

České vysoké učení technické v Praze

Fakulta stavební



Algoritmy v digitální kartografii

Úloha č. 4: Množinové operace s polygony

Skupina:

Sabina Kličková

Martin Vajner

Zimní semestr 2021/2022

I. Obsah

1. Zadání	3
2. Bonusové úlohy	3
3. Popis a rozbor problémů.....	3
4. Popisy algoritmů	4
6. Vstupní data, formát vstupních dat, popis	5
7. Výstupní data, formát výstupních dat, popis	6
Výstupem úlohy je aplikace, ve které lze vyhodnocovat vzájemnou polohu dvou polygonů spolu s výsledky množinových operací.	Chyba! Záložka není definována.
8. Dokumentaci: popis tříd, datových položek a jednotlivých metod	8
10. Citovaná literatura	10
11. Seznam obrázků.....	10

1. Zadání

Vstup: množina n polygonů $P = \{P_1, \dots, P_n\}$.

Výstup: množina m polygonů $P_0 = \{P_{01}, \dots, P_{0m}\}$.

S využitím algoritmu pro množinové operace s polygony implementujte pro libovolné dva polygony $P_i, P_j \in P$ následující operace:

- Průnik polygonů $P_i \cap P_j$,
- Sjednocení polygonů $P_i \cup P_j$,
- Rozdíl polygonů: $P_i \cap P_j$, resp. $P_j \cap P_i$.

Jako vstupní data použijte existující kartografická data (např. konvertované shape fily) či syntetická data, která budou načítána z textového souboru ve Vámi zvoleném formátu. Grafické rozhraní realizujte s využitím frameworku QT. Při zpracování se snažte postihnout nejčastější singulární případy: společný vrchol, společná část segmentu, společný celý segment či více společných segmentů. Ošetřete situace, kdy výsledkem není 2D entita, ale 0D či 1D entita. Pro výše uvedené účely je nutné mít řádně odladěny algoritmy z úlohy 1. Postup ošetření těchto případů diskutujte v technické zprávě, zamyslete se nad dalšími singularitami, které mohou nastat.

Množinové operace, průnik, sjednocení, rozdíl	+20b
---	------

2. Bonusové úlohy

V této úloze nebyly zpracovány žádné bonusové úlohy:

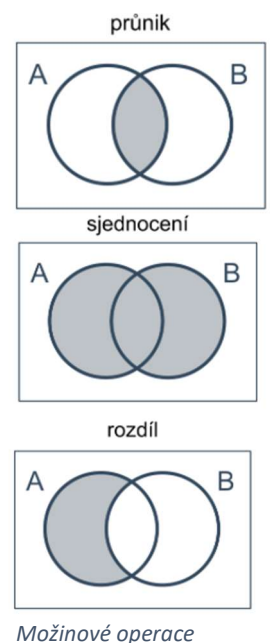
3. Popis a rozbor problémů

Existují-li dva polygony P_i, P_j , je třeba zjistit jejich vzájemnou polohu. K tomu se použijí následující množinové operace:

- Průnik $P_i \cap P_j$,
- Sjednocení $P_i \cup P_j$,
- Rozdíl $P_i \cap P_j$, resp. $P_j \cap P_i$.

Popis jednotlivých operací:

- **Průnik:** Výsledkem operace je množina, která obsahuje pouze prvky, které se nachází v obou množinách zároveň.
- **Sjednocení:** Výsledkem operace je množina, která obsahuje všechny prvky, nacházející se alespoň v jedné z množin, se kterými pracujeme.
- **Rozdíl:** Výsledkem operace je množina, která obsahuje prvky nacházející se v jedné množině, ale nenacházející se v te druhé.

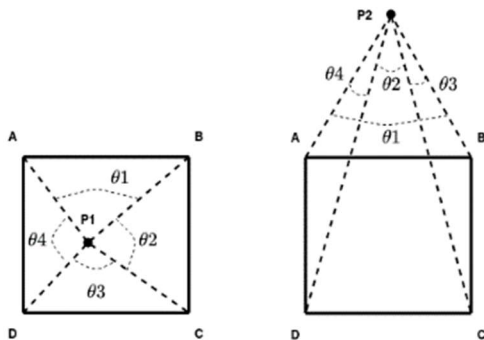


4. Popisy algoritmů

I. Winding Number

Algoritmus Winding Number je definován jako počet rotací křivky, či v našem případě polygonu, v jednom směru okolo daného bodu q . Výpočet je založen na zjištění hodnot úhlů ω mezi daným bodem q a body polygonu a na poloze bodu q vůči segmentu polygonu. Pro všechny segmenty je třeba zjistit, jestli se bod nachází nalevo, či napravo. Pokud jsou nalevo, tak se úhly přičítají, pokud napravo tak se odečítají. Toto platí pro směr ccw (counter clockwise). Pro opačný směr se znaménka prohodí. Pokud se suma všech úhlů nerovná 0, leží bod uvnitř a pokud je suma rovna 0, leží bod mimo polygon. (1)

Princip algoritmu:



Princip Winding Number algoritmu

- Inicializuj $\Omega=0$, tolerance ε .
- Opakuj pro T trojici (p_i, q_i, p_{i+1})

Urči polohu q vzhledem k $p = (p_i, p_{i+1})$

Urči úhel $\omega_i = \angle p_i, q_i, p_{i+1}$

If $q \in \bar{\sigma}_l$, pak $\Omega = \Omega + \omega_i$ //Bod v levé polorovině

Else $\Omega = \Omega - \omega_i$ //Bod v pravé polorovině
- If $||\omega|-2\pi| < \varepsilon$, pak $q \in P$
- Else $q \notin P$

II. Výpočet průsečíku polygonů A,B

Algoritmus slouží k výpočtu polohy průsečíku hran polygonů. Ty jsou ukládány společně s hodnotami α, β , které určují polohu a hodnotu danou průsečákem. S každým dalším nalezeným bodem je seznam aktualizován a řazen podle hodnot α, β . Poledním krokem je implementace algoritmu Winding number, který určuje vzájemnou polohu vrcholů polygonů.

Princip algoritmu:

- for($i=0$; $i < n$; $i++$)
 - Vytvoření mapy: $M = \text{map}(\text{double}, \text{QPointFBO})$
 - for($j=0$; $j < m$; $j++$)
 - Pokud existuje průsečík: if $(b_{ij} = (p_i, p_{(i+1)\%m}) \cap (q_j, q_{(j+1)\%m}) \neq \emptyset)$
 - Přidání do mapy M : $M[\alpha_i] < -b_{ij}$
 - Zpracování prvního průsečíku pro e_j : $\text{processIntersection}(b_{ij}, \beta, B, j)$
- Pokud jsou nalezeny nějaké průsečíky: if($||M|| > 0$)
 - Procházení všech průsečíků v mapě: for($\forall m \in M$):

- i. Získání 2. hodnoty páru: $b \leftarrow m.druhá$
- ii. Zpracování aktuálního průsečíku pro e_i : $\$processIntersection(b, \alpha, A, i)$

processIntersection:

- a. Jestliže ϵ je minimální hodnota: $if(|t| > \epsilon \text{ AND } |t - 1| > \epsilon)$
 - a. Inkrementace pozice: $i \leftarrow i+1$
 - b. Přidání průsečíku na pozici $i+1$: $P \leftarrow (b, i)$

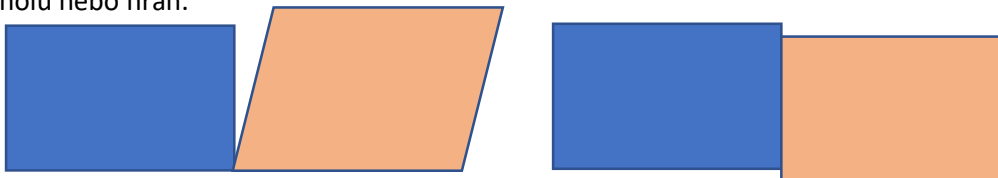
III. Ohodnocení hran

Pro ohodnocení hran byl vytvořen algoritmus `setEdgePositions`. Rozdeluje pozice středů hran ku polygonu na vnitřní či vnější.

- a. $for(i=0; i < n; i++)$ n ..počet vrcholů prvního polygonu
- b. Určení středu hrany $M(x_m, y_m)$:
 - a. $x_m = \frac{x_i + x_{i+1}}{2}$
 - b. $y_m = \frac{y_i + y_{i+1}}{2}$
 - c. Určení pozice: $pos = GetPositionWinding(M, B)$
 - d. Úprava pozice $A(i) \leftarrow pos$

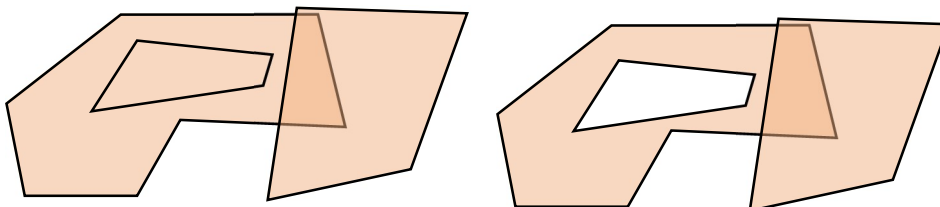
5. Problematické situace

Problematickou situací je stav, kdy dva vstupní polygony mají právě jeden/jednu, či více společných vrcholů nebo hran.



Ukázka problematické situace

Další problematickou situací mohou být nekonvexní polygony s „otvorem“. U těch může docházet k chybnému vyhodnocení množinových operací. Výsledek napravo je ten správný. Naše aplikace není schopna tento problém ovšem řešit, jelikož tato bonusová úloha nebyla řešena.



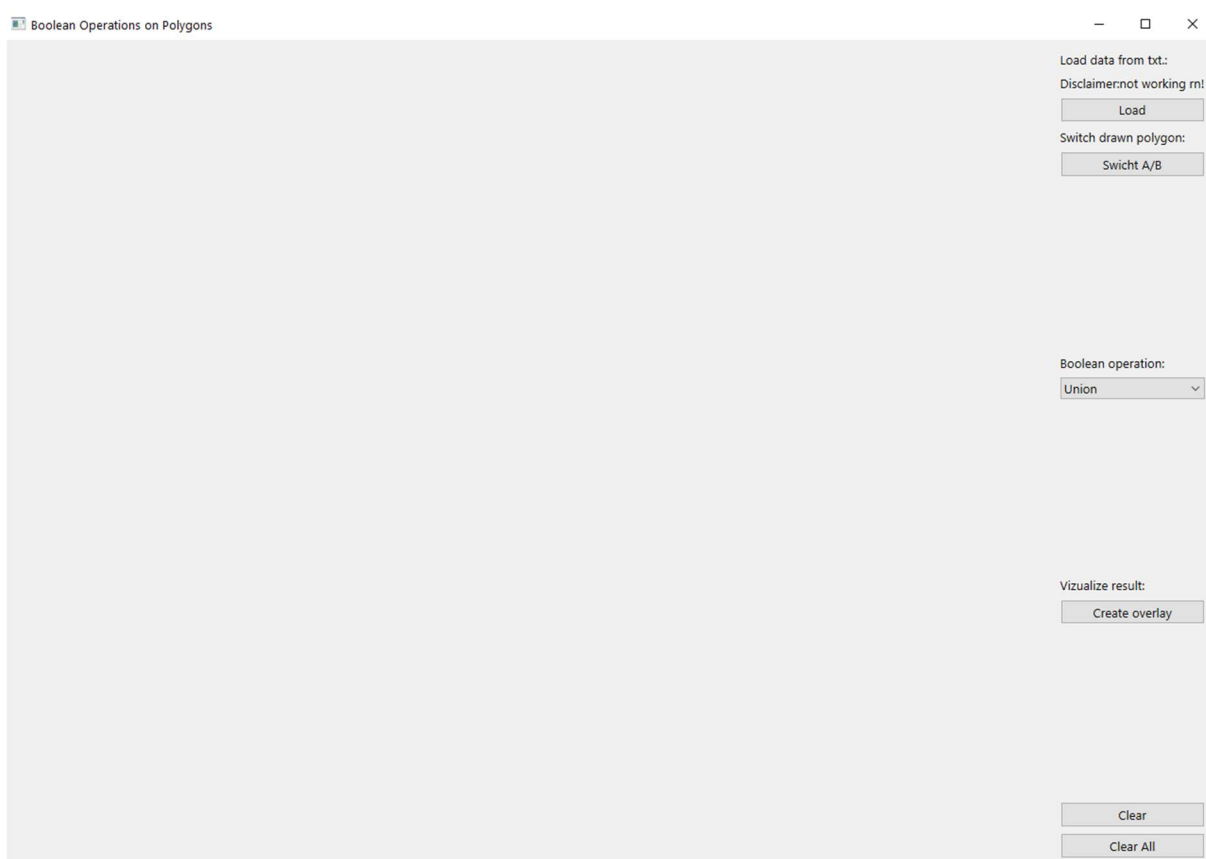
Ukázka problematické situace

6. Vstupní data, formát vstupních dat, popis.

Vstupní data jsou datového typu bod. V současné době lze zadávat data pouze ručním kreslením.

7. Výstupní data, formát výstupních dat, popis

Výstupem je aplikace, která dokáže nad nakreslenými daty spočítat množinové operace a ty následně vizuálně zobrazit. Obsahuje tlačítko Switch pro změnu kreslení polygonu A/B, tlačítko pro výběr množinové operace, tlačítko pro vizuální zobrazení výsledku a dvě tlačítka na smazání obsahu okna a smazání všech hodnot proměnných.



Okno aplikace po spuštění



Okno aplikace po načtení dat



Okno aplikace po vizualizaci množinové operace

8. Dokumentaci: popis tříd, datových položek a jednotlivých metod

class Algorithms

- `TPointLinePosition` `getPointLinePosition`(QPointFBO &a, QPointFBO &p1, QPointFBO &p2);
- zjištění vzájemné polohy bodu a přímky
- `double` `get2LinesAngle`(QPointFBO &p1, QPointFBO &p2, QPointFBO &p3, QPointFBO &p4);
- zjištění hodnoty úhlu mezi dvěma přímkami
- `TPointPolygonPosition` `getPositionWinding`(QPointFBO &q, TPolygon &pol);
- zjištění polohy bodu ku polygonu
- `std::tuple<QPointFBO, T2LinesPosition>` `get2LinesIntersection`(QPointFBO &p1, QPointFBO &p2, QPointFBO &p3, QPointFBO &p4);
- nalezení průsečíku přímek
- `void` `updatePolygons`(TPolygon &A, TPolygon &B);
- aktualizace polygonů novým průsečíkem
- `void` `processIntersection`(QPointFBO &b, double t, int &index, TPolygon &P);
- vložení společných bodů
- `void` `setEdgePositions`(TPolygon &A, TPolygon &B);
- nastavení pozice hrany ku polygonu
- `void` `selectEdges`(TPolygon &P, TPointPolygonPosition pos, TEdges &edges);
- výběr hran
- `TEdges` `createOverlay`(TPolygon &A, TPolygon &B, TBooleanOperation &op);
- vytvoření vrstvy polygonů podle zvolené množinové operace

class Draw

private:

- `TPolygon` A, B;
- zhodnocované polygony
- `TEdges` res;
- vybrané hrany
- `bool` addA;
- změna kreslení polygonu A/B
- public:
- `void` `paintEvent`(QPaintEvent *event);
- vykreslování na plátno
- `void` `mousePressEvent`(QMouseEvent *event);
- metoda pro vyhodnocování kliknutí na myši
- `void` `switchSource`() { addA = !addA; }
- změna vykreslování polygonů A/B
- `void` `drawPolygon`(TPolygon &pol, QPainter &qp);
- vykreslování polygonů
- `TPolygon` `getA`() { return A; }
- získání polygonu A
- `TPolygon` `getB`() { return B; }
- získání polygonu B
- `void` `setEdges`(TEdges &edg) { res = edg; }
- nastavení hran
- `void` `clear`() { res.clear(); }
- Smazání vykreslovacího okna
- `void` `clearAll`() { A.clear(); B.clear(); res.clear(); }
- Smazání všech proměnných a okna
- `void` `loadFile`(std::string &path);
- Načtení souřadnicového textového souboru


```

class Edge
private:
    QPointFBO start, end;
- počáteční a koncový bod hrany
public:
    void setStart(QPointFBO &start_){start=start_;}
- nastavení počátku hrany
    void setEnd(QPointFBO &end_){end=end_;}
- nastavení konce hrany
    QPointFBO getStart(){return start;}
- získání počátku hrany
    QPointFBO getEnd(){return end;}
- získání konce hrany

class QPointFBO
private:
    double alpha, beta;
- hodnoty průsečíku hran
    TPointPolygonPosition pos;
- poloha bodu ku polygonu
public:
    double getAlpha(){return alpha;}
- získání hodnoty alfa
    double getBeta(){return beta;}
- získání hodnoty beta
    TPointPolygonPosition getPosition(){return pos;}
- získání polohy bodu ku polygonu
    void setAlpha(double alpha_){alpha=alpha_;}
- nastavení hodnoty alfa
    void setBeta(double beta_){beta=beta_;}
- nastavení hodnoty beta
    void setPosition(TPointPolygonPosition pos_){pos=pos_;}
- nastavení polohy bodu ku plygonu

class Types
typedef enum{
    LeftHP,
    RightHP,
    On
} TPointLinePosition;
- pozice bodu ku hraně

typedef enum{
    Inner,
    Outer,
    Boundary
} TPointPolygonPosition;
- pozice bodu ku polygonu

typedef enum {
    Union,
    Intersection,
    DifferenceA_B,
    DifferenceB_A
} TBooleanOperation;
- definice množinové operace

```

```
typedef enum{  
    Colinear,  
    Parallel,  
    Intersect,  
    NonIntersect  
} T2LinesPosition;  
- pozice dvou linií vůči sobě  
-  
typedef std::vector<QPointFBO> TPolygon;  
- definice polygonu tvořeného z bodů QPointFBO  
  
typedef std::vector<Edge> TEdges;  
- definice hran
```

9. Závěr

Námi vytvořená aplikace dokáže zhodnotit vzájemnou polohu dvou polygonů P_i , P_j pomocí základních množinových operací. Dokáže pracovat s polygony vytvořenými ručně. Výsledek množinové operace je zvýrazněn červenou barvou.

10. Náměty na vylepšení

V aplikaci se bohužel nepodařilo zprovoznit načítání dat ze souboru. Data se načtou do požadovaných proměnných A, B ale poté se už nevykreslí. Vzhledem k malé dotaci času během zkouškového období nebylo věnováno řešení tohoto problému více času.

11. Citovaná literatura

1. **Tomáš, Bayer.** Perslonal page of Bayer Tomas. *Charles University of Prague*. [Online] [Citace: 05. 12 2021.] <https://web.natur.cuni.cz/~bayertom/index.php/teaching/algoritmy-v-digitalni-kartografii>.

12. Seznam obrázků

<i>Množinové operace</i>	3
<i>Princip Winding Number algoritmu</i>	4
<i>Ukázka problematické situace</i>	5
<i>Okno aplikace po spuštění</i>	6
<i>Okno aplikace po načtení dat</i>	7
<i>Okno aplikace po vizualizaci množinové operace</i>	7

V Praze dne 10.1.2022