Univerzita Karlova
Přírodovědecká fakulta

Algoritmy počítačové kartografie

Technická zpráva k druhé semestrální úloze

Generalizace budov

David Šklíba a Filip Zadražil
1. N-GKDPZ
Praha 2022

1 Zadání a údaje o bonusových úlohách

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	15b
Generalizace budov metodou Longest Edge.	*+5b*
Generalizace budov metodou Weighted Bisector.	*+8b*
Implementace další metody konstrukce konvexní obálky.	*+5b*
Ošetření singulárního případu u při generování konvexní obálky.	*+2b*
Max celkem:	**35b**

V rámci vzniklého programu **byla řešena** první bonusová úloha ***Generalizace budov metodou Longest Edge***.

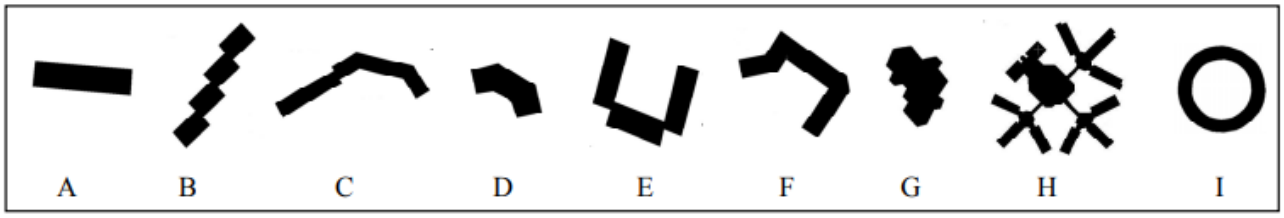
2 Popis a rozbor problému

2.1 Generalizace budov

Generalizace je proces přeměny map velkého měřítka na mapy malého měřítka. Generalizace, jinak také zobecnění map má zlepšit jejich čitelnost a zároveň zachovat důležité základní informace. Typické operátory pro zobecnění mapy zahrnují zjednodušení, přemístění, eliminaci a agregaci. Eliminace je nutná k odstranění malých budov, jako jsou např. izolované budovy. Přemístění je potřeba pro oddělení budov, které by byly příliš blízko u sebe v požadovaném měřítku mapy nebo pro přesun budov dále od silnic. Agregace seskupuje budovy do větších celků zastavěných bloků v případě, že budovy nemají být zobrazeny samostatně (Lee et al. 2017).

Od 60. let 20. století bylo provedeno mnoho studií o automatizované generalizaci map v oblasti GIS a kartografie. Výše uvedené operátory byly vyvinuty právě na základě těchto studií a určily, jak by měly být mapové prvky vhodně reprezentovány v malém měřítku. Ve většině studií zobecnění map výzkumníci vybírají vhodné operátory pro mapové prvky, jakými jsou např. silnice a budovy, a poté je aplikují s různými prahovými hodnotami. Výsledky zobecnění map jsou hodnoceny kvantitativními a kvalitativními měřítky (Lee et al. 2017).

Obecně je budova popsána svou polohou, tvarem, zrnitostí, velikostí a orientací. Studovanou charakteristikou v této práci je zejména orientace. Popis orientace budovy je v rámci automatického procesu generalizace velmi důležitý. Ve skutečnosti je totiž výpočet orientace pro např. liniové prvky poměrně jasný, ale pro případ polygonů se jedná o značně komplikovaný problém. Jednou z metod, kterou lze efektivně využít pro správnou generalizaci budov, je proto nalezení hlavních směrů budov. (Duchene et al. 2003). Ne vždy jsou však hlavní směry, potažmo orientace budovy zcela zřejmé (viz Obrázek 1).



Obrázek 1: Orientace některých budov nemusí být zcela jasná (Duchene et al. 2003)

2.2 Konvexní obálka

Sestrojení konvexní obálky (CH) je důležitý obecný geometrický problém, který lze řešit pomocí výpočetní techniky. Tento problém spočívá v zahrnutí, resp. ohraničení dané sady bodů v rovině polygonálním "pouzdrem", běžně nazývaným právě jako konvexní obálka. Problém konvexní obálky patří mezi kombinatorické, zvláště pak mezi problémy optimalizace. Konvexnost zde odkazuje na vlastnost mnohoúhelníku, který obklopuje dané body a tvoří "pouzdro". Konvexní mnohoúhelník je jednoduchý, bez jakéhokoli vlastního průniku, ve kterém jakákoli úsečka mezi dvěma body na hranách vždy přesahuje mnohoúhelník. Obecně řečeno konvexní mnohoúhelník obsahuje konvexní množinu, což je taková oblast, v níž se každá dvojice bodů nachází v oblasti a úsečka, která takovou dvojici spojuje, je také uvnitř této oblasti. Tato definice říká, že krychle, obdélník, pravidelný mnohoúhelník apod. mají konvexní povahu. Žádný mnohoúhelník, který má duté, promáčknuté nebo prodloužené vrcholy, nemůže být konvexní (Jayaram, Fleyeh 2016).

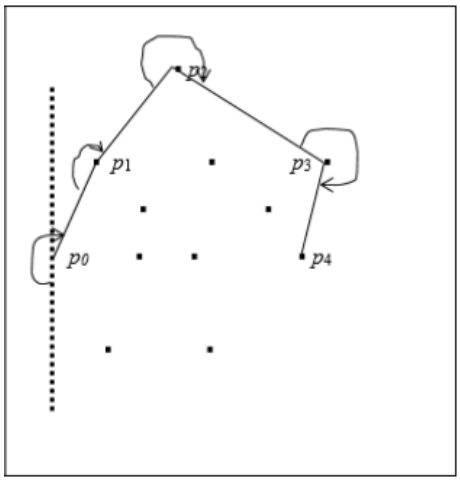
Pro tvorbu konvexní obálky existuje hned několik algoritmů, jakými jsou například:

- *Jarvis Scan (= Gift Wrapping Algorithm)*,
- *Graham Scan*,
- *Quick Hull*,
- *Divide & Conquer*,
- *Sweep Line*,
- *inkrementální konstrukce* atd.

V této práci je využit pouze jeden algoritmus, a tím je Jarvis Scan.

2.2.1 Algoritmus Jarvis Scan

Tento algoritmus je nejstarší a koncepčně nejjednodušší publikovanou metodou výpočtu konvexní obálky. Jejím autorem je R. A. Jarvis, který ji popsal v roce 1973. Její náročnost je označována $O(nlogh)$, kde n je počet bodů ve vstupní množině a h je počet bodů tvořících konvexní obálku. Tento algoritmus je výhodný, když n je malá hodnota nebo se očekává, že h bude velmi malé vzhledem k n. V obecných případech je algoritmus překonán mnoha jinými, sofistikovanějšími. Algoritmus dostal svůj druhý název (*Gift Wrapping Algorithm*) díky specifickému způsobu zpracování bodů, velmi podobnému balení dárku (Jayaram, Fleyeh 2016).



Obrázek 2: Princip fungování algoritmu Jarvis Scan (Jayaram, Fleyeh 2016)

Algoritmus začíná bodem, který je s jistotou součástí obálky, v našem případě bodem nejjižnějším (y_{min}), kterého nazýváme "pivot"a označujeme ho jako p_j. Poté inicalizujeme bod p_{j-1}, který leží na horizontální přímce procházející bodem p_j. V každém dalším kroku jsou pak uvažovány úhly ke všem ostatním bodům p_i v množině P a je hledáno takové p_i, které maximalizuje úhel $\omega(p_{j-1}, p_j, p_i)$. Takový bod následně označíme jako p_{j+1}:

$$p_{j+1} = max\ \omega(p_{j-1}, p_j, p_i)$$

Názorná grafická ukázka hledání bodu p_{j+1}, s tím rozdílem, že pivotem je zde bod s minimální hodnotu souřadnice x, je k vidění na Obrázku 2. Pseudokódem lze tento algoritmus popsat následovně:

Algorithm 1 Jarvis Scan

1: Nalezení p_{min} s minimální hodnotou souřadnice y
2: Přidání bodu p_{min} do konvexní obálky H
3: Inicializace bodů $p_{j-1} = [-10, min(y_i)]$ a $p_j = p_{min}$
4: **Dokud** $p_{j+1} \neq P_{min}$:
5: Pro každý bod $P_i \in P$:
6: Vypočítání $\omega(p_{j-1}, p_j, p_i)$
7: $p_{j+1} = max\ \omega(p_{j-1}, p_j, p_i)$
8: Přidání p_{j+1} do H
9: Aktualizace posledních dvou bodů konvexní obálky: $P_{j-1} = P_j$ a $P_j = P_i$

2.3 Metody natočení budov

Stěžejní fází generalizace je výsledné natočení generalizovaného polygonu. Důležitou podmínkou generalizace je totiž zachování orientace vzhledem k ostatním prvkům v mapě. Je proto nutné detekovat hlavní směry budovy. K tomu slouží hned několik různých metod. Součástí této práce jsou konkrétně metody *Minimum Area Enclosing Rectangle*, *Wall Average* a *Longest Edge*.

2.3.1 Minimum Area Enclosing Rectangle

Algoritmus MAER začíná tvorbou konvexní obálky H a vzápětí pokračuje vytvořením minmax boxu nad konvexní obálko, který se v rámci iterace postupně otáčí o úhel $-\sigma$, který je počítán z vektorů této hrany a osy x podle tohoto vzorce:

$$tan\sigma = \frac{\triangle y}{\triangle x}$$

Dle tohoto vzorce pro matici rotace poté otáčíme svou konvexní obálku:

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} cos(-\sigma) & sin(-\sigma) \\ -sin(-\sigma) & cos(-\sigma) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

Takto vzniká Minmax box nad rotovanou konvexní obálkou. My navíc potřebujeme jeho vrcholy, neboť počítáme jeho plochu Si. Po projití všech iterací tak získáme ten skutečně nejmenší možný minmax box, který lze kolem konvexní obálky sestrojit. Následně je nutné vypočítat původní plochu budovy (S_b), a to pomocí L'Huilierova vzorce:

$$S_b = \frac{1}{2} \sum_{i=1}^{n} x_i(y_{i+1} - y_{i-1})$$

Poté dojde k výpočtu k, což je poměr mezi plochou budovy (S_b) a plochou minimálního MMB (S_{er}):

$$k = \frac{S_b}{S_{er}}$$

Závěrem je nutné spočítat polohu bodů nově vytvořeného *Enclosing Rectangle*. K tomu musíme nejprve za pomoci těžiště C spočítat pro každý vrchol (v_i) obdélníka er vektor u_i:

$$u_i = v_i - C$$

Abychom poté dosadili do posledního vzorce a získali nové souřadnice vrcholů nalezeného *Minimum Area Enclosing Rectangle*:

$$v_i = C + \sqrt{k} u_i$$

Algorithm 2 Minimum Area Enclosing Rectangle

Konstrukce konvexní obálky H
Výpočet aproximace MMB_{min}
Výpočet S_{min}
$\delta_{min} = 0$
Pro každou hranu H:
    Výpočet σ_i
    Otočení H o $-\sigma$
    Konstrukce MMB_i a jeho S_i
    Pokud $S_i < S_{min}$:
        $S_{min} = S_i$
        $\sigma_{min} = \sigma_i$
        $MMB_{min} = MMB_i$
Rotace MMB_{min} o σ_{min}
Výpočet S_b
Úprava velikosti obsahu minmax boxu tak, aby platilo $S_{MMB} = S_b$

2.3.2 Wall Average

Tato metoda, jež byla představena v roce 1998 Hangouetem, vypočítává ukazatel orientace hran budovy. Obecným principem měření je vzít v úvahu orientaci každé hrany $mod(\frac{\pi}{2})$ (tedy mezi 0 a $\frac{\pi}{2}$) a vypočítat průměr těchto směrů vážený délkami hran (Duchene 2003).

U tohoto algoritmu je důležité nejprve zjistit směrnici σ' pro libovolnou hranu a poté spočítat směrnici σ_i pro každou z hran. Poté dojde k redukci těchto hodnot právě o hodnotu σ':

$$\triangle\sigma' = \sigma_i - \sigma'$$

.

Pomocí $\triangle\sigma'$ se následně vypočítá k_i:

$$k_i = \frac{2\triangle\sigma'}{\pi}$$

Poté se přistoupí k výpočtu zbytku r_i. Pokud je jeho hodnota $< \frac{\pi}{4}$, pak je hrana odchýlena od vodorovného směru. V případě, že je hodnota $> \frac{\pi}{4}$, je hrana odchýlena od svislého směru.

Výsledný směr natočení zkoumané budovy (σ) je výpočítán za pomoci již zmíněného váženého průměru hodnot r_i, v němž vahou je suma délek hran (l_i):

$$\sigma = \sigma' + \sum_{i=1}^{n} \frac{r_i l_i}{l_i}$$

Pseudokód pro tento algoritmus je zde:

Algorithm 3 Wall Average

Výpočet směrnice σ' pro libovolnou hranu budovy:
Pro každou hranu budovy:
    Výpočet směrnice σ_i.
    Výpočet $\triangle\sigma'$
    Výpočet k_i
    Výpočet r_i
Výpočet výsledné směrnice σ
Otočení všech hran budovy o $-\sigma$
Vytvoření minmax boxu
Otočení minmax boxu o σ
Úprava velikosti obsahu minmax boxu tak, aby platilo $S_{MMB} = S_b$

2.3.3 Longest Edge

V tomto algoritmu je měřena orientace nejdelšího okraje budovy po odstranění zarovnaných bodů budovy. V principu jde o to nalézt nejdleší hranu polygonu, která určí první hlavní směr orientace. Druhý hlavní směr je kolmý na ten první. Poté už dochází k totožnému postupu jako u zbylých algoritmů. Je vypočítána směrnice σ, každá hrana je rotována o úhel $-\sigma$, dojde k vytvoření minmax boxu, který je narotován o úhel σ a na závěr je změněna velikost minmax boxu, aby měl stejný obsah (S_{MMB}) jako původní polygon (S_b). Přehledněji, viz pseudokód níže:

Algorithm 4 Longest Edge

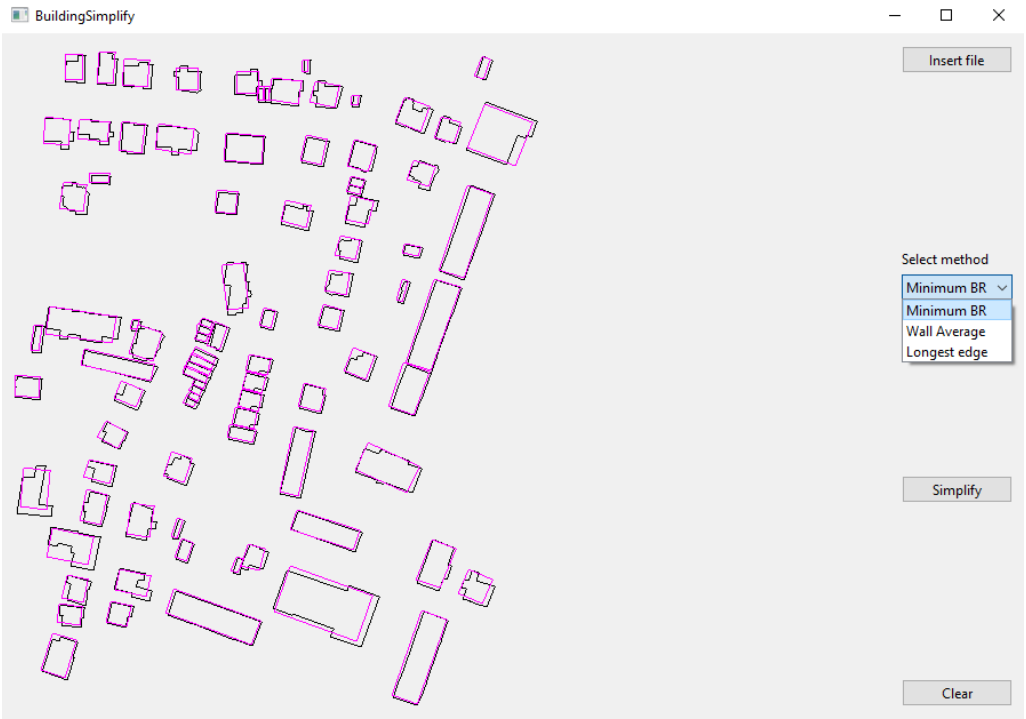
1: $d_{max} = 0$
2: Pro každou hranu budovy:
3: Výpočet délky hrany d_i.
4: **Pokud** $d_i > d_{max}$:
5: $d_{max} = d_i$.
6: Výpočet směrnice σ.
7: Otočení všech hran budovy o $-\sigma$
8: Vytvoření minmax boxu
9: Otočení minmax boxu o σ
10: Úprava velikosti obsahu minmax boxu tak, aby platilo $S_{MMB} = S_b$

3 Vstupní a výstupní data

Data, která vstupují do programu jsou polygony (půdorysy budov) uložené ve formátu *.shp* (shapefile). Lze analyzovat budovy s konvexním i nekonvexním půdorysem, ale podmínkou je, že musí být v souřadnicovém systému S-JTSK. Pro ukázku jsou jako příloha k úloze připojeny soubory s názvy *Sumava.shp*, *Vinohrady_1.shp*, *Vinohrady_2.shp* a *polygon_with_hole.shp*, které obsahují různé tvary polygonů pro porovnání vhodnosti jednotlivých použitých algoritmů.

Výstupem programu je vizualizace generalizovaných polygonů budov z inputované vrstvy. Vizualizace se zobrazuje přímo v okně vytvořeného widgetu.

4 Ukázka vytvořené aplikace



Obrázek 3: Ukázka aplikace generalizace budov metodou Minimum Area Enclosing Rectangle

Na Obrázku 3 lze vidět výsledný widget, neboli grafický výstup vytvořeného kódu. Hlavní část tvoří okno, v němž se po nahrání souboru objeví polygony reprezentující budovy. Ke zmíněnému nahrání souboru *.shp* slouží ikona $InsertFile$, která nás přesměruje do souborového prohlížeče, ve kterém si zvolíme příslušný dataset. Pod ním se nachází *Combo Box* pomocí nějž si uživatel může vybrat jednu z metod, kterou chce pro generalizaci využít. V nabídce jsou tři - *Minimum BR*, *Wall Average* a *Longest edge*. Pod volbou metody se pak nachází tlačítko *Simplify*, jenž zahájí proces výpočtu vybrané metody pro generalizaci budov. Posledním dostupným tlačítkem je *Clear*. To uživateli zajistí odstranění vstupních a výstupních dat z hlavního okna widgetu.

5 Dokumentace

Skript $mainform.py$, přes který se program spouští, obsahuje třídu uživatelského rozhraní $Ui\_mainform$ navrženého pomocí SW *QTCreator*. Pro jednotlivá tlačítka má třída definované metody odkazující na skripty $draw.py$ a $algorithms.py$. Skript $draw.py$ obsahuje třídu $Draw$, která byla navržena pro vizualizaci objektů ve widgetu uživatelského rozhraní.

Parametry třídy $Draw$ jsou:

- seznam $self.polygons$ objektů typu $QPolygon$ vstupních polygonů,
- seznam $self.er$ obsahující generalizované objekty typu $QPolygon$,
- seznam $self.coordinates$ obsahující seznamy souřadnic vstupních polygonů,
- seznam $self.extent$ obsahující souřadnice minmax boxu celého území,
- seznam $self.canvas\_extent$ obsahující šířku a výšku widgetu v pixelech.

Metodami třídy $Draw$ jsou:

- $insertFile$ - načítá vstupní soubor ve formátu *.shp*, ukládá jeho souřadnice do seznamu $self.coordinates$ a hledá souřadnicové extrémy, které ukládá do seznamu $self.extent$
- $rescaleData$ - škáluje souřadnice polygonů tak, aby se celé území vešlo do okna widgetu,
- $paintEvent$ - vykresluje jednotlivé polygony do okna widgetu,
- $getPolygons$ - vrací seznam typu $QPolygon$ vstupních polygonů,
- $delPolygons$ - parametru $self.polygons$ přiřadí prázdný seznam,
- $setEnclosingRectangles$ - parametru $self.er$ přiřadí list polygonů
- $delEnclosingRectangles$ - parametru $self.er$ přiřadí prázdný seznam.

Skript $algorithms.py$ obsahuje třídu $Algorithms$, která byla vytvořena pro implementaci vybraných algoritmů.

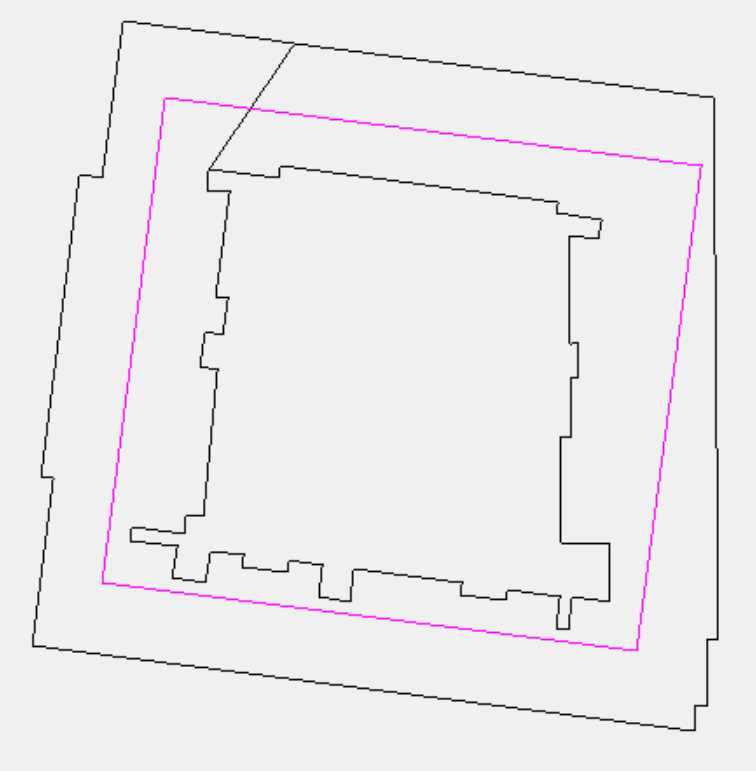
Metodami třídy $Algorithms$ jsou:

- $get2LinesAngle$ – počítá velikost úhlu svíraného mezi dvěma úsečkami,
- $calculateGain$ - počítá velikost směrnice úsečky,
- $createCH$ - vytváří konvexní obálku polygonu ve formě objektu $QPolygon$,
- $rotate$ - otáčí polygon kolem zadaného úhlu
- $minMaxBox$ - vytváří minmaxbox kolem zadaného polygonu, vrací jeho obsah a objekt $QPolygon$,
- $minAreaEnclosingRectangle$ - vytvoří zjednodušený tvar vstupního polygonu ve formě objektu třídy $QPolygon$ metodou $MinimumAreaEnclosingRectangle$,
- $getArea$ – vypočítá plochu obecného mnohoúhelníku pomocí LH vzorce,
- $resizeRectangle$ – změní velikost prvního vstupního polygonu tak, aby měl stejný obsah jako druhý vstupní polygon, a to změnami délek vektorů vedoucích z těžiště do jednotlivých vrcholů,
- $reduceGains$ – vypočítá hlavní směr polygonu redukováním směrnic jednotlivých hran,
- $wallAverage$ – vytvoří zjednodušený tvar vstupního polygonu ve formě objektu třídy $QPolygon$ metodou $WallAverage$,
- $longestEdge$ – vytvoří zjednodušený tvar vstupního polygonu ve formě objektu třídy $QPolygon$ metodou $LongestEdge$.

6 Výsledky

Tato kapitola se zaměřuje na prezentaci výsledků a porovnání výsledků generalizace pomocí třech zpracovávaných algoritmů.

První pokus byl proveden na shapefilu, který obsahuje kruhovou budovu. Pro všechny algoritmy byly výsledky takřka totožné, téměř se nelišily a dopadly velmi slušně (viz Obrázek 4).

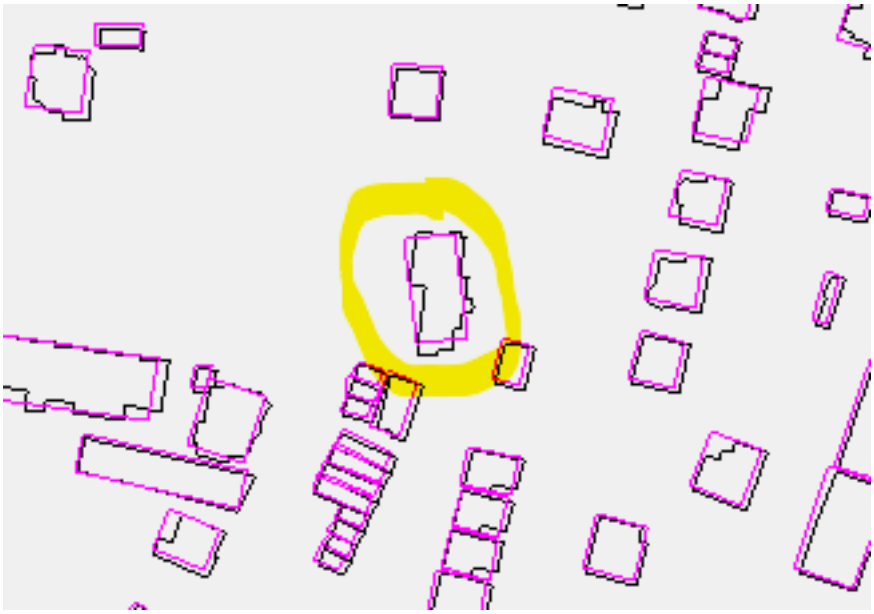


Obrázek 4: Kruhová budova, metoda Longest Edge

Obrázek 5: Vlevo metoda *Minimum Bounding Rectangle*, vpravo metoda *Wall Average*

Druhé porovnání se věnuje nepravidelné budově zhruba čtvercového tvaru, která se nachází na Vinohradech. Z porovnání je zřejmé, že uspokojivější výsledky poskytuje metoda *Wall Average*, která lépe zachovává tvar budovy a nekosí se, na rozdíl od výsledku *Minimum Bounding Rectangle* (viz Obrázek 5).

Obrázek 6 pak ukazuje na problém *Minimum Bounding Rectangle* s podlouhlými členitějšími budovami, kdy generalizovaný polygon se opět stáčí mimo zjevnou hlavní osu budovy. Naopak zbylé dvě metody vykreslily polygon orientovaný vhodným způsobem.



Obrázek 6: Problém *Minimum Bounding Rectangle* s podlouhlými členitými budovami

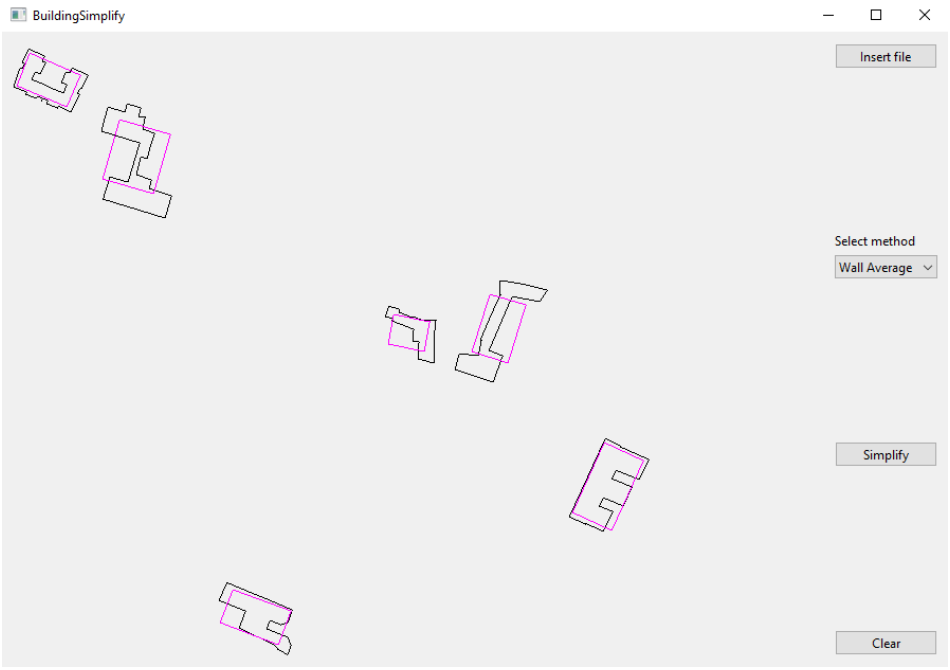
Závěrečné porovnání metod bylo učiněno na datasetu *chynov_vyber.shp* (Obrázek 7, 8 a 9). V něm se nachází budovy různých atypických tvarů, a tak byly na mnohých z nich dobře vidět rozdíly mezi jednotlivými algoritmy.

Nejhorší je zdá se situace u metody *Minimum Bounding Rectangle* v kombinaci s budovou ve tvaru Z. Ta v tomto případě vykresluje opravdu špatné výsledky.

Obrázek 7: *Minimum Bounding Rectangle* a specifické tvary budov

Překvapivě dobře si zase poradily všechny tři algoritmy s budovami tvaru písmen E, U a s budovou ve spodní části okna. Příliš se neliší ani výsledky pro malou budovu ve tvaru L uprostřed okna widgetu.

Ze všech poskytnutých výsledků je zřejmé, že pokud je budova jednoduchého a pravidelného tvaru (čtverec, obdélník) všechny analyzované algoritmy jsou schopné si s její generalizací poradit a dávají slušné výsledky. Problém nastává při nepravidelných tvarech budov.



Obrázek 8: *Wall Average* a specifické tvary budov

Zcela nejzřejmějším problémem je neschopnost metody *Minimum Bounding Rectangle* poradit si s budovami ve tvaru Z. Naopak velmi kladně lze hodnotit zbylé dvě metody u budov, které právě tento tvar mají.

Obrázek 9: *Longest Edge* a specifické tvary budov

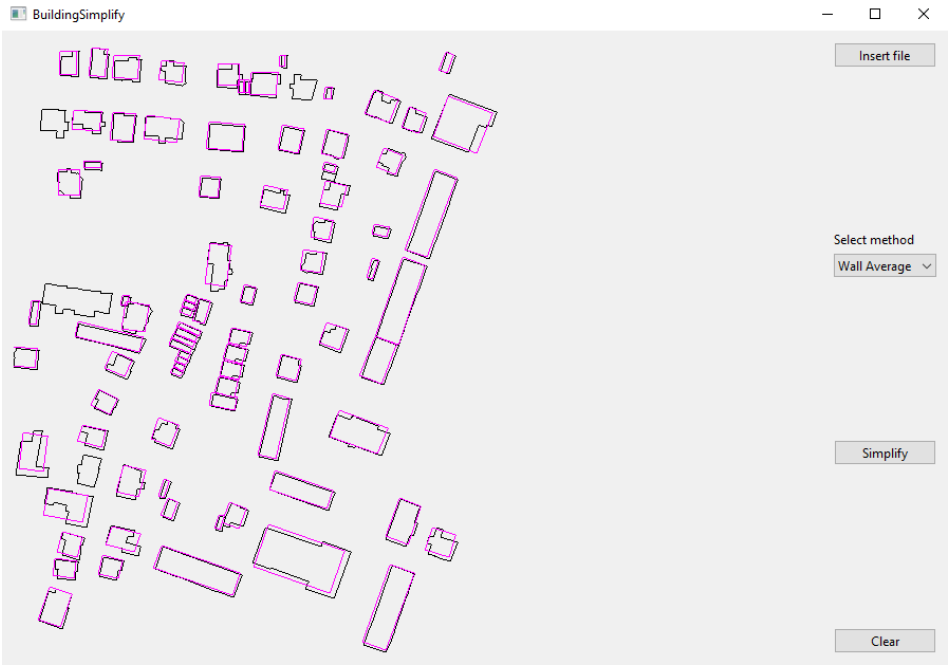
7 Závěr

Zadání práce včetně jednoho z bonusů bylo splněno, čímž se podařilo splnit stanovený cíl. Výsledkem je funkční aplikace, která generalizuje vstupní polygony pomocí tří různých metod. Všechny tyto metody navíc využívají funkčního algoritmu Jarvis Scan pro tvorbu konvexní obálky, bez níž by byla samotná generalizace neproveditelná.

Mimo jiné se také podařilo vizuálně i slovně porovnat vybrané metody. Kvalita porovnání by ale mohla ještě vzrůst, v případě že bychom měli obsáhlejší datasety s dalšími možnými tvary budov.

Program však skýtá mnohé možnosti na zdokonalení, které nebyly řešeny. První možností by byla implementace další z metod, které zajišťují vykreslení konvexní obálky. Další by pak bylo ošetření singulárního případu při generování konvexní obálky. Zcela největším zdokonalením programu by bylo sestrojení generalizačního algoritmu *Weighted Bisector*, který by poskytl další metodu vhodnou k porovnání, což by bylo jistě přínosem.

Další nedokonalostí je neuplná funkčnost algoritmu *Wall Average*, který u jednoho z datasetů nevizualizoval generalizované polygony u všech budov v datasetu (viz Obrázek 7).



Obrázek 10: Vlevo metoda *Minimum Bounding Rectangle*, vpravo metoda *Wall Average*

8 Seznam literatury

Zdrojový kód programu vznikl z velké části na základě informací z přednášky a cvičení vedených doc. Bayerem (2022). Informace byly čerpány z prezenčních lekcí a zároveň z prezentace dostupné na odkazu https://web.natur.cuni.cz/ bayertom/images/courses/Adk/adk4.pdf (cit. 2. 4. 2022)

Duchene, C., Bard, S., Barillot, X., Ruas, A., Trevisan, J., Holzapfel, F. (2003): Quantitative and qualitative description of building orientation. Institut Géographique National, COGIT Laboratory.

Jayaram, M., A., Fleyeh, H. (2016): Convex Hulls in Image Processing: A Scoping Review. American Journal of Intelligent Systems 2016, 6, 2, 48-58

Lee, J., Jang, H., Yang, J., Yu, K.(2017): Machine Learning Classification of Buildings for Map Generalization. ISPRS Int. J. Geo-Inf, 6, 10, 309.