ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ
PROGRAM GEODÉZIE A KARTOGRAFIE
OBOR GEOMATIKA

DIPLOMOVÁ PRÁCE

ROZŠÍŘENÍ ZÁSUVNÉHO MODULU QGIS PRO ZPRACOVÁNÍ GTFS O VIZUALIZACI TARIFNÍCH PÁSEM PID

Vedoucí práce: Ing. Martin Landa, Ph.D.
Katedra geomatiky

červen 2021

Bc. Martin KOUBA

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: Kouba Jméno: Martin Osobní číslo: 468232

Zadávající katedra: Katedra geomatiky

Studijní program: Geodézie a kartografie

Studijní obor: Geomatika

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce: Rozšíření zásuvného modulu QGIS pro zpracování dat GTFS o vizualizaci tarifních pásem PID

Název diplomové práce anglicky: QGIS GTFS Plugin extension with visualization of PID tariff zones

Pokyny pro vypracování:
Cílem diplomové práce je rozšířit zásuvný modul (plugin) platformy QGIS pro zpracování dat ve formátu GTFS, na jehož vývoji se autor spolupodílel v rámci semestrálního projektu předmětu Free Software GIS ve 2. semestru magisterského studia. Primárním úkolem je najít a vhodně implementovat způsob vizualizace tarifních pásem Pražské integrované dopravy (PID) generovaných automaticky z podkladových dat.

Seznam doporučené literatury:
SHERMAN, Gary. The PyQGIS Programmer's Guide: Extending QGIS 3 with Python 3. Ilustrované vydání. Locate Press, 2018. ISBN 9780998547725.
MARK, Pilgrim. Dive into Python 3. Apress, 2009. ISBN 978-1430224150.

Jméno vedoucího diplomové práce: Ing. Martin Landa, Ph.D.

Datum zadání diplomové práce: 10.2.2021 Termín odevzdání diplomové práce: 17.5.2021
Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku

Podpis vedoucího práce Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

Beru na vědomí, že jsem povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v diplomové práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce“ a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací“.

Datum převzetí zadání Podpis studenta(ky)

ABSTRAKT

v ceskem jazyce

KLÍČOVÁ SLOVA

Python, QGIS, zásuvný modul, GTFS, PyQGIS

ABSTRACT

v anglickem jazyce

KEYWORDS

Python, QGIS, plugin, GTFS, PyQGIS

PROHLÁŠENÍ

Prohlašuji, že diplomovou práci na téma „Rozšíření zásuvného modulu QGIS pro zpracování GTFS o vizualizaci tarifních pásem PID “ jsem vypracoval samostatně. Použitou literaturu a podkladové materiály uvádím v seznamu zdrojů.

V Praze dne ....

(podpis autora)

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat panu Ing. Martinu Landovi, Ph.D. za pomoc při ..... Zvlášť bych chtěl poděkovat celé své rodině .....

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Pro cestování po Praze a okolí je často nejlepší volbou použít veřejnou dopravu. Tu zahrnuje metro, tramvaje, železnici, městské a příměstské autobusové linky, lanovou dráhu na Petřín a také některé přívozy. Celý tento systém se nazývá Pražská integrovaná doprava, který je postupně integrován společnými přepravními a tarifními podmínkami a jednotným dopravním řešením včetně koordinace jízdních řádů.

Systém Pražské integrované dopravy je rozdělen do dvanácti tarifních pásem, pro které platí různé jízdní doklady. Rozlohově jsou po celé Praze, na většině Středočeského kraje a dokonce i na částech Plzeňského a Ústeckého kraje, či na Vysočině.

Tarifní pásma Pražské integrované dopravy jsou zatím modelována manuálně a nejsou nikterak zautomatizována. Cílem této diplomové práce bude vytvořit postup pro automatické vyhotovení tarifních pásem a co nejvíce se přiblížit k jejich oficiální podobě. Tento postup bude publikován jako pokračování vývoje zásuvného modulu v softwaru QGIS, u kterého byla započata tvorba v předmětu Free software GIS v 5. semestru magisterského studia.

1 Rešerše

Tarifní pásma Pražské integrované dopravy jsou mimo jiné publikována jako shapefile [1] na portálu *opendata hlavního města Prahy* [2]. Tento shapefile obsahuje vektorové vrstvy polygonů.

První myšlenkou, jak se přiblížit k takovým polygonům, byl algoritmus Delaunay triangulace. Tento algoritmus vytváří liniové spojnice bodů mez jednotlivými body, které si jsou sobě nejblíž, za pomocí opsaných kružnic. Delaunay triangulace má několik vlastností. Jednou z nich je například, že uvnitř kružnice k opsané libovolnému trojúhelníku neleží žádný jiný bod. Zároveň Delaunay triangulace je jednoznačná, pokud žádné čtyři body neleží na kružnici. [3]

Další nápadem byla aplikace Voronoi diagram algoritmu, což je způsob dekompozice metrického prostoru určený vzdálenostmi k dané diskrétní množině bodů v prostoru. Tento algoritmus má taky několik vlastností, jež budou prezentovány v dalších kapitolách (viz 5.2).

[1] formát vektorového datového úložiště Esri pro ukládání umístění, tvaru a atributů geografických prvků [1]

2 GTFS

General Transit Feed Specification, zkráceně GTFS, je dokument, který definuje obecný formát pro jízdní řády veřejné dopravy a příbuzné geografické informace. GTFS „feeds“ umožňují veřejným dopravním agenturám zveřejňovat svá přepravní data a vývojářům psát aplikace, které tato data spotřebovávají interoperabilním způsobem. [4]

2.1 Historie GTFS

V Portlandu ve státě Oregon v USA se společnost TriMet pokusila jako jedna z prvních řešit problém s plánováním tranzitní dopravy pomocí otevřených dat sdílených širokou veřejností. V roce 2005 společnost TriMet oslovila společnost Google s dotazem, zda mají nějaké plány na začlenění tranzitní dopravy do svých plánovačů výletů na základě veřejných údajů TriMet. Google jim kladně odpověděl a spojily síly s implementací jednoho z prvních plánovačů výletů v Portlandu.

Jedním z prvních problémů, kterým TriMet a Google čelily, byl problém udržitelných dat - pro zajištění kvalitních cest by plánovač cest potřeboval kvalitní přepravní řád, trasu a údaje o zastávkách v elektronickém formátu, který by byl neustále aktuální. Společnost TriMet ve spolupráci se společností Google naformátovala svá přepravní data do snadno udržovatelného a spotřebního formátu, který lze importovat do Map Google. Tento formát dat přepravy se stal známým jako Specifikace zdroje Google Transit (anglicky Google Transit Feed Specification (GTFS)). V roce 2005 byla tato služba plánování cesty spuštěna jako Google Transit.

Od tohoto roku se GTFS stal nejpopulárnějším datovým formátem pro přepravní služby na světě. Spousta agentur se rozhodla sdílet své GTFS údaje s veřejností, zatímco některé agentury zůstaly zdrženlivé a přístup k datům nechaly jen některým partnerům. Ke 2. prosinci 2019 uvádí OpenMobilityData 1233 poskytovatelů s veřejně přístupnými kanály GTFS, z nichž 465 je ve Spojených státech.

V důsledku inovací vývojářů třetích stran jsou data GTFS nyní využívána různými softwarovými aplikacemi třetích stran k mnoha různým účelům, včetně plánování výletů, map, vytváření jízdních řádů, mobilních dat, vizualizace, přístupnosti, analytických nástrojů pro plánování a informační systémy v reálném čase. V roce 2010

byl název formátu GTFS změněn na General Transit Feed Specification, aby přesně reprezentoval jeho použití v mnoha různých aplikacích mimo produkty Google. [5]

2.2 GTFS dataset

GTFS dataset[2] obsahuje CSV soubory, jehož úplný seznam je v následující tabulce. Co je to CSV soubor?

CSV, *Comma-separated values*, v překladu *hodnoty oddělené čárkami*, je jednoduchý souborový formát určený pro výměnu tabulkových dat. Data jsou oddělována „oddělovačem“. Ačkoli podle definice by měl být formát oddělen čárkami, oddělovač může být prakticky cokoli. Nejčastějšími oddělovači jsou právě čárky, středníky nebo mezery. CSV soubor se může upravovat v jakémkoliv textovém editoru.

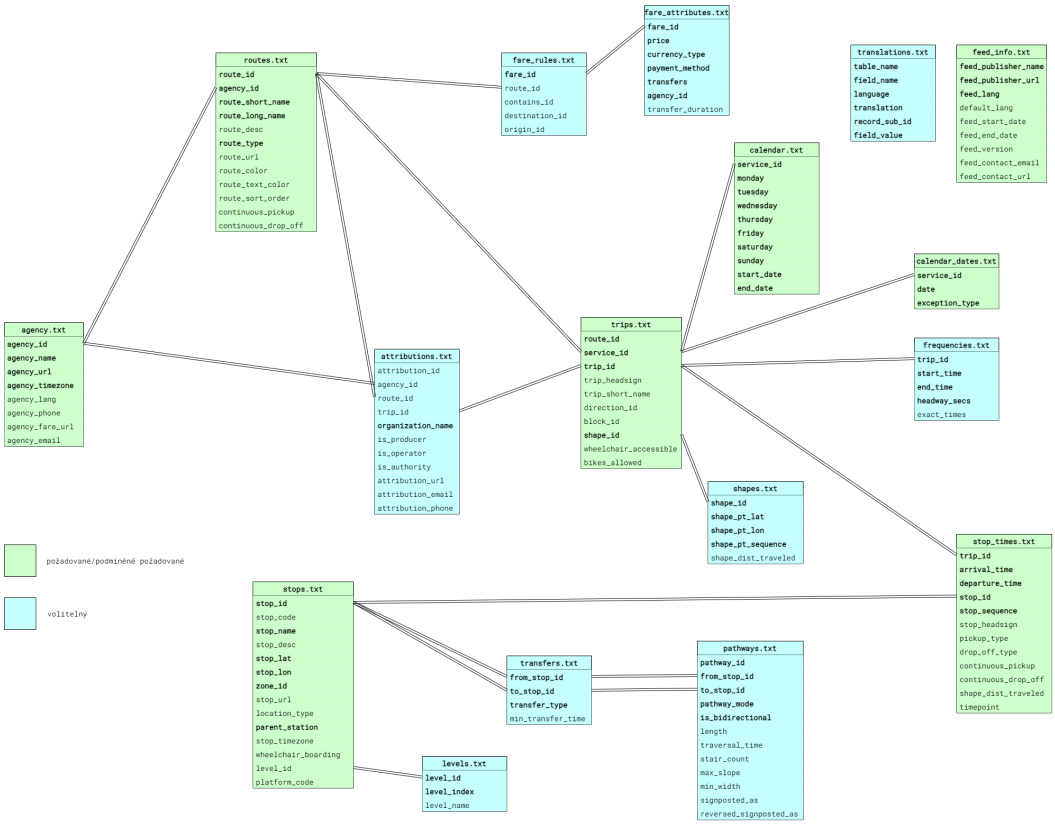
```
shape_id,shape_pt_lat,shape_pt_lon,shape_pt_sequence,shape_dist_traveled
L991V1,50.07553,14.51539,1,0.00000
L991V1,50.07371,14.51461,2,0.20965
L991V1,50.07361,14.51456,3,0.22173
L991V1,50.07351,14.51451,4,0.23382
L991V1,50.07340,14.51447,5,0.24552
L991V1,50.07329,14.51444,6,0.25817
L991V1,50.07319,14.51441,7,0.26988
L991V1,50.07308,14.51437,8,0.28158
L991V1,50.07298,14.51435,9,0.29298
L991V1,50.07288,14.51433,10,0.30438
L991V1,50.07278,14.51431,11,0.31579
L991V1,50.07269,14.51429,12,0.32623
```

Obrázek 2.1: Ukázka CSV formátu ze souboru shapes.txt

V GTFS datasetu je určité množství povinných a nepovinných CSV souborů v textové podobě.

Tučně zvýrazněné pole jsou požadované či podmíněně vyžadované. Ostatní pole jsou volitelné.

[2]kolekce dat, která by měla být publikována na permanantní URL adrese



Obrázek 2.2: Diagram GTFS datasetu

3 Pražská Integrovaná doprava (PID)

Pražská integrovaná doprava (PID) je dopravní systém, který zahrnuje jak pozemní dopravu (tramvaje, železnici, městské a příměstské autobusové linky, lanovou dráhu na Petřín), tak i tu podzemní (metro). Tento dopravní systém zahrnuje i některé přívozy. Systém probíhá integrací společnými přepravními a tarifními podmínkami a jednotným dopravním řešením včetně koordinace jízdních řádů. [6]

Obrázek 3.1: Logo Pražské integrované dopravy [6]

3.1 ROPID

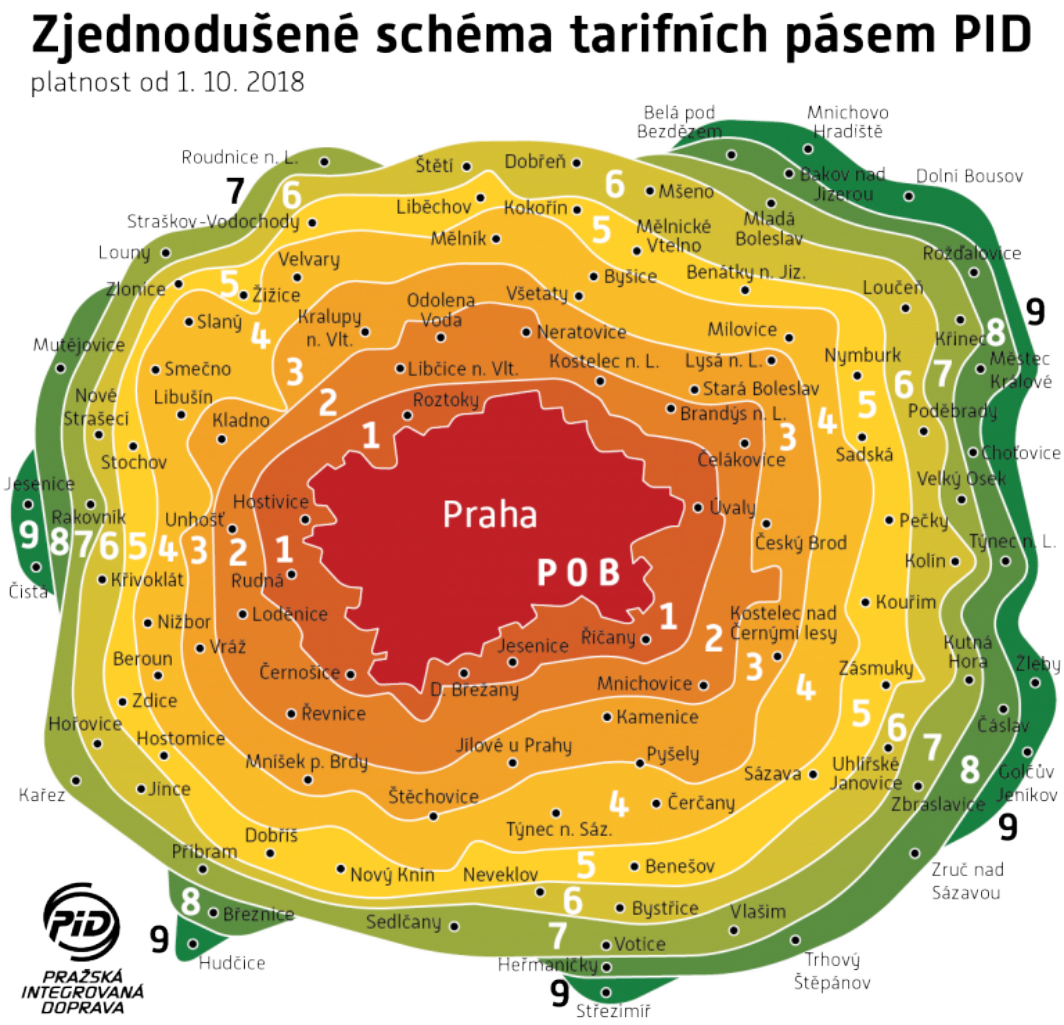
Chod integrace systému zajišťuje Regionální organizátor pražské integrované dopravy (zkráceně ROPID), což je příspěvková organizace hlavního města Prahy. Jeho úloha je organizační a kontrolní a ze své práce se odpovídá orgánům samosprávy a státní správy, které jej zabezpečením dopravy pověřily.

Obrázek 3.2: Logo ROPIDu [6]

ROPID se zabývá vytvářením, rozvíjení, a udržováním systému Pražské integrované dopravy v Praze a okolí, včetně návazností na jiné systémy jako jsou Integrovaná doprava Plzeňského kraje, Doprava Ústeckého kraje, Integrovaný dopravní systém Libereckého kraje nebo Integrovaná regionální doprava Královéhradeckého a Pardubického kraje. Taktéž se zabývá vytvářením zásad a standardů dopravní obsluhy a jejich aplikace v závislosti na dostupných finančních zdrojích a jejich projednání s obcemi, okresními úřady a dopravci. ROPID výbírá dopravce, uzavírání

smluv o závazku veřejné služby jménem města Prahy k zajištění provozu PID s dotčenými obcemi, Středočeským krajem a dopravci a kontrola jejich plnění. Náplní ROPIDu jsou i organizace finančních toků v systému PID, návrh tarifu a jízdného v systému PID a zajištění jednotnosti informačního systému PID. [7]

3.2 Tarifní pásma PID



Obrázek 3.3: Schéma tarifních pásem PID [6]

4 Technologie tvorby

4.1 QGIS

Obrázek 4.1: Logo QGISu [8]

4.2 Python

Python je moderní, dynamický, skriptovací programovací jazyk s veřejným zdrojovým kódem (tzv. open-source). Je snadný na naučení,lehko čitelný a v současné době velmi populární. Tento programovací jazyk využívají softwaroví inženýři, matematici, datovi analytici, vědci, účetní a síťoví inženýři. Dokonce kvůli jednoduchosti tohoto jazyka ho využívají i děti na základních školách.

Důvodů, proč je tento programovací jazyk tak populární, je mnoho. Python je vyšší programovací jazyk, což znamená lépší srozumitelnost než nižších programovacích jazyků a programy zapsané ve vyšších jazycích jsou obvykle kratší a lépe čitelné. Další důvodem je multiplatformnost, což znamená, že se můžou Pythoní aplikace sestavit a běžet na různých platformách jako Windows, Mac či Linux. Komunita Pythonu je obrovská, což je další výhodou tohoto jazyka. Například jenom v Čechách je mnoho skupin na různých sociálních sítích. Python má také spoustu knihoven, frameworků a nástrojů, které lidem usnadňují programování.

Podporuje různá programovací paradigmata [3] jako například objektově orientované, imperativní, procedurální nebo funkcionální.

[3] základní programovací styl, který se liší v pojmech a abstrakcích, které tvoří jednotlivé prvky programu, a krocích, ze kterých se výpočet skládá - objektově orientované, imperativní, procedurální nebo funkcionální [9]

Obrázek 4.2: Logo Pythonu [10]

4.3 PyQt5

Obrázek 4.3: Schéma PyQt5

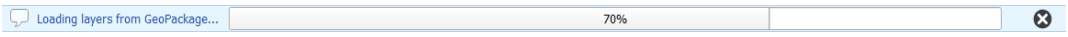
5 Postup

5.1 Aktualizace pluginu do verze 1.0.0

Po dokončení pluginu v předmětu Free Software GIS měl plugin veškerou základní funkcionalitu, kterou měl mít. Tou bylo načítání GTFS ZIP souboru do QGISu, rozbalení ZIP souboru, načtení CSV souborů do geodatabázového kontajneru GeoPackage, vytvoření vektorových vrstev pro soubory *stops.txt* a *shapes.txt*, obarvení vektorové vrty *shapes.txt* a vložení do CSV souborů a vrstev do *layer tree*.

Avšak pro uvedení do "ostrého" provozu musel být plugin více uživatelsky přívětivější. Proto byl veškerý proces přesunut na pozadí, aby celý QGIS program "nezamrzal" a mohla se při jeho výpočtech provádět i jiné akce. To se provedlo díky python třídě *QgsTask* a jejím metodám, které byly zděděny z této třídy. [11]

Pro zobrazování procesu během výpočtu byla použita třída *QProgressBar* a její metody. Zobrazení postupu bylo implementováno do lišty zpráv QGISu spolu s chybovými hláškami.



Obrázek 5.1: ProgressBar v liště QGISu

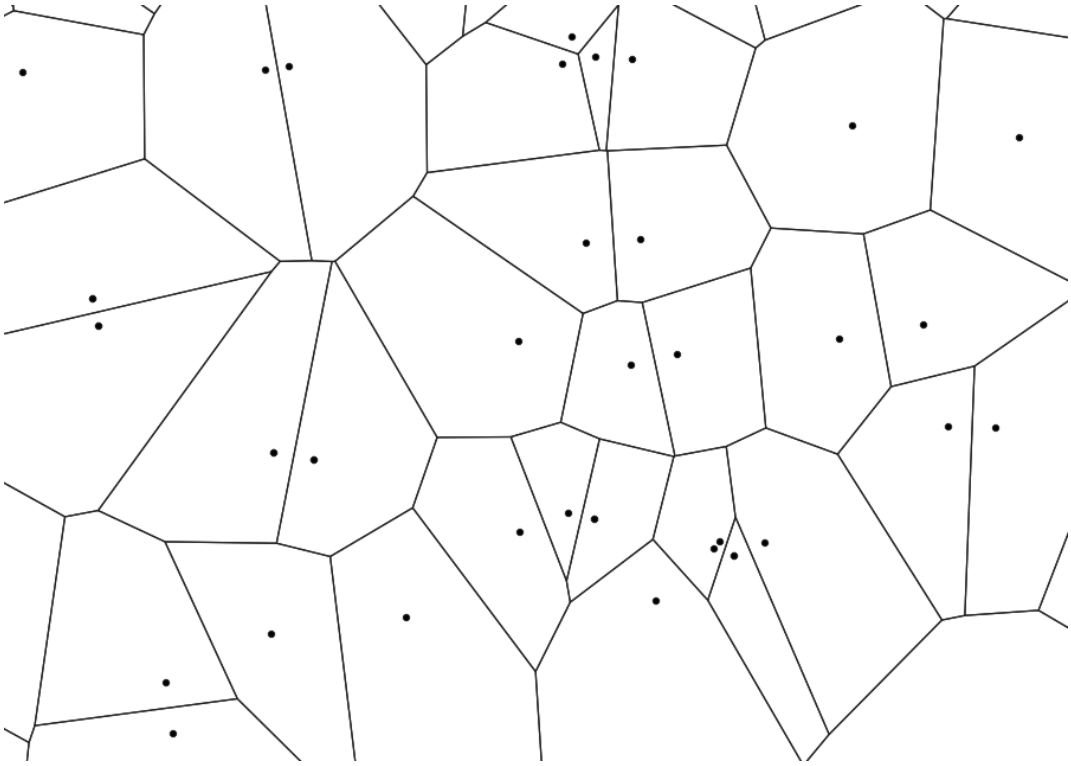
5.2 Postup 1 - pomocí nástrojů QGIS

GTFS obsahuje povinný CSV soubor *stops.txt*. Tento soubor obsahuje mimo jiných polí také pole *zone_id*, které bylo při tvorbě tarifních pásem klíčové. Pole *zone_id* má datový typ *string* a znamená, ve kterém tarifním pásmu daná zastávka leží. Verze GTFS Loaderu 1.0.0 soubor *stops.txt* převádí do vektorové vrstvy ve formě bodů. Z této bodové vrstvy byla vytvořena vektorová vrstva Voroného diagramů nástrojem *Voroného polygony* v programu QGIS.

Pro použití QGIS nástrojů v pythoním skriptu je potřeba importovat modul *processing*, který má funkci *run*, do které se vkládají dva parametry. První parametr je ID nástroje ve formě *stringu* a druhý je *slovník* vstupních parametrů. Vstupní parametry se lze dozvědět z QGIS dokumentace. [12]

Vstupem do tohoto nástroje je vektorová vrstva stops a výstupem jsou právě Voroného polygony.

Voroného diagramy, někdy pod názvy jako Voroného teselace, Voroného dekompozice, Thiessenovy polygony nebo Dirichletova teselace, podle definice představují rozklad množiny bodů P na n uzavřených či otevřených oblastí $V(p) = \{V(p_i), V(p_2), ..., V(p_n)\}$ takových, že každý bod q náléžící množině $V(p_i)$ je blíže k bodu p_i než k jakémukoliv bodu p_j náležící množině P. [13]



Obrázek 5.2: Voroného polygony

Vlastnosti Voroného diagramů

- Voronoi diagram $V(P)$ je planárním grafem.
- Vrchol q Voronoi buňky $\{V(p_i)$ je průnikem 3 hran, právě když je $V(P)$ nedegenerovaný.
- Pokud p_i náležící $H(P)$, pak je $V(p_i)$ otevřený.
- Pro každý bod p_i *náležící* P je $V(P)$ konvexní.
- Bod p_i je nejbližším bodem bodu p jestliže p náleží $\{V(p_i)$.
- Každá strana q_iq_j, *i se nerovná j*, je sdílena právě dvěma sousedními buňkami $V(p)$.

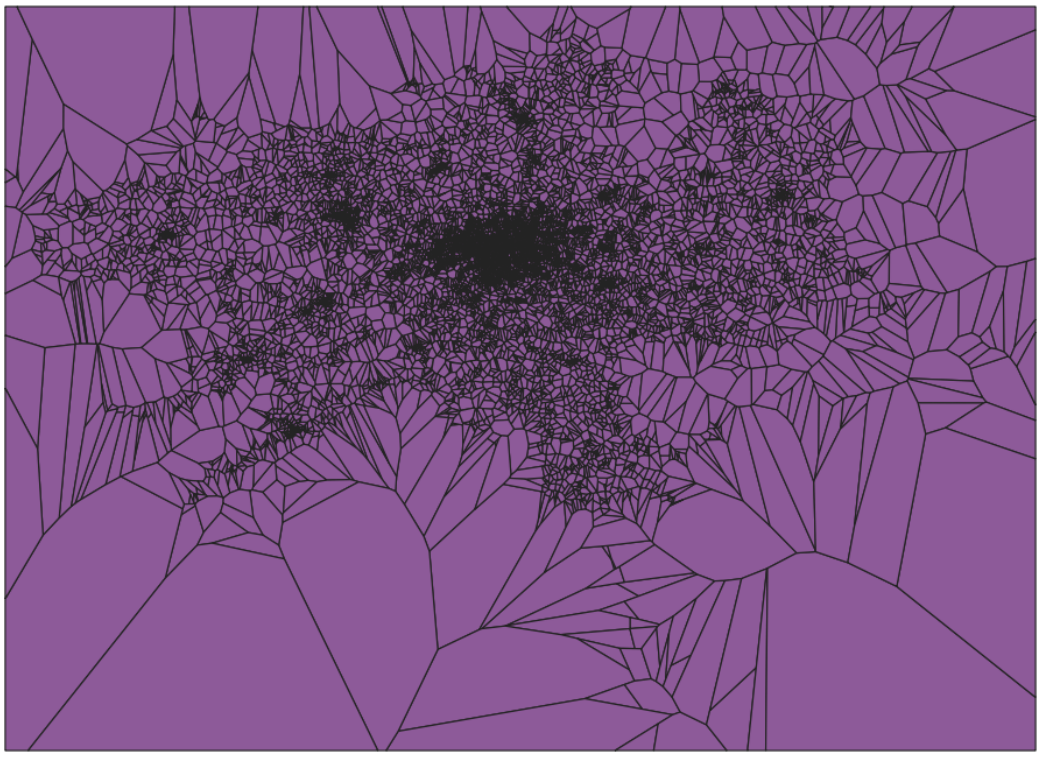
- Bod q je vrcholem *V(p)*, pokud existuje kružnice *k(q,r)* procházející třemi nebo více generátory p_i, p_j, p_k, a neobsahuje žádný další bod P (spojitost s *DT(P)*).
- Kružnici *k(q,r)* označujeme jako největší prázdnou kružnici ze všech prázdných kružnic se středem v bodě *q*.
- Průměrné množství Voronoi hran ve Voronoi polygonu nepřekročí hodnotu 6.
- Vztah mezi počtem bodů n, počtem hran n_h a počtem trojúhelníků n_t teselace *V(P)*:

$$n_h \leq 3n - 6$$

$$n_t \leq 2n - 5$$

- Voronoi diagram *V(P)* představuje ortografickou projekci stěn mnohostěnu tvořeného průsečnicemi všech polorovin A_i do roviny *xy*.
- Nechť bod p_i^* i představuje ortografický průmět bodu p_i na povrch paraboloidu daného rovnicí:

$$z = x^2 + y^2$$



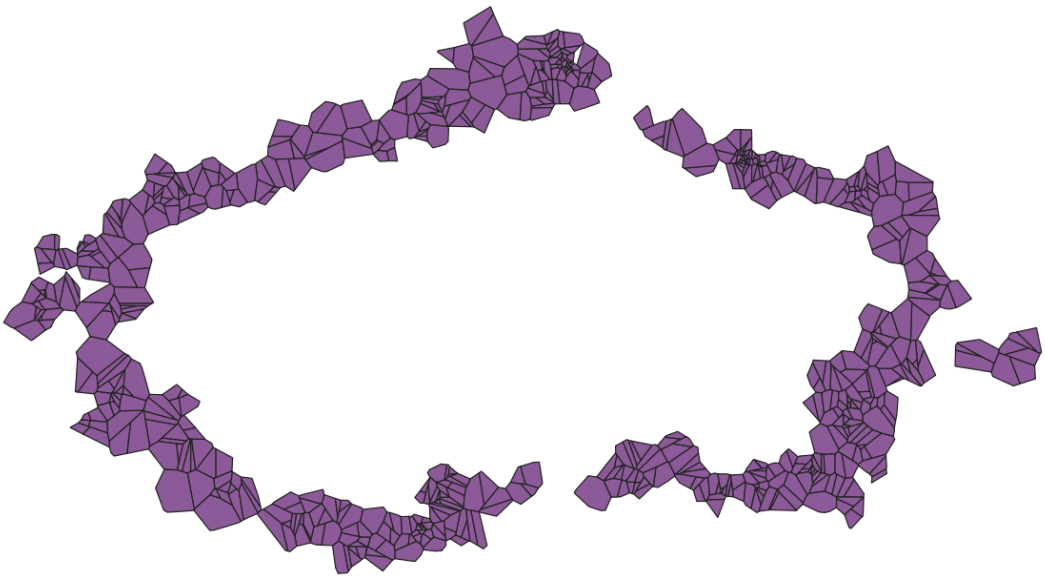
Obrázek 5.3: Voroného polygony pro všechny zastávky

Pro každé tarifní pásmo byly vybrány zastávky pomocí třídy *QgsVectorLayer* a její metody *selectByExpression*, která vybírá prvky podle zadaného výrazu ve formě *string*. Znění zadaného výrazu je následující:

zone_id in "tarifní pásmo" and location_type = 0

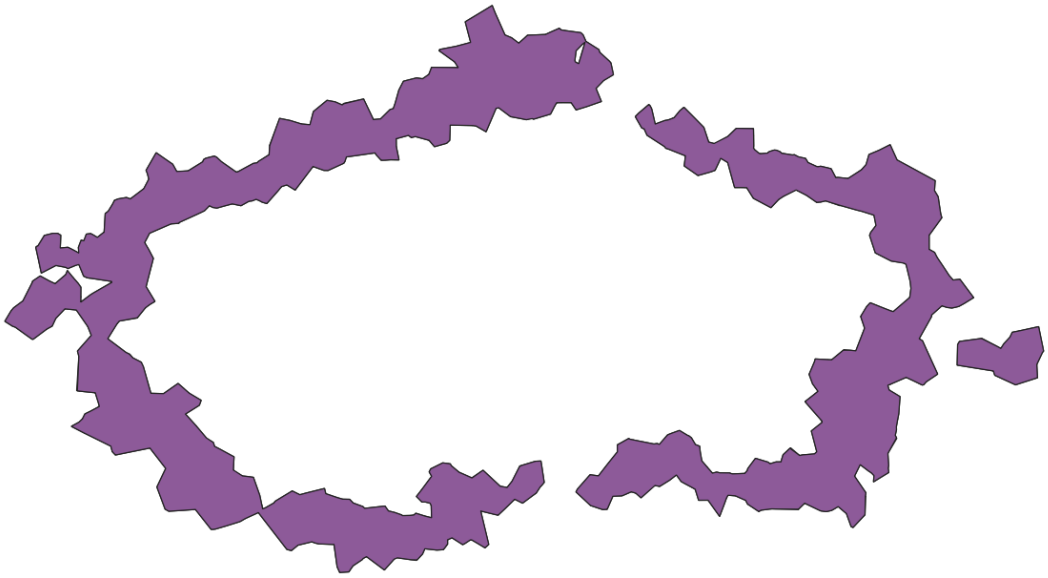
V zadávaném výrazu figuruje taktéž údaj o poli *location_type*, což je typ lokace. Hodnota nuly (nebo prázdná hodnota) je právě lokace zastávky. Třída *QgsVectorLayer* představuje zároveň vektorovou vrstvu, která spravuje vektorové datové sady a která může být považována jako vstup do nástroje QGIS.

Pro vybrané zástavky bylo potřeba vybrat ty Voroného polygony, které svou pozicí dané zastávky protínaly. To bylo provedeno nástrojem *Select by location*, do kterého vstupovala vrstva vybraných zastávek a vrstva Voroného polygonů. Výsledkem tohoto nástroje byla vektorová vrstva vybraných Voroného polygonů. Jako příklad v obrázku zde budu uvádět tarifní pásmo 2.



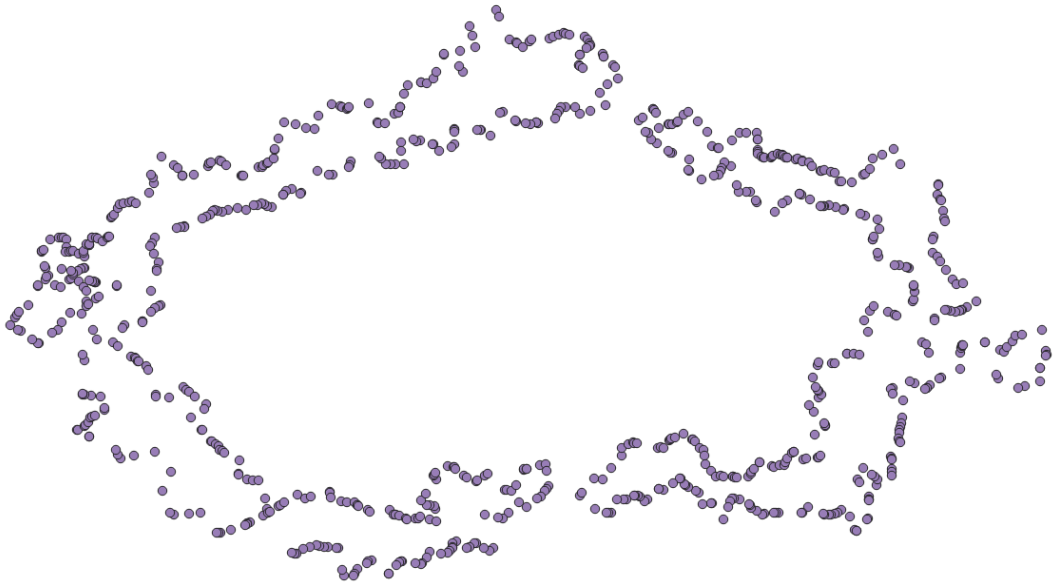
Obrázek 5.4: Vybrané Voroného polygony pro pásmo 2

Tyto polygony byly následně nástrojem *Dissolve* spojeny do jednoho společného polygonu. Vstupem tohoto nástroje byl výstup nástroje *Select by location* a výstupem byl polygon s jednou geometrií.



Obrázek 5.5: Výsledek nástroje Dissolve pro pásmo 2

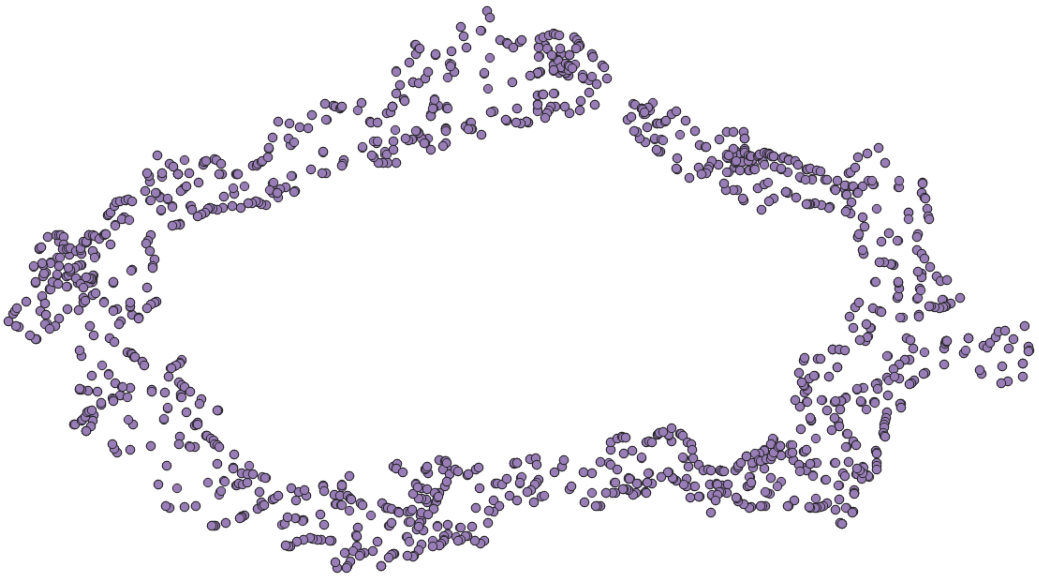
Ze spojených polygonů byla poté vygenerována vektorová vrstva bodů nástrojem *Extract vertices*, která představovala vrcholy spojených polygonů. Pro tuto vrstvu bylo vstupem výstup nástroje *Dissolve* a výstupem byl vektorová vrstva bodů doplněná o pole (mimo původních polí z vektorové vrstvy *stops*) jako *vertex_index, vertex_part, vertex_part_ring, distance* a *angle*. Hodnoty těchto polí avšak nebyly využity v dalším výpočtu.



Obrázek 5.6: Výsledek nástroje Extract vertices pro pásmo 2

Dále byly s pomocí třídy *QgsVectorLayer* a její metody *selectByExpression* vybrány z původní vektorové vrstvy *stops* ty zastávky, které obsahovaly v poli *zone_id* hodnotu tarifního pásma 1,2 nebo 2,3. Takové zastávky ležely na hranici pásma a polygon tarifního pásma měl skrz ně vézt hranici. Z vybraných zastávek byla vytvořena vlastní vektorová vrstva.

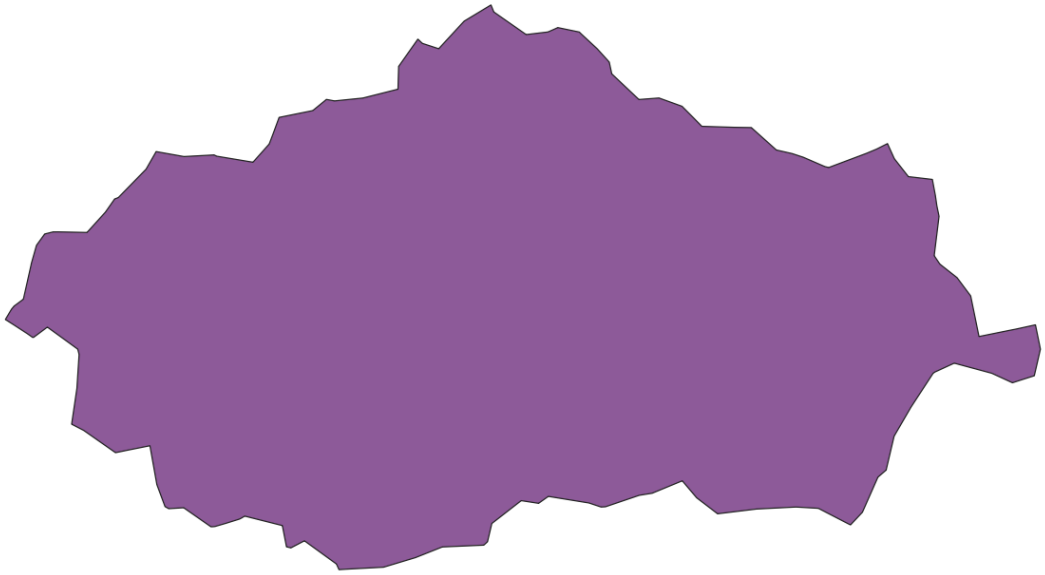
Následně byly spojeny vektorové vrstvy hraniční zastávek, zastávek uvnitř tarifního pásma 2 a bodů z výstupu nástroje *Extract vertices* nástrojem *Merge vector layers*. Tyto tři zmiňované vektorové vrstvy byly vstupem do tohoto nástroje a výstupem byla vektorová vrstva spojených bodů.



Obrázek 5.7: Spojené body třech vektorových vrstev pro pásmo 2

Ze spojených bodů byla vytvořena konkávní obálka. To bylo provedeno nástrojem Concave hull (alpha shapes), který počítá konkávní obálku z vstupních bodů. Vstupem tedy byla vrstva spojených bodů, což byl první parametr nástroje. Dalším parametrem byl *Práh (Threshold)* s datovým typem *čísla*, který byl volen od 0 do 1, kdy 0 znamenala maximum konkávní obálky a 1 konvexní obálky. Po několika testovacích spuštění byla určena hodnota 0,09 jako nejlepší hodnotou pro tvorbu tarifních pásem. Dalším parametrem bylo *Povolení děr (Allow holes)* s datovým typem *boolean*, který byl nastaven na *False*. Posledním parametrem bylo Rozdělit

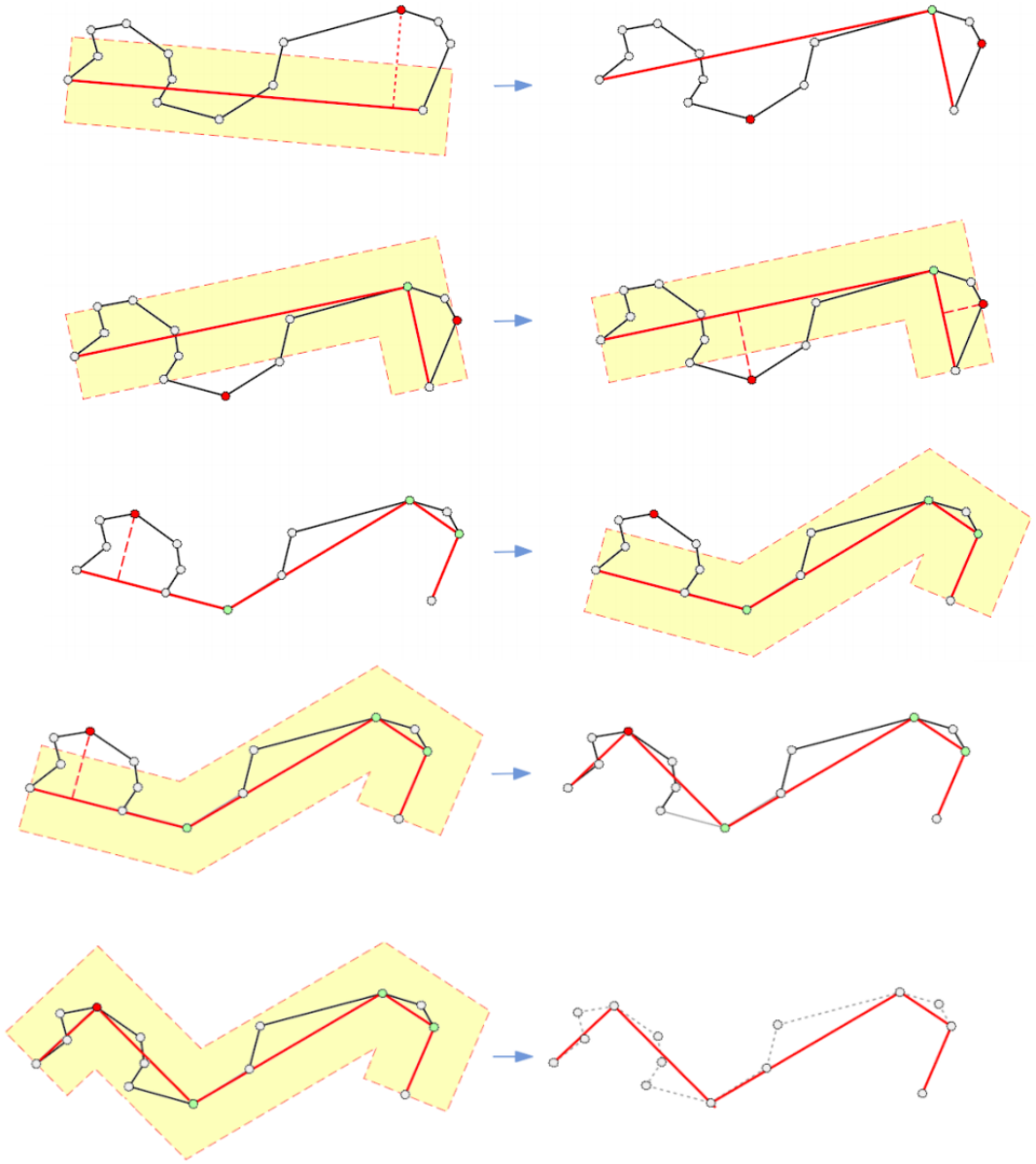
vícedílnou geometrii na jednotlivé části (Split multipart geometry into singlepart geometries) taktéž s datovým typem *boolean*, který byl nastaven na *True*. Výstupem byla polygonová vrstva konkávní obálky.



Obrázek 5.8: Konkávní obálka pro pásmo 2

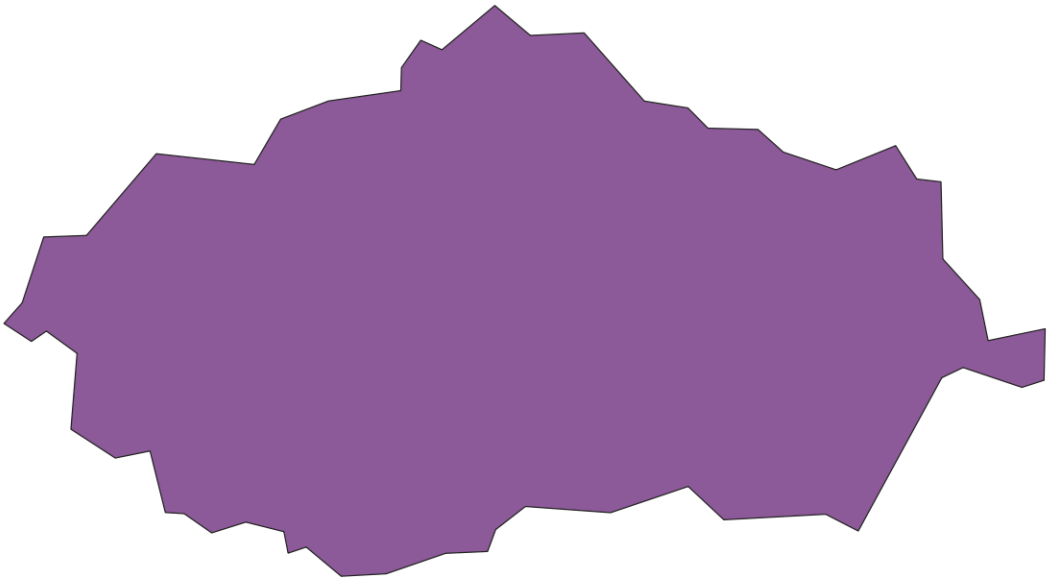
Z konkávní obálky byla pomocí nástroje Simplify "zjednodušena" geometrie polygonové vrstvy. Tento nástroj využívá tři druhy zjednodušení vektorové vrstvy: založené na vzdálenosti (algoritmus „Douglas-Peucker“), založené na ploše (algoritmus „Visvalingam“) a přichytávání geometrií k mřížce. Pro můj postup byl zvolen první volba zjednodušení, pomocí algoritmu „Douglas-Peucker“.

Algoritmus „Douglas-Peucker“ na rozdíl od ostatních algoritmů neodstraňuje vrcholy nesplňující geometrickou podmínku, ale přidává do ní postupně kritické body, které podmínku splnují. Tento algoritmus je jeden z nejlepších generalizacních algoritmů a je velmi často implementován v GIS software. [14]



Obrázek 5.9: Znázornění algoritmu „Douglas-Peucker“ [14]

Do nástroje Simplify byl vstupem výsledek nástroje Concave hull (alpha shapes), parametrem byl výběr algoritmu a nastavení hodnoty tolerance vzdálenosti. Výstupem byla polygonová vektorová vrstva.



Obrázek 5.10: Výsledek nástroje Simplify pro pásmo 2

Výstup nástroje Simplify byl vstupem do nástroje Smooth. Tento nástroj vyhlazuje geometrie liniové nebo polygonové vrstvy pomocí Chaikinova algoritmu.

Chaikin využil fixních poměrů na odříznutí rohů, takže byly všechny rozřezány stejně. Při matematickém zápisu postupuje Chaikinova metoda následovně: Dostaneme kontrolní polygon $\{P_0, P_1, ..., P_n\}$, tento kontrolní polygon vylepšíme vygenerováním nové posloupnosti řídicích bodů

$$\{Q_0, R_0, Q_1, R_1, ..., Q_{n-1}, R_{n-1}\}$$

kde každá nová dvojice bodů Q_i, R_i je třeba brát v poměru $\frac{1}{4}$ a $\frac{3}{4}$ mezi koncovými body segmentu čáry $\overline{P_iP_{i+1}}$.

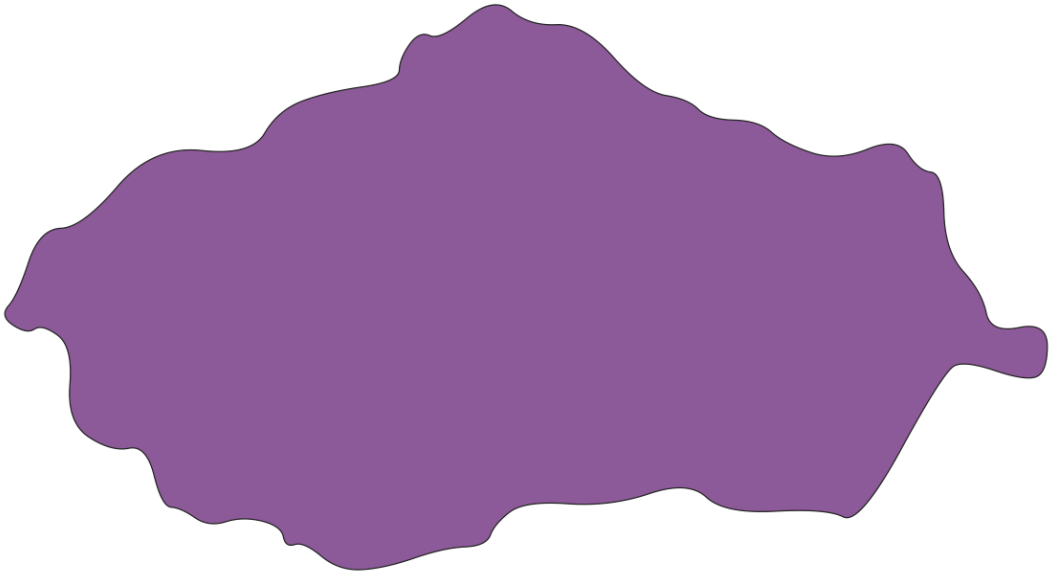
$$Q_i = \frac{3}{4}P_i + \frac{1}{4}P_{i+1}$$

$$R_i = \frac{1}{4}P_i + \frac{3}{4}P_{i+1}$$

Tyto 2 nové body lze považovat za nový řídicí polygon - vylepšení původního řídicího polygonu. [15]

U nástroje Smooth byly zvoleny tři parametry - počet iterací, offset a maximální úhel. Počet iterací znamená, kolik vyhlazovacích iterací bude použito pro každou geometrii. Hodnota počtu iterací byla nastavena na 10, což byla maximální volitelná hodnota (pro co nevětší hladkost). Parametr offset znamená, jak "těsně" vyhlazené

geometrie sledují původní geometrie. Zde byla ponechána výchozí hodnota 0,25. A poslední parametr maximálního úhlu lze použít k zabránění vyhlazení uzlů s velkými úhly. Zde byla také ponechána výchozí hodnota 180°. Výstupem nástroje byla polygonová vektorová vrstva.



Obrázek 5.11: Výsledek nástroje Smooth pro pásmo 2

Závěr

Odkazy

[1] Esri. Shapefiles. [online], [cit. 2021-04-12]. Dostupné z: https://doc.arcgis.com/en/arcgis-online/reference/shapefiles.htm.

[2] Opendata Praha. Pražská integrovaná doprava - tarifní pásma. [online], [cit. 2021-04-12]. Dostupné z: https://opendata.praha.eu/dataset/ipr-prazska_integrovana_doprava_-_tarifni_pasma.

[3] Tomáš Bayer. Rovinné triangulace a jejich využití. [online], [cit. 2021-04-12]. Dostupné z: https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk5.pdf.

[4] Google Developers. Gtfs static overview. [online], [cit. 2021-03-11]. Dostupné z: https://developers.google.com/transit/gtfs/.

[5] TransitWiki users. General transit feed specification. [online], [cit. 2021-04-12]. Dostupné z: https://www.transitwiki.org/TransitWiki/index.php/General_Transit_Feed_Specification#Creating_and_Maintaining_a_GTFS_Dataset.

[6] Pražská integrovaná doprava. O systému|pražská integrovaná doprava. [online], [cit. 2021-03-11]. Dostupné z: https://pid.cz/o-systemu/.

[7] Přispěvatelé Wikipedie. Ropid. [online], [cit. 2021-04-18]. Dostupné z: https://cs.wikipedia.org/wiki/ROPID.

[8] QGIS Development Team. Qgis. [online], [cit. 2021-04-16]. Dostupné z: https://www.qgis.org/.

[9] Přispěvatelé Wikipedie. Programovací paradigma. [online], [cit. 2021-04-16]. Dostupné z: https://cs.wikipedia.org/wiki/Programovac%C3%AD_paradigma.

[10] Python Software Foundation. Python. [online], [cit. 2021-04-16]. Dostupné z: https://www.python.org/.

[11] QGIS. Třída qgstask. [online], [cit. 2021-03-25]. Dostupné z: https://qgis.org/pyqgis/3.16/core/QgsTask.html#qgis.core.QgsTask.

[12] QGIS. Qgis dokumentace. [online], [cit. 2021-03-29]. Dostupné z: https://docs.qgis.org/3.16/en/docs/index.html.

[13] Tomáš Bayer. Voronoi diagram. vlastnosti, použití, konstrukce. zobecněné voronoi diagramy. [online], [cit. 2021-03-23]. Dostupné z: https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk6.pdf.

[14] Tomáš Bayer. Kartografické generalizační algoritmy generace bodových prvků/skupin. generalizace liniových prvků/skupin. generalizace plošných prvků/skupin. [online], [cit. 2021-04-08]. Dostupné z: https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk8.pdf.

[15] Kenneth I. Joy. Chaikin's algorithms for curves. [online], [cit. 2021-04-11]. Dostupné z: https://www.cs.unc.edu/~dm/UNC/COMP258/LECTURES/Chaikins-Algorithm.pdf.

Seznam příloh