$$y_i = \arg\min_{j=1,...,k} \left\| x_i - \mu_j \right\|. \qquad \textbf{(10)}$$

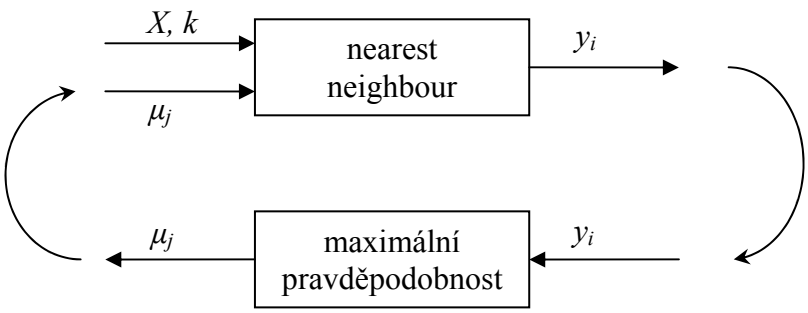
2. **Učení:** V tomto kroku se počítají nové hodnoty vektorů μ_j jako střední hodnoty dat, které byly klasifikovány do třídy určené příslušným vektorem μ_j. Nová hodnota μ_j se spočte podle vztahu

$$\mu_j = \frac{1}{n_j} \sum_{i \in \{i : y_i = j\}} x_i , \qquad \textbf{(11)}$$

kde n_j je počet vzorů x_i klasifikovaných v prvním kroku do třídy určené vektorem μ_j.

Tyto dva kroky se opakují do té doby, dokud se alespoň jeden vektor x_i klasifikuje do jiné třídy, než byl klasifikován v předcházejícím kroku.

Druhý krok algoritmu je v podstatě maximální věrohodný odhad střední hodnoty pro každou třídu zvlášť. Proto celý algoritmus lze znázornit následujícím diagramem:



Obr. 3 Znázornění metody *k-means* pomocí diagramu

Velmi podobnou metodou je tzv. Wishartova metoda, která končí hned po první iteraci, při níž nedošlo ke změně v rozkladu do shluků narozdíl od MacQueenovy, která končí až v následující neměnné iteraci. Algoritmus konverguje v konečném počtu kroků k nějakému řešení [2].

5.4. Metoda K-medoid

Velmi podobná metoda jako *k-means*. Liší se v tom, že centra shluků (vektory μ_j) jsou nahrazena nejbližšími body ze vstupních dat [14]. Tato metoda je vhodná pro takový systém, kde výsledné řešení se musí skládat z bodů ze vstupní množiny dat (pro nespojitá data).

5.5. *Fuzzy C-means algoritmus*

Fuzzy C-means algoritmus [4] hledá minimalizaci funkce, která se nazývá *C-means functional*. Byl definován panem Dunnem jako:

$$J(\boldsymbol{X};\boldsymbol{U},\boldsymbol{V}) = \sum_{i=1}^{c}\sum_{k=1}^{N}(\mu_{ik})^{m}\left\|x_k - v_i\right\|^2, \tag{12}$$

kde

$\boldsymbol{V} = [v_1, v_2, ..., v_c], v_i \in \boldsymbol{R}^n$	jsou centra shluků,
c	je počet shluků,
$\boldsymbol{U} = [\mu_{ik}]$	je matice, kde jednotlivé prvky určují podobnost mezi objekty v rozmezí od 0 do 1,
$\boldsymbol{X} = [x_1, x_2, ..., x_N]$	je vstupní množina dat.

Předešlou rovnici ze statistického pohledu lze chápat jako měření celkové variance. Minimalizaci funkce (12) lze chápat jako nelineární optimalizační problém, který lze jednoduše vyřešit pomocí *Picard iteration* (hledání stacionárních bodů). Stacionární body funkce (12) s následujícími podmínkami

$$\mu_{ij} \in [0,1], 1 \le i \le N, 1 \le k \le c, \tag{13}$$

$$\sum_{k=1}^{c}\mu_{ik} = 1, 1 \le i \le N, \tag{14}$$

$$0 < \sum_{i=1}^{N}\mu_{ik} < N, 1 \le k \le c \tag{15}$$

lze vypočítat Lagrangeových multiplikátorů jako

$$\overline{J}(\boldsymbol{X};\boldsymbol{U},\boldsymbol{V},\lambda) = \sum_{i=1}^{c}\sum_{k=1}^{N}(\mu_{ik})^{m}\left\|x_k - v_i\right\|^2 + \sum_{k=1}^{N}\lambda_k\left(\sum_{i=1}^{c}(\mu_{ik} - 1)\right). \tag{16}$$

Výsledkem minimalizace (12) je

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{D_{ik}}{D_{jk}}\right)^{2/(m-1)}}, 1 \le i \le c, 1 \le k \le N, \tag{17}$$

a

$$v_i = \frac{\sum_{k=1}^{N}\mu_{ik}^{m}x_k}{\sum_{k=1}^{N}\mu_{ik}^{m}}, 1 \le i \le c, \tag{18}$$

kde $D_{ik}^2 = \|x_k - v_i\|^2$.

Fuzzy C-means algoritmus (FCM) počítá jednotlivé shluky pomocí standardní Euklidovské normy. Podrobněji v [5].

5.6. The Gustafson–Kessel algoritmus

Gustafson a Kessel rozšířili standardní fuzzy metodu c-means a to tak, že používají jinou vzdálenostní normu [6]. Každý shluk má vlastní normovou matici A_i a norma je pak dána

$$D_{ikA}^2 = (x_k - v_i)^T A_i (x_k - v_i), 1 \le i \le c, 1 \le k \le N. \quad \textbf{(19)}$$

Matice A_i jsou použity jako optimalizační proměnné v *c-means functional*, pomocí těchto proměnných lze přizpůsobit vzdálenostní normu každého shluku. Označíme-li matici $\boldsymbol{A}$ jako c-násobnou normovou matici $\boldsymbol{A} = (A_1, A_2, ..., A_c)$, pak funkce GK algoritmu je definována jako

$$J(\boldsymbol{X};\boldsymbol{U},\boldsymbol{V},\boldsymbol{A}) = \sum_{i=1}^{c}\sum_{k=1}^{N}(\mu_{ik})^m D_{ikAi}^2 . \quad \textbf{(20)}$$

Abychom získali přípustné řešení, musí být A_i pozitivně definitní. Toho docílíme tak, že

$$\|A_i\| = \rho_i, \rho > 0, \quad \textbf{(21)}$$

kde ρ_i je pevně daný pro každý shluk. Použitím Lagrangeových multiplikátorů dostaneme:

$$A_i = [\rho_i \det(\boldsymbol{F}_i)]^{1/n} \boldsymbol{F}_i^{-1}, \quad \textbf{(22)}$$

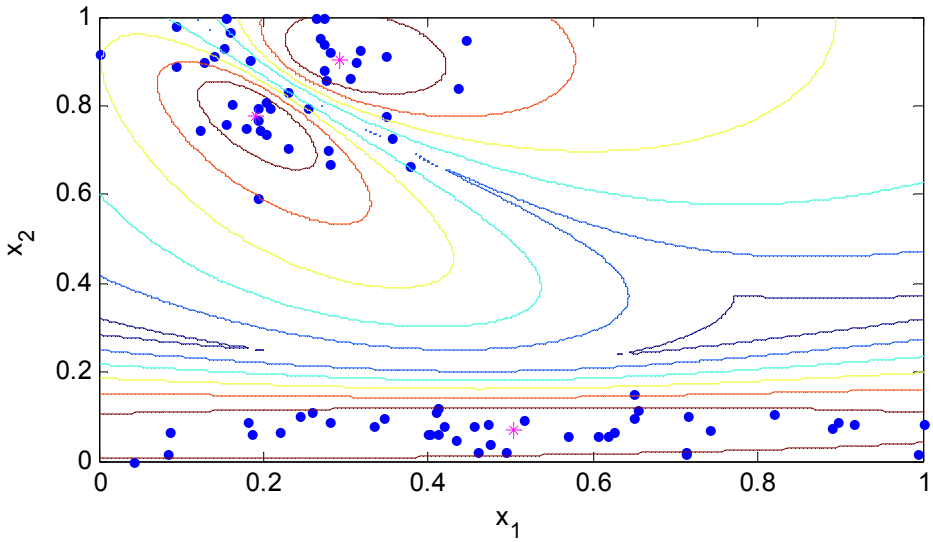
kde $\boldsymbol{F}_i$ je fuzzy kovarianční matice i-tého shluku a pro kterou platí:

$$\boldsymbol{F}_i = \frac{\sum_{k=1}^{N}(\mu_{ik})^m (x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^{N}(\mu_{ik})^m} . \quad \textbf{(23)}$$

Substituce (23) a (22) do rovnice (19) zanáší kvadratickou Mahalanobis vzdálenostní normu definovanou mezi x_k a středem shluku v_i, kde kovarianční váhy jsou obsaženy v matici $\boldsymbol{U}$.

Robustní variantu tohoto algoritmu popsali R. Babuška, P. J. van der Veen a U. Kaymak [7] a implementaci tohoto algoritmu lze nalézt v Fuzzy toolboxu [5].

Tento toolbox také obsahuje metody pro výpočet tzv. *contour map* (Obr. 4). Pomocí těchto vrstevnic lze rozpoznat, jak jsou jednotlivé body závislé na středu shluku. Anebo lze efektivně podle nich přiřadit nové body k patřičnému shluku (tzv. „unseen data“).



Obr. 4 Výsledek GK shlukovacího algoritmu (pro 3 shluky) a „*contour map*“

6. Porovnání metod

Při porovnání metod jsem se soustředil na dvě hlavní věci, a to na determiničnost a na velikost vstupní množiny bodů.

Nevýhoda hierarchického shlukování spočívá v tom, že není vhodné tyto metody používat na velké množství dat, neboť koeficient nepodobnosti se vypočítává mezi každou dvojicí vstupních dat. Na velké množství dat je tedy efektivnější využít nehierarchické shlukování. Výhoda je v rovnoměrném rozprostření shluků, kdežto u nehierarchického shlukování dochází k nahušťování shluků tam, kde se nachází více dat. Deterministickému kriteriu nevyhovují pouze dvě metody K–means a K–medoid.

III. Získání virtuální mapy z množiny GPS bodů

1. Úvod

V současné době dostupným digitálním mapám chybí mnoho upřesňujících (užitečných) vlastností např. informace o tom, do kterého pruhu se má řidič zařadit, aby dosáhl svého cíle aniž by v poslední chvíli musel měnit jízdní pruh, nebo aby byl včas informován o uzavírkách silnic, o nehodách anebo průměrné hustotě provozu v daném místě. O těchto problémech je pojednáváno v [1].

Obr. 1 Kamerový systém

Obr. 2 GPS přijímač

Vstupní množinu dat se nejčastěji získává pomocí GPS přijímačů (Obr. 2) v podobě GPS bodů (zeměpisná šířka a délka) anebo pomocí kamery (Obr. 1). Tyto data jsou následně analyzována – vytvoření shluků (pomocí shlukovacích algoritmů), spojení shluků. Výsledné řešení je virtuální mapa.

2. Získání vstupních dat pomocí kamery

Jedná se zařízení pomocí kterého jsme schopni vytvořit model virtuální mapy. Kamera musí být schopna detekovat trasy jednotlivých dopravních prostředků [31]. Jak tato vstupní data mohou vypadat je znázorněno na Obr. 3.

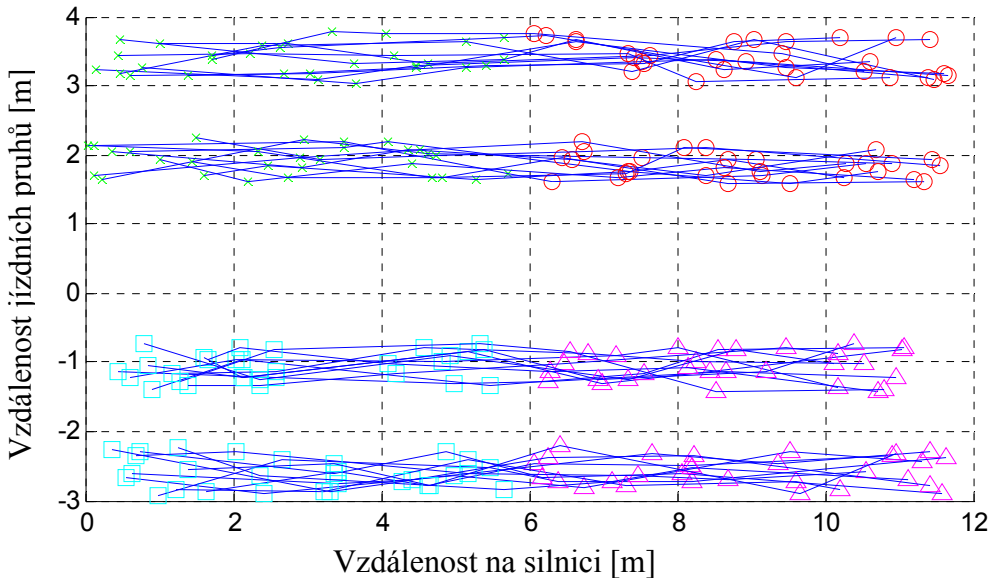
Na tato naměřená data je vhodné použít *fuzzy C-means algoritmus* [31] anebo *two–layer hierarchical clustering*, který v první vrstvě vyhledá dominantní cesty a v druhé tyto cesty dále dělí na jednotlivé jízdní pruhy [32], dále v této práci je diskutováno o anomálii detekce (zjišťování netypických cest, překročení rychlostního limitu).

Obr.3 Zachycení tras (černé čáry) pomocí kamery

3. Algoritmy pro zpracování GPS data

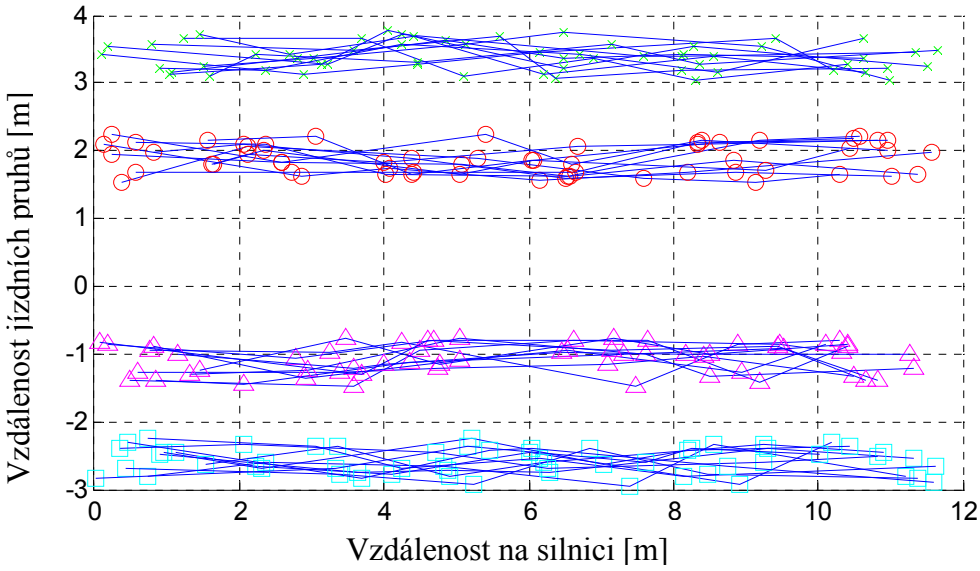
3.1. Algoritmus COP–k-means

Tento algoritmus je vhodný pro zjišťování počtu pruhů v konkrétní části silnice. Je ho vhodné použít jen v případě, že máme přesná data - na první pohled jsou patrné jízdní pruhy. V opačném případě (při cestě docházelo k přejíždění mezi pruhy, nepřesnost přijímaného GPS signálu je větší jak šířka jízdního pruhu) dostaneme špatné výsledky, jako je tomu i když použijeme jen *k*-means algoritmus viz. Obr. 4.



Obr. 4 *K*-means data pro $k = 4$, jednotlivé shluky jsou označeny rozdílnými symboly

Při použití metody COP–*k*–means [16] docílíme správného výsledku – 4 shluky odpovídající jednotlivým jízdním pruhům. Principem algoritmu je, že si nejprve nadefinujeme patřičná omezení (více o algoritmech s omezením je v [17] a v Eatonově diplomové práci [18]). Omezení spočívají v tom, že si stanovíme, které body musí být v daném shluku. Pro data na Obr. 5 je vhodné přidat podmínky, že jednotlivé body z každé cesty musí patřit do stejného shluku – na Obr. 5 označeno modře. Další část algoritmu je stejná jako algoritmus *k*-means. V případě, že zkoumáme jízdní pruhy na datech (jednotlivé silnice), která jsou příliš rozlehlá, je vhodné si rozdělit data na několik segmentů a algoritmus použít pro každý zvlášť. Efektivita tohoto algoritmu byla zkoumána na několika rozdílných datech a pro různé podmínky. Vždy bylo dosaženo lepšího výsledku při použití metody COP–*k*–means než algoritmů bez podmínek, podrobněji [15] a [19].



Obr. 5 Generování shluků pomocí *Cop–k–means algoritmu*

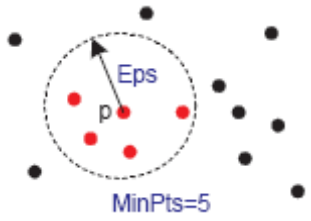
Existuje řada dalších shlukovacích algoritmů, které také zahrnují problematiku omezení např. MPCK–Means algoritmus [29], který se především používá pro více rozměrná data (např. pro shlukování proteinů).

3.2. Algoritmus DJ–Cluster

DJ-Cluster algoritmus [20] patří mezi deterministické algoritmy. Používá se především pro vyhledávání často navštěvovaným míst během určitého časového intervalu [21] (např.: místo našeho zaměstnání, oblíbená restaurace, sportovní klub, …). Výhoda tohoto algoritmu oproti předešlým algoritmům je v tom, že robustněji ignoruje šum, velké chyby měření a

nepodstatné body. Má jednodušší výběr vstupních parametrů (*Eps* - poloměr kruhu a *MinPts* - minimální počet bodů uvnitř kruhu) a je deterministický (narozdíl od *k*-means algoritmu).

Pro každý bod počítá tzv. hustotu bodů v jeho sousedství (density-based neighborhood of a point), tak aby nedošlo k překročení vstupního parametru *Eps* viz. Obr. 6. Pokud daný bod nesplňuje podmínku *MinPts*, tak je označen jako nepodstatný bod neboli jako šum a neovlivní výstup algoritmu.



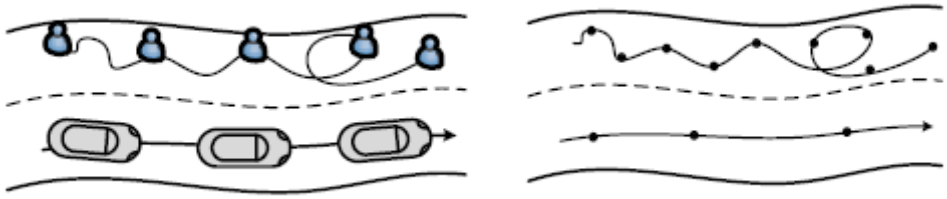
Obr. 6 Density-based

Tento postup by bylo vhodné použít pro filtraci vstupních dat pro virtuální mapu. Mělo by dojít k odfiltrování nepřesných bodů a cest, kterým auto projelo jednou až dvakrát za celou dobu měření.

Chceme-li získat přesné informace o denním programu, je nutné přidat další vstupní parametr (*deltaT* - časový interval ⇒ *R-TDJ algoritmus* [24]). Pomocí tohoto parametru docílíme toho, že shluky (významná místa navštívená několikrát za den) budou rozděleny. Tyto algoritmy se především používají v ekonomických oblastech (např. pro zjištění zda by nebylo výhodné postavit další supermarket nebo sportovní areál, aby zbytečně zákazníci daleko nedojížděli).

Výsledná řešení těchto algoritmů mohou být i implementována do GPS navigací, kde informují uživatele o významných místech anebo o častých nehodách [25].

V případě, že vstupní data jsou sbírána jak při chůzi, jízdě na kole, autem, atd. je vhodné shlukovací proceduru provádět pro jednotlivé druhy pohybu [30], neboť dochází ke změně jednotlivých parametrů. Především při chůzi nebo jízdě na kole lze velmi snadno změnit směr pohybu viz. Obr. 7, proto je zapotřebí tyto změny eliminovat, aby nenastalo špatné vyhodnocení významných míst.



Obr. 7 Změna směru při chůzi

3.3. Algoritmus CLATIN

Algoritmus CLATIN [33] (Clustering Large Applications with Triangular Irregular Network) je efektivnější verzí *k-medoid* algoritmu. Hlavní kroky algoritmu lze definovat takto:

1) Inicializace
   - vybrání *k* reprezentativních objektů – počáteční body (medoids),
   - konstrukce TIN **Chyba! Nenalezen zdroj odkazů.** z těchto bodů.
2) Výpočet celkové ceny TC_{ih} pro všechny dvojice objektů O_i, O_h, kde O_i je současný medoid a O_h není medoid objekt.
3) Vybrání objektů O_i, O_h, které odpovídají *min* O_i, O_h (TC_{ih}). Jestliže minimum je záporné,
   - nahraď O_i s O_h,
   - aktualizuj TIN,
   - jdi zpět na krok 2,

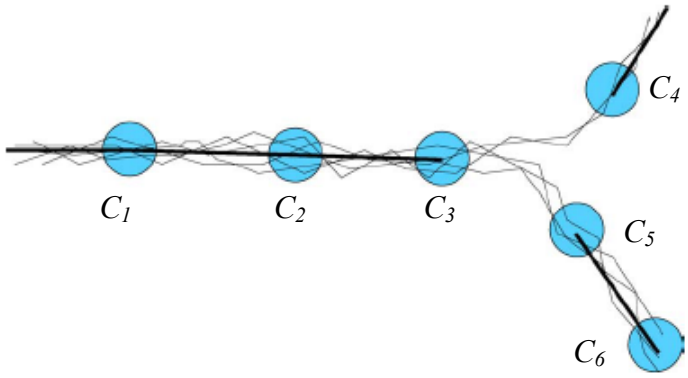
   jinak pro každý nevybraný objekt najdi nejvíce podobný objekt. Ukonči algoritmus.

Efektivita spočívá v tom, že je schopen dosáhnout stejných výsledků jako jeho předchůdci v kratším čase, proto je vhodný pro objemná data (např.: silniční infrastruktura). Tento algoritmus byl zatím vyzkoušen na jednoduchých datech, ale je plánováno jeho použití a případné doladění pro složitou silniční strukturu. Dále se pracuje na automatickém spojování jednotlivých shluků.

4. Schroedelova metoda

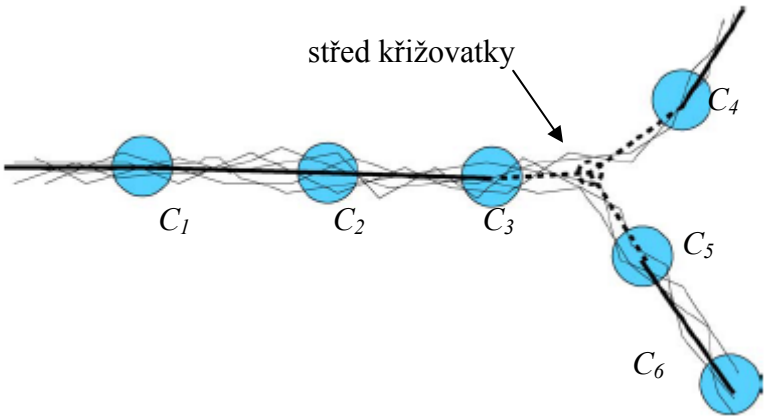
Tato metoda se používá k vytváření virtuální mapy ze získaných GPS bodů. Při vytváření virtuální mapy je vhodné si vytvořit důkladný rozbor shlukování. Podle Schroedla [1] je

potřebné si v prvním kroku rozdělit vstupní data do segmentů odpovídající jednotlivým úsekům cest. V ideálním případě by bylo vhodné v každém úseku mít jeden shluk, který splňuje požadované parametry – vzdálenost mezi jednotlivými shluky by měla být menší než je požadováno. Průměr shluku by měl být větší než nejširší cesta (aby nedocházelo k rozdělování cest) a zase by neměl překročit takovou vzdálenost, aby docházelo k slučování jednotlivých cest – průměrně se uvádí kolem 10–20 metrů [1]. Takto vytvořené shluky jsou podle Schroedla podobné algoritmu k-means. V dalším kroku je zapotřebí spojit vygenerované shluky tak jak je ukázáno na Obr. 8.



Obr. 8 Spojení center (označeno modrým kruhem) shluků (cluster seeds)

Kriterium spojení dvou shluků C_1 a C_2, kde C_1 předchází C_2, je takové, že všechny trasy patřící C_1 musí procházet C_2. Všechny dvojice shluků, které splňují tuto podmínku lze propojit křivkou (v případě, že chceme určit přesnější tvar silnice, můžeme aproximovat shluky metodou nejmenších čtverců – LS).



Obr. 9 Spojení shluků do křižovatky

Při spojování shluků bychom měli počítat s tím, že ve vstupních datech mohou být mezery způsobené např. ztrátou GPS signálu. Proto je důležité si stanovit určité toleranční pásmo a následně zjišťovat, zda všechny trasy patřící C_1 jdou do C_2. V jednoduchých infrastrukturách (např. data s jednou křižovatkou) můžeme tyto cesty úplně vypustit. Schroedl již neuvádí co by se stalo v případě, že šířka silnice je větší jak průměr shluku a jak efektivně vyhledávat jednotlivé cesty patřící k danému shluku. V posledním kroku je zapotřebí zjistit střed křižovatky a propojit jej s patřičnými shluky. Ke stanovení středu můžeme využít koncept *snake*, který hledá minimum z kvadrátu vzdáleností od vygenerovaného bodu – střed křižovatky a příslušnými středy shluků C_{3-5} (Obr. 9) anebo se střed křižovatky hledá pomocí všech bodů patřící shlukům C_{3-5}.

5. Zjišťování hustoty provozu

Z výzkumu [22] by bylo vhodné pro budoucí práci využít hlavní myšlenku na zjišťování hustoty provozu. Výzkum [22] se zabýval zjišťováním počtu nehod v konkrétní infrastruktuře silnic. Pro tento účel byly navrženy dvě shlukovací metody (*two-dimensional clustering* a *linear clustering*).

První metoda počítá hustotu nehod (D) tak, že pro danou část silnice (většinou kruhová oblast s poloměrem r) nejprve spočte počet nehod X_i, a poté vydělí danou plochou (1).

$$D = \frac{\sum_{i=1}^{n} X_i}{\pi r^2}. \qquad \textbf{(1)}$$

Druhá metoda využívá tzv. autokorelačního indexu (konkrétně Moran´s index) ke zjištění hustoty nehod. Podrobněji o těchto metodách viz. [22].

Pokud bychom zjišťovali místo počtu nehod v daném úseku průměrnou rychlost, mohli bychom získat přehled rychlostního rozložení na dané silnici. Abychom zjistili, kdy dochází ke zvýšení hustoty provozu během jednoho dne, je nutné provádět analýzu rychlostního rozložení pro několik časových intervalů (čím menší průměrná rychlost, tím větší hustota provozu).

6. Spojování shluků podle směru vektoru

Podle pánů Worralla a Nebota [23] je vhodné si definovat každý shluk jako vektor:

$$c_{\mathrm{i}} = \{x_i, y_i, \varphi\}, \qquad \textbf{(2)}$$

kde:

x_i, y_i jsou souřadnice středu shluku,

φ je směr vektoru.

Na Obr. 10 jsou znázorněny takto vypočítané shluky pro vstupní množinu dat (Obr. 11). Bohužel ve výzkumu těchto pánů se již neuvádí jak efektivně tento směr vypočítat, ani jak upravit shluky, když se prolínají oba jízdní směry a jsou vygenerovány pouze společné shluky.

Druhou fází je spojování jednotlivých shluků, každý shluk může mít maximálně jednoho předchůdce a jednoho následovníka. Pro daný shluk a nejbližší sousedy počítáme tzv. funkci vzdáleností - *metric*, která je definována následovně:

$$metric = \alpha \Delta S + \beta \phi + \gamma \Delta \varphi, \qquad \textbf{(3)}$$

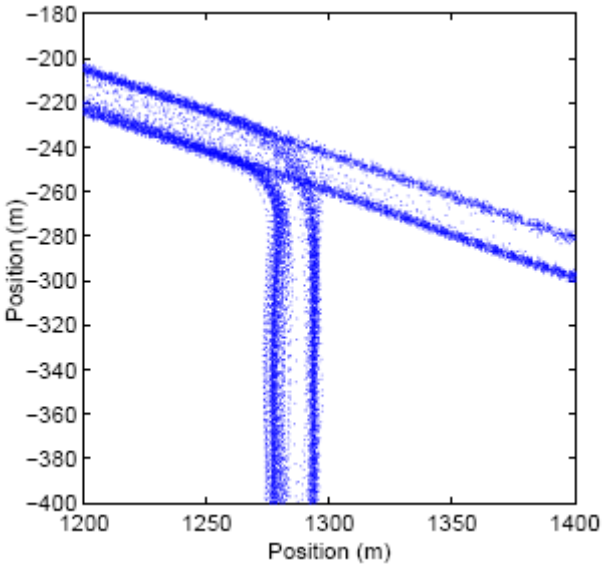
kde:

ΔS je vzdálenost mezi daným shlukem a jeho sousedem,

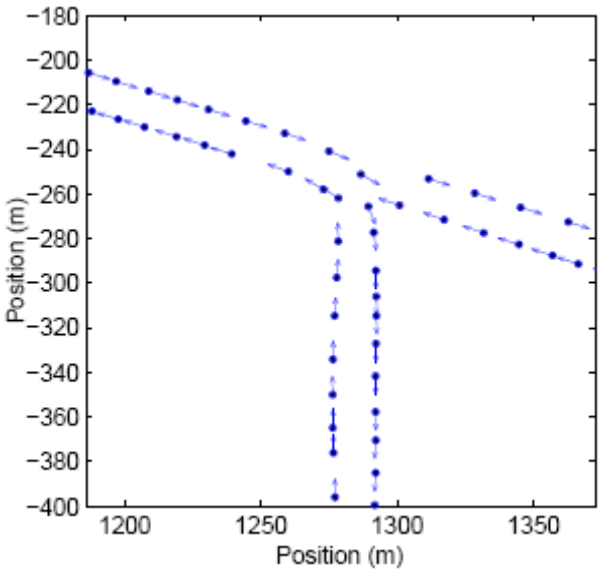
Δφ je rozdíl mezi směry shlukových vektorů,

α, β, γ jsou koeficienty jednotlivých proměnných.

Pozn.: koeficienty α, β, γ volíme inženýrským odhadem.



Obr. 11 Vstupní data



Obr. 10 Vygenerované směrové shluky

Nakonec ještě můžeme výsledné křivky aproximovat *metodou nejmenších čtverců*.

7. Zpřesňování virtuálních map

V případě použití diferenční GPS lze měřením získat velmi přesná data a následně zpřesnit komerční digitální mapu, u které se uvádí přesnost v jednotkách metrech.

Zpřesňování probíhá tak, že pomocí digitální mapy si rozdělíme vstupní data do konkrétních segmentů. Tyto segmenty odpovídají jednotlivým cestám. Dále pak pro každý segment vykonáváme tyto operace. Pro každý mapový bod m se směrodatnou odchylkou m_σ nalezneme nejbližší změřený bod n. Jeho směrodatná odchylka n_σ se vypočítává pomocí okolních GPS bodů, podrobněji v [26]. Nový mapový bod p se vypočte jako průměr bodů m a n. Jeho nová odchylka je

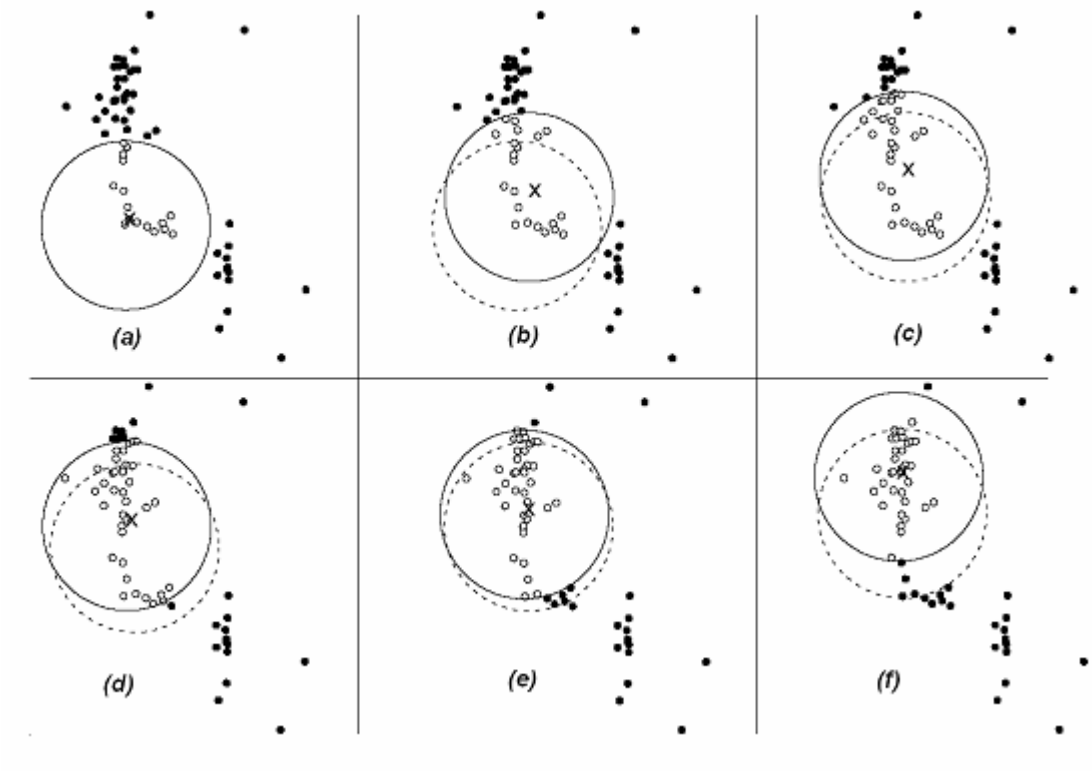
$$p_\sigma = \sqrt{\frac{m_\sigma^2 n_\sigma^2}{m_\sigma^2 + n_\sigma^2}} \,. \qquad \textbf{(4)}$$

Tento postup se provádí pro každou trasu zvlášť. Ve výsledku získáme novou tzv. *centerline* (středovou dělící čáru). Tento postup velmi pozvolna konverguje.

8. Prediktivní určování pohybu

Prediktivní určování polohy je vhodné např. pro uživatele, kteří si chtějí naplánovat schůzku. Předpokládá se, že znají základní informace o dané osobě (jméno,…).

V první řadě je zapotřebí rozpoznat často navštěvovaná místa konkrétní osobou. Na naměřená GPS data použijeme tento shlukovací algoritmus: zvolíme libovolný bod a poloměr kruhu. Označíme všechny body uvnitř kruhu a vypočteme z nich střední hodnotu (střed shluku). V případě, že se střední hodnota téměř nemění (Obr. 12 – *(e)*) tak všechny body uvnitř kruhu sloučíme do jednoho bodu (často navštěvované místo) viz. Obr. 12. Tyto sloučené body již neuvažujeme při dalším průběhu algoritmu. Nalezenému místu přiřadíme jednoznačný identifikátor. Algoritmus je ukončen až všechny body jsou přiřazeny k určitým místům. Poté následuje fáze přiřazení jednotlivým identifikátorům názvy (např.: domov, obchodní centrum, škola …). Toto přiřazení nelze provést bez znalostí od uživatelů. Nakonec se z jednotlivých míst sestrojí Markův model a dopočítají se pravděpodobnosti přechodu. Tento model je využíván k určování predikce [27].



Obr. 12 Ilustrace algoritmu (x znamená střed shluku, který je tvořen bílými body, body uvnitř čárkovaného shluku tvoří předešlý shluk)

Jiný prediktivní algoritmus lze nalézt v [28]. Jeho hlavní myšlenka je nalézt jednotlivé cesty mezi významnými místy a predikovat další pozici, kam se uživatel chystá. Predikce je odhadována z jednotlivých pravděpodobností konkrétních významných míst. Tato myšlenka se iterativně řeší pomocí dvou algoritmů, které lze nalézt v [28].

IV. Specifikace pro mapu

1. Úvod

Analýzu GPS dat jsem rozdělil na několik fází, které na sebe navazují. Prvotní fáze je pouze výběr dat pro analýzu z celkové vstupní množiny GPS bodů. Poté následuje filtrace, pomocí které dojde k odstranění nepřesných dat. V třetí fázi se z vyfiltrovaných dat vygenerují shluky, které se v následné fázi pospojují, a tím vznikne virtuální mapa. Přesnost analýzy si můžeme zkontrolovat na mapovém podkladu staženého z Google Maps. Podrobněji o Google mapách je v *kapitole V*.

2. Filtrace dat

Téměř všechna reálná data jsou zatížena chybou měření, a proto je nezbytně nutné tyto nepřesnosti v datech odstranit. Nepřesnosti neboli chyby z lokalizace dat vyplývají zvláště z nepřesného měření GPS způsobeného odlehlostmi geoidu a elipsoidu WGS 84 (World Geodetic System 1984), atmosférickými efekty, chybami ve výpočtu drah družic atp. Ukázka nepřesných dat je zobrazena na Obr. 1, kde jsou také vykreslena vyfiltrovaná data (zelená barva). Na Obr. 2 jsou tato data zobrazena na mapovém podkladu. Z obrázků plyne, že při jednom průjezdu vozidla došlo k nepřesnému uložení jeho GPS souřadnic.

Obr. 1 Originální a vyfiltrovaná data

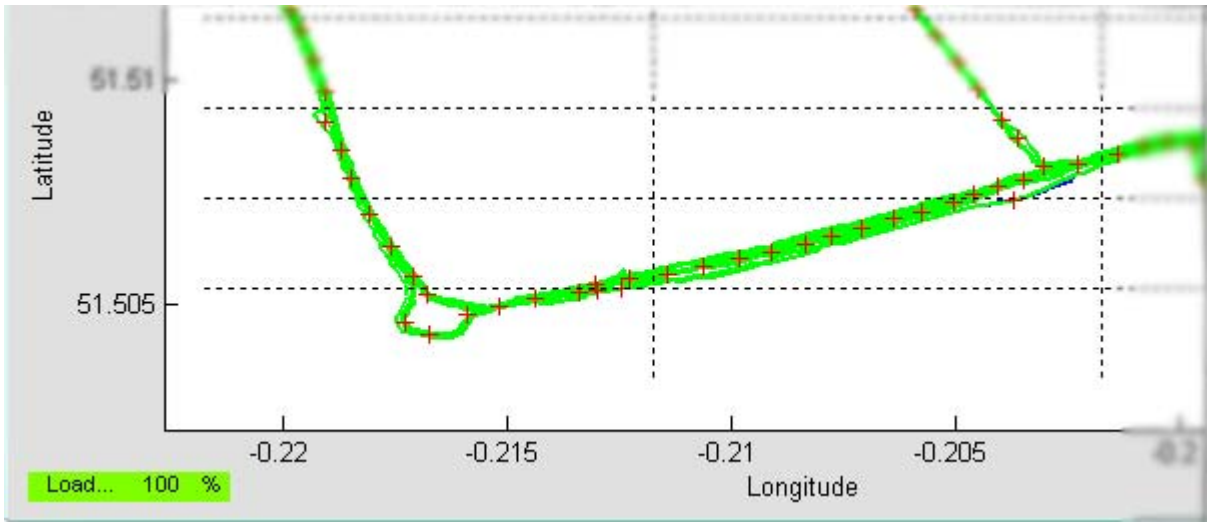
Obr. 2 Data na mapovém podkladu

Filtrovací metodu lze rozdělit na dvě části. V první části algoritmu se vyhledávají takové body, které jsou stejné a jsou bezprostředně změřené po sobě. Tyto body kromě prvního jsou označeny jako redundantní body, jsou odebrány ze vstupní množiny a již se s nimi nepracuje. Tyto body vznikají tehdy, když se vozidlo nepohybuje např. při čekání na křižovatce na zelenou. V druhé fázi algoritmu dochází k odstranění bodů, vzniklých chybou měření (tzv. outliers) a bohužel také cest, které byly navštívené jednou až dvakrát, protože nelze tyto cesty rozpoznat od chyb měření. Pro každý bod ze vstupní množiny hledáme v jeho okolí, které je určené konkrétním poloměrem, předem definovaný počet bodů. V případě, že nalezneme alespoň tolik bodů, kolik je požadováno, tak daný bod není označen jako outlier a rovněž není vyjmut ze vstupní množiny dat. Na předchozích dvou obrázcích a na následujícím jsou zobrazeny zelenou barvou výsledky po filtraci pro reálná data. Na Obr. 3 je možné vidět, co se stane, když špatně nastavíme vstupní parametry algoritmu (poloměr a počet bodů v kruhu). V tomto případě dojde jen k částečnému odfiltrování nepřesností. Bohužel může nastat i takový případ, že při změně parametrů zase nedojde k odfiltrování chyb měření, ale některých cest. V tomto případě vstupní data obsahují velké množství nepřesných bodů, a proto doporučuji vstupní data rozdělit na několik částí a pro každou část provádět analýzu. Tímto postupem dosáhneme toho, že můžeme pro jednotlivé části nastavovat různé vstupní parametry.

Obr. 3 Špatně a dobře vyfiltrovaná data

3. Generování shluků

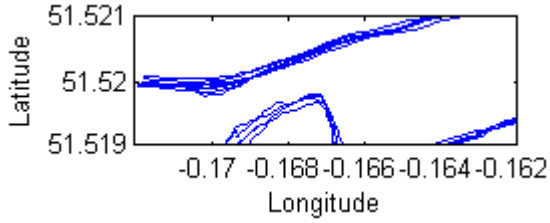
Z časového hlediska je efektivnější generovat shluky pro část dat než rovnou aplikovat metodu na celá vstupní data, proto si vstupní data rozděluji do obdélníkové sítě a v každém obdélníku aplikuji danou shlukovou metodu. Ukázka obdélníkové sítě je zobrazena na Obr. 4 čárkovanou černou čarou. Výhoda tohoto generování spočívá v tom, že méně častěji dochází k špatnému vygenerování shluků pro cesty, které vedou blízko u sebe.



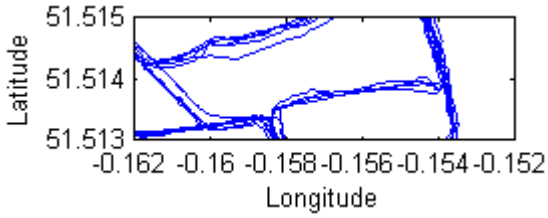
Obr. 4 Ukázka obdélníkové sítě

3.1. Optimální počet shluků

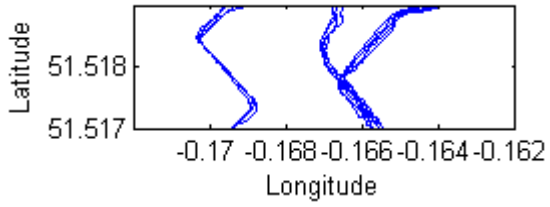
Při použití nehierarchických metod je zapotřebí před výpočtem znát počet shluků. Jak stanovit tento optimální počet není pevně stanoveno. Existuje několik indexů, pomocí kterých lze stanovit počet shluků (ne vždy optimální). Pro fuzzy metody jsem porovnal těchto šest indexů: Partition coefficient (PC), Classification Entropy (CE), Partition index (SC), Separation index (S), Xie and Beni index (XB), Dunn index (DI) a Alternative Dunn index (ADI). Podrobněji o těchto indexech lze nalézt v [5]. Poslední dva indexy lze použít i pro zjištění počtu shluků pro metodu *K-means*. Jednotlivé indexy jsem srovnával na vstupních datech viz. Obr. 5–9. Hodnoty daných indexů závisí na počtu shluků. Počet shluků jsem měnil od dvou do čtrnácti.



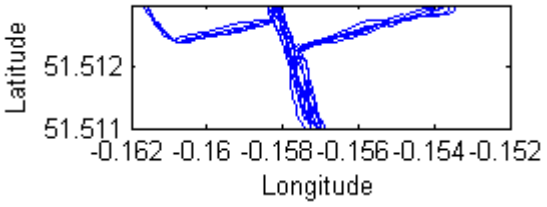
Obr. 5 Data 1



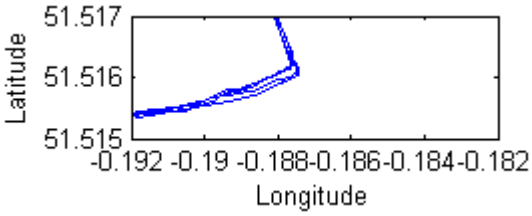
Obr. 6 Data 3



Obr. 7 Data 2

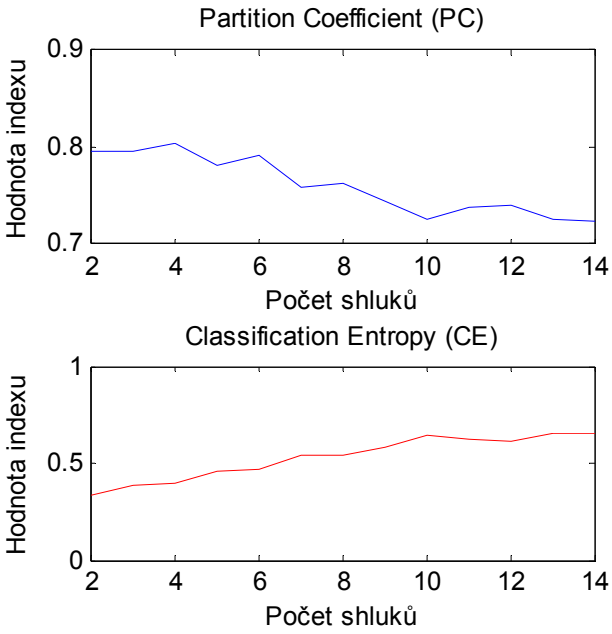


Obr. 8 Data 4

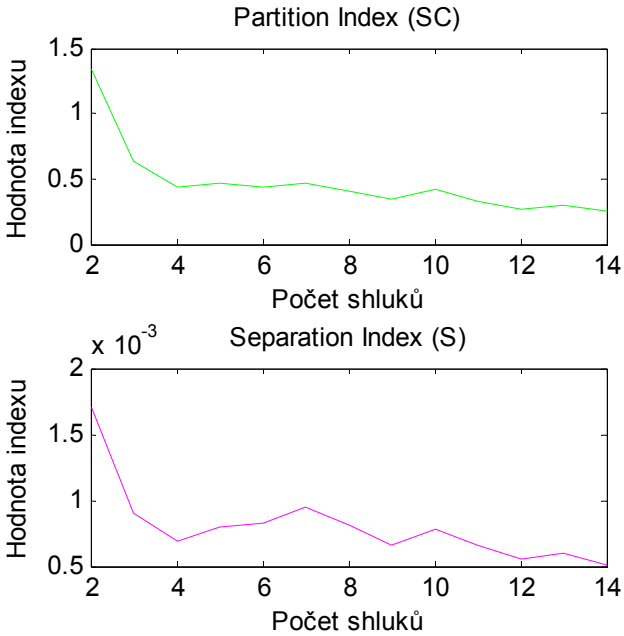


Obr. 9 Data 5

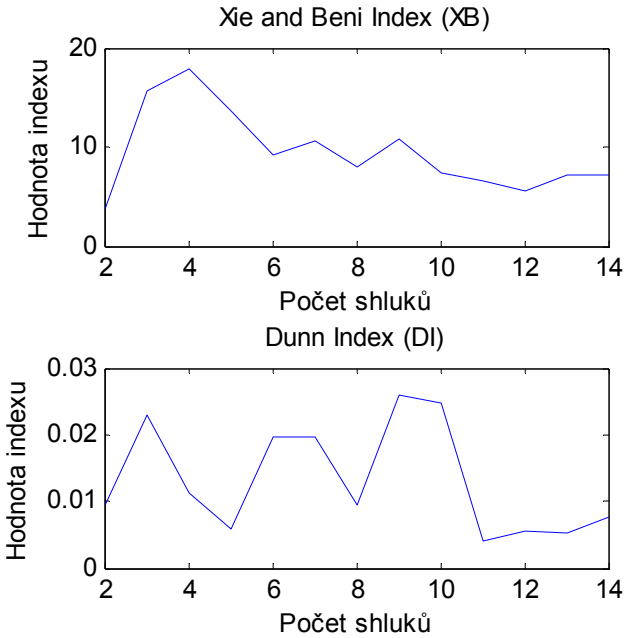
Na Obr. 10–12 jsou znázorněny výsledky pro **Data 1** při použití algoritmu *Fuzzy C-means*. Vstupní data jsou normalizovaná na hodnoty [0, 1] kvůli lepší efektivitě indexů [5].



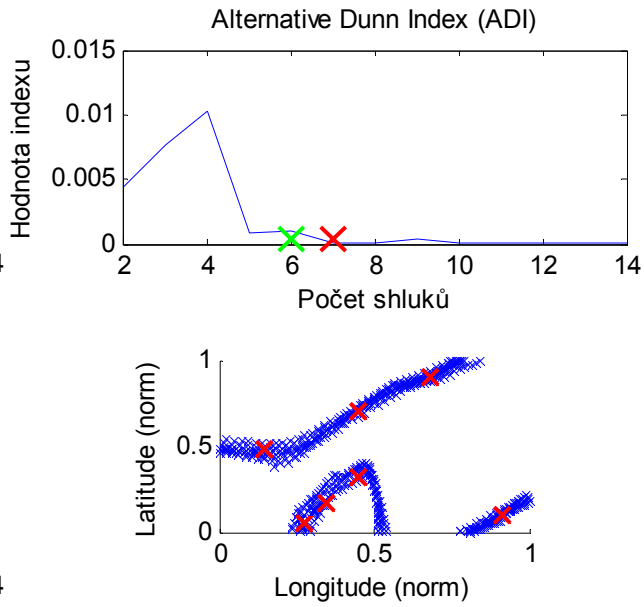
Obr. 10 Hodnoty indexů PC a CE



Obr. 11 Hodnoty indexů SC, S



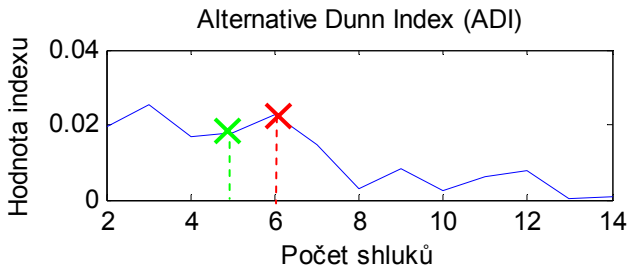
Obr. 12 Hodnoty indexů XB a DI



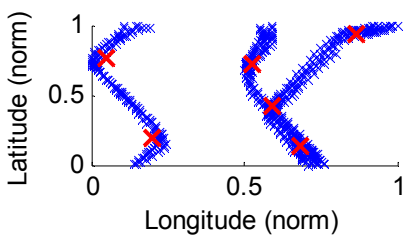
Obr. 13 Index ADI a sedm vygenerovaných shluků

Z prvních dvou indexů Obr. 10 je patrné, že pro vstupní data nelze efektivně stanovit počet shluků, neboť hodnota PC monotónně klesá a u indexu CE monotónně roste. Pro další dva indexy (SC a S) se hodnota velmi rychle ustálí, což by ve výsledku znamenalo splynutí jednotlivých cest. Indexy XB a DI se vůbec neustálily, proto jsou také nevhodné pro výpočet počtu shluků. Jako nejlepší index pro GPS data se jeví ADI, který se ustálil na počtu šesti shluků (zelený křížek). Jako optimální počet shluků beru až následující hodnotu (červený křížek). Výsledné shluky jsou znázorněny na Obr. 13. Za ustálenou hodnotu považuji takovou hodnotu, kde rozdíl mezi dvěma následujícími hodnotami nepřesáhne hodnotu indexu 0,0015.

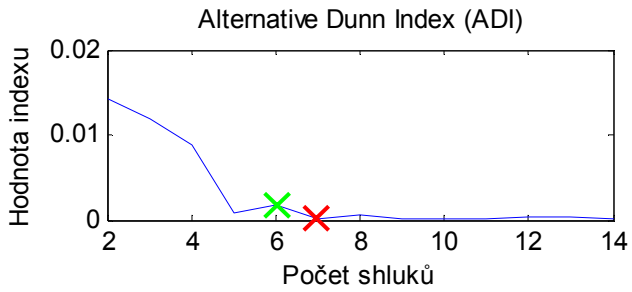
Pro data Obr. 6–9 vychází hodnoty indexů obdobně, proto zde uvádím jen hodnoty indexu ADI a výsledné vygenerované shluky.



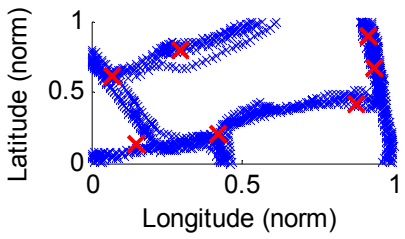
Obr. 14 Hodnota indexu ADI pro Data 2



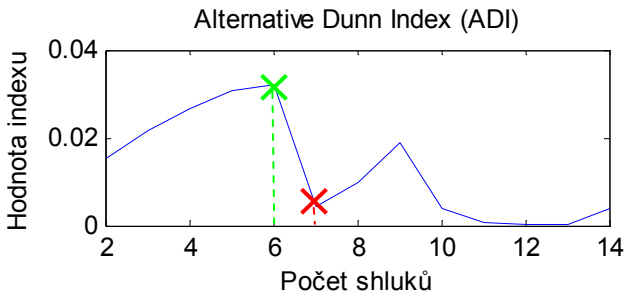
Obr. 15 Šest shluků pro Data 2



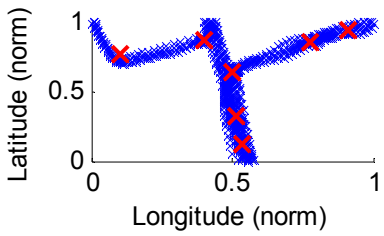
Obr. 16 Hodnota indexu ADI pro Data 3



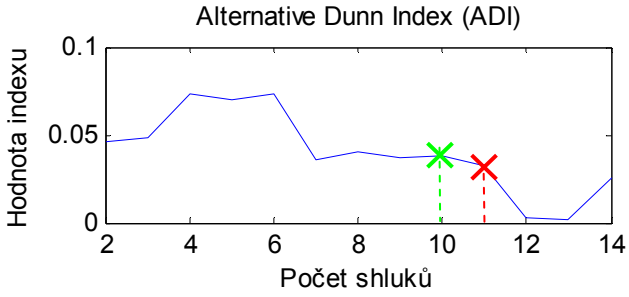
Obr. 17 Sedm shluků pro Data 3



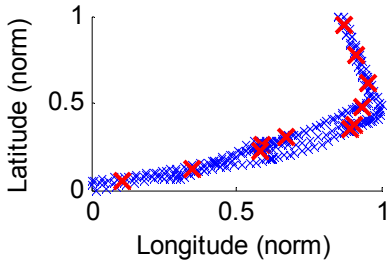
Obr. 18 Hodnota indexu ADI pro Data 4



Obr. 19 Sedm shluků pro Data 4



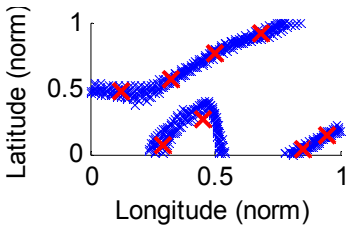
Obr. 20 Hodnota indexu ADI pro Data 5



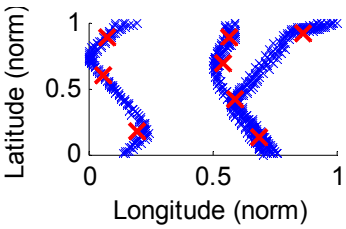
Obr. 21 11 shluků pro Data 5

Z obrázků viz. výše plyne, že ne vždy dojde k ustálení hodnoty indexu. I v tomto případě beru jako optimální počet shluků následující hodnotu, poté co rozdíl mezi sousedními hodnotami nepřesáhne požadovanou hodnotu např. Obr. 18 a 19.

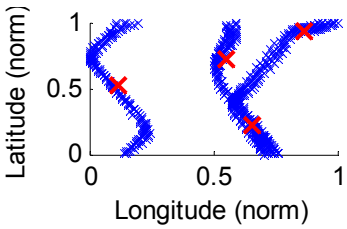
Pro algoritmus *K-means* je také vhodné použít ADI index na zjišťování optimálního počtu shluků, neboť z ostatních možných indexů není patrné, kdy dochází k ustálení (obdobně jako u fuzzy shlukování). Výsledné shluky pro Data 1–5 jsou vykreslena na Obr. 22–27.



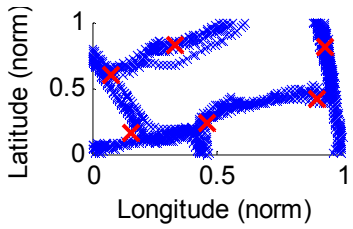
Obr. 22 Shluky pro Data 1



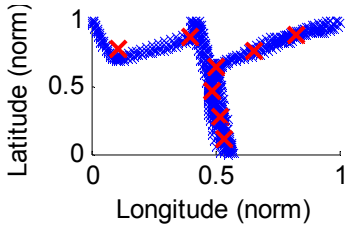
Obr. 23 Shluky pro Data 2



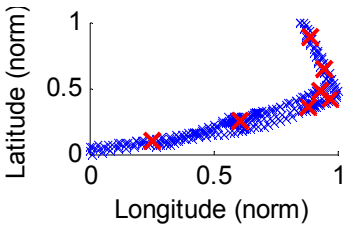
Obr. 24 Shluky pro Data 2



Obr. 25 Shluky pro Data 3



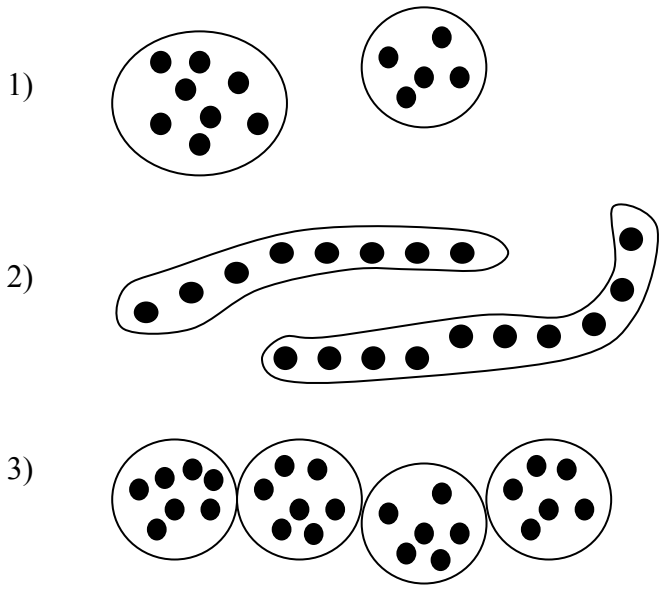
Obr. 26 Shluky pro Data 4



Obr. 27 Shluky pro Data 5

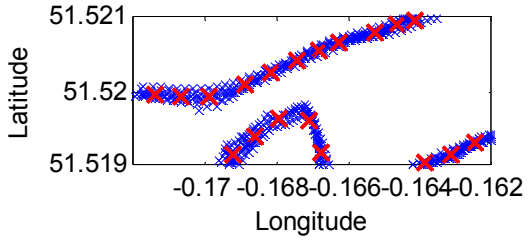
Nevýhoda algoritmu *K-means* spočívá v tom, že je nedeterministický. Jeho nedeterminičnost ovlivňuje výsledný počet shluků. Výsledek lze vidět na Obr. 23–24, kde na Obr. 24 došlo při opětovném spuštění algoritmu k špatnému vygenerování shluků. Pro správné řešení je nezbytné tento proces spustit několikrát a porovnat výsledky.

Pro aglomerativní hierarchické shlukování je nutné určit správný koeficient nepodobnosti objektů a hranici, kde uříznout strom v dendrogramu. Tato hranice závisí především na tom, jak jsou vstupní data daleko od sebe. Tento algoritmus není vhodné používat na datech, která nejsou přibližně stejně hustě rozložena, protože by mohlo dojít ke špatnému generování shluků neboli ke ztrátě detailů v části, kde data jsou více nahromaděná. Existuje několik typických shluků Obr. 28, pro které již byli nalezeny nejvhodnější koeficienty nepodobnosti objektů. Pro první typ se používá metoda průměrné nepodobnosti objektů, pro druhý typ se používá metoda nejbližšího souseda a pro poslední typ se využívá metoda nejvzdálenějšího souseda. Z Obr. 28 plyne, že pro náš případ použijeme metodu nejvzdálenějšího souseda.

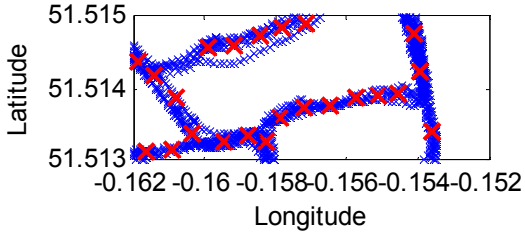


Obr. 28 Typy shluků: (1) Kulovitý, (2) Podlouhlý a (3) Složený („špatně oddělený“)

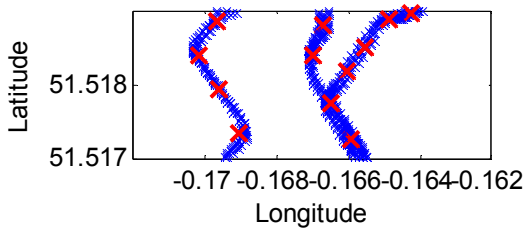
Zobrazíme-li si jednotlivé dendrogramy pro vstupní data (Obr. 5–9), zjistíme, že je vhodné volit hranici ve stromu kolem jedné tisíciny. Pro tuto hranici jsou výsledné shluky zobrazeny na Obr. 29–33.



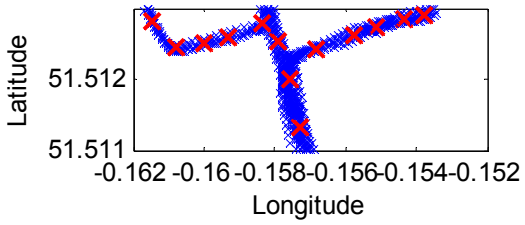
Obr. 29 Shluky pro Data 1



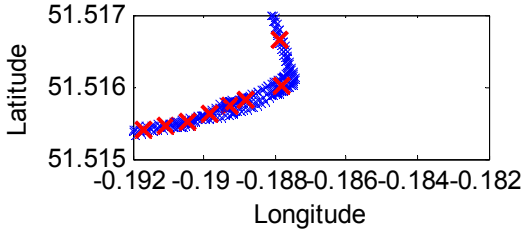
Obr. 30 Shluky pro Data 3



Obr. 31 Shluky pro Data 2



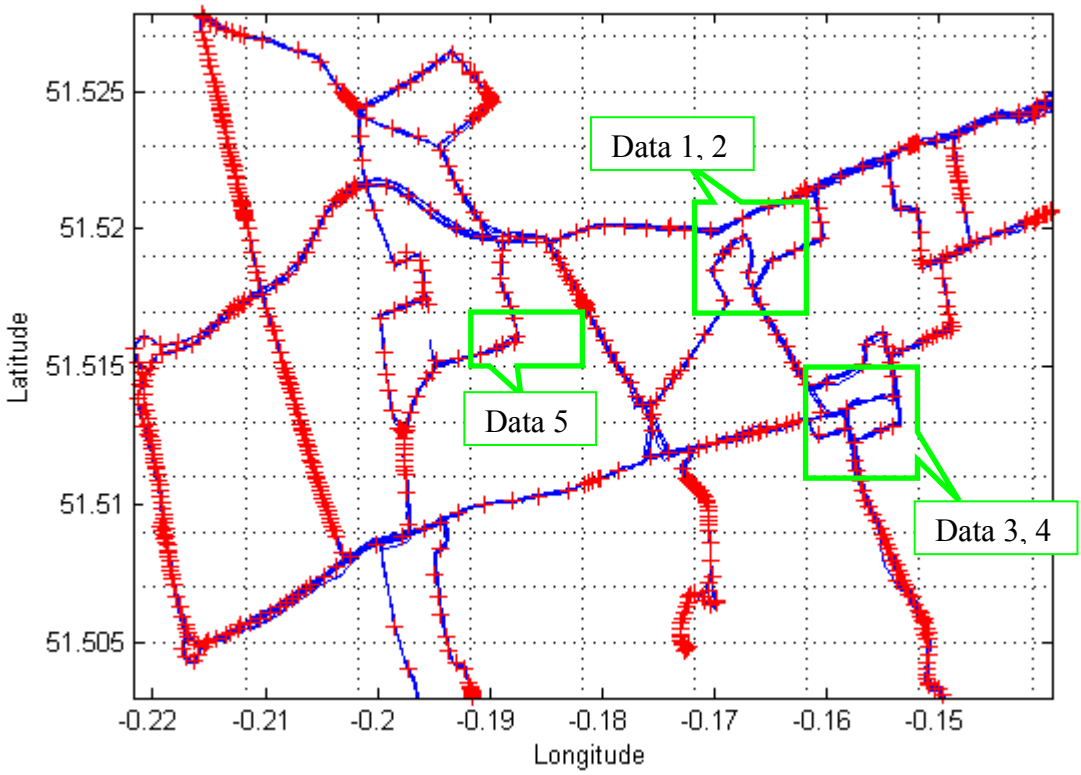
Obr. 32 Shluky pro Data 4



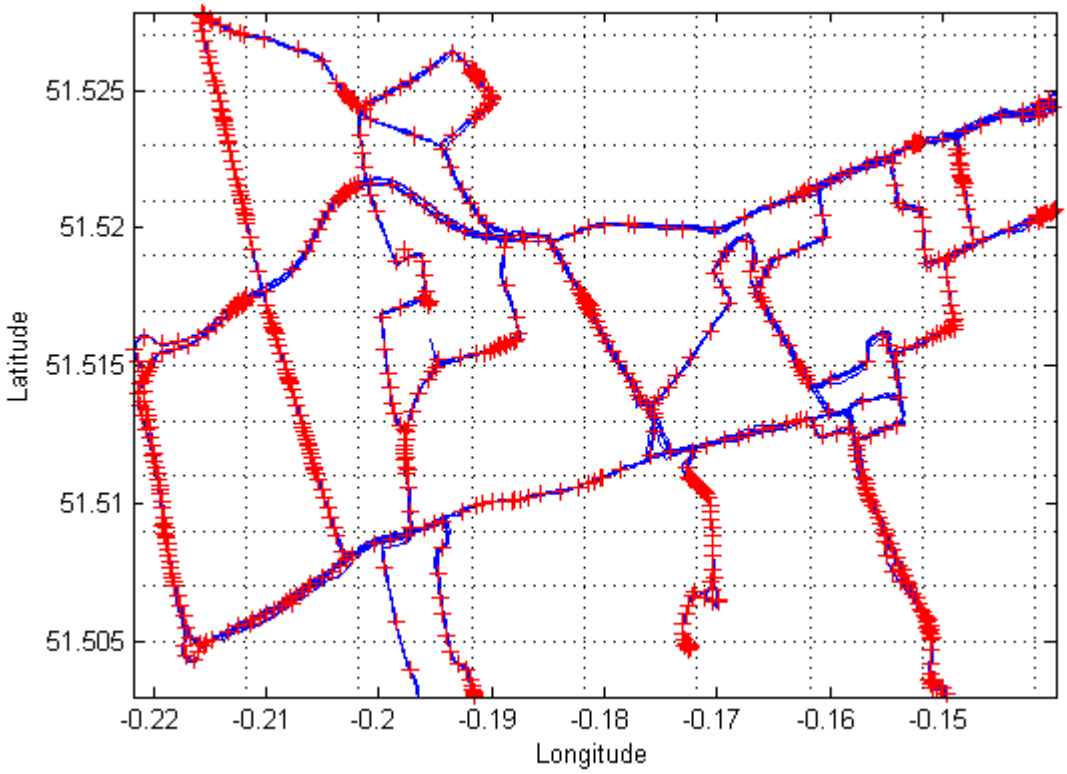
Obr. 33 Shluky pro Data 5

Z vygenerovaných shluků pomocí hierarchického shlukování je vidět plynulé rozložení shluků na jednotlivých cestách. Další výhoda od předchozích metod je v tom, že nedošlo k vygenerování přebytečných shluků jako je tomu na Obr. 27 (v ohybu cesty). Menší nevýhoda tohoto algoritmu je, že ne vždy dojde k vygenerování shluku poblíž křižovatky.

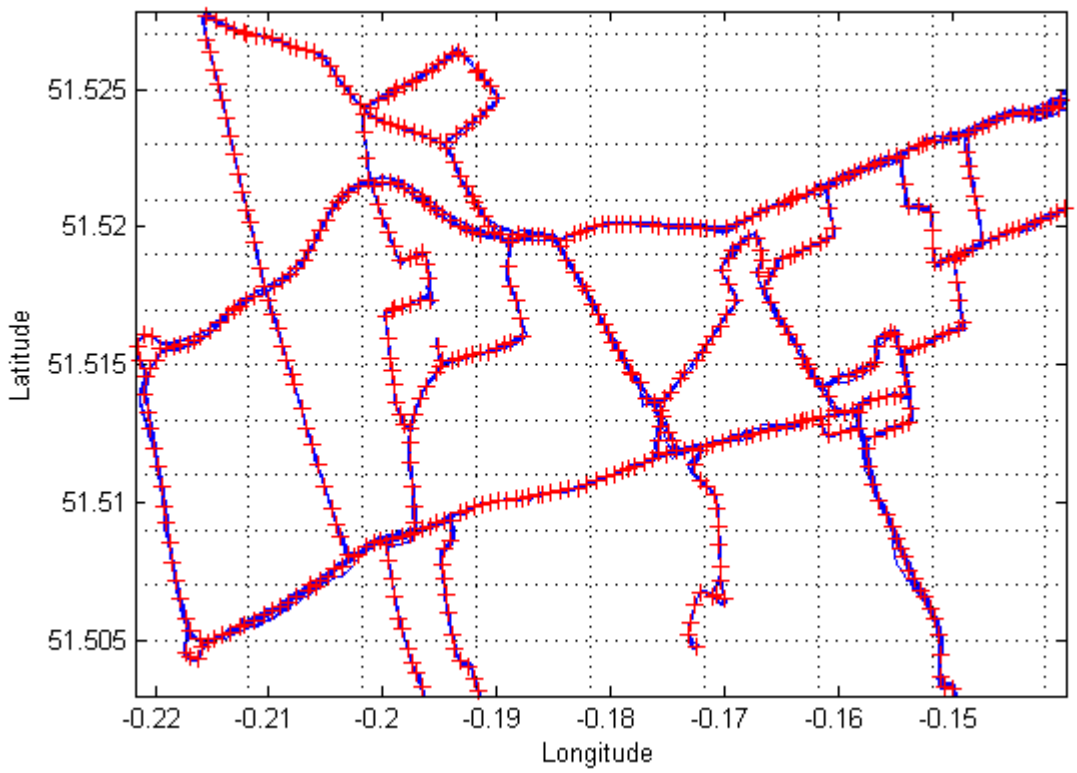
Na následujících obrázcích jsou zobrazeny vygenerované shluky pro složitější silniční infrastrukturu při použití předešlých shlukovacích metod. Jako nejlepší metoda pro generování shluků se jeví aglomerativní hierarchické shlukování, které plynule rozprostírá shluky na silniční strukturu Obr. 36. Zbylé dvě metody generují v některých částech infrastruktury přebytečné shluky, což je způsobeno neustálením indexu a špatným určením optimálního počtu shluků pro danou oblast. V tomto případě se bere jako optimální hodnota maximální možný počet shluků.



Obr. 34 Vygenerované shluky při použití algoritmu *Fuzzy C-means*



Obr. 35 Vygenerované shluky při použití algoritmu *K-means*



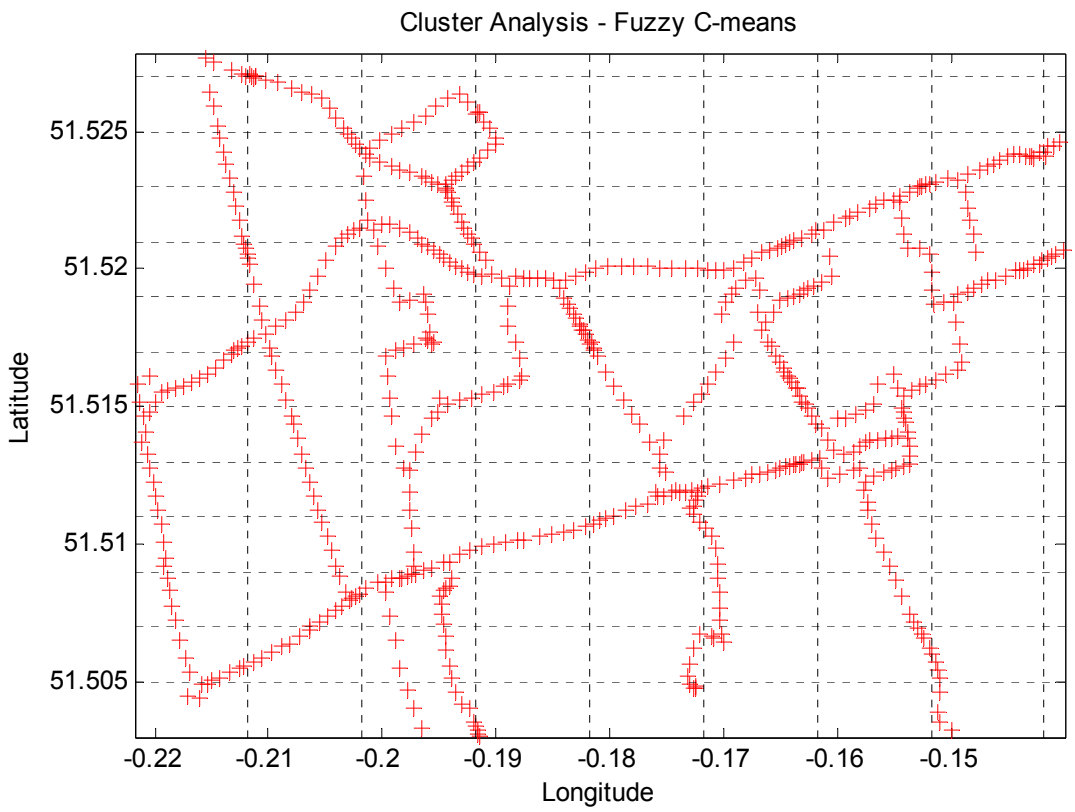
Obr. 36 Vygenerované shluky při použití aglomerativního hierarchického shlukování

Abychom i pomocí nehierarchického shlukování dosáhli obdobných výsledků jako u aglomerativního hierarchického shlukování, je zapotřebí navrhnout nový index pro vyhledávání optimálního počtu shluků. Při návrhu jsem využil poznatků z hierarchického shlukování, kde klíčovým parametrem je vzdálenost mezi jednotlivými body. Nový index vypočítává v Euklidovském prostoru průměrnou hodnotu kvadrátu vzdálenosti mezi vygenerovanými shluky a body patřícími do konkrétního shluku (24). Počet generovaných shluků se mění od zadané minimální hodnoty shluků až do té doby, než je splněný požadavek na průměrnou hodnotu anebo výpočet končí při dosažení maximální hodnoty shluků pro danou oblast. V tomto případě se ještě zkouší proces zopakovat s dvojnásobnou průměrnou hodnotou a začíná se od dvounásobného minimálního počtu shluků. Pokud ani v tomto případě není nalezeno řešení, tak jako optimální počet shluků se bere maximální možný.

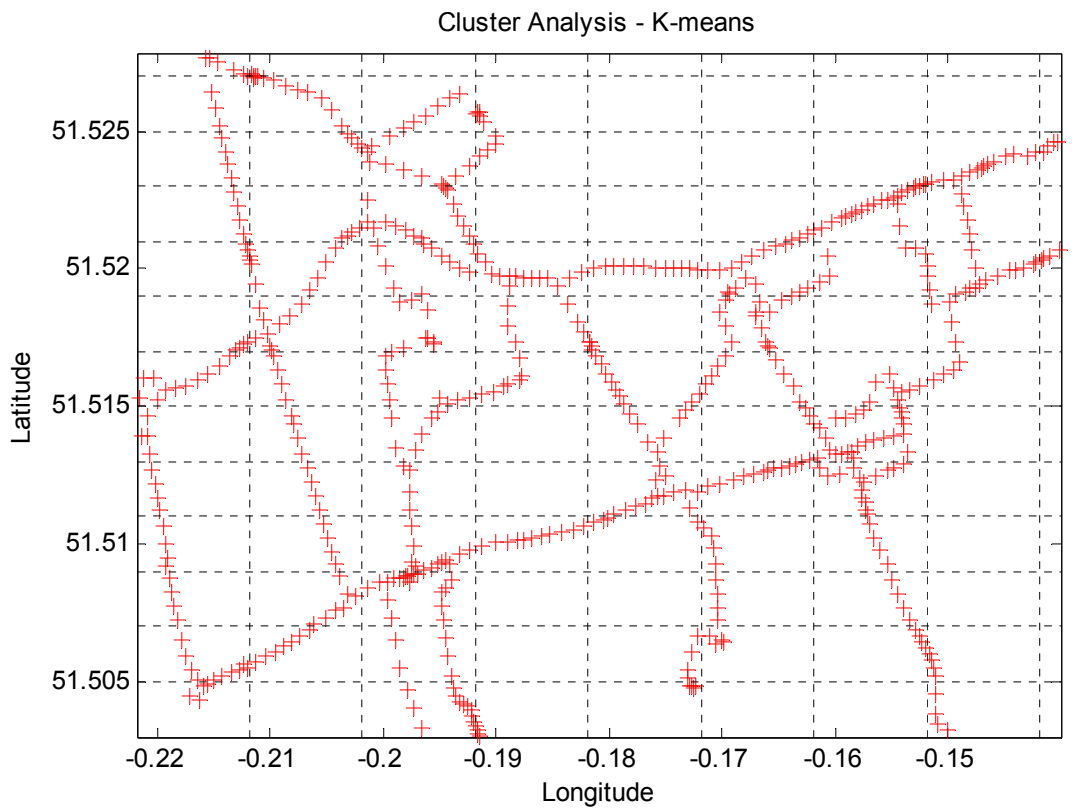
$$Index(\boldsymbol{X};\boldsymbol{V}) = \sum_{i=1}^{c}\sum_{k=1}^{N}\frac{1}{N}\left(x_j - v_i\right)^2 , \qquad \textbf{(24)}$$

kde význam symbolů je stejný jako u (12).

Vygenerované shluky při použití tohoto indexu jsou na Obr. 37–38.



Obr. 37 Vygenerované shluky při použití algoritmu *Fuzzy C-means*



Obr. 38 Vygenerované shluky při použití algoritmu *K-means*

3.2. *Výpočetní náročnost algoritmů*

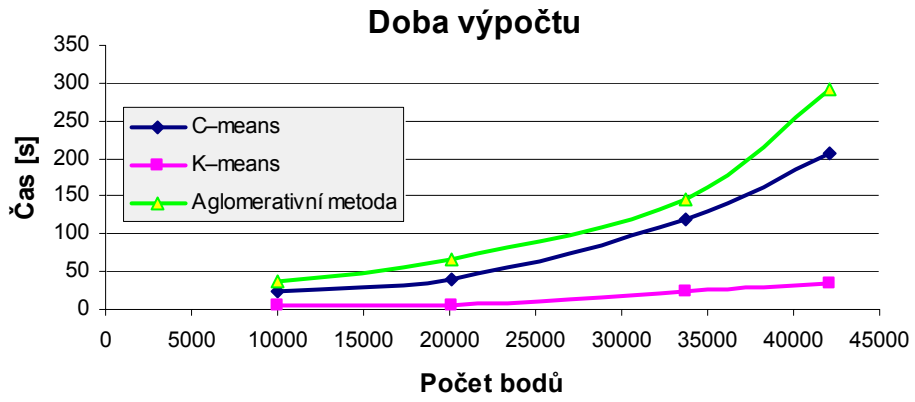
Počet bodů	*C-means* [s]	Počet shluků	*K-means* [s]	Počet shluků	*Aglomerativní metoda* [s]	Počet shluků
516	0,4	7	0,46	9	1,97	8
786	10,18	19	1,75	23	1,57	19
977	1,86	11	0,69	13	3,39	12
1901	31,47	26	3,09	25	26,2	23

Tabulka 1 Časová náročnost shlukových metod

Z Tabulky 1 plyne, že čas potřebný k vygenerování shluků ze vstupních dat, na které lze přímo použít shlukové metody, nezávisí jen na počtu těchto dat, ale i na počtu vytvořených shluků. Pro data, která si rozdělujeme do obdélníkové sítě a následně na každou část aplikujeme shlukovou metodu, je doba výpočtu patrná z Tabulky 2 a Obr. 39. Z výsledků plyne, že nejrychlejší metoda je K-means.

Počet bodů	*C-means* [s]	*K-means* [s]	*Aglomerativní metoda* [s]
10000	22,57	4,1	37,92
20152	39,79	6,37	67,61
33677	119,32	22,59	144,91
42096	207,49	34,42	292,14

Tabulka 2 Časová náročnost shlukových metod pro složitější infrastrukturu

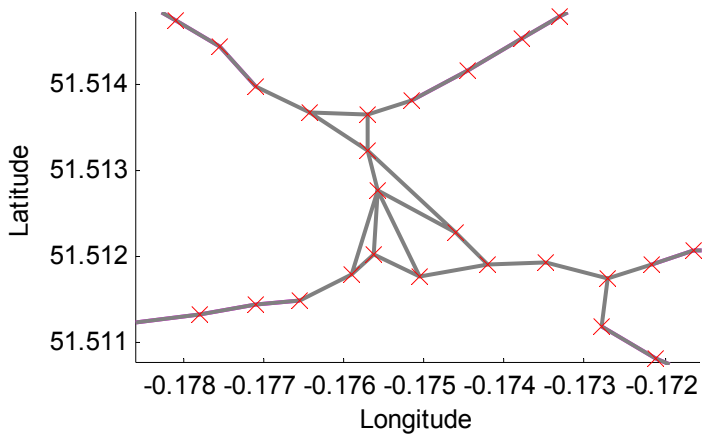


Obr. 39 Doba výpočtu

Nejvíce náročnou metodou na měření je hierarchická metoda, protože musí vypočítávat pro každou dvojici dat koeficient nepodobnosti, a proto při jejím použití je nutné volit správně parametry pro obdélníkovou síť, aby nedošlo k překročení paměti v paměťovém prostoru Matlabu. Pokud dojde k překročení paměti, tak pro danou oblast se nevytvoří shluky. Chceme-li pro tuto oblast zobrazit shluky, musíme zmenšit parametry pro obdélníkovou síť a znovu spustit shlukovací metodu.

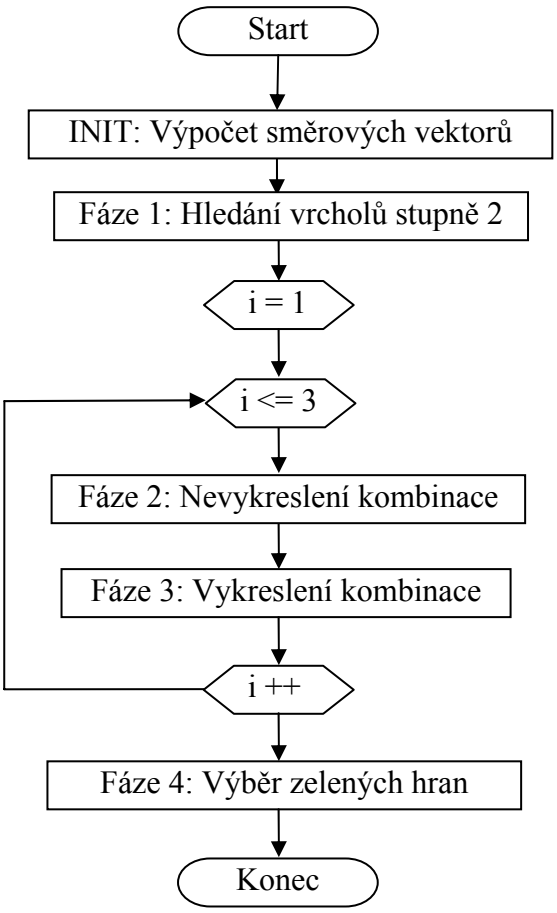
4. Spojovací (linkage) metoda

Poslední část analýzy spočívá ve spojení vygenerovaných shluků a vytvoření mapy oblasti. Vstupní data lze popsat pomocí neorientovaného grafu, kde jednotlivé vrcholy grafu jsou dány vygenerovanými shluky a hrany mezi vrcholy odpovídají všem souslednostem mezi body patřícími do různých shluků viz. Obr. 40. Jednotlivé hrany se vytvářejí tak, že při generování shluků se ukládá informace o tom, který bod patří do kterého shluku a díky tomu, že známe následnost naměřených dat, lze podle jízdy najít potřebné kombinace neboli informaci o tom, z kterého shluku a do kterého shluku se vozidlo přesunulo.



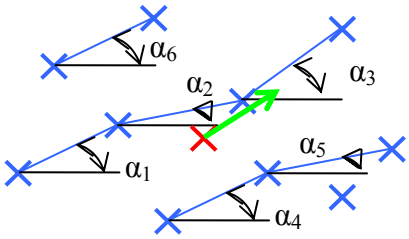
Obr. 40 Vstupní data – neorientovaný graf

Mapa oblasti je tedy dána vrcholy grafu a podmnožinou hran, jenž hledá následující algoritmus. Tento spojovací algoritmus lze rozdělit na několik fází viz. Obr. 41. U každé fáze se obarvují hrany barvami a to tak, že hrany obarvené zelenou barvou reprezentují cesty, které budou tvořit virtuální mapu a hrany obarvené modrou barvou jsou považovány za přebytečné cesty a ve výsledku nebudou vykresleny.



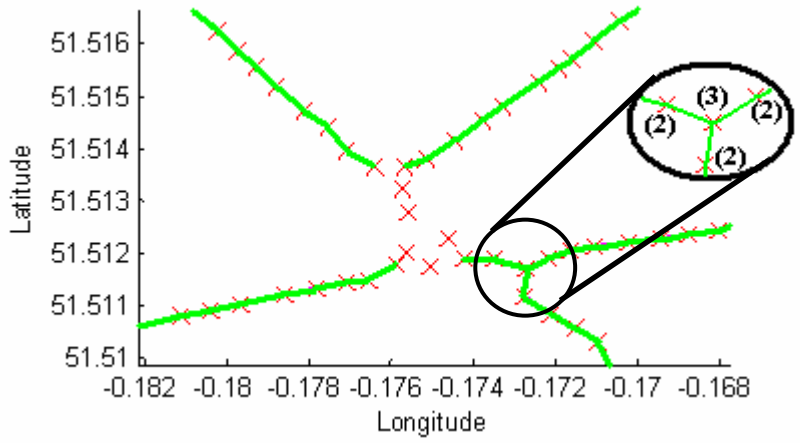
Obr. 41 Vývojový diagram

V rámci inicializace algoritmu dochází k výpočtu směrových vektorů vrcholů (shluků). Směrový vektor ke každému shluku se vypočítává tak, že pro dva po sobě následující body náležící danému shluku se vypočte směrový úhel viz. Obr. 42. Průměrováním všech úhlů získáme úhel pro vektor k danému shluku. Délka vektoru je brána jako jednotková. Výsledný vektor pro shluk (červený křížek) je na Obr. 42 zobrazen zelenou barvou. V případě, že pro daný shluk jsou vypočteny dílčí směrové úhly z různých kvadrantů, tak výsledný úhel se počítá pouze z kvadrantu, ve kterém je nejvíce vypočtených úhlů.



Obr. 42 Ukázka dílčích směrových vektorů

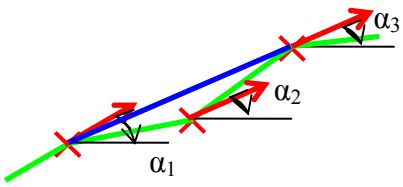
První fáze algoritmu hledá takové vrcholy, které mají stupeň roven dvěma, to znamená, že daný vrchol má pouze dva sousední vrcholy neboli vrchol je spojen pouze dvěma hranami. V případě nalezení takovýchto vrcholů, dojde k obarvení hran, které jsou spojeny s tímto vrcholem zelenou barvou. Ukázka této fáze se nachází na Obr. 43. V této fázi dojde i ke spojení jednoduchých křižovatek, na Obr. 43 označeno v kroužku, protože shluky, z kterých vede cesta do shluku, který označuje křižovatku, mají pouze dva sousední vrcholy.



Obr. 43 Spojení shluků po druhé fázi *„Hledání vrcholů stupně 2“*

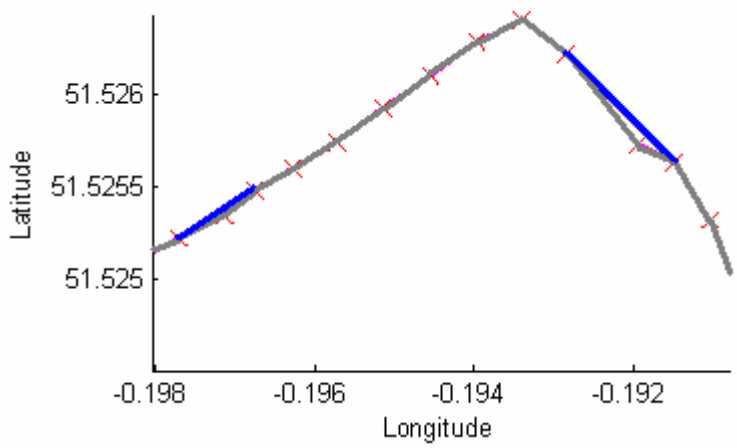
V dalších dvou fázích výpočtu dochází k výslednému dobarvení grafu. Tyto dvě fáze *„Vykreslení kombinace“* a *„Nevykreslení kombinace“* se postupně střídají a mohou obarvit pouze hrany, které ještě nejsou obarvené.

Nejprve začíná fáze *„Nevykreslení kombinace“*. V této fázi se hledají takové vrcholy, které mají být spojeny do trojúhelníku. V případě, že se naleznou patřičné vrcholy a již dvě hrany spojující tyto vrcholy jsou obarveny zelenou barvou, tak ke každému vrcholu se dopočítá opačný vektor a jako výsledný směrový vektor vrcholu se vezme ten, který má menší úhel. Tento výběr se provádí kvůli tomu, abychom docílili toho, že vozidla mezi danými shluky jezdila stejným směrem. Dále pak se zjišťuje jaká je úhlová odchylka mezi každou dvojicí směrových vektorů těchto tří vrcholů Obr. 44. Pokud jsou všechny odchylky



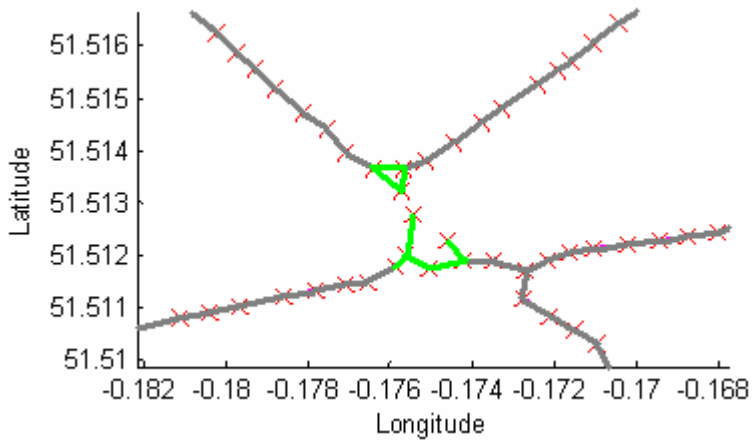
Obr. 44 Hledání třetí hrany trojúhelníka

menší než vstupní koeficient *difAngle*, tak třetí hrana trojúhelníku je označena modrou barvou. Na Obr. 45 je ukázka této fáze. Modrou barvou jsou zobrazeny kombinace, které byly označeny a ve výsledné mapě se nebudou vykreslovat.



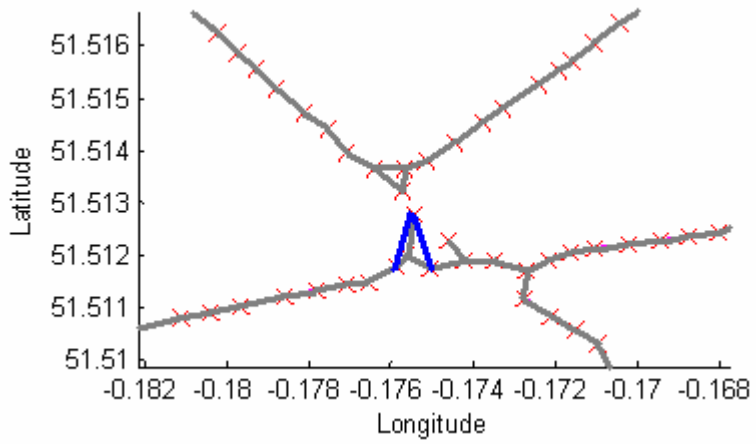
Obr. 45 Ukázka fáze *„Nevykreslení kombinace“*

Následuje fáze *„Vykreslení kombinace“*, která pro neobarvené hrany zjišťuje, zda je úhlová odchylka dvou vrcholů, které spojuje neobarvená hrana menší než koeficient *difAngle2*. Pokud ano a daná cesta mezi shluky byla projeta alespoň tolikrát, kolik udává koeficient *countWay*, tak tato hrana bude zahrnuta do množiny přímek, které vytváří virtuální mapu, a je obarvena zelenou barvou. Na Obr. 46 je možné vidět, které přímky budou vykresleny touto fází.



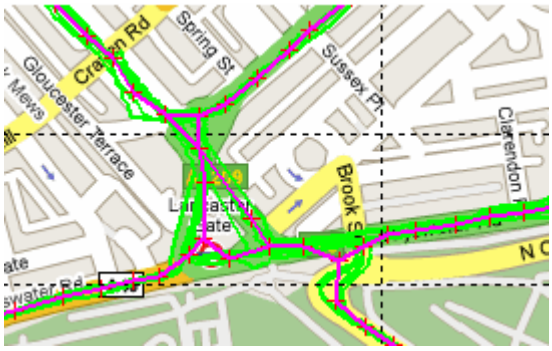
Obr. 46 Ukázka fáze *„Vykreslení kombinace“*

V každé iteraci dojde ke změně vstupních koeficientů těchto fází. Koeficient *difAngle* fáze *„Nevykreslení kombinace“* se postupně zvyšuje a při posledním průchodu je již tak velký, že neovlivní výsledek této fáze (90°). Na Obr. 47 je ukázka fáze *„Nevykreslení kombinace“* po druhé iteraci. Koeficient *countWay* fáze *„Vykreslení kombinace“* je stejný pro všechny iterace kromě poslední, kde je nulový a tedy na něm nezáleží. Koeficient *difAngle2* postupně roste až do poslední iterace a začíná na hodnotě rovné čtyřnásobku počáteční hodnoty *difAngle*. V poslední iteraci je jeho hodnota třetinová oproti původní – v této fázi se hledají takové cesty, které mají malou odchylku ve směrových vektorech a byly alespoň jednou projety.



Obr. 47 Ukázka fáze *„Nevykreslení kombinace“* – druhá v pořadí

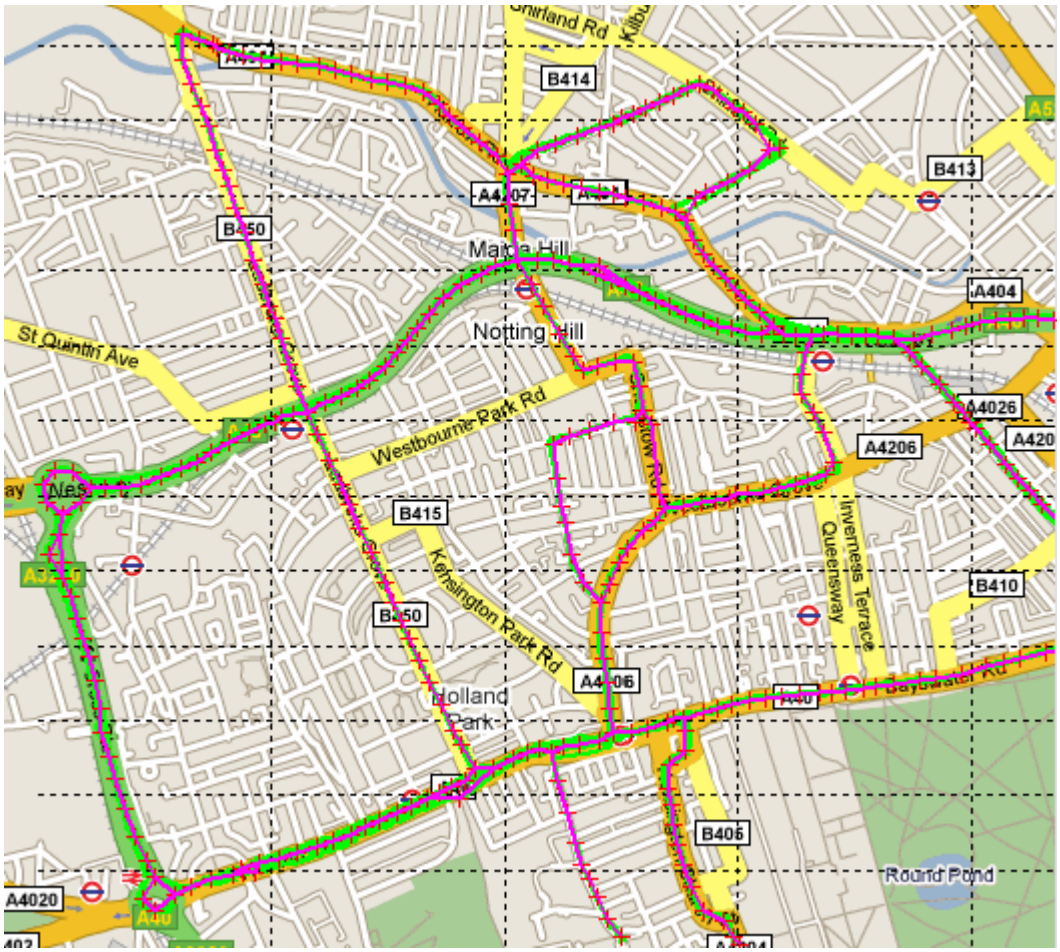
Poslední fáze algoritmu pouze vybere hrany obarvené zelenou barvou a zobrazí je. Na Obr. 48 je zobrazen výsledný graf nalezené mapy pro ukázkovou křižovatku.



Obr. 48 Ukázková křižovatka na mapovém podkladu

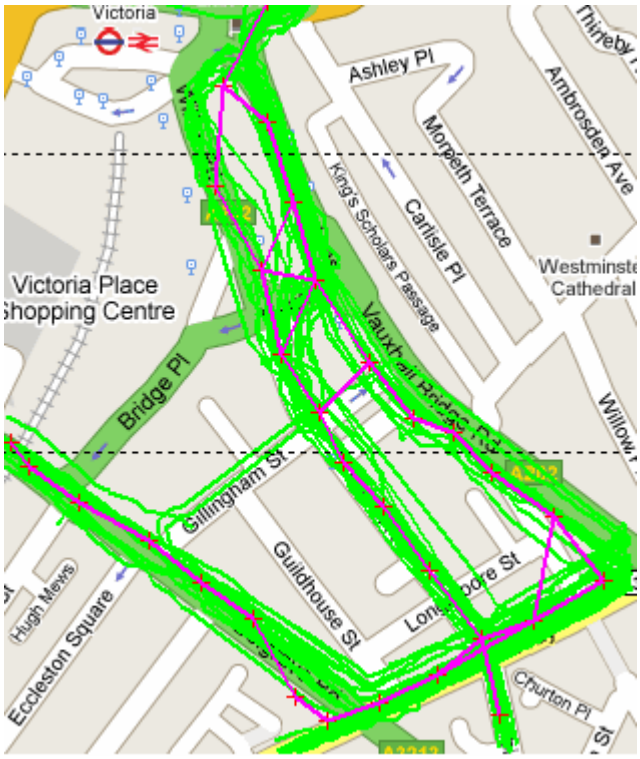
4.1. *Příklady výsledné virtuální mapy*

Na Obr. 49 se nachází ukázka virtuální mapy. Vygenerované shluky jsou velmi dobře spojeny linkage metodou. Jen v několika místech došlo ke vzniku přebytečných cest, obzvláště na hranicích obdélníkové sítě a také u víceproudých silnic.



Obr. 49 Virtuální mapa na mapovém podkladu

Na Obr. 50 se nachází druhá ukázka virtuální mapy. Tato ukázka zobrazuje špatné spojení shluků, ke kterému dojde díky tomu, že vygenerované shluky obsahují body z obou rovnoběžných a blízko u sebe položených silnic.



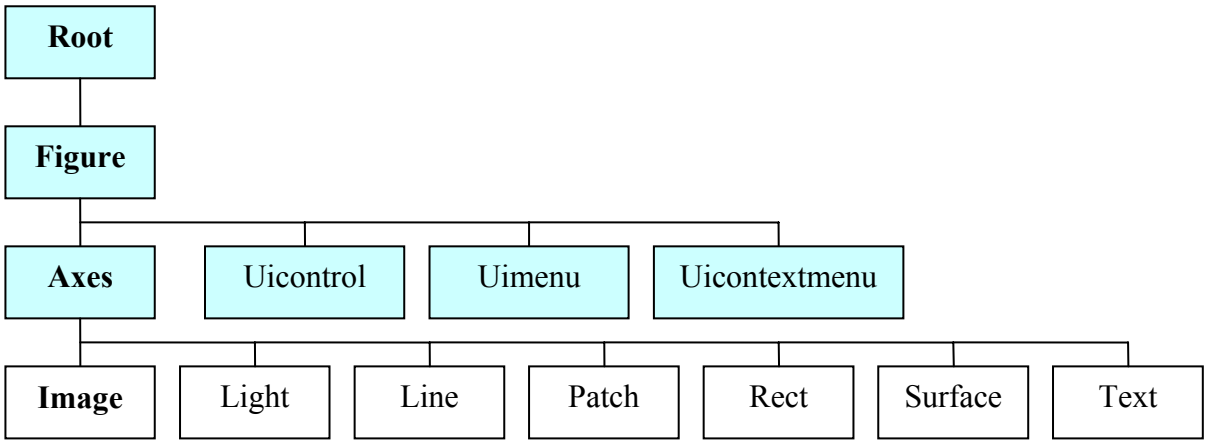
Obr. 50 Virtuální mapa na mapovém podkladu

V. Matlab GUI (Graphical User Interface)

1. Systém Handle Graphics

Systém *Handle Graphics* je nástroj, s jehož pomocí lze efektivně pracovat s grafickými objekty v systému Matlab. Zahrnuje animaci, zpracování signálů, příkazy pro 2D a 3D vizualizaci dat a grafiku obecně. Přináší efektivnost a mnoho možností při práci s grafikou.

Grafickými objekty se rozumí všechny základní elementy, používané k zobrazení jakéhokoliv grafického výstupu. Hierarchie grafických objektů je znázorněna na Obr. 1. Podrobněji o jednotlivých objektech lze nalézt v literatuře [34].



Obr. 1 Hierarchie základních grafických objektů systému Matlab

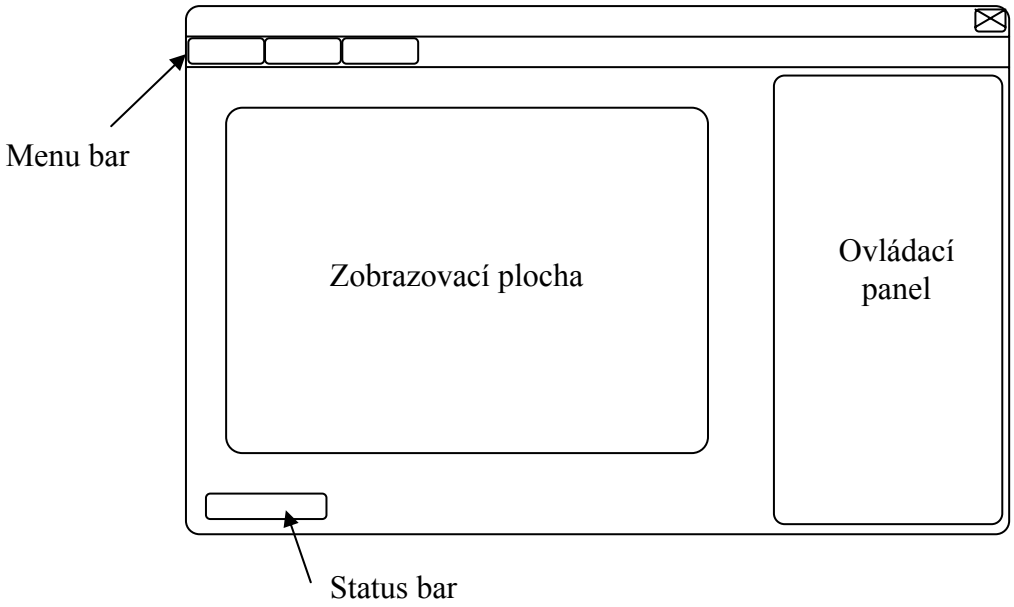
Každý grafický objekt lze vytvořit jako proměnnou, která se označuje *Handle* a je naplněna číslem, které jednoznačně identifikuje grafický objekt. Nastavení vlastností jednotlivých grafických objektů se vykonává pomocí funkcí *get* a *set*.

Při programování interakcí grafických objektů používáme *Callback* funkce, jedná se o naprogramování reakce tlačítek, zaškrtávacích políček apod. na uživatelský zásah (stisk tlačítka, označení, vepsání textu apod.).

Pro tvorbu jednoduchých grafických prvků je přímo v Matlabu zabudován nástroj GUIDE (Graphical User Interface Development Environment), který usnadňuje uživateli generovat GUI. Pro tento projekt je vhodné tento nástroj využít a jednoduše rozmístit grafické objekty na budoucí okno aplikace. Nevýhoda tohoto generování prvků je, že kód z důvodu univerzálnosti není optimální a modifikovatelnost GUI je značně omezená. Proto jsem další změny a *callback* funkce doplňoval přímo v editoru.

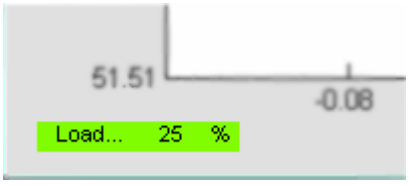
2. Vlastní realizace GUI, verifikace – Google mapy

Celá aplikace se nechá z grafického hlediska rozdělit na několik hlavních částí Obr. 2. První důležitou částí je zobrazovací plocha představována grafickým objektem *axes*, která slouží k vykreslování jednotlivých dat, ať už originálních nebo vypočtených.



Obr. 2 Wireframe (drátěný model) aplikace

Přibližný průběh jednotlivých výpočtů je možné sledovat na grafickém prvku *Static text*, který je umístěn v levém dolním rohu figury Obr. 3. Aby v průběhu výpočetního procesu docházelo k jeho změně, je nezbytně nutné volat funkci *drawnow*, která aktualizuje okno dané figury.

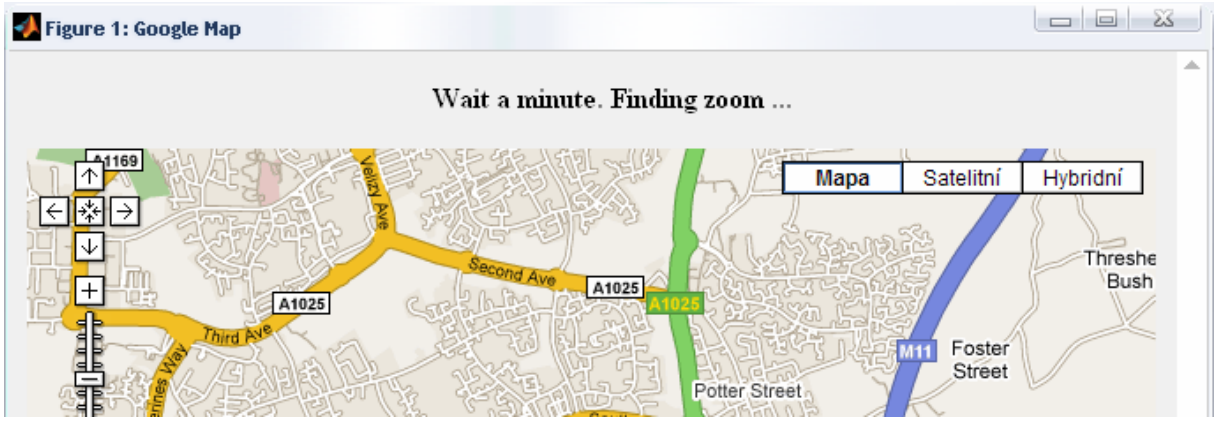


Obr. 3 Zobrazení průběhu výpočtu

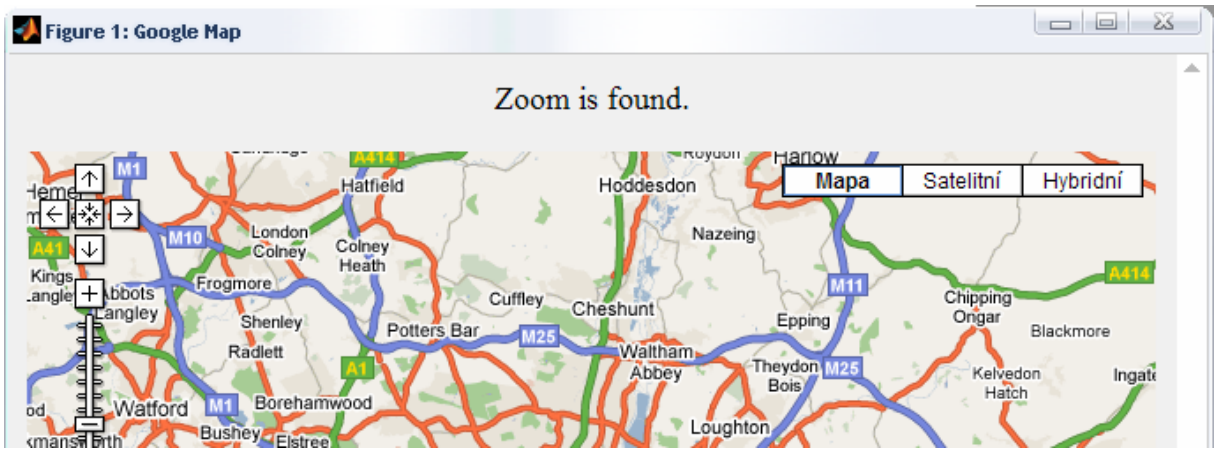
Další částí jsou ovládací panely Obr. 7 a 8. Panel *Functions* obsahuje šest vnořených panelů s grafickými prvky, které se postupně aktivují dle jednotlivých výpočtů. První slouží

k výběru vstupních dat ze souboru (originální data nebo uložená data do *mat*–souboru). Po načtení dat se vykreslí na zobrazovací plochu a aktivují se patřičné prvky na dalších panelech, které umožňují další práci s daty. Druhý vnořený panel slouží k výběru analyzovaných dat ze vstupní množiny bodů. Hraniční hodnoty výběru se nastavují automaticky při použití nástrojů ručička (pan) nebo zoom a jsou zobrazovány ve stejném panelu. Výsledný výřez se vykresluje v poměru 1:1. Následující tři panely slouží pro výběr filtrovacích, shlukovacích a spojovacích metod. Jednotlivé výsledky vyjmenovaných metod se vykreslují v různých barvách. Jednotlivé barvy odpovídají barvám textu patřičných zaškrtávacích políček.

Pomocí posledního panelu je možné získat z Google Map mapový podklad pro právě analyzovaná data a přesvědčit se jak daný výpočet je přesný neboli provést verifikaci předešlých výpočtů. Okrajové hodnoty pro získání mapy se automaticky nastavují na témže panelu a jsou stejné jako hraniční hodnoty os. Po zaškrtnutí mapového zaškrtávacího políčka se vytvoří nové modální okno *Google Map* Obr. 4, ve kterém se automaticky pomocí prohlížeče Internet Explorer načte internetová stránka se vstupními parametry shodnými s hraničními hodnotami. Pomocí JavaScriptu se začne vyhledávat potřebný mapový zoom. Při nalezení zoomu Obr. 5 dojde k přenesení mapy na pozadí zobrazovací plochy jako obrázek, k nastavení doplňujících parametrů na mapovém panelu (zoom, xPix a yPix) a k zavření modálního okna. Poslední dva parametry udávají hodnoty zeměpisné šířky a délky, která odpovídá pro danou oblast velikosti 640 pixelům. K nalezení zoomu dojde tehdy, když je možné na mapu zobrazit data, která mají hraniční hodnoty shodné jako jsou vstupní parametry internetové stránky.

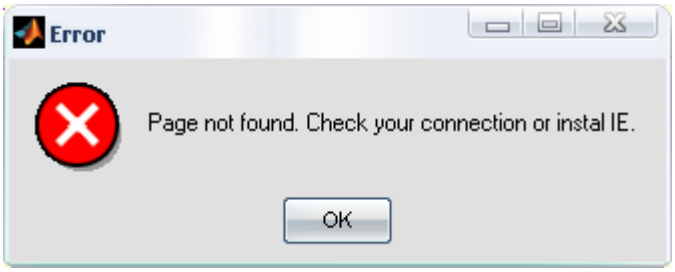


Obr. 4 Hledání zoomu



Obr. 5 Výsledná mapa s patřičným zoomem

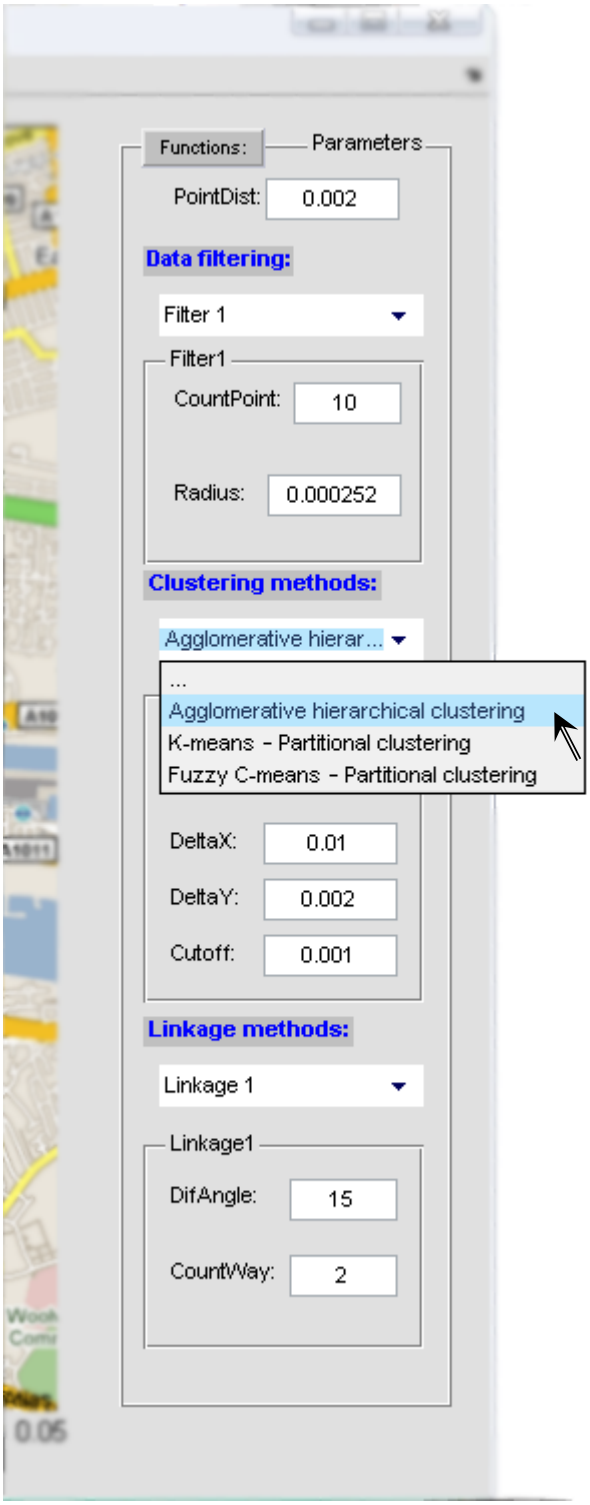
Aktualizace mapy probíhá při každé změně velikosti os, ale jen pokud je zatrhnuto mapové zaškrtávací políčko. Nemáme-li k dispozici prohlížeč Internet Explorer nebo nelze navázat spojení, objeví se místo modálního okna Obr. 4 chybové okno Obr. 6, které nás informuje



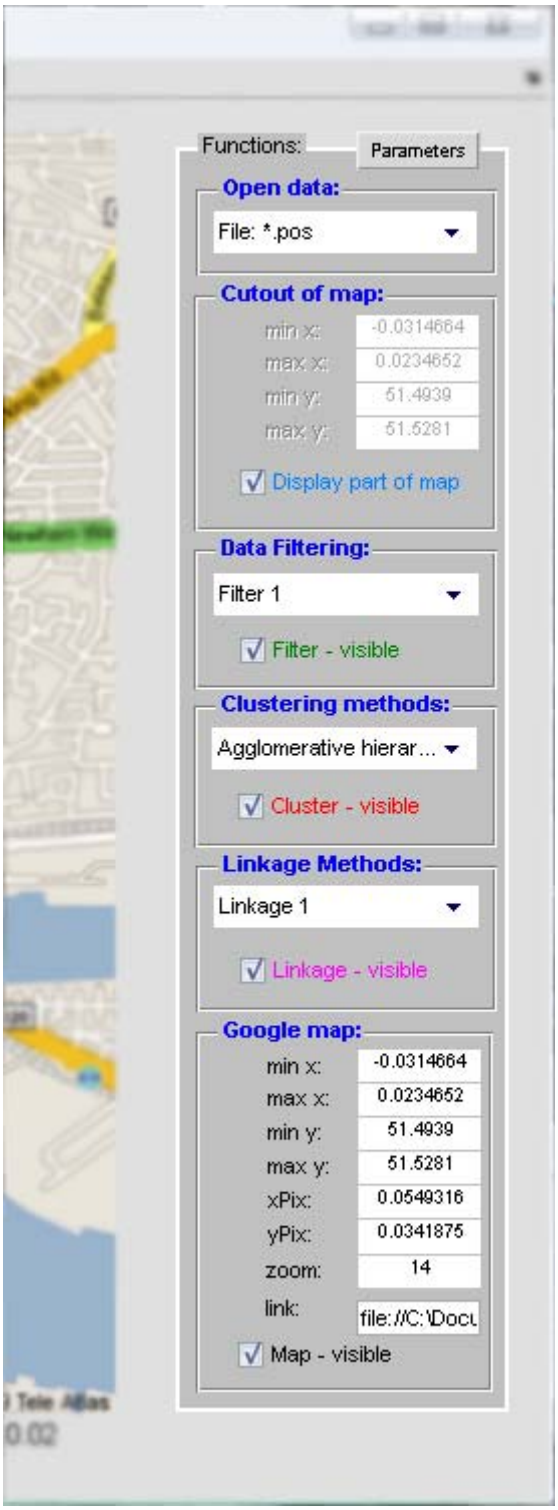
Obr. 6 Chybové okno

o tomto problému. Pokud máme nainstalovaný jiný internetový prohlížeč, lze zkopírovat adresu, která je zobrazena v spodní části mapového panelu do tohoto prohlížeče a po proběhnutí JavaScriptu zpětně překopírovat změněnou URL (Uniform Resource Locator) adresu. V případě úspěšného překopírování dojde ke stažení patřičného mapového podkladu s vyhovujícím zoomem na pozadí zobrazovací plochy.

Jednotlivé hladiny výpočtu se nechají pomocí zaškrtávacích políček nastavovat do stavu *Visible on/off*. V případě zaškrtávacího políčka pro Google mapy je napřed ještě zkontrolováno zda nedošlo ke změně některého parametru a pokud ano, tak se stáhne nová mapa, jinak se jen zviditelní předchozí mapa.



Obr. 7 Panel s parametry



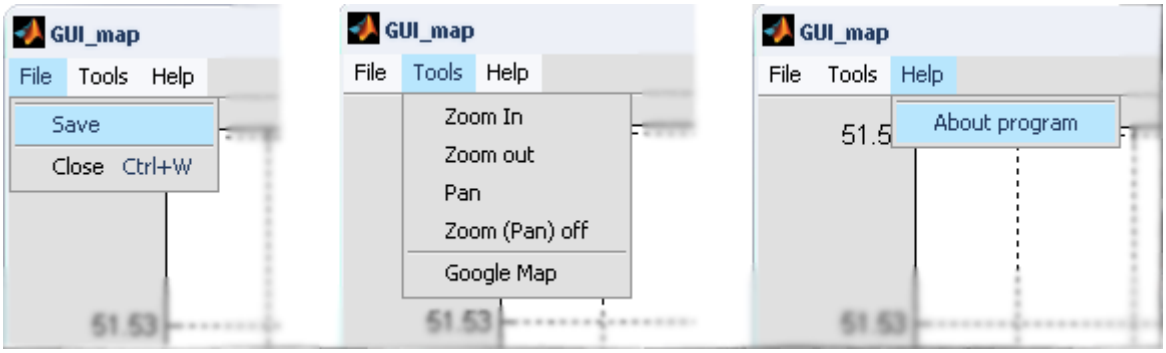
Obr. 8 Panel s funkcemi

Druhý hlavní panel *Parameters* se používá pro nastavování konkrétních parametrů k jednotlivým výpočetním funkcím Obr. 7. Jednotlivé parametry jsou vysvětlené v přiložené dokumentaci na CD ke grafickému programu. Při počátečním spuštění programu jsou tyto hodnoty nastaveny na defaultní hodnoty.

Poslední důležitou částí je Menu Bar Obr. 9. Obsahuje tři položky *File, Tools* a *Help*. V podmenu položky *File* nalezneme položky pro zavření aplikace a pro uložení doposud zobrazených (vypočtených) dat. Data se ukládají do mat–souboru, který obsahuje datové pole. Jednotlivé prvky se ukládají jako struktura viz Kód Matlabu 1 a obsahují jak původní data, tak i vypočítaná v průběhu chodu aplikace. Délka datového pole je maximálně šest – vypočtené hodnoty z každého vnořeného panelu z hlavního panelu *Functions* znamenají jeden prvek v datovém poli.

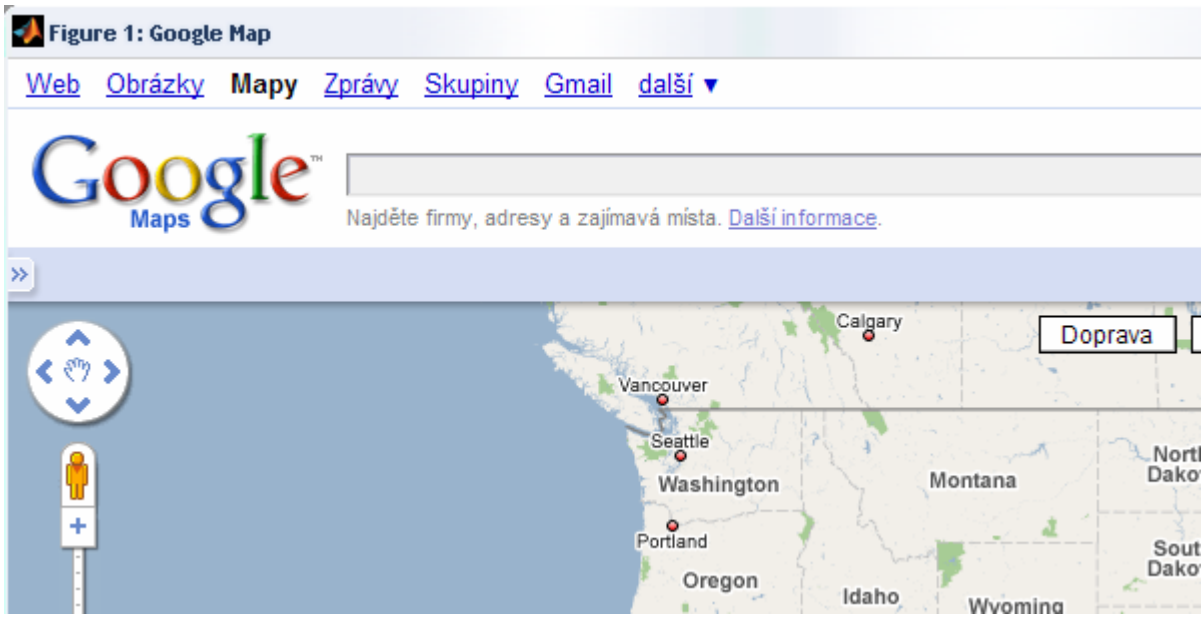
```
dataM = struct('data_ID',  id, ...
               'data_Data',  data, ...
               'data_Visible',  visible, ...
               'data_Name',  name, ...
               'data_Group',  group, ...
               'data_Processor',  processor);
```

Kód Matlabu 1 Struktura uložených dat



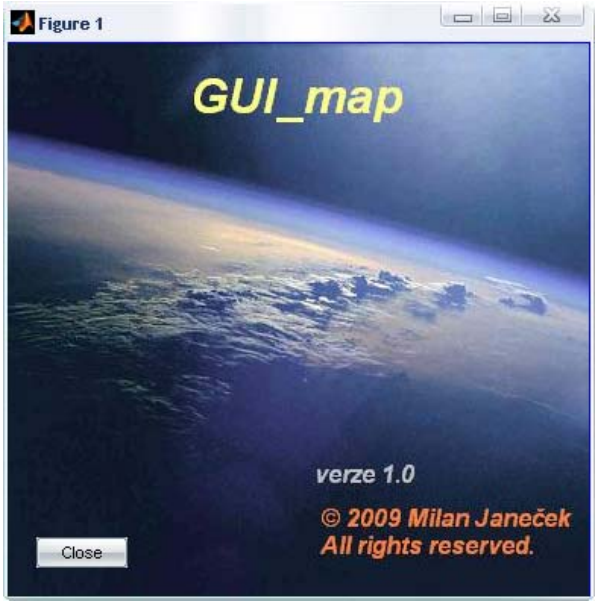
Obr. 9 Menu Bar

V podmenu položky *Tools* se nachází grafické nástroje zoom a ručička sloužící pro zobrazování detailů. Pomocí položky *Google Map* se otevře nové okno s internetovou stránkou **http://maps.google.com/** Obr. 10.



Obr. 10 Hlavní část okna figury Google Map

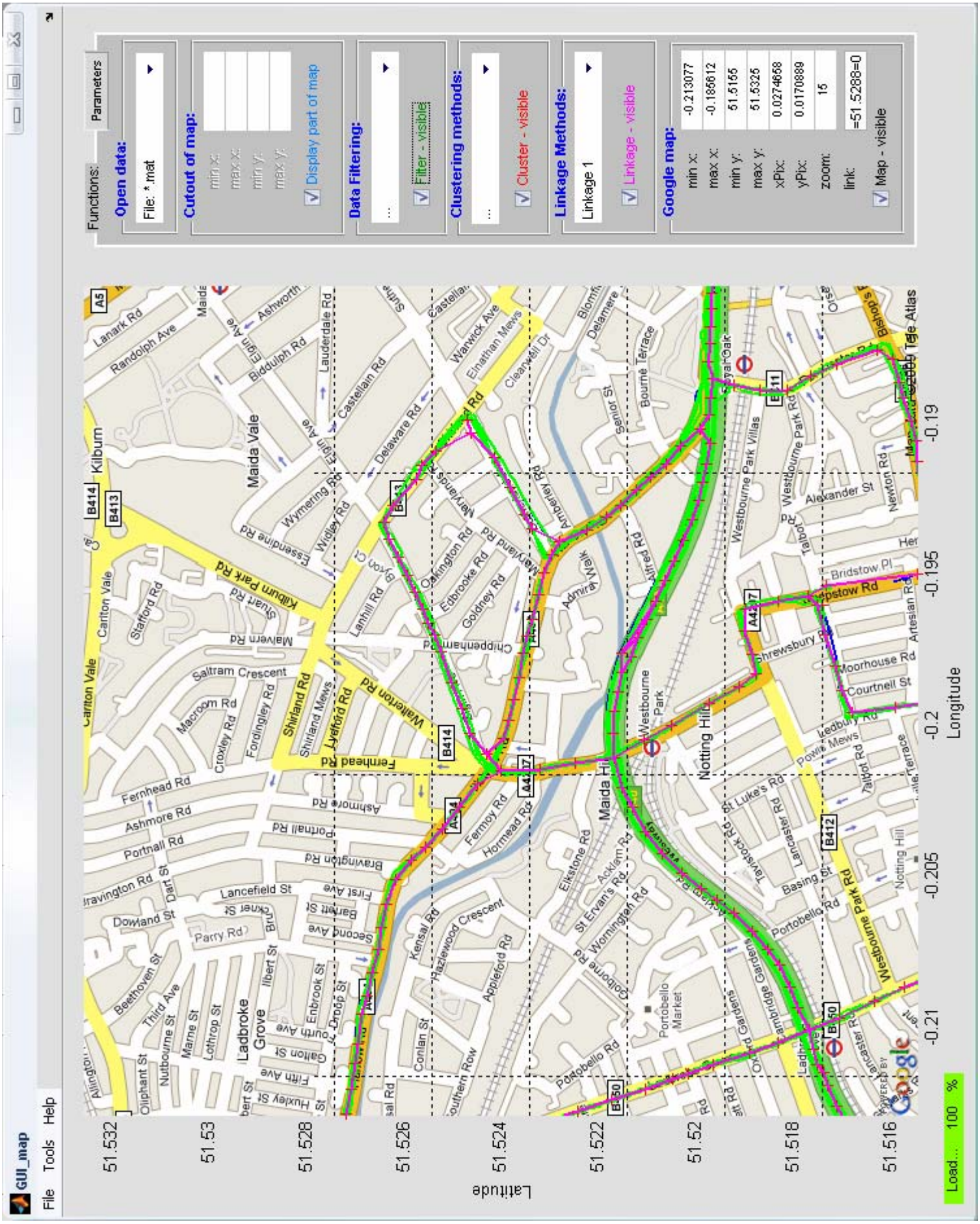
Po rozkliknutí poslední položky *Help* v menu se objeví podmenu About program Obr. 11.



Obr. 11 O programu

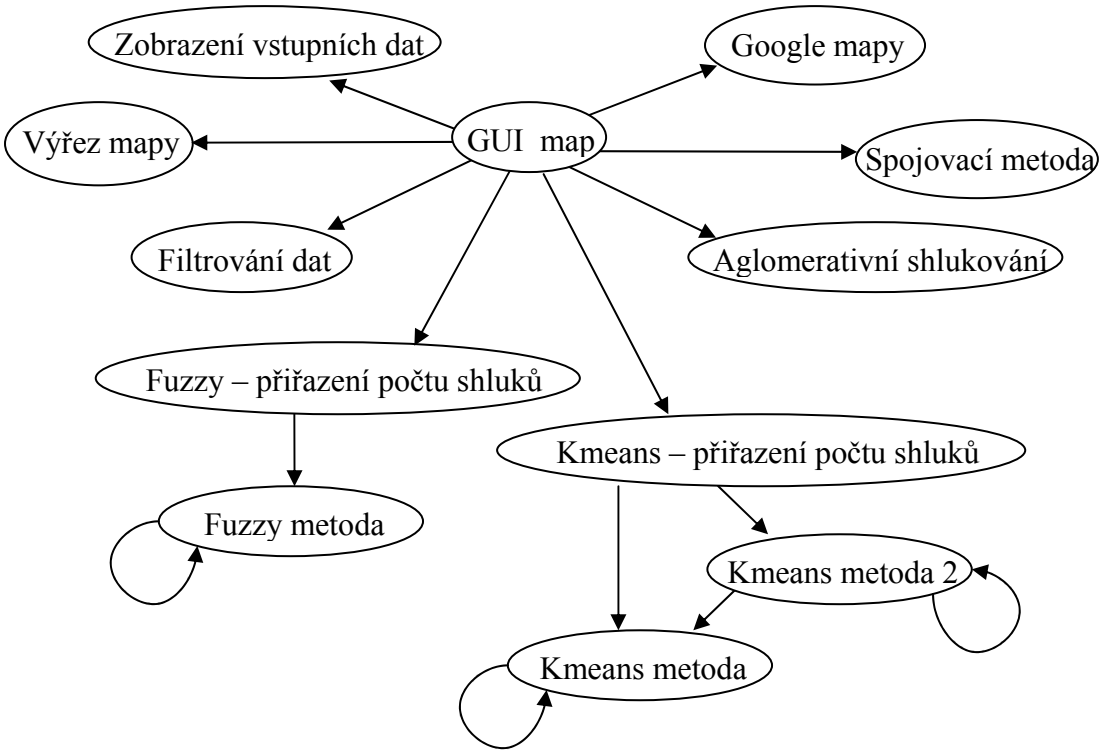
Na Obr. 12 je znázorněna celá hlavní figura s vypočtenými daty i s mapou z Googlu.

Pozn.: V poslední řadě bych chtěl upozornit na to, že pokud budete chtít vyzkoušet libovolnou jinou metodu na stejná data, tak dojde ke smazání všech závislých dat na původně vybrané metodě. Pro porovnání dvou metod, je nutné spustit paralelně další instanci aplikace.



Obr. 12 GUI s výslednými daty

Na posledním obrázku Obr. 13 je zobrazen graf závislosti jednotlivých funkcí, které je možné volat z GUI.



Obr. 13 Graf závislosti jednotlivých funkcí

pozn.: Pro správnou funkčnost programu je nutné mít nainstalovaný *Statistics Toolbox*, protože shlukovací metody využívají některé jeho funkce.

Dále pak nakopírovaný Fuzzy toolbox v adresáři *FuzzyClusteringToolbox_m*, který se nachází ve stejném adresáři jako jednotlivé skripty (toolbox lze stáhnout z **http://www.fmt.vein.hu/softcomp/fclusttoolbox/**).

VI. Závěr

Cílem práce bylo analyzovat naměřená data získaná z měření pomocí plovoucích vozidel vybavených GPS zařízením a verifikovat získanou infrastrukturu pomocí mapových podkladů. Pro tyto účely bylo vytvořeno v prostředí Matlab grafické uživatelské rozhraní. Pomocí grafického rozhraní lze vykonávat jednotlivé části analýzy naměřených dat, a to konkrétně výběr dat, filtraci, generování shluků a spojování shluků neboli vytvoření virtuální mapy. Rozhraní je navrhnuto tak, aby bylo snadné do něho implementovat nové metody ať už filtrovací, shlukovací nebo spojovací. Dále pak byla navrhnuta metoda pro filtraci dat a pro spojování vygenerovaných shluků. Vytvořené rozhraní umožňuje uživateli výběr ze tří shlukovacích metod, dvou nehierarchických a jedné hierarchické. Pro nehierarchické shlukování byl navrhnut nový index na zjišťování optimálního počtu shluků. Grafické rozhraní také umožňuje měnit vstupní parametry jednotlivých metod nebo uložení dat do *mat* souboru a jeho opětovné načtení a možnost pokračování v analýze.

Pro verifikaci byla vytvořena webová stránka, která se spouští z grafického rozhraní při požadavku zobrazení mapového podkladu a pomocí JavaScriptu vyhledává patřičná vstupní data, pomocí kterých dojde ke stažení požadovaného mapového podkladu.

V praxi tato práce může posloužit jako podklad k účelům monitorování hustoty dopravy a k následnému jejímu efektivnímu řízení. Aby tomu tak mohlo být bylo by vhodné doplnit načítání vstupních dat z databázového serveru, ukládání výsledných dat do databáze a vymyslet, jak jednotlivé uložené výsledky analýz spojit dohromady pokud na sebe navazují anebo se navzájem překrývají.

VII. Literatura

[1] Schroedl, S., Wagstaff, K., Rogers, S., Langley, P., & Wilson, C. (2001). *Mining GPS traces for map refinement.* DaimlerChrysler Research and Technology North America, pp. 59–87.

[2] Alena Lekasová, Jana Šarmanová, *Metody shlukové analýzy*, 1985

[3] Leonard Kaufman, Peter J. Rousseeuw, *Finding Groups in Data*, 2005

[4] J. C. Bezdek, J. C. Dunn, *Optimal fuzzy partitions: A heuristic for estimating the parameters in a mixture of normal dustrubutions.* IEEE Transastions on Computers, pages 835–838, 1975

[5] B. Balasko, J. Abonyi, B. Feil, *Fuzzy Clustering and Data Analysis Toolbox*.

[6] D. E. Gustafson, W. C. Kessel, *Fuzzy clustering with fuzzy covariance matrix*, in Proceedings of the IEEE CDC, San Diego, pages 761–766, 1979.

[7] R. Babuška, P. J. van der Veen, U. Kaymak, *Improved covariance estimation for Gustafson–Kessel clustering*, IEEE International Conference on Fuzzy Systems, pages 1081–1085, 2002.

[8] P. H. A. Sneath, *Evaluation of Clustering Methods*, in: Numerical Taxonomy London and New York, 1969.

[9] T. Sörensen, *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and its Application to Analysis of the Vegetation on Danish Commons*, Biol. Skr., pages 1–34, 1948.

[10] R. R. Sokal, C. D. Michener, *A Statistical Method for Evaluating Systematic Relationships*, Univ. Kansas Sci., pages 1409–1438, 1958.

[11] P. MacNaughton–Smith, *Some Statistical and Other Numerical Techniques for Classifying Individuals*, London, Home Office Research Unit Report 1965.

[12] F. R. Fromm, R. A. Northouse, *A Nonparametric Clustering Algorithm. Pattern Recognition*, pages 107–114, 1976

[13] J. B. MacQueen, *Some Method for Classification and Analysis of Multivariate Observations*, in 5th Proc. Symp. Math. Stat. and Prob. Berkeley, pages 281–297, 1967.

[14] L. Kaufman, P. J. Rousseeuw, Clustering by means of medoins, in Statistical Data Analysis based on the L_1 Norm, Amsterdam, pp. 405–416, 1987.

[15] K. Wagstaff, S. Rogers, and S. Schroedl, *Constrained K-means clustering with background knowledge*, in Proc. 8th Int. Conf. Machine Learning, 2001, pp. 577–584.

[16] Yi Hong, and Sam Kwong, *Learning Assignment Order of Instances for Constrained K-means Clustering Algorithm*, 2007.

[17] Zhengdong Lu, Todd K. Leen, *Semi-supervised Clustering with Pairwise Constraints: A Discriminative Approach*, Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, 2007, San Juan, Puerto Rico, pp. 299-306.

[18] Eric Robert Eaton, *Clustering with Propagated Constraints*, Master Thesis 2005.

[19] Julia Ferraioli, *User-Guided Interactive Grapf Layout*, 2005.

[20] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen. *Discovering personal gazetteers: An interactive clustering approach*. In Proc. ACMGIS, 2004, pp. 266-273.

[21] Zhou, C., Frankowski, D., Ludford, P., Terveen, L. *An Experiment in Discovering Personally Meaningful Places from Location Data*, Extended Abstracts, Proceedings of CHI 2005.

[22] Steenberghen, T., Dufays, T., Thomas, I. and Flahaut, B., 2004, *Intra-urban location and clustering of road accidents using GIS: a Belgian example*. International Journal of Geographical Information Science, 18, pp. 169–181.

[23] Stewart Worrall and Eduardo Nebot, *Automated Process for Generating Digitised Maps through GPS Data Compression*, Proceedings of the 2007 Australasian Conference on Robotics & Automation, 2007.

[24] Zhou, C., Shekar, S., and Treveen, L. *Discovering Personal Paths from Sparse GPS Traces,* In Proc. of the JCIS 2005 Workshop on Data Mining, 2005.

[25] Torkkola, K., et al., *Traffic Advisories Based on Route Prediction*, in Workshop on Mobile Interaction with the Real World (MIRW 2007), Singapore 2007.

[26] Rogers, S., Langley, P., Wilson, C., *Mining GPS data to augment road models*, Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining, pages 104–113, San Diego, 1999.

[27] Daniel Ashbrook, Thad Starner, *Using GPS to learn significant locations and predict movement across multiple users*, in Personal and Ubiquitous Computing, October 2003.

[28] K. Laasonen, *Clustering and prediction of mobile user routes from cellular data*, in Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), ser. Lecture Notes in Artificial Intelligence, vol. 3721. Springer-Verlag, pp. 569–576, 2005.

[29] M. Bilenko, S. Basu, and R. J. Mooney, *Integrating constraints and metric learning in semi–supervised clustering*, in Proceedings of ICML, pages 81–88, Banff, Canada, 2004.

[30] Zheng Y., Li Q., Xie X.. *Understanding mobility based on GPS data*, in Proc. of UbiComp2008, ACM Press(2008).

[31] D. Xie, W. Hu, T. Tan, and J. Peng, *A multi-object tracking system for surveillance video analysis*, in Proc. ICPR, 2004.

[32] Zhouyu Fu, Weiming Hu, Tieniu Tan, *Similarity based vehicle trajectory clustering and anomaly detection*, Int. Conf. on Image Processing,Vol.2, pp.602-605, 2005.

[33] Zhang Q., Couloigner I., *A new and efficient k-medoid algorithm for spatial clustering*, in International Conference on Computational Science and Its Applications, volume 3482 of LNCS, pages 181–189, Singapore. Springer-Verlag, 2005.

[34] Bourke P, *Efficient Triangulation Algorithm Suitable for Terrain Modeling* [online], January 1989 [cited 19.5. 2009], <http://local.wasp.uwa.edu.au/~pbourke/papers/triangulate/>.

[35] Karel Zaplatílek, Bohuslav Doňar, MATLAB – tvorba uživatelských aplikací, BEN, Praha 2004

VIII. Příloha CD

1. Diplomová práce v elektronické podobě
2. Vytvořené skripty
3. Dokumentace ke skriptům
4. Volně dostupné zdroje v elektronické podobě