Univerzita Karlova
Přírodovědecká fakulta

Algoritmy počítačové kartografie

Úloha 2: Generalizace budov

Adam Kulich, Markéta Růžičková, Eliška Sieglová
N-GKDPZ

Praha 2023

1 Zadání

Vstup: množina budov $B = \{B_i\}_{i=}^{n}$, *budova* $B_i = \{P_{i,j}\}_{j=1}^{m}$.

Výstup: $G(B_i)$.

Ze souboru načtěte vstupní data představovaná lomovými body budov. Pro tyto účely použijte vhodnou datovou sadu, např. ZABAGED.

Pro každou budovu určete její hlavní směry metodami:

- Minimum Area Enclosing Rectangle
- Wall Average

U první metody použijte některý z algoritmů pro konstrukci konvexní obálky. Budovu nahraďte obdélníkem se středem v jejím těžišti orientovaným v obou hlavních směrech, jeho plocha bude stejná jako plocha budovy. Výsledky generalizace vhodně vizualizujte.

Odhadněte efektivitu obou metod, vzájemně je porovnejte a zhodnoťte. Pokuste se identifikovat, pro které tvary budov dávají metody nevhodné výsledky, a pro které naopak poskytují vhodnou aproximaci.

Hodnocení:

Krok	**Hodnocení**
Generalizace budov metodami Minimum Area Enclosing Rectangle a Wall Average.	10b
Generalizace budov metodou Longest Edge.	*+5b*
Generalizace budov metodou Weighted Bisector.	*+8b*
Implementace další metody konstrukce konvexní obálky.	*+5b*
Ošetření singulárního případu při generování konvexní obálky.	*+2b*
Max celkem:	**35b**

Čas zpracování: 3 týdny.

2 Údaje o bonusových úlohách + body

3 Generalizace budov: popis a rozbor problému

3.1 Formulace problému

Generalizace budov je hojně řešený problém v kartografii při převádění map do menších měřítek. Menší plocha mapového pole znamená nutnost zjednodušit tvary nacházející se v mapě. Obecně by se tento problém mohl nazvat tvorbou konvexní obálky množiny bodů, který je využívaný nejen v kartografii, ale také například v analýze shluků, statistické analýze rozptylů, či analýze tvarů.

Jedním z problémů, kterými se generalizace budov zabývá, je problému tvorby konvexní obálky množiny bodů. Cílem je vytvořit co nejmenší konvexní množinu (v angličtině Convex Hull Problem) a k tomu je možné dospět více metodami. Mějme množinu bodů S a konvexní obálku H. Jednou z definicí je naleznout takové H, aby plocha konvexní obálky byla co nejmenší. Některé z metod tvorby konvexní obálky, které si blíže popíšeme, jsou Jarvis Scan, Graham Scan a Quick Hull.

Dalším problémem, který generalizace řeší je orientace budov a zachování jejího hlavního směru vůči ostatním prvkům v mapě, například vůči ulicím. Algoritmy, které využívají hlavních směrů budov jsou například Maximum Area Enclosing Rectangle, Wall average, Longest Edge či Weighted Bisector.

A třetím problémem v rámci generalizace je škálování zjednodušených budov. Může to být nalezení například nejmenší možné obálky, či budovu přeškálovat tak, aby měla stejný obsah jako originální polygon.

3.2 Metody konstrukce konvexní obálky

3.2.1 Jarvis Scan

Metoda Jarvis Scan je jednou z prvních metod řešení Convex Hull Problem. Pochází z roku 1973 a někdy je také nazývána Gift Wrapping Algorithm, protože postup připomíná balení dárku do dárkového papíru.

Je triviální na implementaci, je třeba data předzpracovat, naprogramovat cyklus a funkci na počítání úhlů. Její nevýhodou je ovšem kvadratická složitost.

V Jarvis Scanu se nejdříve data setřídí, a poté se nalezne pivot, tedy počáteční bod, od kterého začneme počítat konvexní obálku. Tento bod označíme q a rovnou ho přidáme do konvexní obálky H. Před započetím for-cyklu musíme vytvořit segment rovnoběžný s osou x, od kterého budeme počítat úhly. Od bodu q tedy vedeme úsečku qr se stejným y. Poté procházíme postupně všechny body množiny S a do konvexní obálky H přidáme ten, který s našim segmentem qr svírá největší úhel. Segment qr nahradíme úsečkou p_iq a opět hledáme další bod, který s tímto segmentem svírá největší úhel. Takto postupujeme až do momentu, kdy nalezneme počáteční bod.

S konstrukcí počátečního segmentu qr bychom měli být opatrní, protože při výpočtu úhlů by mohlo docházet k chybě ze zaokrouhlení. Například kdyby bod, ke kterému počítáme úhel, byl velmi blízko pivotu q, dělili bychom malé číslo velkým číslem, a mohlo by dojít ke ztrácení desetinných míst. Tento segment by tedy neměl být ani moc krátký, ani moc dlouhý, ideálně by jeho délka měla být určena na základě statistického posouzení datové sady S (například se dá použít $x_{max} - x_{min}$).

Algorithm 1 Jarvis Scan

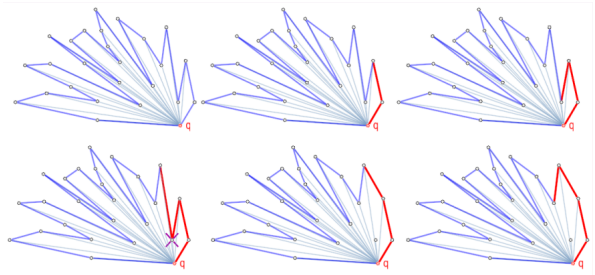
```
setřídit data podle y
najít pivot q s nejmenším y                    ▷ pivot: počáteční bod
vytvořit segment qr
inicializuj q = p_i
for body v množině S do                        ▷ spojení pivotu q s bodem o stejné hodnotě y

end for
```

3.2.2 Graham Scan

Algoritmus Graham Scan má logaritmickou složitost a není těžký na implementaci, nicméně mnoho singulárních případů ho dělá poměrně komplikovaným. Podobně jako u algoritmu Jarvis Scan je potřeba body množiny S nejprve setřídit a vybrat vhodný pivot q, například bod s nejnižší hodnotou y. Po vybrání pivotu je zkonstruován tzv. start-shaped polygon (je ve tvaru hvězdice).

Poté se body postupně prochází, a z množiny se odstraňují body neležící na konvexní obálce pomocí tzv. kritéria levotočivosti. Jednoduše se dá říci, že úsečka spojující body p_i a p_{i+1} dělí prostor na pravou a levou polorovinu. Pokud bod p_{i+2} leží v pravé polorovině, je bod p_{i+1} z množiny odstraněn pomocí *S.pop*.



Ukázka konstrukce konvexní obálky s využitím algoritmu Graham Scan. Z obrázku je patrné, proč se polygonu říká star-shaped. Obrázek je převzatý z přednášky Konvexní obálka množiny bodů od pana doc. Bayera (2023).

3.2.3 Quick Hull

Tento algoritmus využívá strategie Divide and Conquer, tedy rozdělí si úlohu na menší podúlohy, které postupně řeší. Ve většině případů je rychlý, nicméně v některých případech může nabývat až kvadratické složitosti a je velice neefektivní, například v případě vzorkované kružnice.

I v tomto algoritmu se začíná setřízením bodů dle x a určeny body q_1 (x_{min}) a q_3 (x_{max}). Spojením těchto dvou bodů je množina S rozdělena na S_u (upper) a S_l (lower). Konvexní obálka se poté počítá pro tyto dvě poloroviny zvlášť, proto je to zařazeno do strategie Divide and Conquer.

Tento algoritmus se dělí na lokální a globální proceduru, v lokální proceduře hledáme bod, který je od úsečky nejvzdálenější. Dále se postupuje rekurzivně, kdy vždy hledáme nejvzdálenější bod poloroviny.

3.3 Hlavní směry budov

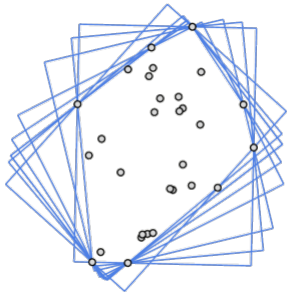
Pro otočení polygonu ve směru σ se využívá vzorec pro matici rotace.

$$\begin{pmatrix} \text{x'} \\ \text{y'} \end{pmatrix} = \begin{pmatrix} \cos(\sigma) & \sin(\sigma) \\ \text{-sin}(\sigma) & \cos(\sigma) \end{pmatrix} \begin{pmatrix} \text{x} \\ \text{y} \end{pmatrix}$$

3.3.1 Minimum Area Enclosing Rectangle

Tato metoda hledá obdélník o co nejmenší ploše, který je natočený ve úhlu hlavního směru původního polygonu. Pro generalizaci využívá konvexní obálku, je tedy nutné předchozí předzpracování dat.

V této metodě se opakovaně otáčí minmaxbox o směrnici jednotlivých hran konvexní obálky a hledá se minmaxbox (obdélník obsahující všechny body) o minimální ploše. Dle Freeman a Shapiro (1975) je dokázané, že takovýto obdélník bude mít vždy jednu hranu kolineární s hranou konvexní obálky. Tato metoda je poměrně úspěšná až na případy, kdy je budova ve tvaru písmene L či Z.



Ukázka opakovaného otáčení o úhel σ. Obrázek převzatý z DataGenetics (2013).

3.3.2 Wall Average

Wall Average je další metoda určující orientaci na základě určení hlavního směru budovy. Nejprve je spočtena směrnice pro libovolnou hranu budovy σ. K této hodnotě poté redukujeme další hrany budovy, čímž získáme $\Delta\sigma_i$. Poté spočteme zaokrouhlený podíl k_i a přejdeme k výpočtu zbytku r_i a odchylky od $0 \pm k\pi$. Pokud zbytek po dělení vyjde $r_i < \frac{\pi}{4}$, jde o odchylku od vodorovného směru, pokud vyjde $r_i > \frac{\pi}{4}$, jde o odchylku od svislého směru.

$$\Delta\sigma_i = \sigma_i - \sigma$$

$$k_i = \frac{2\Delta\sigma_i}{\pi}$$

$$r_i = \Delta\sigma_i - k_i\frac{\pi}{2}$$

Výsledný hlavní směr budovy je váženým průměrem dle následujícího vztahu:

$$\sigma = \sigma' + [\sum_{i=1}^{n}] \frac{r_i s_i}{s_i}. \quad (1)$$

3.3.3 Longest Edge

Metoda Longest Edge otáčí budovu o úhel σ, který je směrnicí nejdelší hrany. Tato metoda je relativně jednoduchá, protože jde pouze o nalezení nejdelší hrany a spočtení jejího úhlu, nicméně nejde o metodu spolehlivou. Nejdelší hrana budovy nemusí totiž vůbec určovat její hlavní směr.

3.3.4 Weighted Bisector

Metoda Weighted Bisector dává naopak velmi dobré výsledky. Hlavní směr je určen na základě váženého průměru směrů dvou nejdelších úhlopříček polygonu. Délky úhlopříček jsou značeny s_1 a s_2, jejich směrnice jsou σ_1 a σ_2. Vztah pro výpočet hlavního směru je dán následovně:

$$\sigma_1 = \frac{s_1\sigma_1 + s_2\sigma_2}{s_1 + s_2} \quad (2)$$

4 Popisy algoritmů formálním jazykem

Algorithm 2 Minimum Area Enclosing Rectangle

```
vytvoř konvexní obálku
inicializuj minmaxbox mmb_min a jeho plochu a_min
inicializuj σ_min = 0
for všechny hrany konvexní obálky do
    spočti úhel σ
    otoč konvexní obálku o úhel σ
    spočti minmaxbox mmb a jeho plochu a
    if a > a_min then
        aktualizuj a_min = a, mmb_min = mmb, σ_min = σ
    end if
end for
otoč a přeškáluj obdélník s nejmenší plochou (minmaxbox)
```

Algorithm 3 Wall Average

```
inicializuj zbytek r = 0
spočti σp_1 − p_0
for každá hrana do
    spočti σ_i
    spočti rozdíl úhlů Δσ_i = σ_i − σ
    spočti zaokrouhlený podíl k_i
    spočti zbytek r_i
end for
spočti průměrný úhel σ
otoč polygon o σ
přeškáluj polygon
```

Algorithm 4 Longest Edge

inicializuj nejdelší hranu $l_{max} = 0$ a úhel $\sigma = 0$
for každá hrana **do**
    spočti délku hrany
    if délka hrany > než l_{max} **then**
        aktualizuj l_{max}
        spočti nový úhel σ
        aktualizuj úhel σ
    end if
end for
otoč polygon o σ
přeškáluj polygon

Algorithm 5 Graham Scan

vytvoř prázdný list *sorted*
najdi pivot q s nejmenším y ▷ pivot: počáteční bod
nastav $pj1 = [q_x$ - 1, $q_y]$
setřiď data podle ω
vytvoř list Q o délce počtu vertexů a inicializuj na 1
do *sorted* přidej q
for každý vertex kromě q **do**
    for každý vertex v **do**
        if $v \neq q$ and $Q[v]$=1 **then**
            spočítej úhel ω mezi úsečkami $q - pj1$ a $q - v$
            if $\omega > \omega_{max}$ **then**
                aktualizuj ω_{max}
                nastav $v_{max} = v$
            end if
        end if
    end for
    do *sorted* přidej v_{max}
    nastav $Q[v_{max}] = 0$
end for
inicializuj $j = 2$
while $j <$ délka *sorted*
if v_j je v levé polorovině vůči v_{j-1} a v_{j-2} **then**
    nastav $j = j + 1$
end if
if v_j je v pravé polorovině vůči v_{j-1} a v_{j-2} **then**
    ze *sorted* odeber vertex v_{j-1}
    nastav $j = j$ - 1
end if

5 Problematické situace a jejich rozbor + ošetření kódu

6 Vstupní data, formát vstupních dat, jejich popis

Vstupními daty pro aplikaci jsou tři multipolygonové soubory ve formátu .shp, z nichž každý má reprezentovat jeden z typů rozložení zástavby ve městě. První soubor, označen v aplikaci "city center", obsahuje polygony budov z historického centra Prahy (Vyšehradu). Druhý soubor, označený jako "residential quarter", reprezentuje polygony zástavbu vilové obytné čtvrti (Dolní Počernice) a poslední soubor ("blocks of flats") zahrnuje polygony budov z panelového sídliště (sídliště Krč). Data jsou převzata z Digitální technické mapy Prahy.

7 Výstupní data, formát výstupních dat, jejich popis

Výstupem programu je aplikace v grafickém rozhraní Qt, která reprezentuje polygony jako datovou strukturu QPolygonF a provádí na nich zadané analýzy. Aplikace dovoluje výběr ze tří typů dat dat k analýze, umožňuje provést 5 generalizačních analýz (generalizace metodami Minimum area enclosing rectangle, Wall average, Longest edge a Weighted bisector a tvorbu konvexní obálky metodou Graham scan). Dále aplikace obsahuje funkce "Clear results", která vymaže výsledky analýzy a "Clear all data", která vymaže nahraná data a umožňuje zvolit k analýze jiný dataset.

8 Printscreen vytvořené aplikace

9 Dokumentace: popis tříd, datových položek, jednotlivých metod

9.1 Třída Algorithms

Třída Algorithms se nachází ve scriptu *algorithms.py* a v této třídě jsou definovány metody, analyzující vstupní polygon. V první řadě jsou zde pomocné funkce *get2LinesAngle*, která vrací úhel mezi dvěma vektory, *createCH*, která tvoří konvexní obálku polygonu pomocí algoritmu Jarvis scan, funkce *rotate*, která rotuje vstupní polygon podle zadaného úhlu a funkce *minMaxBox*, která vytvoří pro zadaný polygon a úhel nejmenší nenatočený obdélník. Další pomocné funkce ve třídě jsou *computeArea*, která počítá obsah útvaru a *resizeRectangle*, která přepočítá velikost obdélníku, nahrazujícího původní polygon. Nakonec tato třída obsahuje metody pro výpočet každé z analýz, obsažených v aplikaci (MAER, Wall average, Longest edge, Weighted bisector a tvorbu konvexní obálky metodou Graham scan.

9.2 Třída Draw

Tato třída se nachází ve scriptu draw.py a jejími parametry jsou self.__q (objekt typu QPointF, bod se dvěma souřadnicemi x a y, jehož poloha se bude vyšetřovat), self.__pol (objekt typu QPolygonF, který se bude vyšetřovat), self.__pols (seznam polygonů), self.polLoad (nahraná data) a self.polRes (polygony generalizovaných budov). Tato třída má definované metody grafického rozhraní na vykreslování vstupů, výsledků, a interakce uživatel-grafické rozhraní. V této třídě jsou definovány metody mousePressEvent (mění polohu bodu q na základě toho, kam uživatel klikne), input (načte data polygonů), paintEvent (umožňuje uživateli nakreslit bod), a getPolygon (vrací seznam polygonů), ClearAll (vymaže obsah analýzy), ClearLoadedData (vymaže načtená data z rozhraní), polisEmpty (vrací, jestli rozhraní obsahuje generalizované obdélníky), polLoadisEmpty (vrací, jestli jsou v rozhraní nahraná data)

9.3 Třída Load

Tato třída se nachází ve skriptu load.py. V této třídě dochází k načtení dat ve formátu shapefile pomocí knihovny geopandas, a získání x a y souřadnic bodů v grafickém rozhraní, které tvoří polygony.

9.4 Třída Ui_Mainform

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

10 Závěr, možné či neřešené úlohy, náměty na vylepšení

V rámci druhé úlohy z předmětu Algoritmy počítačové kartografie byl řešen problém generalizace budov pomocí metod Minimum area enclosing rectangle, Wall average, Longest edge a Weighted bisector. Byly rovněž implementovány dvě metody konstrukce konvexní obálky, Jarvis scan a Graham scan. Kód byl psán v jazyce Python v prostředí PyCharm s využitím grafického rozhraní QT.

Výstupem je aplikace, která uživateli umožní vizualizovat a porovnávat výsledky několika generalizačních metod na datasetech různých typů zástavby.

Prvním datasetem, který je pro aplikaci připraven, jsou budovy sídliště. Jde o budovy velmi pravidelné a z pohledu generalizace nepředstavují větší výzvu. Všechny 3 metody generalizovaly budovy shodně, s vyjímkou jediné drobné budovy ve tvaru hokejky, kterou vizuálně nejlépe generalizovala metoda Minimum area enclosing rectangle.

Druhým datasetem jsou samostatné budovy vilové čtvrti. Tyto budovy mají pouze drobné tvarové nehomogenity. Všechny metody dávaly opět shodné výsledky, s výjimkou dvou budov. V obou případech šlo o budovy s tvarem připomínající písmeno L, jednou se od ostatních metod lišil výsledek metody Minimum area enclosing rectangle, ve druhém případě se odlišovala metoda Wall average. Ani v jednom případě se ale nedá hovořit o špatném provedení generalizace.

Poslední dataset, který je pro aplikaci připraven, obsahuje budovy v historické části Prahy. Historická zástavba se často vyznačuje nepravidelnými a nepravoúhlými tvary, tudíž z hlediska generalizace nejvíce problematická. Vizuálně nejpříjemnějšího výsledku dosáhla metoda Minimum area enclosing rectangle. Naopak metoda Longest edge nedávala příliš uspokojivé výsledky u několika budov v datasetu, které byly (mírně) obloukovitého tvaru. Podle očekávání zde byla shoda mezi výstupy všech metod nejmenší, obzvláště výše zmíněních budov do tvaru L, obloukových budov a budov konvexních tvarů, avšak nepravoúhlých.

Aplikace rovněž umožňuje uživateli vlastní polygon nakreslit, a následně provádět generalizační metody nad ním. Toto ovšem funguje jen za situace, kdy jsou v aplikaci již nějaké data nahrána. Generalizační metody se nespustí, pokud je v aplikaci pouze nakreslený polygon.

Co se týče dalších možných vylepšení: Zdrojový kód aplikace je navržen tak, aby se vizualizace dat přizpůsobila charakteru dat zvolených uživatem. Do aplikace by bylo tudíž vhodné přidat tlačítko, které by uživateli umožnilo datset vyhledat v paměti svého zařízení, a nahrát ho. V současné chvíli, aby si mohl uživatel do aplikace nahrát

vlastní data, musí přepsat název souboru ve zdrojovém kódu. Uživatelsky zajímavá by také mohla být možnost přblížení dat (zoom), sloužící k lepšímu prohlížení výsledků generalizace u malých budov.

11 Seznam literatury

Bayer, Tomáš. 2023. "Konvexní obálka množiny bodů." Praha, February 23. https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk4_new.pdf.

"Best Fit Rectangles." n.d. Accessed March 30, 2023. http://datagenetics.com/blog/march12014/index.html.

H.Freeman, R.Shapira. Determining the Minimum-Area Encasing Rectangle for an Arbitrary Closed Curve, 1975.

Geoportál Praha: Digitální technická mapa Prahy. https://www.geoportalpraha.cz/cs/data/otevrena-data/C170F739-D27C-4556-9138-CAF7C14FB01B