Technická zpráva úkol č. 4: Energetické spliny

Miroslav Hrubeš, Lucie Peterková

Praha 2023

1 Zadání

Úloha č. 4: Energetické spliny
S využitím generalizačního operátoru Partial Modification realizujte odsun a částečnou změnu tvaru jednoho blízkého prvku vůči blízkému pevnému prvku (bariéře), tak aby v generalizované mapě nedošlo k jejich grafickému konfliktu. Hodnotu minimální vzdálenosti prvků d volte v závislosti na měřítkovém čísle mapy (např. 1 mm v mapě). Pro implementaci použijte metodu energetických splinů.

Jako vstupní data použijte existující kartografická data (např. silniční síť, železniční síť, vodstvo), která budou načtena ze dvou textových souborů ve vašem zvoleném formátu.

Grafické rozhraní realizujte s využitím frameworku QT, výsledky generalizačních operací vizualizujte. Porovnejte dosažené výsledky s ruční generalizací prováděnou kartografickým expertem.

2 Bonusové úlohy

Nebyly řešeny žádné bonusové úlohy.

3 Popis a rozbor problému

3.1 Energetické spliny

Energetické spliny jsou v digitální kartografii využívány k transformaci původních dat do nových, generalizovaných dat. Cílem je minimalizovat deformace v původních datech, zatímco zajišťujeme, aby výsledná generalizovaná mapa měla předem stanovenou minimální vzdálenost mezi prvky.

Tento problém vzniká při generalizaci digitálních map, kdy je potřeba zmenšit měřítko mapy, a tedy zobecnit jednotlivé prvky, aby mapa byla stále čitelná. Zároveň je ale nutné zachovat obecnou podobu původních dat a minimalizovat deformace, které mohou nastat při změně velikosti prvku.

Tato metoda měří míru ohybu nebo zakřivení křivky pomocí energie. Cílem je vytvořit co nejhladší křivku, která přesně reprezentuje datové body.

Pro úpravu funkce energie se využívá operátoru částečné modifikace, který upravuje tuto funkci v lokální oblasti kolem vybraného bodu, aby se výsledný energetický spline lépe přizpůsobil lokální geometrii modelované křivky. Na tvar výsledné křivky má vliv vnitřní (E_i) a vnější (E_e) energie splinu. Energetický model pro křivku L o délce l lze zapsat jako

$$E(d) = \int_0^l E_i(s), ds + \int_0^l E_e(s), ds$$

kde ds představuje parametrické vyjádření křivky.

Vnitřní energie (E_i) splinu definovaná vztahem ovlivňuje průběh splinu a jeho tvar. První člen měří vzdálenost splinu od původního elementu, druhý napětí (elasticitu) splinu a poslední tuhost (křivost) splinu. Vliv těchto faktorů je modelován s využitím trojice parametrů $\alpha(s)$, $\beta(s)$, $\gamma(s) \in \mathbb{R}^+$. Spline tedy může více či méně sledovat původní prvek.

Vnější energie (E_e) popisuje deformaci splinu způsobenou vnějšími silami. Energetická funkce, která popisuje silový model, může mít mnoho forem. Z matematického hlediska by měla být spojitá v bodě, diferencovatelná a mít jednoduchý průběh bez zbytečných oscilací. Její minimum je blízko svislé osy bufferu. Rozhodující faktory ovlivňující míru deformace představují gradient (strmost) a omezenost funkce shora. Čím větší jsou funkční hodnoty, tím silnější je jejich vliv na deformaci tvaru. Existuje mnoho způsobů, jak navrhnout přidruženou energetickou funkci. Z pohledu kartografické generalizace, jejíž cílem je realizace generalizační operace částečného posunutí, se snažíme nepřiblížit se k jinému prvku na vzdálenost menší než $\underline{d}$.

Požadavek minimalizace celkové energie splinu je vyjádřen následující rovnicí

$$E(d(s)) = \int_l F(s, d(s), d'(s), d''(s)), ds$$

Eulerova-Lagrangova rovnice pro tuto energii je

$$\frac{\partial y}{\partial F} - \frac{\partial}{\partial x}\left(\frac{\partial y'}{\partial F}\right) + \frac{\partial^2}{\partial x^2}\left(\frac{\partial y'}{\partial F}\right) - \ldots + (-1)^n \frac{\partial^n}{\partial x^n}\left(\frac{\partial y^n}{\partial F}\right) = 0$$

Po aplikaci Eulerovy-Lagrangovy rovnice na E(d) dostaneme následující rovnice

$$\frac{\partial y}{\partial F} = \alpha(s)d(s) + \nabla Ee(x(s), y(s))$$

$$\frac{\partial}{\partial x}\left(\frac{\partial y'}{\partial F}\right) = \frac{\partial}{\partial s}\left(\beta(s)\frac{\partial s}{\partial d(s)}\right) = \beta(s)\frac{\partial^2 d(s)}{\partial s^2}$$

$$\frac{\partial^2}{\partial x^2}\left(\frac{\partial y''}{\partial F}\right) = \frac{\partial^2}{\partial x^2}\left(\gamma(s)\frac{\partial^2 s}{\partial^2 d(s)}\right) = \gamma(s)\frac{\partial^4 d(s)}{\partial s^4}$$

kde je optimální řešení při rozepsání ve tvaru dvou diferenciálních rovnic

$$\alpha(s)\frac{dx(s)}{ds} + \beta(s)\frac{\partial^2 d^2 x(s)}{\partial s^2} - \gamma(s)\frac{\partial^4 d^4 x(s)}{\partial s^4} + \frac{\partial}{\partial x}E_e(x(s), y(s)) = 0,$$

$$\alpha(s)\frac{dy(s)}{ds} + \beta(s)\frac{\partial^2 d^2 y(s)}{\partial s^2} - \gamma(s)\frac{\partial^4 d^4 y(s)}{\partial s^4} + \frac{\partial}{\partial y}E_e(x(s), y(s)) = 0.$$

Při považování hodnot $\alpha(s), \beta(s), \gamma(s) \in \mathbb{R}^+$ za konstantní, Eulerovy rovnice jsou ve tvaru

$$\alpha x + \beta\frac{\partial^2 x}{\partial s^2} - \gamma\frac{\partial^4 x}{\partial s^4} + \frac{\partial}{\partial x}E_e(x(s), y(s)) = 0,$$

$$\alpha y + \beta\frac{\partial^2 y}{\partial s^2} - \gamma\frac{\partial^4 y}{\partial s^4} + \frac{\partial}{\partial y}E_e(x(s), y(s)) = 0.$$

Pokud je spline vzorkován s konstantním krokem h, lze použít také jeho diskrétní aproximaci, která je pro praktické výpočty vhodnější. Parciální derivace lze nahradit centrálními diferencemi

$$\frac{\partial^2 x_i}{\partial s^2} = \frac{1}{h^2}(x_{i-1} - 2x_i + x_{i+1}),$$

Po dosazení Eulerových rovnic získáme soustavu lineárních rovnic

$$\alpha x_i + \beta(x_{i-1} - 2x_i + x_{i+1}) + \gamma(x_{i-2} - 4x_{i-1} + 6x_i - 4x_{i+1} + x_{i+2}) + E_{e,x} = 0,$$

$$\alpha y_i + \beta(y_{i-1} - 2y_i + y_{i+1}) + \gamma(y_{i-2} - 4y_{i-1} + 6y_i - 4y_{i+1} + x_{i+2}) + E_{e,y} = 0,$$

kde E_{ix} a E_{iy} představují parciální derivace vnější energie podle proměnných x_i a y_i.

Aby bylo diskretizované řešení funkční, předpokládáme, že generalizovaná polylinie by měla mít co nejhladší průběh s dostatečně hustým a pokud možno konstantním krokem vzorkování.

3.2 Operace Partial Displacement

Tato generalizační operace, jejíž český ekvivalent je "částečná změna tvaru", provádí komplexní korekci tvaru a geometrické polohy generalizovaného prvku. Zahrnuje posun a změnu tvaru takových částí prvku, které se přiblíží k jinému prvku pod určitou mez danou hodnotou d. Tento generalizační operátor se často používá u prvků, které se v generalizované mapě ocitnou příliš blízko, a může tak dojít k jejich vzájemnému grafickému konfliktu (slití). Energetická funkce je navržena tak, aby nedocházelo k přiblížení na vzdálenost menší než $\underline{d}$

$$E_e(x, y) = \begin{cases} c(1 - \frac{d}{\underline{d}}), & \text{if } d < \underline{d}, \\ 0, \text{jinak.} \end{cases}$$

$\underline{d}$ v dané situaci značí minimální vzdálenost, kdy nedochází ke "slévání"dílčích prvků.Pro diskretizovanou variantu splinu se využívá parciálních derivací E_e podle x a y, kdy $q_n = [x_n, y_n]$ je nejbližším bodem k bodu $p = [x, y]$.

$$\frac{\partial E_e(x,y)}{\partial x} = -c\frac{x - x_n}{d}$$

$$\frac{\partial E_e(x,y)}{\partial y} = -c\frac{y - y_n}{d}$$

4 Vstupní a výstupní data

Vstupní data jsou v podobě množiny bodů uložené v textových souborech, kde jsou souřadnice x, y a z oddělené tabulátorem. Konkrétně se jedná o 2 textové soubory představující liniové prvky. Konkrétně se jedná o linie železnice mezi Častolovicemi a Rychnovem nad Kněžnou a řeky Kněžné.

Výstupem je grafické znázornění odsunu linií pomocí energetických splinů.

5 Aplikace

Pomocí skriptu *mainform.py* se spustí aplikace. Na widgetu (obr. 1) se nachází panel nástrojů, kde je možné pomocí ikon v tomto pořadí provést následující akce: *Element*: vložení liniového prvku elementu, *Barrier*: vložení liniového prvku elementu, *Displace 1 element*: odsun elementu od bariéry, *Settings*: nastavení dílčích parametrů energetických splinů a *Clear*: odstranění provedené analýzy a datového souboru z aplikace.

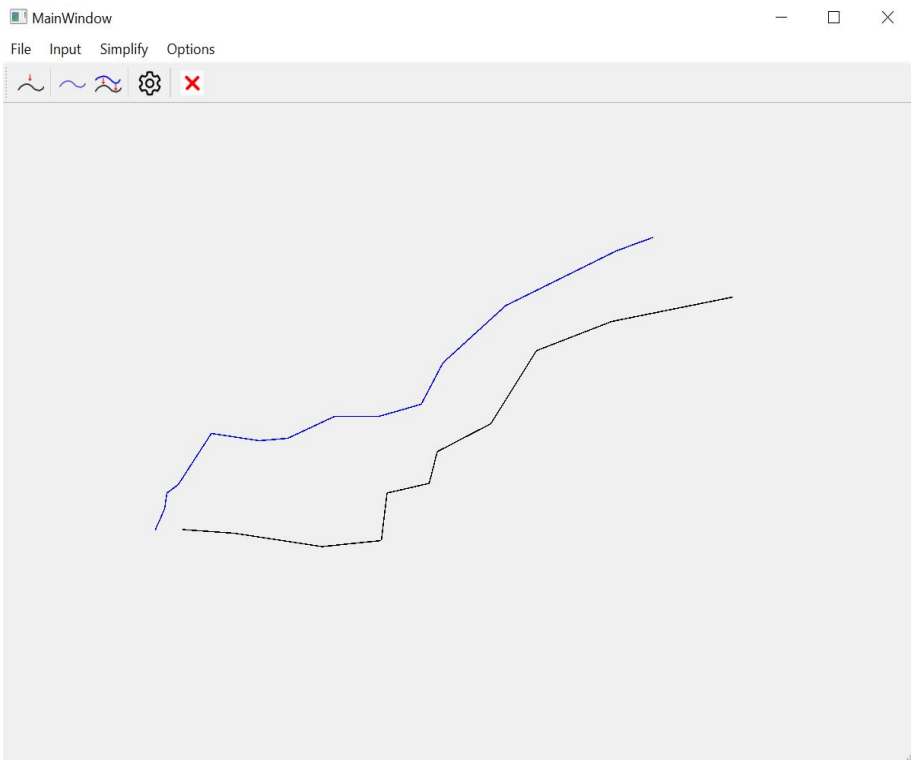
Obrázek 1: Ukázka aplikace se vstupními liniemi, zdroj: autoři.

6 Výsledky

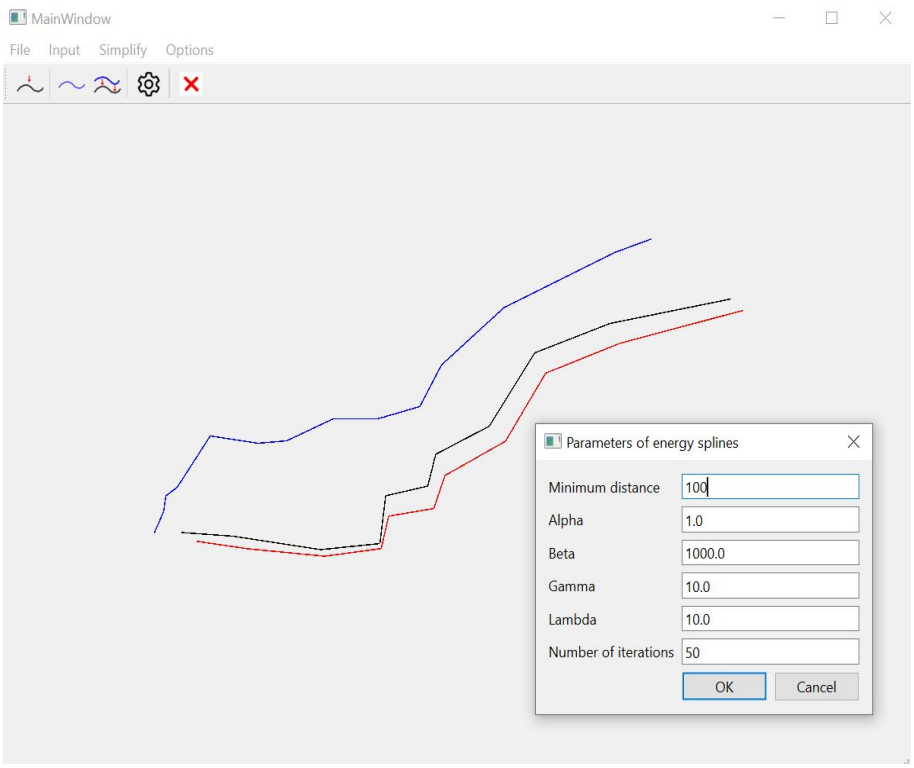
Aplikace programu na vstupní data nebyla bohužel úspěšná a vedla ve všech případech k pádu programu. Autoři zkusili část programu odpovědnou za odsunutí linie opravit a použít jiná vstupní dat. Také byla provedena změna velikosti načtených vstupních dat vůči oknu aplikace (v domnění, že pro odsunutí chybí v oknu aplikace prostor). Tyto pokusy však skončili bez výsledku.

Podařilo se alespoň identifikovat pravděpodobný důvod pádu programu. V případě, že se vstupní linie vzájemně kříží program spadne. Avšak při nekřížení (obr. 2) k pádu nedojde a program provede úspěšné odsazení (obr. 3).

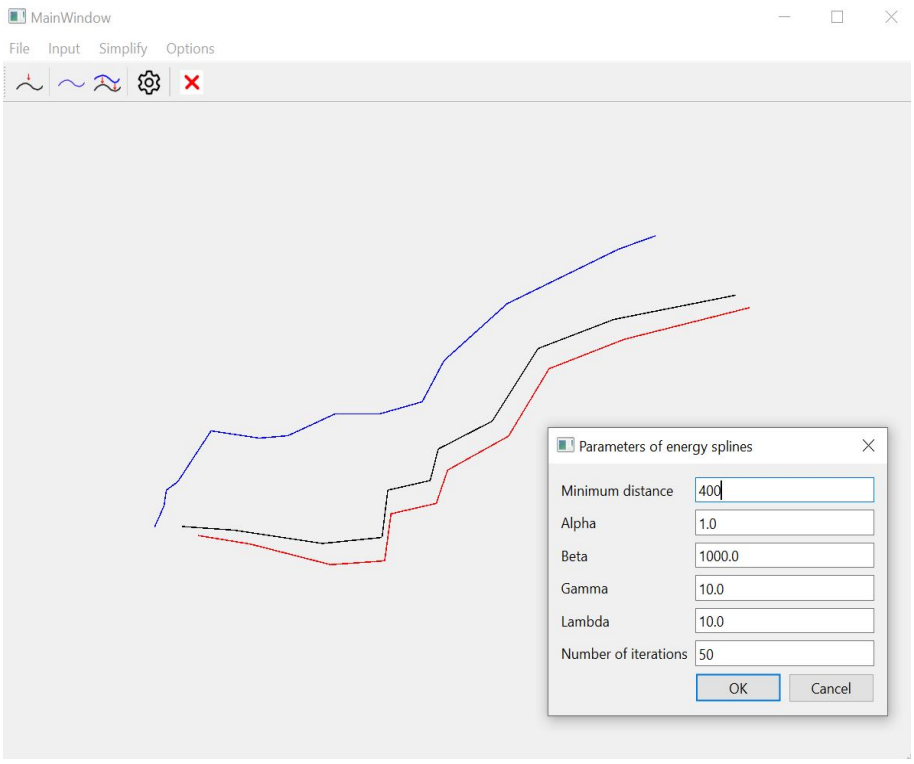
Dále byly provedeny změny parametrů a zhodnoceny změny ve vykreslení odsunuté linie. Při změně minimální vzdálenosti (obr. 4) ze 100 m na 400 m došlo k většímu odsazení od původní linie pouze v levé dolní části. Nejvýraznější změna (obr. 5) proběhla při snížení parametru Alfa, kdy došlo k významnému odsunu ve střední a pravé části linie. Zvláštní je však že nedošlo k odsunu v levé části linie. Při další změně parametru, v tomto případě zdvojnásobení parametru Beta, došlo pouze k nepatrné změně, a to k "vyhlazení"stočení v levé části linie. Parametr Gama, resp. jeho změna, neměla na následné vykreslení linie žádný vliv. U parametru Lambda (obr. 6) došlo k velice výrazné změně vykreslení. Při desetinásobné hodnotě se odsunutá linie po celé délce "přimknula"k původní linii a odsazení jako takové je velice nevýrazné. V případě poslední parametru, počtu iterací, nebyly změny jednoznačné. Při velmi malé hodnotě je odsazení linie velice malé. Pokud ale hodnotu postupně zvyšujeme do meze 20 nebo 30, odsazení velice výrazně narůstá. Při dalším zvyšování se další odsazení zvětšuje už velmi nepatrně.



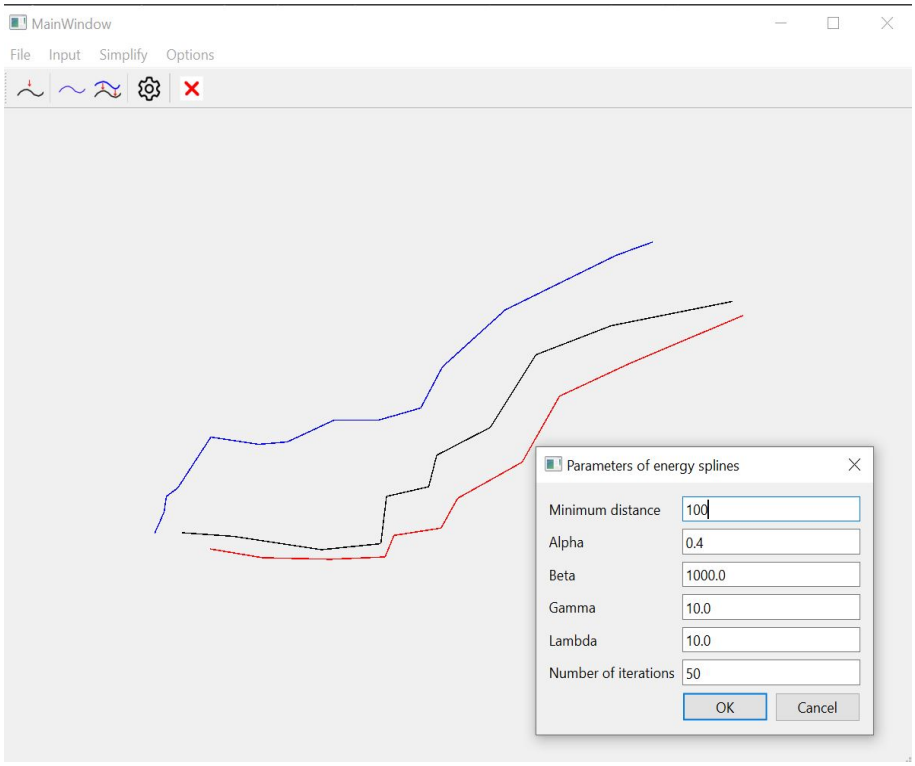
Obrázek 2: Ukázka aplikace se vstupními liniemi bez překřížení, zdroj: autoři.



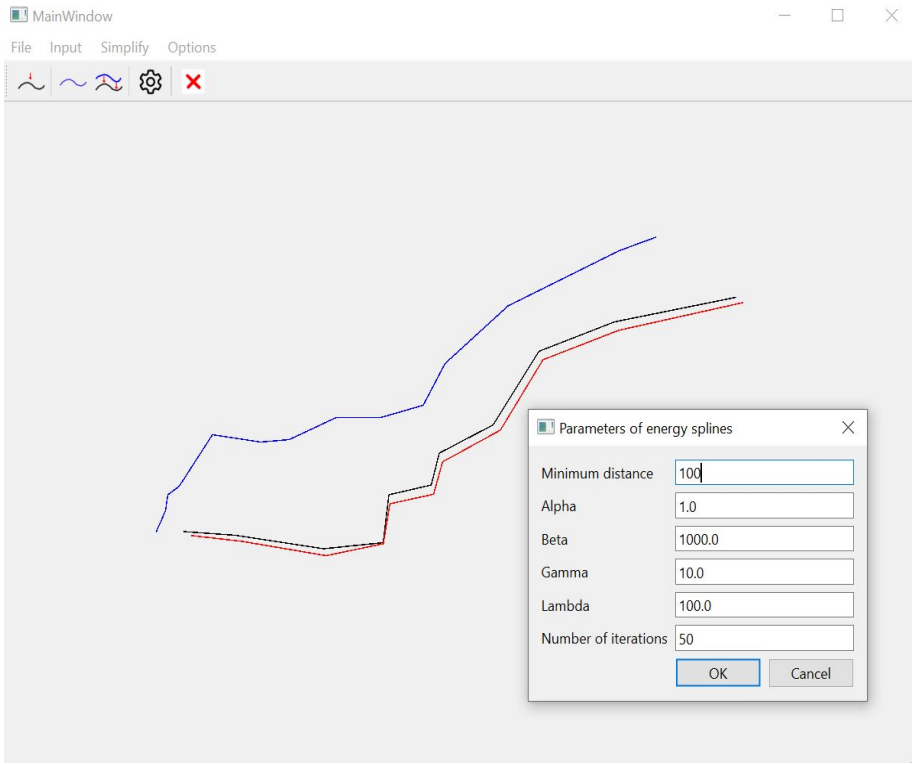
Obrázek 3: Výsledné odsazení (červeně) při spuštění programu, zdroj: autoři.



Obrázek 4: Výsledné odsazení (červeně) při zvýšení minimální vzdálenosti, zdroj: autoři.



Obrázek 5: Výsledné odsazení (červeně) při snížení paramteru Alfa, zdroj: autoři.



Obrázek 6: Výsledné odsazení (červeně) při zvýšení parametru Lambda zdroj: autoři.

7 Dokumentace

Aplikace je tvořena osmi skripty. Uživatelské rozhraní je tvořeno skriptem *mainform.py*, jehož základ byl vytvořen pomocí softwaru QT Creator. Vizualizaci a vykreslování objektů zajišťuje skript *draw.py*. Skript *algorithms.py* definuje matematické metody pro správné provedení zvolených analýz. Skript *QPoint3DF* definuje třídu QPoint3DF, která reprezentuje bod v 3D prostoru pomocí tří souřadnic (x, y, z). Skript *edge.py* definuje třídu Edge pro reprezentaci hrany mezi dvěma body v trojrozměrném prostoru pomocí třídy QPoint3DF a implementuje několik metod pro práci s touto třídou, jako jsou získání počátečního a koncového bodu, vytvoření nové hrany s opačnou orientací a porovnání dvou hran. Skript *setting.py* definuje dialogové okno pro nastavení parametrů energetických splinů. *mainwindow.py* slouží k vytvoření uživatelského rozhraní aplikace. *triangle.py* slouží k počítání sklonu a orientace trojúhelníku.

Metody skriptu *mainform.py* :

- **setupUi**: inicializuje a nastavuje různé prvky uživatelského rozhraní, jako např. velikost okna, hlavní widget, menu, lištu nástrojů a akce (např. otevření souboru, ukončení aplikace, vykreslení polygonu, analýza pozice bodů a polygonů apod.).
- **retranslateUi**: nastavení textů pro různé prvky uživatelského rozhraní.
- **settings**: Zobrazuje dialogové okno, které umožňuje uživateli nastavit parametry energiových splinů. Po potvrzení změn se získají hodnoty z dialogového okna a uloží se do příslušných proměnných.
- **inputL a inputB**: jsou volány při kliknutí na položky menu "Element"a "Barrier". Tyto metody slouží k nastavení cesty v závislosti na zvolené položce.
- **displaceClick**: provádí posunutí jednoho prvku. Nejprve se získají souřadnice prvku a bariéry. Poté se používá algoritmus pro výpočet minimální energie spline.
- **drawLineClick**: slouží k aktivaci kreslení elementu.
- **drawBarrierClick**: slouží k aktivaci kreslení bariéry.
- **clearClick**: slouží k vymazání všech elementů a bariér.
- **exit**: ukočuje celou aplikaci.

Metody skriptu *algorithms.py* :

- **getEuclidDistance**: vypočítá euklidovskou vzdálenost mezi dvěma body (x_1, y_1) a (x_2, y_2). Vrací vypočtenou vzdálenost.
- **getPointLineDistance**: vypočítá vzdálenost mezi bodem A $[x_a, y_a]$ a přímkou (p_1, p_2). Nejprve se vypočítá hodnota d_n a d_d pomocí metody *getEuclidDistance*. Poté se vypočítává hodnota d_1 a k. Nakonec se vypočítávají souřadnice bodu na přímce (x_q, y_q) nejbližšího bodu A na přímce (p_1, p_2). Metoda vrací hodnoty d (vzdálenost bodu A od přímky) a souřadnice x_q, y_q.
- **getPointLineSegmentDistance**: vypočítá vzdálenost mezi bodem $A[x_a, y_a]$a úsečkou (p_1, p_2). Nejprve se vypočítávají hodnoty souřadnic bodů $P_3 a P_4$. Poté se pomocí metody getPointLineDistance vypočítávají hodnoty $d_1 3, xq_3, yq_3, d_2 4, xq_4, yq_4$. Dále se testuje podmínka t, a pokud platí, vrací se hodnoty vypočtené pomocí metody getPointLineDistance pro přímku (p_1, p_2). Pokud neplatí, testují se podmínky $d_1 3$ a vrací se hodnoty vypočtené pomocí metody *getEuclidDistance* pro body p_1 a p_2.

- **getNearestLineSegmentPoint**: hledá na překážce bod nejblíže bodu A. Nejprve se inicializují proměnné $imin$ a $dmin$. Poté se vypočítává vzdálenost d_i a souřadnice x_i, y_i pro každou úsečku na překážce. Pokud je vzdálenost d_i menší než aktuální minimum $dmin$, aktualizuje se minimum a uloží se index úsečky a souřadnice nejbližšího bodu. Nakonec se vrací hodnoty $dmin$, $imin$, $xmin$, $ymin$.
- **createA**: vytváří matici A pro počítání energie. Nejprve se vypočítávají koeficienty a, b, c. Poté se vytvoří matice A a nastavují se hodnoty pro hlavní diagonálu a další diagonály.
- **getEx**: vypočítá parciální derivaci vnější energie podle x. Nejprve se vypočítává hodnota c. Pokud je vzdálenost d menší než $dmin$, vrací se hodnota parciální derivace. Jinak se vrací 0.
- **getEy**: vypočítá parciální derivaci vnější energie podle y. Nejprve se vypočítává hodnota c. Pokud je vzdálenost d menší než $dmin$, vrací se hodnota parciální derivace. Jinak se vrací 0.
- **minEnergy**: implementuje algoritmus pro výpočet minimální energie splinu. Nejprve se vytváří matice pro reprezentaci polylinie a překážky. Poté se vypočítává velikost kroku h. Následně se vytváří matice A pomocí metody $createA$ a vypočítává se inverze matice A. Dále se vytvářejí matice pro rozdíly. Iterační proces probíhá v cyklu, ve kterém se vypočítávají parciální derivace potenciálu a posuny bodů. Nakonec se výsledná polylinie převede z matice na seznam bodů a vrátí se jako výstup.

Metody skriptu *edge.py* :

- **getStart**: metoda, která vrací počáteční bod hrany.
- **getEnd**: metoda, která vrací koncový bod hrany.
- **switchOrientation**: metoda, která vytvoří novou hranu s opačnou orientací, tedy vymění počáteční a koncový bod.
- **eq**: metoda, která porovnává dvě hrany a vrací True, pokud jsou stejné (mají stejný počáteční a koncový bod), jinak vrací False

Metody skriptu *QPoint3DF.py* :

- **getZ**: Metoda, která vrací hodnotu souřadnice z objektu QPoint3DF.

Metody skriptu *triangle.py* :

- **getP1**: vrátí první vrchol trojúhelníku.
- **getP2**: vrátí druhý vrchol trojúhelníku.
- **getP3**: vrátí třetí vrchol trojúhelníku.
- **getSlope**: vrátí sklon trojúhelníku.
- **getAspect**: vrátí orientaci trojúhelníku.

Metody skriptu *mainwindow.py* :

- **loadui**: načtení uživatelského rozhraní ze souboru form.ui a jeho zobrazení v okně aplikace.

Metody skriptu *draw.py* :

- **setPath**: slouží k načtení datového souboru pomocí dialogového okna. Přijímá parametry width a height, které určují rozměry okna. Dále určuje, zda se mají načíst body pro L

(False) nebo B (True). Metoda umožňuje uživateli vybrat soubor pomocí dialogového okna a načte souřadnice bodů z vybraného souboru. Následně přepočítá souřadnice bodů tak, aby se vešly do okna aplikace. Načtené body jsou uloženy v příslušném polygonu.

- **switchSource**: sloužík přepnutí mezi pohybem bodu a přidáváním vrcholů.
- **getL**: metoda pro získání polygonu L.
- **getL**: metoda pro získání polygonu B.
- **setLD**: metoda pro nastavení polygonu LD.
- **setSource**: metoda pro nastavení zdroje bodů. Přijímá parametr status, který určuje, zda jsou body přidávány do polygonu L (True) nebo do polygonu B (False).
- **clearAll**: metoda pro vymazání všech polygonů.

8 Závěr

V tomto úkolu byla vytvořena aplikace využívající generalizační operátor Partial Modification. Ten umožňuje provést odsun a částečnou změnu jednoho blízkého prvku vzhledem k bariéře, aby nedocházelo k jejich grafickému konfliktu. Uživatel má možnost aplikovat tuto metodu na svá data, která načte do aplikace. Součástí aplikace je také možnost nastavit hodnoty parametrů pro tvorbu energetických splineů. V současné verzi aplikace je partial modification dostupná pouze pro jeden prvek a překážku.

Aplikace je bohužel funkční pouze částečně. Dochází k pádu programu při vstupních dat, které obsahují vzájemně se křížící linie. V případě nekřížících se linií je však odsun linie proveden správně a různá změna parametrů vytváří různé výsledky odsunu. Jako další možné vylepšení programu lze spatřovat v opravě programu, aby dokázal zpracovat i křížící se linie. Dalším vylepšením může být také možnost načítat mnohem více vstupních dat a provádět několik odsunů najednou.

9 Zdroje

Přednášky z předmětu *Algoritmy z počítačové kartografie.*